T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI İLE YOKLAMA AMAÇLI YÜZ TANIMA SİSTEMİ

DEMET HANİFE SUNGUR

YÜKSEK LİSANS TEZİ
ELEKTRİK ELEKTRONİK VE BİLGİSAYAR MÜHENDİSLİĞİ
ANABİLİM DALI

DANIŞMAN
PROF. DR. PAKİZE ERDOĞMUŞ

DÜZCE, 2022

T.C.
DÜZCE ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI İLE YOKLAMA AMAÇLI YÜZ TANIMA SİSTEMİ

Demet Hanife SUNGUR tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Prof. Dr. Pakize ERDOĞMUŞ

Düzce Üniversitesi

Jüri Üyeleri

Prof. Dr. Pakize ERDOĞMUŞ
Düzce Üniversitesi _____

Doç. Dr. Abdullah Talha KABAKUŞ
Düzce Üniversitesi _____

Doç. Dr. Devrim AKGÜN
Sakarya Üniversitesi _____

Tez Savunma Tarihi: 10/08/2022

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

10 Ağustos 2022

Demet Hanife SUNGUR

TEŞEKKÜR

Yüksek lisans öğrenimimde ve bu tezin hazırlanmasında gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Prof. Dr. Pakize ERDOĞMUŞ'a en içten dileklerimle teşekkür ederim.

Bu çalışma boyunca yardımlarını ve desteklerini esirgemeyen sevgili aileme ve çalışma arkadaşlarıma sonsuz teşekkürlerimi sunarım.

10 Ağustos 2022 **Demet Hanife SUNGUR**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ vii

ÇİZELGE LİSTESİ viii

KISALTMALAR ix

ÖZET x

ABSTRACT xi

1. GİRİŞ 1

2. LİTERATÜR ÖZETİ ve GENEL BİLGİLER 4

2.1. YAPAY SİNİR AĞI 8

2.1.1. Aktivasyon Fonksiyonu 11

2.1.1.1. *Sigmoid Aktivasyon Fonksiyonu 12*

2.1.1.2. *Hiperbolik Tanjant (Tanh) Aktivasyon Fonksiyonu 13*

2.1.1.3. *ReLu Aktivasyon Fonksiyonu 13*

2.1.1.4. *Leaky (Sızdıran) ReLU Aktivasyon Fonksiyonu 14*

2.1.1.5. *Softmax Aktivasyon Fonksiyonu 14*

2.2. DERİN ÖĞRENME 15

3. MATERYAL VE YÖNTEM 17

3.1. EVRİŞİMLİ SİNİR AĞI (CONVOLUTIONAL NEURAL NETWORK, CNN) 17

3.1.1. Evrişim (Konvolüsyon, Convolution) Katmanı 19

3.1.2. ReLU (Doğrultulmuş Doğrusal Birim, Rectified Linear Unit) 20

3.1.3. Havuzlama (Pooling) Katmanı 20

3.1.4. Tam Bağlı Katman (Fully Connected Layer) 21

3.1.5. Softmax Katmanı 21

3.2. EVRİŞİMLİ SİNİR AĞI MİMARİLERİ 22

3.2.1. AlexNet 22

3.2.2. GoogLeNet 23

3.2.3. ResNet 25

3.3. EVRİŞİMLİ SİNİR AĞI VE YÜZ TANIMA 28

3.4. BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (REGION BASED CONVOLUTIONAL NEURAL NETWORK, R-CNN) 33

3.5. HIZLI BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (FAST REGION BASED CONVOLUTIONAL NEURAL NETWORK, FAST R-CNN) 36

3.6. DAHA HIZLI BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (FASTER REGION BASED CONVOLUTIONAL NEURAL NETWORK, FASTER R-CNN) 38

3.7. SADECE BİR KEZ BAK (YOU LOOK ONLY ONCE, YOLO) 40

3.8. VERİ SETİ 45

3.9. DENEYSEL ÇALIŞMALAR 46

3.9.1. Doğrulama 47

3.9.2. Eğitimde Kullanılan Hiperparametreler 49

3.9.2.1. *Filtre Boyutu 49*

3.9.2.2. *Öğrenme Hızı (Initial Learn Rate) 49*

3.9.2.3. *Optimizasyon Algoritmaları* *50*
3.9.2.4. *Mini Yığın Boyutu (Mini Batch Size)* *50*
3.9.2.5. *Eğitim Turu (Epoch) Sayısı* *51*
3.9.3. Transfer Öğrenme (Transfer Learning) **51**
3.9.4. Eğitim **54**
3.9.5. Test **69**
3.10. SONUÇLARIN DEĞERLENDİRİLMESİ **69**
4. BULGULAR **72**
4.1. R-CNN İLE ALEXNET, GOOGLENET VE RESNET50 SONUÇLARI **72**
4.2. GOOGLENET İLE ADAM VE SGDM ALGORİTMALARININ SONUÇLARI **76**
4.3. ALEXNET İLE ARTTIRMA SONUÇLARI **78**
4.4. ALEXNET İLE 350/50 VERİ SETİ SONUÇLARI **82**
4.5. FASTER R-CNN SONUÇLARI **83**
4.6. YOLOv4 SONUÇLARI **85**
4.7. TÜM UYGULAMA SONUÇLARININ KARŞILAŞTIRILMASI **87**
4.8. KULLANICI ARAYÜZÜ **89**
5. SONUÇLAR VE ÖNERİLER **92**
6. KAYNAKLAR **95**
ÖZGEÇMİŞ **108**

ŞEKİL LİSTESİ

Sayfa No

Şekil 2.1. Yapay sinir ağına örnek bir yapı 8
Şekil 2.2. Yapay sinir ağında aktivasyon fonksiyonu 12
Şekil 2.3. Sigmoid aktivasyon fonksiyonu 13
Şekil 2.4. ReLU aktivasyon fonksiyonu 14
Şekil 2.5. Softmax aktivasyon fonksiyonu 15
Şekil 3.1. Evrişimli sinir ağı genel mimarisi 18
Şekil 3.2. AlexNet mimarisi [8] 22
Şekil 3.3. GoogLeNet mimarisi [35] 24
Şekil 3.4. ResNet mimarisi [63] 26
Şekil 3.5. ResNet50 mimarisi [65] 28
Şekil 3.6. R-CNN dedektör algoritmasının çalışma prensibi 35
Şekil 3.7. Fast R-CNN dedektör algoritmasının çalışma prensibi 37
Şekil 3.8. Faster R-CNN dedektör algoritmasının çalışma prensibi 40
Şekil 3.9. YOLO ağ mimarisi [97] 41
Şekil 3.10. Veri setinden örnek bir fotoğraf 46
Şekil 3.11. Etiketleme örneği 47
Şekil 3.12. Çalışmada kullanılan AlexNet mimarisi 57
Şekil 3.13. Çalışmada kullanılan GoogleNet mimarisi 60
Şekil 3.14. Çalışmada kullanılan ResNet50 mimarisi 65
Şekil 4.1. Test veri setinde başarılı tanınan öğrencilerin görüntü ve etiketleri 72
Şekil 4.2. AlexNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği 73
Şekil 4.3. GoogleNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği 73
Şekil 4.4. ResNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği 73
Şekil 4.5. AlexNet, GoogleNet ve ResNet50 mimarilerinin karşılaştırılması 75
Şekil 4.6. GoogleNet SGDM algoritması ile elde edilen ortalama kesinlik (AP) grafiği 76
Şekil 4.7. GoogleNet mimarisinde ADAM ve SGDM algoritmalarının karşılaştırılması 78
Şekil 4.8. AlexNet arttırma 1 ile elde edilen ortalama kesinlik (AP) grafiği 78
Şekil 4.9. AlexNet arttırma 2 ile elde edilen ortalama kesinlik (AP) grafiği 79
Şekil 4.10. AlexNet arttırma 3 ile elde edilen ortalama kesinlik (AP) grafiği 79
Şekil 4.11. AlexNet arttırma 4 ile elde edilen ortalama kesinlik (AP) grafiği 79
Şekil 4.12. AlexNet arttırma uygulamalarının karşılaştırılması 81
Şekil 4.13. AlexNet 350/50 veri seti ile elde edilen ortalama kesinlik (AP) grafiği 82
Şekil 4.14. Faster R-CNN ile elde edilen ortalama kesinlik (AP) grafiği 85
Şekil 4.15. YOLOv4 ile elde edilen ortalama kesinlik (AP) grafiği 86
Şekil 4.16. YOLOv4 350/50 veri seti ile elde edilen ortalama kesinlik (AP) grafiği 86
Şekil 4.17. Farklı uygulamalar ile elde edilen ortalama kesinlik (mAP) değerleri 89
Şekil 4.18. Kullanıcı arayüzü giriş ekranı 90
Şekil 4.19. Kullanıcı arayüzü görüntü yükleme 91
Şekil 4.20. Kullanıcı arayüzü sonuç ekranı 91

ÇİZELGE LİSTESİ

Sayfa No

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri 29
Çizelge 3.2. Deneysel çalışmalarda kullanılan bilgisayar konfigürasyonu 47
Çizelge 3.3. R-CNN Object Detector eğitim seçenekleri 54
Çizelge 3.4. AlexNet mimarisinde kullanılan katmanlar 55
Çizelge 3.5. GoogleNet mimarisinde kullanılan katmanlar 58
Çizelge 3.6. ResNet50 mimarisinde kullanılan katmanlar 63
Çizelge 3.7. Faster R-CNN Object Detector eğitim seçenekleri 68
Çizelge 3.8. YOLOv4 Object Detector eğitim seçenekleri 69
Çizelge 4.1. AlexNet, GoogleNet ve ResNet50 mimarileri ile elde edilen ortalama kesinlik (AP) değerleri 74
Çizelge 4.2. GoogleNet mimarisinde ADAM ve SGDM algoritmaları ile elde edilen ortalama kesinlik (AP) değerleri 77
Çizelge 4.3. AlexNet mimarisinde arttırma uygulama ve farklı hiperparametre değerleri ile elde edilen ortalama kesinlik (AP) değerleri 80
Çizelge 4.4. AlexNet 350/50 veri seti ile elde edilen ortalama kesinlik (AP) değerleri 83
Çizelge 4.5. Faster R-CNN ile elde edilen ortalama kesinlik (AP) değerleri 84
Çizelge 4.6. YOLOv4 ile elde edilen ortalama kesinlik (AP) değerleri 87
Çizelge 4.7. Farklı uygulamalar ile elde edilen ortalama kesinlik (mAP) değerleri 88

KISALTMALAR

ANN	Yapay Sinir Ağı (Artifical Neural Network)
AP	Ortalama Kesinlik (Average Precision)
CDN	Evrişimli Derin Ağ (Convolutional Deep Network)
CNN	Evrişimli Sinir Ağı (Convolutional Neural Network)
H-CNN	Hiyerarşik Evrişimli Sinir Ağı (Hierarchical Convolutional Neural Network)
mAP	Ortalama Kesinlik Değerlerinin Ortalaması (Mean Average Precision)
R-CNN	Bölge Tabanlı Evrişimli Sinir Ağı (Region based Convolutional Neural Network)
ReLU	Doğrultulmuş Doğrusal Birim (Rectified Linear Unit)
RNN	Tekrarlayan Sinir Ağı (Recurrent Neural Network)
RPN	Bölge Önerisi Ağı (Region Proposal Network)
RoI	İlgilenilen Bölge (Region of Interest)
SVM	Destek Vektör Makinesi (Support Vector Machine)
YOLO	Sadece Bir Kez Bak (You Look Only Once)

ÖZET

BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI İLE YOKLAMA AMAÇLI YÜZ TANIMA SİSTEMİ

Demet Hanife SUNGUR
Düzce Üniversitesi
Lisansüstü Eğitim Enstitüsü
Elektrik-Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi
Danışman: Prof. Dr. Pakize ERDOĞMUŞ
10 Ağustos 2022, 107 sayfa

İnsanların göz veya parmak izi gibi biyometrik özelliklerinin yanı sıra müdahale veya temas olmadan kullanılabilecek en belirgin ve temel özelliği yüzüdür. Bu çalışmanın amacı, derin öğrenme tekniği ve evrişimli sinir ağı (convulational neural network, CNN) algoritması kullanarak bir yüz tanıma sistemi geliştirmektir. Bunu sağlamak için araştırmacılar tarafından on iki öğrencinin 400 farklı fotoğrafını içeren özgün bir veri seti oluşturulmuştur. Öğrencilerin fotoğrafları, sınıf ortamında farklı oturma düzeninde farklı açılar oluşturacak şekilde kameraya karşı yerleri ve açıları değiştirilerek çekilmiştir. Veri seti eğitim ve test setlerine ayrılmış ve her bir öğrenci fotoğrafta ayrı ayrı etiketlenmiştir. Eğitim süreci gerçekleştirildikten sonra test seti üzerinde yüz tanıma süreci test edilmiştir. AlexNet, GoogleNet ve ResNet50 mimarileri ile oluşturulan R-CNN modellerinde, faster R-CNN ve YOLOv4 modellerinde eğitim ve test işlemleri tamamlanmıştır. Veri setindeki görüntü sayısının ağın başarısına etkisine incelemek amacıyla 300/100 ve 350/50 olmak üzere iki farklı eğitim/test oranı kullanılmıştır. Önerilen yaklaşım, öğrencilerin yüz tanıması ile kimliklerinin belirlenmesinde umut verici sonuçlar vermiştir. 300/100 eğitim/test oranı kullanıldığında başarı oranı %59 iken, 350/50 eğitim/test oranında %98'e yükselmiştir. Sonuç olarak yüzleri kolayca tanımak için uygun derin öğrenme modellerinin tatmin edici düzeyde kullanılabileceği gösterilmiştir. Ayrıca bu modeller sadece yoklama veya güvenlik amaçlı değil, öğrencilerin duygularını tespit etmek için de kullanılabilir. Böylece öğrenme ortamının iyileştirilmesi ve öğrencilerin hazır bulunuşluklarının değerlendirilmesi mümkün olabilir.

Anahtar sözcükler: Görüntü işleme, Nesne tespiti, Yüz tanıma, R-CNN, YOLOv4.

ABSTRACT

FACE RECOGNITION SYSTEM FOR ATTENDANCE WITH REGION BASED CONVOLUTIONAL NEURAL NETWORK

Demet Hanife SUNGUR
Düzce University
Institute of Graduate Education
Department of Electrical-Electronics and Computer Engineering
Master's Thesis
Supervisor: Prof. Dr. Pakize ERDOĞMUŞ
August 10, 2022, 107 pages

In addition to biometric features of people such as eyes or fingerprints, the most distinctive and basic feature that can be used without intervention or contact is the face. The aim of this study is to develop a face recognition system using deep learning technique and convulational neural network (CNN) algorithm. To avhieve this, an original data set including 400 different photos of twelve students was created by the researchers. Photos of the students were taken in their own classroom by changing their places and angles to the camera to create different angles with different sitting patterns. The data set divided into training and test sets, and each students were labeled individually in each photos. After the training process was carried out, face recognition process was testing on the test set. Training and testing of R-CNN models created with AlexNet, GoogleNet and ResNet50 architectures, faster R-CNN, and YOLOv4 models have been completed. To examine the effect of the number of images in the data set on the success of the network; two different training/test ratio; 300/100, and 350/50; were used. The proposed approach yielded promising results in identifying students using face regions. A success rate of 59% was obtained when the 300/100 training/test ratio was used, while the success rate increased to 98% for the 350/50 ratio. As a result, it has been shown that appropriate deep learning models can be used to recognizing the faces easily with a satisfying level. In addition, these models can be used not only for attendance or security purposes, but also for detecting the emotions of the students. Thus, it may be possible to improve the learning environment and evaluate the readiness of students.

Keywords: Image processing, Object detection, Face recognition, R-CNN, YOLOv4.

1. GİRİŞ

İnsanların otomatik olarak tanınması önemi giderek artan bir konu haline gelmiştir. Otomatik insan tanıma sistemlerinde iris, parmak izi ve yüz gibi biyometrik veriler kullanılabilmektedir. Herhangi bir kişinin tanınmasını sağlayan en belirgin ayırt edici özelliklerinden biri de kişinin yüzüdür. Ancak parmak izi, iris tanıma gibi özelliklerle çalışan sistemlerde, kullanıcıların sistemi etkin kullanabilmesi için belirli bazı özelliklere sahip olması ve belki de kullanıcılara temel düzeyde eğitim verilmesinin gerekli olması veya sistem kullanımında kullanıcılara müdahale gerekli olabileceğinden, bu alternatiflere göre otomatik yüz tanıma çok daha cazip ve pratik olabilmektedir [1]. Yüz tanıma askeri, sanayi, sosyal medya, pazarlama, sağlık alanı gibi farklı alanlarda gerek güvenlik, gerek bilimsel ve gerekse hizmet sektörlerinde kullanılabilmektedir [1], [2].

Standart bir yüz tanıma işlemi üç aşamadan oluşmaktadır; Yüz tespiti, özellik çıkarma ve yüz tanıma [1], [3]. Verilen herhangi bir görüntüdeki yüzün bulunması ile başlayan yüz tanıma süreci yüzün konumunun tespiti ve segmentlerine ayrılması ile devam eder. Yüzün ayırt edici özelliklerinden oluşan bir özellik vektörü kullanılarak bölümlenmiş yüzün temsili oluşturulur [1], [3], [4]. Veri tabanındaki özellik şablonu ile belli yüz özellikleri tek bir karşılaştırma yapılmasına doğrulama, veri tabanındaki tüm özellik şablonlarıyla karşılaştırma ise tanımlama olarak ifade edilebilir. Yüz tanıma bu iki alternatifle yapılabilir.

İnsan yüzü yapısal olarak birbirine benzerdir. Farklı insan yüzleri arasında ufak sayılabilecek farklılıklar mevcuttur. Bunun yanı sıra insanlar makyaj ve çeşitli aksesuarlar yardımıyla farklı görünümler oluşturabilirler. Dolayısıyla insan yüzü herhangi sabit bir nesne gibi değerlendirilemez.

Yüz tanımada, yaş, yüz ifadeleri (jestler, mimikler), aydınlatma, poz açısı, çözünürlük, gürültü, bulanıklık, görüntü tahrifatı ve sahteciliği gibi faktörlerden kaynaklanan iç ve dış faktörler olarak sınıflandırılan çeşitli zorluklar mevcuttur. Bu zorluklar içsel ve dışsal faktörler olarak kategorize edilmiştir [1].

İçsel faktörler de kendi içinde yüz ifadeleri (mimik), yaş, yüz oklüzyonu ve görüntü sahteciliği şeklinde 4 gruba ayrılabilir:

1. Yüz İfadeleri: Öfke, korku, kızma, gülümseme gibi duyguların yüz ifadelerine yansıması doğaldır. Yüzün farklı ifadelere bürünebilmesi yüz tanıma için zorluklar doğurabilmektedir.
2. Yaş: İnsan yüzü yıllara bağlı olarak değişmektedir. Her bireyin çocukluk gençlik yetişkinlik ve ileri yetişkinlik dönemlerinde yüz hatlarında değişimler olabilmektedir. Bu da yüz tanıma için aşılması gereken bir problemdir.
3. Yüz Oklüzyonu: Kişiler mevsimsel olarak çeşitli aksesuarlar kullanabilir, örneğin atkı, bere, gözlük gibi aksesuar kullanımı da çeşitli zorluklar doğurmuştur. Ayrıca makyaj ve sakal gibi detaylarda bu zorluklardandır.
4. Görüntü Sahteciliği: Yüz tanıma üzerine umut verici çalışmalar yapıldıkça ve yüz tanımanın popülerleşmesi ile birlikte bu alanda da sahtecilik ortaya çıkmıştır. Yapay zeka kullanılarak hazırlanmış var olmayan bir yüz tasarlanarak sisteme saldırılar planlanabilir [1]. Dolayısıyla, geliştirilen yüz tanıma sistemlerinin güvenlik için de önlemler geliştirmesi gerekmektedir.

Dışsal faktörler ise aydınlatma, çözünürlük, poz açısı, gürültü ve bulanıklık olarak gruplandırılabilir.

1. Aydınlatma: Işığın geliş açısı, yoğunluğu oldukça önemlidir. Herhangi bir fotoğrafta dahi ışığın kişinin arkasında kalması görüntünün net olmamasına sebep olacaktır. Dolayısıyla yüzü net şekilde belirlenememiş bir kişinin tanıma yapılması oldukça güçtür. Işık yoğunluğu, ışık açısı insan yüzünün tanınması için oldukça önemli parametrelerdir.
2. Çözünürlük: Çözünürlüğün yüksek olması elbette çok daha net görüntüler elde edilmesini sağlayacaktır. Ancak bilgisayar kaynaklarının kısıtlı olması çok yüksek çözünürlük üzerinde işlem yapılmasını zorlaştırmaktadır. Bilgisayar kaynaklarını optimum seviyede kullanabilmek için çözünürlüğü daha düşük dosyalar üzerinde işlemler yapılmasını gerektirir. Bu durum görüntü netliğinde çeşitli zorluklar ortaya çıkarmıştır. Ayrıca farklı mesafelerden görüntü almanın da çözünürlük üzerinde etkisi mevcuttur [1].
3. Poz Açısı: İnsan yüzünün görüntüsünün farklı açılardan yakalanması, aynı kişiye ait olsa bile farklı görüntüler oluşturur. Beklenen insanların yüzünün görüntü yakalama düzlemine dönmesidir.

4. Gürültü: Görüntü yakalama aygıtından kaynaklı gürültüler oluşabilir. Bu gürültüleri gidermek için çeşitli yüz tanıma yöntemleri mevcuttur [1].
5. Bulanıklık: Kişilerin hareketi, kameraların yüksek pozlama süresi gibi nedenlerden kaynaklanarak ortaya çıkabilir.

Belirtildiği üzere yüz tanımada aşılması gereken problemler mevcuttur. Bu problemleri aşarak en yüksek doğrulukta ve güvenilir yüz tanıma için derin öğrenme teknikleri kullanılmaktadır [2], [5]. Derin öğrenme yapılarından biri olan evrişimli sinir ağı (convolutional neural network, CNN), görüntülerin sınıflandırılması üzerine başarılı sonuçlar elde etmiştir [6], [7].

Bu çalışmanın amacı, derin öğrenme tekniği ve CNN algoritması kullanarak bir yüz tanıma sistemi geliştirilmesidir. Önerilen sistem dört ana aşamadan oluşmaktadır. Birinci aşamada, bir sınıftaki öğrencilerin farklı oturma düzenlerinde farklı açılardan fotoğrafları çekilerek bir veri seti oluşturulmuştur. İkinci aşamada, öğrencilerin her biri, veri setindeki her bir fotoğraf için ayrı ayrı etiketlenmiştir. Üçüncü aşamada ise veri setinin eğitime ayrılan kısmı ile eğitim süreci gerçekleştirilmiştir. Son olarak eğitim süreci tamamlandıktan sonra test seti üzerinde yüz tanıma süreci test edilmiştir.

Tezin devam eden bölümleri şu şekilde yapılandırılmıştır: Derin öğrenme, CNN, görüntü işleme ve yüz tanıma ile ilgili çalışmalar ve literatür ikinci bölümde özetlenmiştir. Üçüncü bölümde, önerilen yaklaşım için kullanılan materyal ve yöntemler hakkında açıklamalar yapılmıştır. Dördüncü bölümde veri seti üzerinde gerçekleştirilen test çalışmalarının sonuçlarından bahsedilerek bulgular verilmiş, son bölümde ise bulguların değerlendirilmesi, tartışılması ve sonuçlandırılmasına odaklanılmıştır.

2. LİTERATÜR ÖZETİ VE GENEL BİLGİLER

Görüntü işleme, doğal dil işleme, ses işleme, biyomedikal vb. alanlarda kullanılan çok katmanlı algılayıcıların bir türü olan CNN, görüntü işleme alanında en iyi sonuçları vermiştir. Özel bir derin öğrenme mimarisi olan CNN'ler kullanıldığı birçok çalışma yapılmış ve özellikle görüntü işlemede başarılı birçok sonuç elde edilmiştir.

Krizhevsky ve ark. [8], evrişimli derin ağ (convolutional deep network, CDN) ile ImageNet sınıflandırması yaptıkları 2012 yılındaki çalışmalarında, 1 milyon 200 bin yüksek çözünürlüklü nesneyi 1000 ayrı sınıfa bölen bir derin CNN eğitilmiştir. Öğrenme sürecini hızlandırmak için evrişimli işlemin oldukça etkili olduğu bir GPU uygulaması ve doymamış nöronlar kullanılmıştır. Ağdaki tam bağlı katmanlarda bunu uygun bir düzeye düşürmek için oldukça etkili olduğu kanıtlanmış yeni bir hazır bırakma yöntemi kullanılmıştır. Bu yaklaşımın diğer yaklaşımlardan daha iyi olduğunu ispat etmek için ise daha geniş bir veri seti, daha iyi teknikler ve daha güçlü öğrenme modelleri kullanılmış ve sonuç olarak, sınıflandırma işlemi çok az (0.15) hata oranı ile gerçekleştirilerek CNN'in sınıflandırmadaki başarısı gösterilmiştir.

Zhou ve ark. [9] 2013 yılında, derin ağlarla görüntü sınıflandırma problemini ele alan CDN adı verilen yarı denetimli bir öğrenme algoritması geliştirmiştir. İlk olarak, görüntünün bilgisini etkili bir şekilde izole etmek ve azaltmak için, evrişimsel kısıtlı Boltzmann makineleri kullanılarak birkaç gizli katman oluşturulmuştur. Daha sonra yine kısıtlı Boltzmann makineleri kullanılarak görüntünün bilgileri hızlı bir şekilde ayrıştırılabilmiştir. Son olarak, derin öğrenme mimarisi, üstel kayıp fonksiyonlu denetimli tabanlı bir eğim-iniş ile ince ayar yapılmıştır. CDN'ler, bilgileri nesnelerden ayırabilmekte ve boyutlarını küçültebilmektedir. Ek olarak, CDN izolasyon ve sınıflandırma süreci, aynı parametrenin farklı adımlarda optimizasyonu için aynı derin mimariyi kullanmaktadır ve bu nedenle öğrenme yeteneğini de etkin bir şekilde geliştirebilmektedir. İki standart veri setinde çeşitli uygulamalar yapılmış ve CDN'nin yarı denetimli sınıflandırma uygulamaları ve mevcut öğrenme teknikleri ile rekabet edebildiği görülmüştür.

Kukharenko ve Konushin [10] tarafından 2015 yılında yapılan çalışmada, bir kişinin görüntüsünden aynı anda birçok özelliğin çıkarılmasına olanak sağlayan bir model

tanımlanmıştır. Modelde, çeşitli olasılıksal çıktılara ve ortak girdi katmanlarına sahip bir CDN yapısı önerilmiştir. Bu ağ, doğruluğu yüksek olan bir CNN'ye sahip olup bir özellik çıkarımı için gerekli olan sürede bir dizi özelliği aynı anda elde edebilir. Bu ağ, 13233 adet fotoğrafın yer aldığı LWF veri seti ile eğitilmiştir. Bireyin görünüş özellikleri için cinsiyete ek olarak sakal ve bıyık da kullanılmıştır. Doğru tanıma oranı, özelliklerin her biri için LWF veri seti için en iyi sonuçlardan biri olan %91,5'ten daha az olmamıştır. Bu model, işlem süresinde önemli bir artış olmaksızın insan görünümünün aynı anda daha fazla sayıda özelliğinin tanınması için kullanılabilir.

Büyük kalabalık dağılımlarını hesaplamak veya yoğunluğu tahmin etmek, özellikle kamu güvenliği için başka bir ilgi alanı olmuştur. Bu konudaki mevcut uygulamalar, önceden tanımlanmış özelliklere dayanmaktadır ve çoğunun alanı daralmış veya kısıtlı durumdadır. Bu kısıtlamalardan biri; birey sayısı düzinelerce olduğunda uygulanabilir olmalarıdır, ancak kalabalıktaki birey sayısı yüzlerle veya binlerle ifade edildiğinde kalabalık hesaplaması yerine yoğunluk tahmini yapılabilmektedir. Diğeri ise; genellikle videolardaki zamansal bir sıraya bağlı olmalarıdır ve durağan (hareketsiz) nesnelere uygulanamamalarıdır. Hu ve ark. [11] 2016 yılında, bu problemlere atıfta bulunarak, CNN'leri kullanarak hareketsiz bir görüntüden yoğun kalabalık sayma çalışması yürütmüşlerdir. Orta veya üst düzey bir kalabalığa katılan insan sayısını tek bir görüntüde tahmin etmek için bir CDN yaklaşımı kullanmışlardır. İlk olarak, kalabalığın özelliklerini çıkarmak için bir CNN yapısı kullanmışlardır. Ardından, belirli sayıyı tahmin etmek ve kalabalık özelliklerini öğrenmek için kalabalık sayımı ve kalabalık yoğunluğu olmak üzere iki denetim sinyali kullanılmıştır. Yaklaşım 45.000 açıklamalı kişi ve her birisi 58 ila 2201 arasında değişen kafa sayısına sahip 107 adet kalabalık görüntüden oluşan veri seti ile test edilmiştir. Bu yaklaşımın etkisi, sayım performansının çoklu değerlendirme kriterleri ile hesaplandığı kapsamlı deneylerde kanıtlanmıştır. Çalışma, video izleme ve güvenlik yönetiminde daha iyi sonuçlar verecek şekilde geliştirilebilir.

Cireşan ve ark. [12] tarafından 2011 yılında CNN yaklaşımını kullanarak yapılan GPU uygulaması yüksek doğrulukta görüntü işlemenin başka bir örneği olarak gösterilebilir. Eğitim süreci olasılıksal gradyan iniş yöntemi ile yapılan bu sinir ağının kullanılması ile düşük çözünürlüklü doğal resimler için %19,51 doğruluk oranı elde edilirken, nesne tanıma için %2,53 ve el yazısı tanıma için %0,35 doğruluk oranı elde edilmiştir. 2012 yılı itibariy ile literatürde en iyi sonuçların elde edildiği belirtilen bu çalışma ile CNN mimarilerinin yeterli bir veri tabanı ile eğitildiğinde görüntü tanıma için en doğru

sonuçları vereceği de gösterilmiştir. Sonuçların doğruluğunun yanı sıra yüksek öğrenme hızı da bu GPU tabanlı uygulamada büyük bir avantajdır. GPU tabanlı uygulamalar, CPU tabanlı uygulamalardan 10 ila 60 kat aralığında değişen miktarda daha yüksek öğrenme hızlarına sahiptir [12].

Bir başka uygulamada, 2019 yılında yaptıkları çalışmalarında Sarıgül ve ark. [13], hata oranını en aza indirmek için bir geri yayılım algoritması önermiştir. Geri yayılım algoritması, filtre sayısını değiştirmeden aktivasyon hatalarını dikkate alarak sınıflandırma performansını arttırır. Algoritma 4 farklı deney seti ile test edilmiştir. İlk deneyde doğruluk oranı %55,29 oranında artmıştır. Diferansiyel CNN uyarlaması, ikinci deney setinde, AlexNet'in ilk 1 ve 5 testinin doğruluğunu ImageNet veri setinde %5,3 ve %4,75 oranında artırmıştır. Üçüncü deney setinde ise önceki tüm modeller aşılmış, CIFAR-10 veri seti ve CIFAR-100 veri seti için, diferansiyel evrişimli ağ modeli, en yakın düzeyde başarı göstermiş modelden %4,33 ve %7,72 oranında daha da yüksek doğru sınıflandırma oranı göstermiştir. Son deneyde CIFAR-10 veri seti ve CIFAR-100 veri seti için VGGNet modeli %93,58 ve %75,06 doğruluğa ulaşmıştır.

Gradyan iniş yöntemi, yapay sinir ağının (artificial neural network, ANN) veritabanı ile eğitilmesine olanak veren bir algoritmadır. Makine öğrenme alanında oldukça yaygın olan algoritma, eğitim verileri üzerinden maliyet hesaplayarak bu hesaplanan maliyetin en aza indirilmesini sağlamaktadır. Literatürde olasılıksal gradyan iniş yöntemi ile ilgili birçok örnek bulunmaktadır.

Traore ve ark. [14], 2018 yılındaki çalışmalarında mikroskobik görüntüleri tanımlamak için bir CNN modeli sunmuşlardır. Bu şekilde, sıtma ve kolera patojenlerinin tespiti için karar verme sürecinde destek olmak üzere, bu ağın akıllı mikroskopların düzeneğine eklenerek yardımcı olması amaçlanmıştır. Eğitim veri seti olarak olasılıksal gradyan iniş yöntemi kullanılmış ve CNN'nin eğitim sürecinde 200 kolera patojeni ve 200 sıtma patojeninden oluşan toplam 400 görüntü kullanılmıştır. Seksen adet test görüntüsü ile yapılan çalışma sonucunda %94 oranında doğru sınıflandırma başarısı elde edilmiştir. Önerilen CNN modeli, veri setinin boyutuna göre değişiklik gösterebilmektedir. Çalışmanın 14 milyon hiyerarşik görüntüden oluşan bir veri seti ile eğitilmiş olan bir CNN modeli ile tekrarlanması amaçlanmıştır.

Sınıflandırılması gereken veri çok fazla olduğunda, hiyerarşik bir yapıda görüntülerin sınıflandırılması gereken durumlar söz konusu olabilir. Hiyerarşik görüntü sınıflandırma

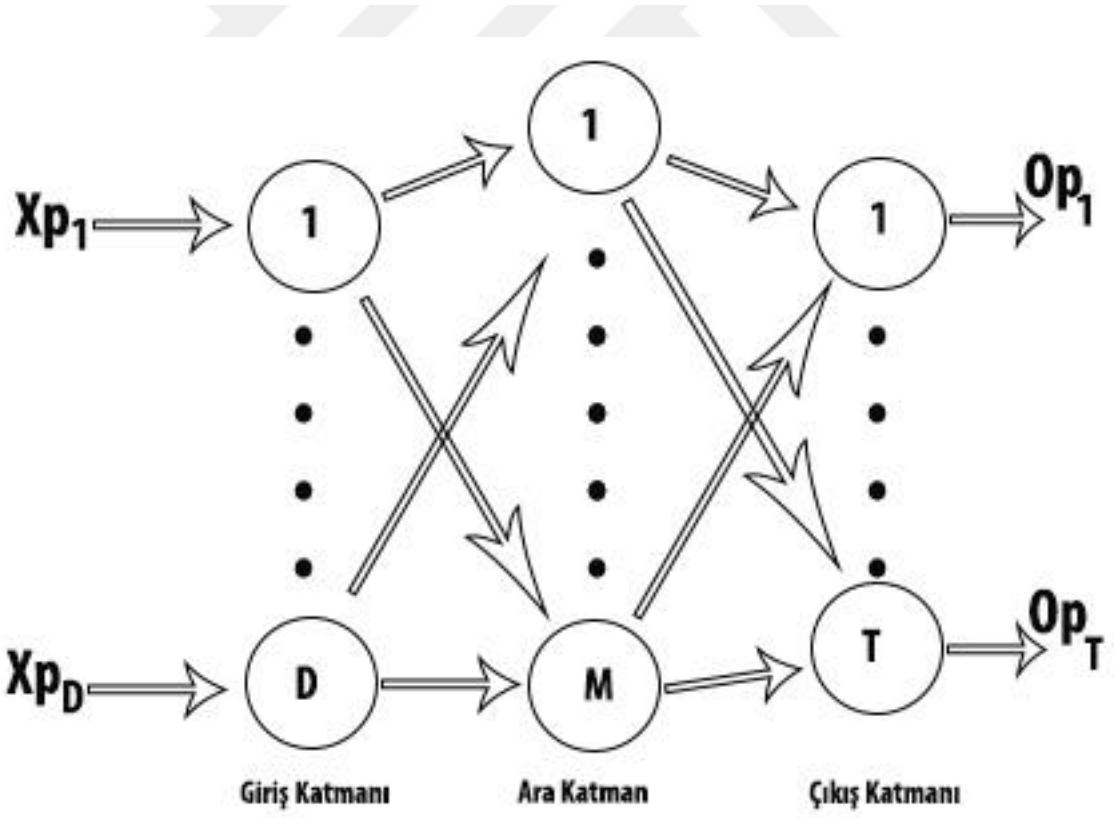
yaklaşımının temel sınıflandırmaya göre daha iyi düzeyde sonuçlar verdiği gözlenmiştir. Seo ve Shin [15], maksimum havuzlama yöntemini de kullandıkları 2019 yılındaki çalışmalarında, görüntü sınıflandırması için bir hiyerarşik evrişimli sinir ağı (hierarchical convolutional neural network, H-CNN) önermiştir. Maksimum havuzlama yöntemi, elde edilen verilerin belli bir bölümünün seçilerek o bölümdeki en yüksek olan değerin dikkate alınıp verileri yeni bir şekilde göstermek için kullanılan bir yöntemdir. Çok fazla bilgi var olduğunda anlamı çok fazla kaybetmeden sadece bilginin boyutunu küçültmek amacıyla kullanılmaktadır. Çanta, ayakkabı, giysi gibi moda resimlerinin sınıflandırıldığı uygulamada, veri tabanında bulunan görselleri ilk olarak 10 temel gruba ayırmışlar, daha sonra da bu 10 grubu hiyerarşik yapılarına göre tekrar iki gruba daha ayırmışlardır. Sinir ağını eğitmek için olasılıksal gradyan iniş yöntemi kullanıldı. H-CNN olarak adlandırılan bu CNN, tam bağlı katmanlardan oluşmakta olan VGGNet'e dayanmaktadır. Bu sinir ağını eğitmek için 28x28 piksel boyutunda 50.000 eğitim görüntüsü ve test için 10.000 test görüntüsü kullanılmıştır. H-CNN modeli, temel modelle karşılaştırılmış ve temel modele göre daha yüksek doğruluk ve daha az kayıpla daha iyi başarı göstermiştir. 52. eğitim döngüsü sonunda, 0,9 momentum katsayısındaki öğrenme hızı 0,00005 olmuştur.

Olasılıksal gradyan iniş yöntemiyle birlikte kullanılmakta olan başka yöntemler de vardır. Bu farklı yöntemlerden biri de ince ayardır. İnce ayar yöntemi, daha önce eğitilmiş olan bir ağın, yeniden benzer ikinci bir görevi yapmak üzere tekrardan eğitilmesi işlemine denilmektedir. Özellikle, az sayıda görüntü olan veri kümeleri ile eğitilen ağlarda performansı daha çok etkileyebildiği gözlemlenmiştir.

Hassasiyet tahmini için ince ayar ağlarının ön eğitimi, daha önce nesne tanıma için eğitilmiş ağlardan daha iyi sonuçlar vermiştir. Cetinic ve ark. [16], sanatsal içeriklerle ilgili görüntülerin sınıflandırılması amacıyla görüntü benzerliğinin farklı yönlerini analiz ettikleri 2018 yılındaki çalışmalarında, görsel görevlerde başarılı sonuçlar gösteren CNN'nin uygulanabilirliğini incelemişlerdir. Çalışmada CNN, 0,0005 öğrenme hızında olasılıksal gradyan iniş yöntemi ile eğitilmiştir. Çalışma sonucunda, ağdaki bağlantıların ağırlıklarının başlangıç değerleri atamasının anlamsal sonuçları ile ilgili daha kesin sonuçlar çıkarılabilmesi için bu konuda daha ileri araştırmalar yapılmasının gerektiği sonucuna ulaşılmıştır. Ek olarak, ince ayar ağlarından elde edilen özelliklerin, tarz ve içerik açısından benzer görüntüler elde edebilmek için de kullanılabileceği ve sanat koleksiyonları için çevrimiçi arama sistemlerinin özelliklerini geliştirmede de kullanılabileceği sonucuna varılmıştır.

2.1. YAPAY SİNİR AĞI

Yapay sinir ağı (artificial neural network, ANN), insan beynindeki nöronların işleyişinden ve beyindeki öğrenme sürecinden esinlenilerek geliştirilmiştir. İlk olarak 1943 yılında McCulloch ve Pitts tarafından beyin hücrelerinin yapısının matematiksel olarak modellenmesi ile ortaya çıkmıştır [17], [18]. ANN ile insan beynindeki öğrenmeyi sağlayan nöronların modellenmesi ile matematiksel olarak benzer bir yaklaşım ortaya koymak ve bu şekilde bilgisayar sisteminin bu öğrenme yaklaşımını gerçekleştirmesi amaçlanmıştır. Nöronlar dentrit adı verilen uzantıları ile giriş verilerini almakta, her bir veriyi önem düzeyi oranında ağırlıklandırarak taşımakta ve bu ağırlıklandırmadan elde edilen toplam değer belirli bir eşiğin üzerindeyse çıkış verisi üretilmektedir. İnsanda ağırlıklar öğrenilmekte, öğrenme nöronların birbiriyle kurdukları bağlantılar (sinapslar) sayesinde gerçekleşmekte ve bu sayede çıkış verisi üretilmektedir. Tekrar eden durumlar söz konusu olduğunda öğrenme de artmaktadır [19].



Şekil 2.1. Yapay sinir ağına örnek bir yapı.

Şekil 2.1'de çok katmanlı bir ANN örneği verilmiştir. Giriş katmanı, gizli katman(lar) ve çıkış katmanı olmak üzere üç farklı katman bulunmaktadır. Bu katmanlı yapı sayesinde ağırlıklandırma ve matematiksel işlemler yapılarak ağın eğitilmesi sağlanmaktadır. ANN giriş verileri ve ağırlıklar çarpılarak modele girecek şekilde tasarlanmıştır. Giriş katmanı Xp_1 den Xp_D'ye kadar olan giriş verilerini alır ve ara katmanlara 1'den M'ye kadar olan gizli katmanlara gönderir. Bu katmanlarda giriş katmanından gelen veriler üzerinden matematiksel işlemler yapılır. Öğrenme, matematiksel modeldeki ağırlıkların öğrenilmesi ile gerçekleşir ve eşik değer ile matematiksel modelde elde edilen toplam değerin kontrol edilmesi aktivasyon fonksiyonu ile yapılır. Matematiksel işlemler ile girdi verileri incelenerek bağlantıların ağırlık değerleri aktivasyon fonksiyonları ile yeniden düzenlenir. Ağın eğitimi, bu bağlantılara ait ağırlık değerlerinin yeniden düzenlemesi sonucu ortaya çıkan süreçtir. Dolayısıyla, ANN'de eğitim, belirli öğrenme kurallarına dayalı olarak yapılan bu ağırlıkların yeniden hesaplanması ve güncellenmesi ile olmaktadır. Bir ANN için sonuca yönelik çıkarım ve genelleme yeteneği olabilmesi doğru ağırlık değerlerinin öğrenilmesi ile mümkün olmaktadır [20]. Bu şekilde, ANN için insan beynine benzer şekilde çıkarım yapma, genelleme, sınıflama vb. işlemleri yapabilme yeteneği kazanma süreci, giriş verileri ve bu giriş verilerinin ağırlıkları ile yapılan matematiksel işlemler ve uygulanan aktivasyon fonksiyonu ile olur [21]. Son olarak çıkış katmanı ise Op_1 ve Op_T isimli çıkış verilerini verir.

Bağlantı şekillerine göre ileri beslemeli ANN ve geri beslemeli ANN olmak üzere iki çeşit yapıda ANN söz konusudur.

İleri beslemeli ANN'de giriş katmanına gelen girdi verileri herhangi bir işlem yapılmadan doğrudan gizli katmanlara verilir. Oradan da tek yönlü bağlantı ile çıkış katmanına iletilir. Doğrusal olarak girişten çıkış yönüne bir akış vardır ve etkileşim söz konusu değildir. Geri beslemeli ANN'lerde ise girişten çıkış yönüne doğru bir hesaplama yapılırken, aynı zamanda geriye doğru da hesaplama söz konusudur. Giriş katmanından giren veriden elde edilen çıktı değeri, çıktının beklenen değeriyle karşılaştırılarak aradaki fark (hata) hesaplanır ve bu fark (hata) miktarı tekrar geriye doğru gönderilerek ağırlık değerlerinin yeniden hesaplanması yapılır [22]. Bu geriye doğru gönderme sürecinde hesaplanan hata miktarının azaltılması mevcut ağırlıklara dağıtılarak sağlanır. Bu süreçte geri beslemeli sinir ağlarında elde edilen çıkış verisi ile aynı anda ilk giriş verileri ve bu verilerinden elde edilen çıkış verileri ile oluşan giriş verileri eş zamanlı olarak dinamik bir şekilde işlenmektedir [23].

ANN teorisiyle birlikte makine öğrenmesi ile ilgili çalışmalar konusunda yeni bir çağa girilmiş olmuştur. İlk geliştirilen ANN modeli tek katmanlı ve ileri beslemeli bir ANN modeli olarak geliştirilmiştir [24]. Ancak bu model ile elde edilen sonuçlar belirli bir başarı oranının üzerine çıkamamıştır. İlerleyen dönemlerde geri beslemeli ANN modeli geliştirilmiş ve bu model üzerinde yapılan düzenlemelerle daha başarılı sonuçlar elde edilebilmiştir.

ANN'nin eğitimi ağırlıkların yeniden hesaplanması ile gerçekleştirilir. Girişten sinir ağına verilen değişkenler aktivasyon fonksiyonundan geçirilir. Elde edilen çıkış değeri bilinen çıkış değeri ile karşılaştırılarak hata bulunur. Her bir iterasyondaki hata miktarına göre aktivasyon fonksiyonu güncellenir. Çeşitli yöntemlerle buradaki hata azaltılmaya çalışılır. Her girişin ağırlıkları farklı olabilir. Ağırlıklar yeniden hesaplanarak hedeflenen sonuca ulaşılması sürecinde öğrenme gerçekleşir. En uygun ağırlık tespiti gerçekleştirilir ve bu değişken ağırlıkları saklanır. Öğrenme aşamasından sonra farklı girişlerle test yapılır. Eğer test aşamasındaki verilerde de beklenen sonuçları veriyorsa sinir ağı öğrenmeyi gerçekleştirmiştir. Ancak eğitim aşamasında sinir ağına optimum veri sağlanmalıdır. Eğer eğitim aşamasında optimum sayıda veri yerine daha fazla örnek verilirse öğrenme değil ezberleme söz konusu olacaktır [25].

Doğrusal problemlerin çözümünde başarılı bir şekilde çalışan ANN modelleri ile artan çalışma sayısı ve yapılan düzenlemelere rağmen halen doğrusal olmayan modellerin çözümünde sağlıklı bir şekilde uygulanamamakta ve doğru sonuçlar elde edilememekteydi [26]. Çok katmanlı ANN modellerinin ortaya çıkmasıyla birlikte doğrusal olmayan denklem sistemlerinde de geri beslemeli ve çok katmanlı ANN modelleri sayesinde başarılı sonuçlar elde edilmeye başlanmış ve yeniden ANN ile yapılan çalışmalar artmıştır [27]. Daha sonraları katman sayısı artırılarak geliştirilen ve gizli katmanlar içeren CNN modelleri ile daha başarılı sonuçlar elde edilmeye başlanmıştır [28].

ANN modelleri ile mevcut değişkenlerle, verilerle çeşitli analizler yaparak gelecekle ilgili tahminler üretebilmektedir. Örneğin çok yağış alan bir bölgede yağış miktarı ve toprak yapısına göre heyelan olma olasılığı gibi problemlerde başarılı sonuçlar elde edilebilir. Kısacası doğrusal olmayan problemlerin çözümünde de oldukça başarılıdır.

ANN paralellik, gerçekleme kolaylığı, öğrenebilirlik ve adaptasyon gibi özelliklere sahiptir [22], [29].

Paralellik: Pek çok örnek modellenebilir. Hata sistemin tümüne etki etmez. Görüntüler gibi yüksek boyutlu verilerde paralellik önemlidir. Aynı anda birden fazla çıkış üretebilmek hız sağlamaktadır.

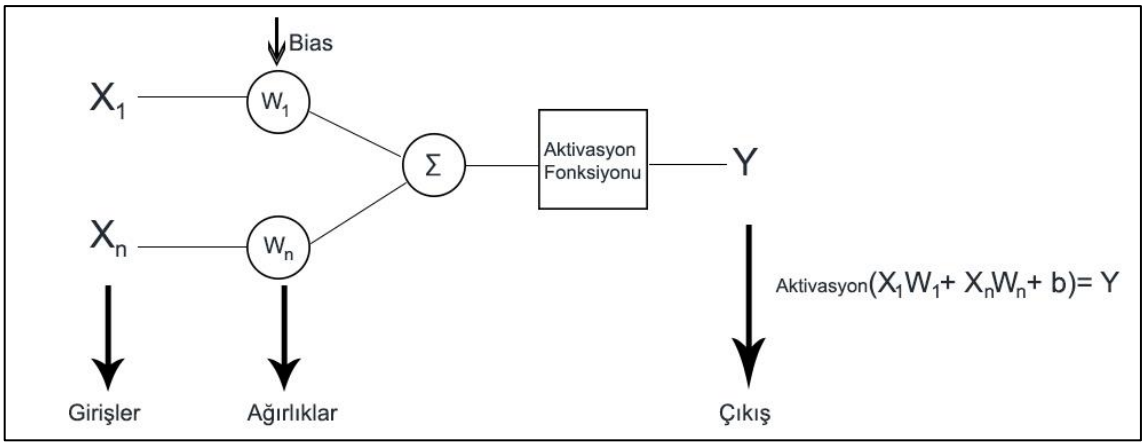
Gerçekleme Kolaylığı: Yapay sinir ağlarında problemleri fonksiyonlar şeklinde yazmak pratiktir.

Öğrenebilirlik ve Adaptasyon: Her türlü probleme uygulanabilir. Sayısal bir girdi mevcutsa tahmin, kümeleme, adaptasyon çözebilir.

2.1.1. Aktivasyon Fonksiyonu

Doğrusal olmayan özelliklerin ANN tarafından tanınması için aktivasyon fonksiyonuna gerek duyulmaktadır. Aktivasyon fonksiyonu kullanılmaması durumunda, öğrenme sürecinden ziyade temel bir doğrusal regresyon modeli şeklinde bir süreç ortaya çıkacaktır. Dolayısıyla çıkış değeri de doğrusal bir işlev sonucu olarak elde edilecektir. Oysa, ANN modellerinde sadece doğrusal değil doğrusal olmayan durumlar da söz konusu olacaktır ve doğrusal olmayan bu durumların da ANN tarafından öğrenilebilmesi için aktivasyon fonksiyonları kullanılmaktadır. Basit bir ANN modelinde, genel olarak bakıldığında aktivasyon işlemi, X girdileri ve W ağırlıkları kullanılarak çıkış katmanına aktarılan değerlere uygulanmaktadır. Aktivasyon işleminin uygulanması Şekil 2.2'de modellenmiştir.

Eğitim süreci açısından düşünüldüğünde, ANN yapısı giriş verileri, ağırlıklar, ağırlıklandırma ve toplama işlevi ve aktivasyon fonksiyonu elemanlarından oluşmaktadır. Temel olarak n sayıda girişe sahip olan bir sinir ağında girişler ağırlıkları ile çarpılır, çarpımların tamamı toplanır ve bias değeri eklenir, son olarak aktivasyon fonksiyonuna gönderilir. Elde edilen çıkış değeri hedeflenen çıkış değeri ile karşılaştırılır, eğer elde edilen ve hedeflenen çıkış değeri aynı ise herhangi bir güncelleme yapılmaz. Farklıysa ağırlıklar ve bias değerleri yeniden güncellenir. Kısaca, aktivasyon fonksiyonu Y çıkışının aktif olup olmayacağının kontrolünü yapar. Yani temel olarak bir ANN modelinde girdilerin ağırlıkları toplamı hesaplanarak bias değeri eklenip aktivasyon fonksiyonu ile çıkış üretme prensibi söz konusudur. Bu çıkış da başka bir katman için girdi verisi olabileceği gibi sonuç olarak üretilen çıktı değeri de olabilir. ANN için giriş verileri bir önceki sinirin çıkışı olabileceği gibi dış çevreden gelen girişler de olabilir.



Şekil 2.2. Yapay sinir ağında aktivasyon fonksiyonu.

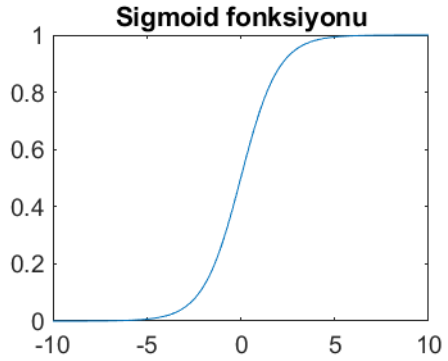
Öğrenme sürecinin gerçekleştirilmesinde kullanılan geri yayılım algoritmasında türev alma işlemi de yapıldığından aktivasyon fonksiyonu türevi alınabilir bir fonksiyon olmalıdır [30]. Bununla birlikte, ANN ile sınıflandırma yapılırken genellikle doğrusal olmayan durumların da öğrenilmesi gerektiğinden aktivasyon fonksiyonu doğrusal olmayan bir fonksiyon olmalıdır. Eğer ANN'nin tüm katmanlarında doğrusal bir fonksiyon kullanılırsa giriş ile çıkış katmanı arasında doğrusal olarak sürekli aynı sonuç elde edilecektir. Doğrusal fonksiyonların birbiriyle doğrusal olarak birleşiminde ortaya çıkacak olan başka bir doğrusal fonksiyon olacaktır. Bu durumda da ara katmanların pasif kalması söz konusu olacak, dolayısıyla çok katmanlı yapıda çalışma durumu olmayacaktır. Yani çoğunlukla doğrusal olmayan sınıflandırma yapmak için kullanılan ANN'de aktivasyon fonksiyonu da yine doğrusal olmayan fonksiyon olarak seçilmektedir.

Doğrusal olmayan aktivasyon fonksiyonları arasında en çok kullanılanlar sigmoid, hiperbolik tanjant (tanh), doğrultulmuş doğrusal birim (Rectified Linear Unit, ReLU), leaky (sızdıran) ReLU ve softmax aktivasyon fonksiyonlarıdır.

2.1.1.1. *Sigmoid Aktivasyon Fonksiyonu*

Girdi verilerini kullanarak her zaman [0, +1] aralığında çıktı değeri üreten bir fonksiyondur. Değerin 0 olması durumunda öğrenme olayı hiç gerçekleşmezken 1 olması durumunda ise tam olarak gerçekleşmesi söz konusudur. Sigmoid aktivasyon fonksiyonu 1990'lı yılların başında sinir ağlarında varsayılan olarak tercih edilen aktivasyon fonksiyonuydu. Anlamlı sonuçları -5 ve +5 girdi aralığında sağlamaktadır. Bu aralıkta

geri yayılım algoritması ile öğrenim devam edebilir. Ancak sonucun her zaman 0 çıkması durumunda geri yayılım algoritması ile hataların güncellenmesi mümkün olmaz ve öğrenme durur. Derin ağlarda büyük çıktılar üretilir. Sigmoid aktivasyon fonksiyonu kullanılan sinir ağlarında gradyan kaybolma (vanishing gradient) problemi ortaya çıkmıştır. Başlangıçtaki ağırlık değerleri çok büyük olduğunda öğrenmenin gerçekleşmemesi gibi sorunlar ortaya çıktığından ve daha verimli alternatif aktivasyon fonksiyonları da olması sebebiyle sigmoid aktivasyon fonksiyonu günümüzde daha az tercih edilen bir aktivasyon fonksiyonu haline gelmiştir [31]. Şekil 2.3'te sigmoid aktivasyon fonksiyonu gösterilmiştir.



Şekil 2.3. Sigmoid aktivasyon fonksiyonu.

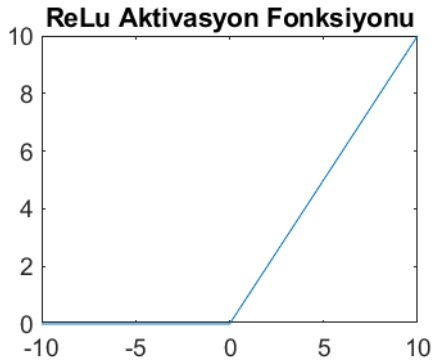
2.1.1.2. *Hiperbolik Tanjant (Tanh) Aktivasyon Fonksiyonu*

Sigmoid fonksiyonuna benzer yapıda bir fonksiyon olup farklı olarak fonksiyonun aralığı [-1, +1] aralığında çıktı değeri üretmektedir. Sigmoid aktivasyon fonksiyonuna göre avantajlı olan yönü ise öğrenme ve sınıflama işleminin daha hızlı olabilmesini sağlayan daha geniş aralığa sahip olma özelliğidir. Bu, fonksiyonun türevinin daha dik olmasından yani sigmoid fonksiyona göre daha çok değer alabilmesinden dolayı öğrenme ve sınıflama işleminde daha verimli olabileceği anlamındadır. Yine bu fonksiyonda da uç değerlerde gradyan kaybolma sorunu ortaya çıkmaktadır.

2.1.1.3. *ReLu Aktivasyon Fonksiyonu*

Doğrultulmuş doğrusal birim (Rectified Linear Unit) şeklinde ifade edilen ReLU aktivasyon fonksiyonu, X pozitif girdi verileri için X değerini alan ve negatif girdiler için 0 değerini alan, dolayısıyla [0, +∞] aralığında çıktı değerleri üreten doğrusal olmayan bir

fonksiyondur. Pozitif her girdi için hataların geri yayılım algoritması kullanarak öğrenme devam eder. Ancak negatif girdiler için her zaman 0 sonucunu verdiği için öğrenme durur. Negatif değerler için her zaman 0 değerini alması hesaplama ve işlem yükünün daha az olması ve dolayısıyla ağın da daha hızlı çalışması manasına gelmektedir. Bu durum sigmoid ve hiperbolik tanjant (tanh) aktivasyon fonksiyonlarına kıyasla çok katmanlı ağlarda bu fonksiyonun daha fazla tercih edilme nedeni olmuştur [32]. Şekil 2.4'te ReLU aktivasyon fonksiyonu gösterilmiştir.



Şekil 2.4. ReLU aktivasyon fonksiyonu.

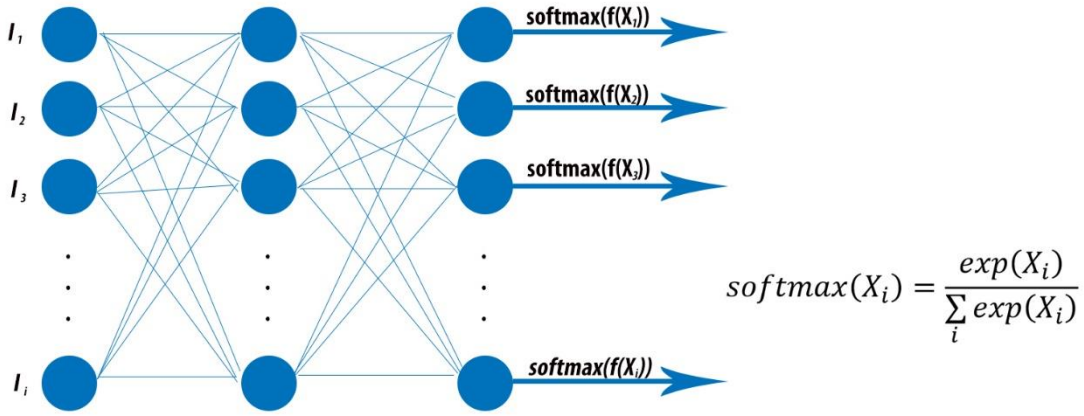
2.1.1.4. *Leaky (Sızdıran) ReLU Aktivasyon Fonksiyonu*

Negatif değerler için genellikle 0,01 düzeyinde ayarlanan bir sızdırma değeri ile gradyan kaybolma sorununa bir çözüm getirilerek negatif değerler için de öğrenmenin gerçekleşmesi sağlanmaya çalışılmaktadır. Leaky (sızdıran) ReLU aktivasyon fonksiyonunda negatif değerler sıfıra çok yakındır ancak tam sıfır değildir. Sıfıra yakın olarak verilen bu sızdırma değerinden dolayı leaky (sızdıran) ReLU aktivasyon fonksiyonunun tanım aralığı da -∞ düzeyine kadar devam etmektedir.

2.1.1.5. *Softmax Aktivasyon Fonksiyonu*

Sigmoid aktivasyon fonksiyonuna benzer bir yapıya sahip olan ve ikiden fazla sınıfın söz konusu olduğu çoklu sınıflandırma olan durumlarda kullanılan softmax aktivasyon fonksiyonu her giriş verisinin bir sınıfa ait olma olasılığını gösterir. Olasılık açısından yorumlama yapmayı sağlayan bu fonksiyon [0, +1] aralığında çıktı değerleri üretmektedir. Sigmoid aktivasyon fonksiyonundan farklı olan tarafı çıktı değerlerinin normalize edilmesidir. Girdi verisinin sıfır, sıfırdan küçük veya büyük olması fark

etmeksizin softmax aktivasyon fonksiyonu giriş verisini 0 ile +1 arasında olasılık değerine dönüştürmektedir. Giriş verisinin negatif ya da küçük bir değer olması durumunun ayrımı, fonksiyonun bu değer için çıktı olarak düşük olasılık değeri vermesi ile sağlanmaktadır. Giriş değeri büyüdükçe olasılık da artmaktadır. Şekil 2.5'te softmax aktivasyon fonksiyonuna bir örnek gösterilmiştir.



Şekil 2.5. Softmax aktivasyon fonksiyonu.

2.2. DERİN ÖĞRENME

Çok değişkenli ancak herhangi bir denkleme sahip olmayan problemlerin çözümünde makine öğrenmesi kullanılabilir. Derin öğrenme makine öğrenmesinin bir dalıdır. Derin öğrenme kavramı Hinton ve Salakhutdinov [33] tarafından 2006 yılında yayınlanan çalışma ile ortaya çıkmış ve derin öğrenme alanındaki çalışmalar başlamıştır. Bu çalışmalarda ANN'ye getirilen yeni yaklaşım, derin evrişimli sinir ağı (deep convolutional neural network) adını almıştır [33]. Çok katmanlı ANN modelleri ile elde edilen başarılı sonuçlardan sonra derin evrişimli sinir ağı ile daha yüksek başarı seviyelerine ulaşılmıştır [8], [34], [35].

Derin öğrenme, eldeki bir veri seti ile ANN'nin eğitilerek çıktıların tahmin edildiği, çok sayıda veriden öznitelik çıkarmak için kullanılan çok katmanlı bir yapıya sahip bir öğrenme metodolojisidir [36]. Bu öğrenme sisteminde bilgisayarların kapasitesi, işlemci hızı vb. gibi özelliklerine bağlı olarak büyük veri setleri kolayca işlenebilmektedir. Eğitim sürecinde denetimli (supervised) öğrenme ve denetimsiz (unsupervised) öğrenme kullanılabilir [37], [38].

Denetimli Öğrenme: Girdi olarak kullanılan tüm veriler etiketlidir ve girdi olarak kullanılan verilerden çıktının tahmin edilmesine dayanır. Çıktının beklenen değeri bellidir ve algoritmanın çıktıyı doğru ya da yanlış tahmin etmesine bağlı olarak öğrenme süreci devam eder. Tahmin yanlışsa algoritma hesaplamaları yeniden düzenler ve tahmin yapmaya devam eder. Bu işlem tahmindeki hata oranı minimum düzeye inene kadar devam eder [37].

Denetimsiz Öğrenme: Girdi olarak kullanılan veriler etiketsizdir ve girdi olarak kullanılan verilerden doğal yapının öğrenilmesi prensibine dayanır. Yalnızca girdi vardır ve çıktıların beklenen değeri yoktur. Bu öğrenme şeklinde algoritmanın verilerden mantıksal bir sınıflandırma yapması, temel yapının veya dağılımın modellenmesi söz konusudur. Girdi olarak bir veri seti verildiğinde, alt gruplar oluşturarak veri setinin yapısını ve aralarındaki ilişkileri bulup bu yapıları birbirinden ayırt eder, ancak oluşturulan kümenin otomatik olarak etiketlenmesi söz konusu değildir [38].

Derin öğrenme medikal uygulamalardan, siber güvenliğe, deniz canlıların saptanmasından, araç tespitine, doğal dil işlemeye kadar pek çok alanda çalışılmıştır. Derin öğrenme ile metin, ses ve görüntü gibi verilerle sınıflandırma, tanıma gibi işlemler gerçekleştirebilmektedir. Kullanılacak alanın farklılığı ve kullanılacak verinin yapısına göre kullanılacak YSA modeli de farklılık göstermektedir [39], [40].

3. MATERYAL VE YÖNTEM

Bu çalışmada, derin öğrenme tekniği ve CNN algoritması kullanarak bir yüz tanıma sistemi geliştirilmesi amaçlanmıştır. Çalışmada kullanılacak veri seti araştırmacılar tarafından bir sınıftaki öğrencilerin farklı oturma düzenlerinde farklı açılardan fotoğrafları çekilerek oluşturulmuştur. Veri seti, eğitim seti ve test seti olmak üzere iki kısma ayrılmıştır. Her bir öğrenci, veri setindeki her bir fotoğraf için ayrı ayrı etiketlenmiştir. Dedektör nesnesi, eğitim süreci için bölge tabanlı evrişimli sinir ağı (region based convolutional neural network, R-CNN) Object Detector kullanılarak oluşturulmuştur. Eğitim sürecinden sonra test seti üzerinde algılama işlemi gerçekleştirilerek dedektör nesnesinin başarı oranı incelenmiştir. Yeni oluşturulan bu nesne algılama modeli kullanılarak veri setinin test kısmı ile test işlemi gerçekleştirilmiştir. Modelin doğruluk oranlarını karşılaştırmalı olarak derinlemesine inceleyebilmek için üç farklı evrişimli sinir ağı mimarisi kullanılmıştır.

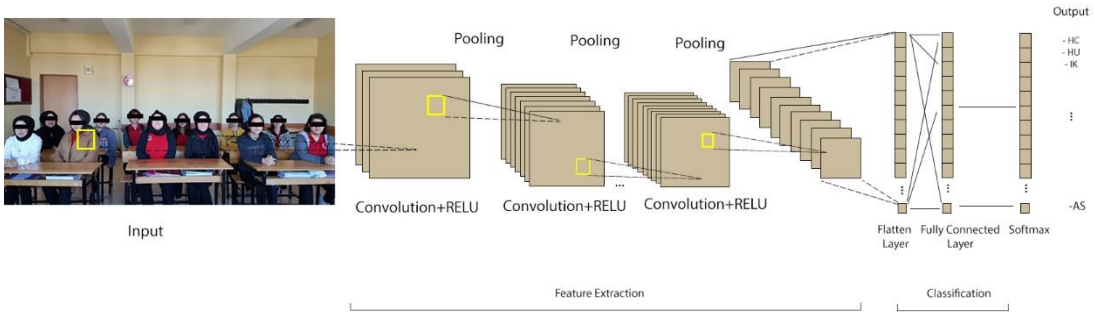
Fotoğraf çekmek ve kullanmak için öğrencilerden ve velilerinden yazılı izin alınmıştır.

3.1. EVRİŞİMLİ SİNİR AĞI (CONVOLUTIONAL NEURAL NETWORK, CNN)

Derin öğrenme için kullanılan en popüler modellerden biri CNN'dir. CNN, biyolojik yapılardan ve süreçlerden esinlenerek bilgisayarla görme uygulamaları için geliştirilmiş çok katmanlı ANN'nin özel bir modelidir. CNN'ler, öznitelik çıkarma ve sınıflandırma işlemlerini birleştiren, görüntü piksellerinden (veya başka diğer sinyallerinden) doğrudan kalıpları tanımak için dizayn edilmiş olan ileri beslemeli sinir ağlarıdır [41]. Görüntü işleme ile ilgili problemlerin çözümü için ortaya atılan, nesne tespiti, tanıma ve sınıflandırma gibi amaçlarla kullanılan CNN görüntü işlemede çok iyi uyum sağlayabilen bir derin öğrenme mimarisidir [42] - [47]. Bu sinir ağlarının adı karmaşık işlemleri kolay bir şekilde gerçekleştirmek için kullanılan evrişim (konvolüsyon, convolution) işlemini yapan evrişim filtresinin kullanılmasından ileri gelmektedir [48]. Görüntü işleme sürecinde kullanılan daha önceki yöntemlerde, görüntü vektörlere dönüştürülerek ele alınmakta ve görüntünün pikselleri birbirinden bağımsız şekilde vektör konumlarına göre incelenmektedir. Dolayısıyla, görüntüde yer alan nesnelerin özelliklerine ait bilgilerin

kaybolması sorunu ortaya çıkmaktadır. Derin öğrenmede en çok kullanılan ağ modeli olan CNN mimarisinde kullanılan filtreler ile bu soruna çözüm üretilmiş olmaktadır. CNN mimarisindeki öğrenme sürecinde, görüntüye filtrelerin uygulanmasıyla görüntüden elde edilen özgün özellikler ortaya çıkarılarak bu özellikler işlenmekte ve öğrenilmektedir [49] - [51]. Mimari yapısında çok sayıda parametre yer alan CNN mimarisi, maymunlardaki katmanlardan oluşan görme yapısı dikkate alınarak yapılandırılmıştır [8], [52]. CNN mimarisinde de maymunlardaki görme işlevine benzer şekilde, katmanlar direk olarak birbirine bağlı olmayıp görüntü katmanlar arasında çeşitli bağlantılar ile iletilmektedir.

Derin öğrenme metodolojisinde bir çok çeşitli sınıflandırma modeli vardır ve bu modeller de bir çok alanda kullanılmaktadır. CNN, görüntü tanıma ve sınıflandırma işlemlerinde en güçlü model olarak ortaya çıkmaktadır [53]. Yapısında, evrişim (konvolüsyon, convolution) katmanı, aktivasyon (doğrultulmuş doğrusal birim, Rectified Linear Unit, ReLU) katmanı, havuzlama (pooling) katmanı ile tam bağlı katman (fully connected layer) ve softmax olmak üzere her birinin kendine özel görevi olan ayrı katmanlar içerir. Bu katmanlardan bir kısmı özellik çıkarımı ve bu özelliklerin öğrenilmesi ile ilgiliyken bir kısmı da sınıflandırma süreci ile ilgilidir. Bir CNN'nin genel mimarisi Şekil 3.1'de gösterilmektedir [54]. Genel olarak bir CNN, giriş, evrişim katmanı, havuzlama (pooling) katmanı, tam bağlı katmanlar ve sınıf kararından oluşur. Bir CNN, bir ya da birden fazla evrişim katmanı içerir ve bu evrişim katmanlarını izleyen çok katmanlı bağlantılı standart bir ağ yapısından oluşur. Genellikle, evrişim katmanlarından sonra parametre sayısı havuzlama işlemi ile azaltılır.



Şekil 3.1. Evrişimli sinir ağı genel mimarisi.

CNN'ler, standart sinir ağlarından daha kolay eğitilebilir ve daha az parametreye sahip olabilir. Görüntüler CNN ile işlendiğinde görüntüyü oluşturan pikseller düzleştirilir ve ağ girişine bir vektör olarak uygulanır. Vektörün boyutu, görüntünün kanal sayısına ve piksel sayısına bağlı olarak değişebilir. Oluşturulan vektörün sinir ağına uygulanmasıyla sınıflandırma işlemi yapılır. Bir görüntü düzleştirildiğinde görüntüdeki köşeler, renkli çizgiler ve oval kısımları kaybolur. Bu nedenle görüntüde nesne tanıma işleminde önemli olan bu detayların kaybolmaması adına CNN kullanılması daha uygun olmaktadır [55].

CNN'de görüntüyü düzleştirmek yerine filtreler kullanılarak görüntüye ait farklı özellikler ortaya çıkarılmaktadır. Her bir filtre, görüntünün farklı özelliklerinin temsil edilmesini sağlamaktadır. Evrişim işleminden sonra değerler aktivasyon fonksiyonundan geçirilmektedir. Filtrelerin havuzlama işleminden sonra da süreç uygulanır. Havuzlamadan sonra ortay çıkan katmanlı yapı düzleştirilir ve bir ağa uygulanır. Sınıflandırma işlemi de bu ağda yapılır. Bu şekilde görüntünün detay özellikleri kaybolmadan ağa uygulanmış olur. CNN'de özellikleri belirlemek için evrişim katmanı kullanılırken, sistemde doğrusal olmayan değerlerin işlenmesi ise aktivasyon katmanı ile sağlanarak yalnızca aktif olan özellikler sonraki katmana taşınmaktadır. Havuzlama (pooling) katmanında ise ağın parametre sayısının azaltılmasıyla sürecin hızlanması sağlanmış olur. Tam bağlı katman, sınıf sayısına göre karar süreçlerinin yürütüldüğü bir sinir ağıdır. Tam bağlı katman bir sinir ağıdır ve sınıf sayısına göre karar süreçleri bu katmanda gerçekleştirilir.

3.1.1. Evrişim (Konvolüsyon, Convolution) Katmanı

Sinir ağının ilk katmanı olan ve aynı zamanda ağın CNN adını almasının da nedeni olan bu katman CNN mimarisinin temelini oluşturmaktadır [56]. Bu katman giriş görüntüsünü evrişimli filtrelerden geçirerek özellik çıkarımı yapmakta ve ilgili bölgenin özelliklerini öğrenmektedir. Bu katmanda incelenen her bölgeden elde edilen giriş verileri ağırlık değerleri ile çarpılarak ağırlıklandırılmakta ve evrişim işleminden geçirilerek sıradaki katmanlara aktarılmaktadır. Filtreler görüntülerin üzerinde yatayda ve dikeyde özellik taraması gerçekleştirirler. Sonuç olarak özellikleri tespit ederek özellikleri için bir matris üretirler. Soldan sağa yukarıdan aşağıya belirtilen matris filtre değerine göre tarama yapması, özellik çıkarımı için güçlü bir yöntemdir. Görüntünün herhangi bir yerinde özellik çıkarımı sağlar. Filtre girdi dizesine birden çok kez uygulandığında, filtrelenmiş girdiyi temsil eden iki boyutlu çıktı dizisi özellik haritası olarak adlandırılır. Yapılan

evirişim işlemi ile özelliklere ait matrisler görüntü üzerinde kaydırma yapılarak sıradaki katmanlara iletilecek olan özelliklerin sayısı azaltılmaktadır [57]. CNN yapısında birden fazla özellik tespiti için birden fazla sayıda evrişim katmanı bulunabilir ve dolayısıyla bir CNN yapısında birden çok evrişim katmanı yer alabilir.

3.1.2. ReLU (Doğrultulmuş Doğrusal Birim, Rectified Linear Unit)

Negatif giriş verisi için sıfır değerini almasını veya giriş verisi negatif değilse aynı değeri almasını sağlayan işlemlerin yapıldığı aktivasyon sürecinin uygulanmasıdır. Her bir giriş verisi için bir eşik değer hesaplaması yapılmaktadır. Bu noktada, genellikle eğitim sürecinin birkaç kat daha hızlı olmasını sağlayan ReLU aktivasyon fonksiyonu daha çok kullanılmakla birlikte, sigmoid veya hiperbolik tanjant (tanh) fonksiyonu gibi aktivasyon fonksiyonları da kullanılabilmektedir [8].

3.1.3. Havuzlama (Pooling) Katmanı

Bu katmanda ağın eğitilmesi için gereken parametre sayısının azaltılması sağlanarak hem eğitim süresinin daha kısa olması hem de daha az örnek kullanılarak daha etkin bir eğitim yapılması için özellik seçim işlemi uygulanmaktadır [58]. Böylece ağda ortaya çıkabilecek ezberleme veya aşırı öğrenme probleminin de önüne geçilmiş olmaktadır. Havuzlama işlemindeki parametre sayısı, havuzlama işlemine alınan matris boyutlarına göre ve yapılan havuzlama tipine göre belirlenmektedir. Giriş verisi olarak alınan görüntüye ait tüm parametreler kullanılmamakta işlem havuzlama tipine göre yapılmaktadır.

Ortalama havuzlama ve maksimum havuzlama en çok kullanılan havuzlama teknikleri olarak ortaya çıkmaktadır. Her iki yöntemde de herhangi bir parametre içermeyen pencere boyutu belirlenmektedir. Bu pencereler sadece giriş matrisinde işleme alınacak parametre sayısını belirlemekte kullanılır. Maksimum havuzlama yönteminde bu pencereye ait alandaki en yüksek değere sahip piksel değeri seçilirken ortalama havuzlama yönteminde ise bu alandaki piksel değerlerinin ortalaması hesaplanır. Her iki yöntemde de maksimum değer ya da ortalama alınarak belirlenmiş olan bu değerler sıradaki katmana iletilir. Bu şekilde yapılan işlemler ile matrisin boyutu oldukça küçülmesine rağmen önemli özellikler halen korunmaktadır. Dolayısıyla bu yöntemlerin en önemli avantajı matrislerin boyutlarının küçülmüş olmasına rağmen hiçbir bilgi kaybı olmaması, önemli özelliklerin halen korunuyor olmasıdır.

3.1.4. Tam Bağlı Katman (Fully Connected Layer)

Bu katman bir CNN mimarisindeki en son ve önemli katmandır. Bu katmanda yer alan nöronların hepsi daha önceki katmanda yer alan nöronlara bağlıdır. Katmana adını veren de bu şekilde tüm nöronların önceki katmandaki nöronlara tam olarak bağlı olmasıdır. Önceki katmanlarda öğrenilen özelliklerin tamamı tam bağlı katmanda birleştirilir [8]. Bu katmanın bir başka özelliği de çıktı sayısının ayarlanabilmesidir. Tam bağlı katmandan önceki süreçte yapılan işlemler verinin sadeleştirilmesi ve mantıksal işlem süreçlerine hazırlanması amacıyla geçirilmiş olduğu süreçlerdir. Mantıksal işlemler tam bağlı katmanlarda gerçekleşmektedir. Ağa giriş katmanından giren görüntü ağ boyunca iki boyutlu matrisler şeklinde işlenerek ilerlemektedir. Bu şekilde, özellik haritaları da her katmanın çıkışında yine matris şeklinde olmaktadır. Girişten itibaren ağ boyunca matris olarak ilerleyen tüm özelliklere ait matrislerin taşıdığı değerler birbiri altına sıralanarak tek sütun haline gelmekte, böylece özelliklere ait matris şeklindeki yapılar bu katmanda sadeleşerek vektör haline dönüştürülmekte ve öğrenme süreci özelliklere ait bu vektör ile gerçekleşmektedir. Bu şekilde sınıflandırma yapılması sağlanmaktadır [58].

Görüntü sınıflandırma sürecinde iki boyutlu matris yapısındaki özellik haritalarını tek boyutlu vektöre dönüştürmek için CNN yapısında çoğunlukla birkaç tane tam bağlantılı katman kullanılır. Bu tam bağlantılı katmanların her biri geleneksel sinir ağı gibi düşünülebilir ve bu katmanlar tüm CNN parametrelerinin yaklaşık %90'ını kapsarlar. Ancak bu durum eğitim için gerekli olan sürecin kayda değer miktarda artmasına neden olur. Bu sorunla başa çıkmak için yaygın olarak uygulanan bir çözüm, bu tam bağlı katmanlardaki bağlantıların azaltılmasıdır. Bu amaçla, önerilmiş olan bir çok CNN mimarisi olup en çok kullanılan CNN mimarilerinden bazıları LeNET, AlexNet, GoogLeNet, VGGNet mimarileridir [59] - [61].

3.1.5. Softmax Katmanı

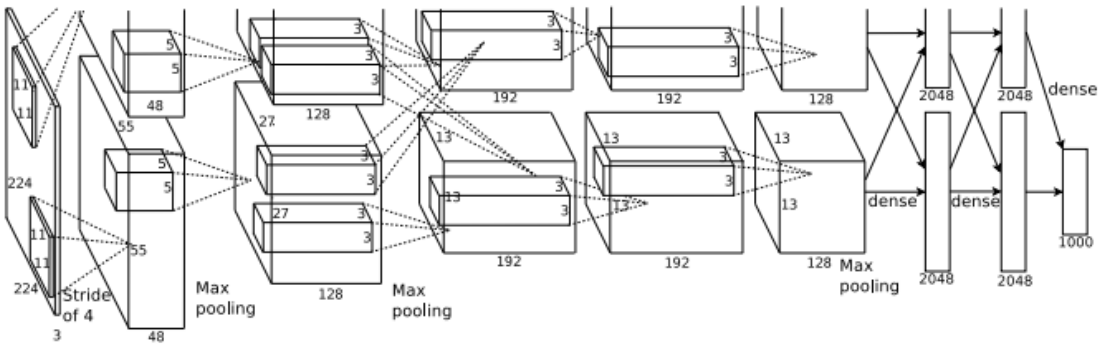
Bu katmanda giriş verisine softmax fonksiyonu uygulanır. Bu fonksiyon ile normalizasyon yapılarak her bir giriş verisinin bir sınıfa ait olma olasılığı, [0, +1] aralığında değerler üretilerek olasılıksal yorumlama şeklinde elde edilmektedir. Sınıflandırma probleminin ortadan kalkması için bu katmanın son tam bağlı katmandan daha sonra yer alması gereklidir. Çıktı biriminin aktivasyon fonksiyonu olan softmax fonksiyonu, giriş verisi vektörünün her elemanı için uygulanır ve genel üstel fonksiyon ile elde edilen toplam üstele bölünme ile normalleştirme yapılmış olur. Bu şekilde,

normalleştirme işlemi ile çıkış vektörünün her biri olasılık değeri ifade eden elemanlarının toplamının bire eşit olması sağlanmış olur [62].

3.2. EVRİŞİMLİ SİNİR AĞI MİMARİLERİ

3.2.1. AlexNet

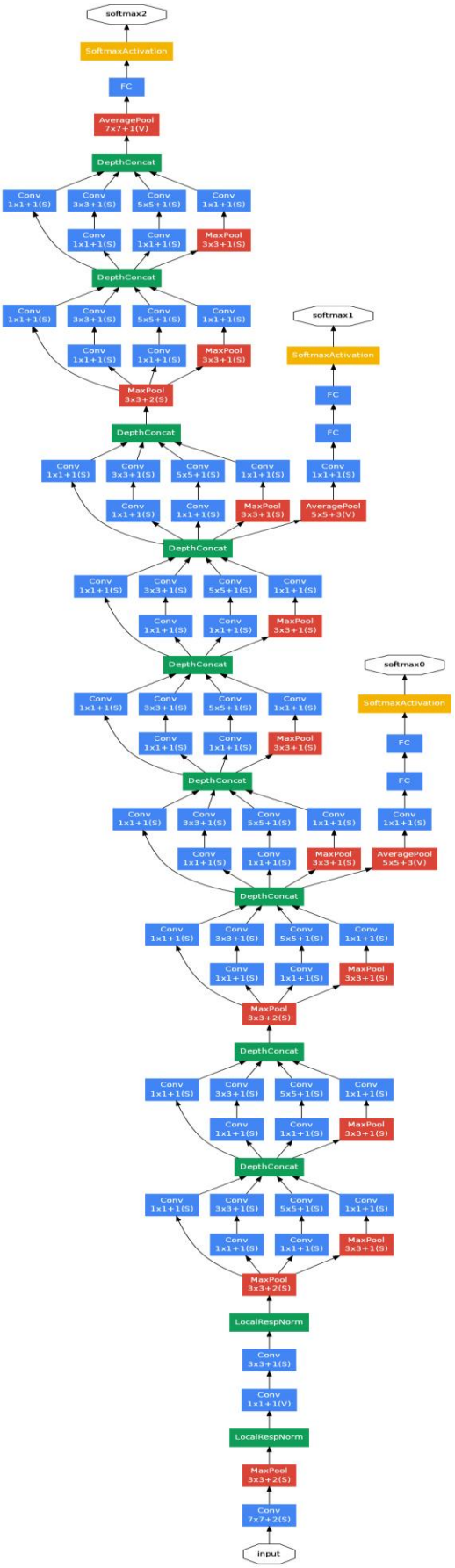
Görüntü işleme problemleri için popüler bir mimaridir. AlexNet mimarisinin geliştirilmesinden önce özellik çıkarma işlemleri manuel olarak uygulanıyordu. Alex Krizhevsky ve arkadaşları tarafından geliştirilen ve ImageNet veri setindeki bir milyondan daha fazla sayıdaki görüntü ile eğitilmiş olan bir CNN mimarisi olan AlexNet mimarisinin geliştirilmesinden sonra görüntü işlemeye yönelik çalışmaların seyri de değişmiştir. Yapısı itibariyle LeNet mimarisine benzer olan AlexNet daha fazla katman içeren daha derin bir CNN mimarisidir ve bu sayede daha çok özellik filtresi içermektedir. Beş tanesi evrişim ve üç tanesi tam bağlı katman olmak üzere toplam sekiz adet derin katmanı bulunmaktadır [56]. 224×224 piksel boyutundaki renkli görüntülerin sınıflandırılması için uygun olan AlexNet mimarisi sahip olduğu 1000 sınıf etiketine sahip softmax çıkış katmanı ile görüntüleri 1000 kategoride sınıflandırabilmektedir. Altmış milyon parametre ve 650.000 nörondan oluşmaktadır. İki GPU üzerinde eğitilmiş olan bu ağda, üstteki katman parçaları bir GPU tarafından, alttaki katman parçaları ise diğer GPU tarafından çalıştırılmaktadır. GPU'lar birbiriyle yalnızca belirli katmanlarda iletişim kurmaktadır. Günümüzdeki GPU teknolojileri sayesinde bu mimari hızlı bir biçimde eğitilmektedir. Şekil 3.2'de AlexNet mimarisi görülmektedir, her bir şeklin altında verilen rakamlar o katmandaki nöron sayısını ifade etmektedir [8].



Şekil 3.2. AlexNet mimarisi [8].

3.2.2. GoogLeNet

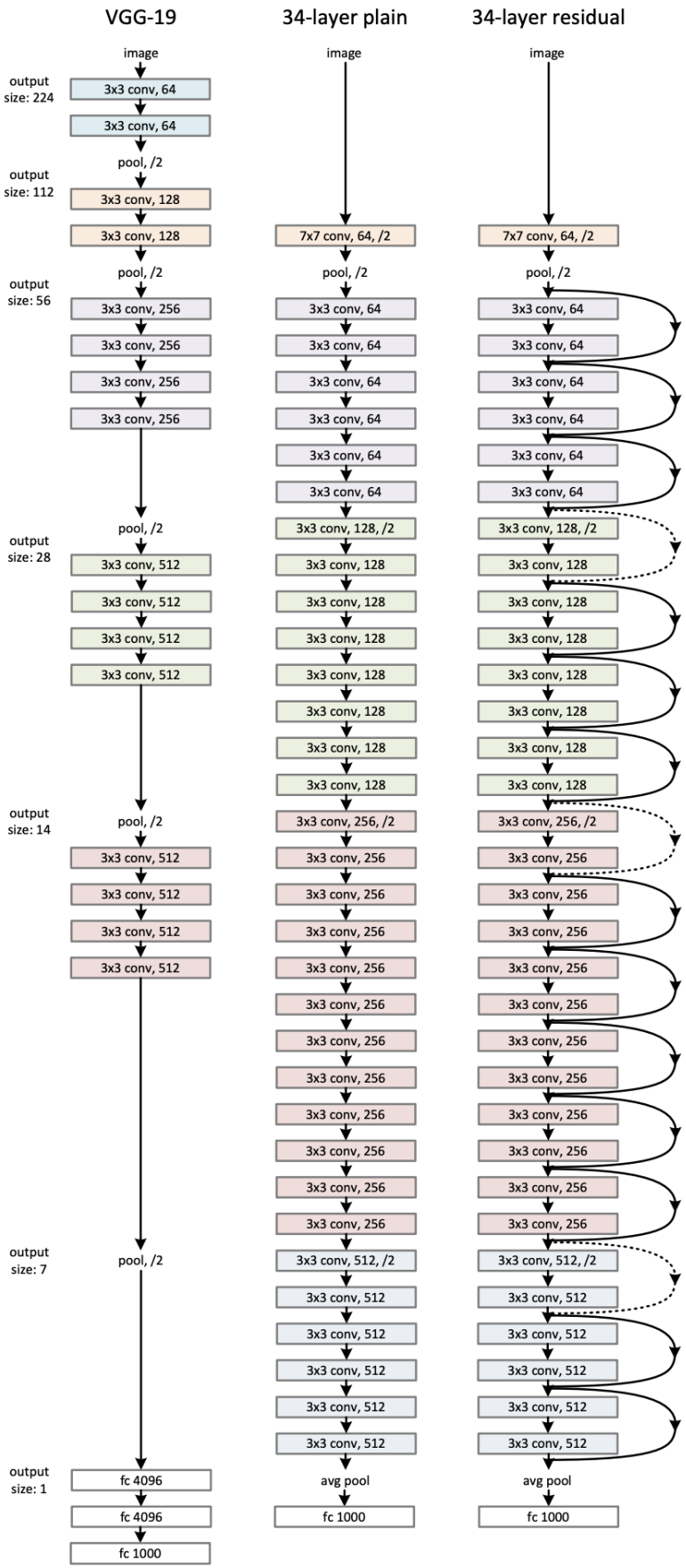
Sınıflandırma problemlerinin çözümü için LeNet ve AlexNet mimarilerinin yapısından etkilenerek geliştirilmiş olan GoogLeNet mimarisinde asıl amaç aşırı uyum problemine çözüm üretmektir [35]. Bu amaç, yapısındaki çok küçük evrişim katmanları sayesinde ağdaki parametre sayısının azaltılması ile sağlanmaktadır. Ağın mimarisinde yer alan 1x1 boyutlu evrişim katmanları doğrusallığın bozulması ve ağdaki parametre sayısının azaltılmasını sağlamaktadır. Yirmi iki tane evrişim katmanı olan GoogLeNet 4 milyon parametre içerirken AlexNet ise yaklaşık 60 milyon parametre içermektedir. Diğer CNN mimarilerinden farklı olarak GoogLeNet bir de başlangıç (inception) katmanı içermektedir. Bu başlangıç katmanı farklı boyutlardaki evrişim katmanları ile etkileşerek tüm özellikleri birleştirmekte ve sınıflandırma başarısı da bu sayede artmaktadır. Bu başarı yanında, GoogleNet CNN mimarisi küçük görüntü alanlarında daha iyi çözünürlük elde edilmesini de sağlamaktadır. GoogleNet mimarisi Şekil 3.3'te verilmiştir [35].



Şekil 3.3. GoogLeNet mimarisi [35].

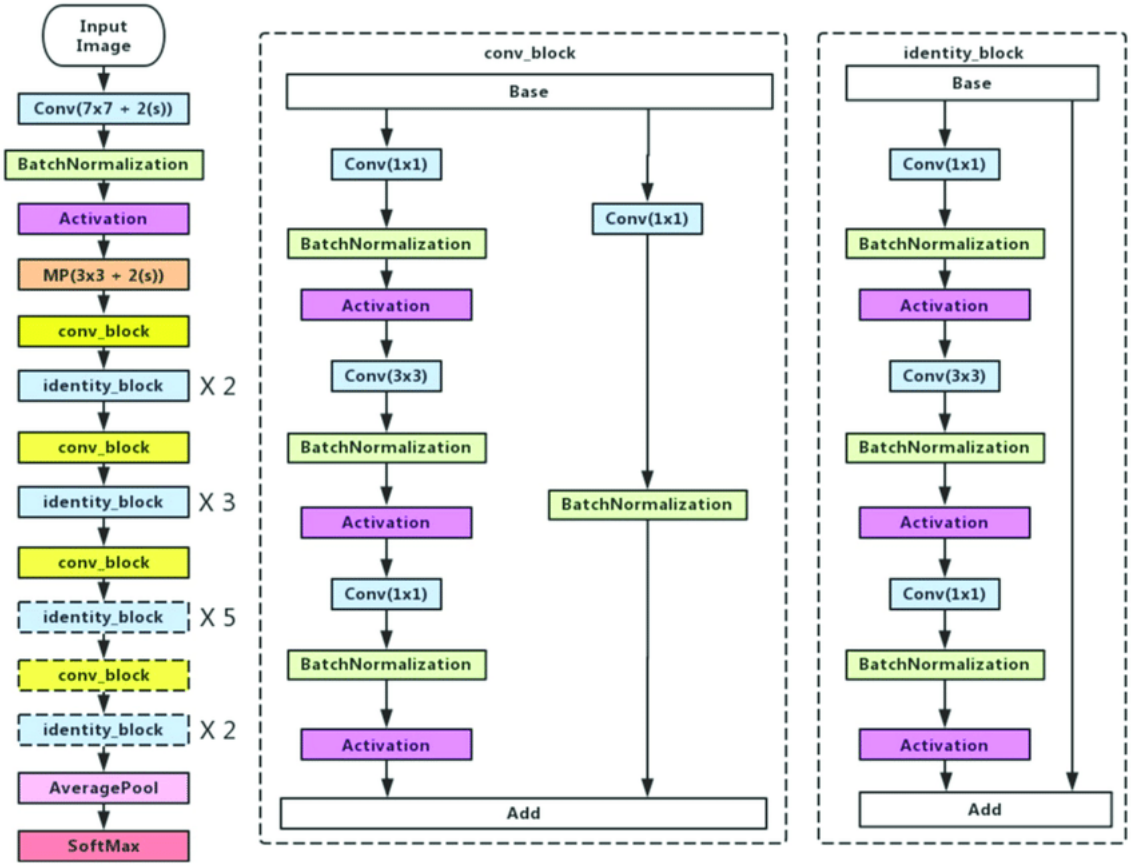
3.2.3. ResNet

Kendinden önceki mimarilerden daha çok derinliğe ve farklı bir mantığa sahip olan, ağ mimarisinin gerçek anlamda derinleşmeye başladığı bir mimari olan ResNet; iki ReLU ile bir doğrusal katman arasında artık değerlerin (residual value) bir kez sıradaki katmanlara beslendiği bir artık bloktan (residual block) oluşmuştur. Sahip olduğu bu özellik ile klasik bir model olmaktan çıkan bu mimari ile öğrenmenin daha hızlı bir şekilde gerçekleşeceği düşünülmüştür [63]. Daha önceki mimarilerden daha derin bir derinliği olan bu mimari katman sayısı açısından da diğer mimarilerdekinden daha fazla sayıda katmana sahiptir [64]. 152 katmandan oluşan ResNet mimarisinin ilk önerilen ağ mimarisi olan 34 katmanlı yapısı Şekil 3.4'te görülmektedir [63].



Şekil 3.4. ResNet mimarisi [63].

Önceden eğitilmiş bir ağ, farklı nesnelerin özelliklerinin belirli bir ağırlık ve yanlılık setini içeren bir ağdır. Daha sonra bu tür veri seti bazında eğitim yapılır. ResNet50 önceden eğitilmiş ağlardan biridir. ResNet50, farklı kategorilerdeki milyonlarca görüntü ile halihazırda eğitilmiş bir sinir ağıdır. Bu ağ, araba, hayvanlar, günlük kullanım aksesuarları, dizüstü bilgisayarlar, doğa ve daha pek çok farklı kategori dahil olmak üzere 1000'den fazla görüntüyü sınıflandıran 50'ye kadar birçok katmana sahiptir. Sonuç olarak, bu ağın zengin özelliklerle ve farklı kategorilerdeki çok çeşitli görüntülerle eğitildiği oldukça açıktır. Derin sinir ağının kendi kendini eğitmesi zor olduğundan ve birçok fayda sağladığından ResNet50'yi tercih etmek iyi bir seçenektir. Çünkü her şey zaten bir standartta ayarlandığından herhangi bir düğüm, katman veya filtre tasarlamaya gerek yoktur ve yalnızca bu ağ üzerinden bir giriş görüntüsü geçirmek yeterlidir. Bununla birlikte ResNet50, gereksinime göre değiştirmek için sınıflandırma katmanlarına erişim de sağlamaktadır. Bu şekilde, çıktının ihtiyaca göre olmasını sağlamak için ağ üzerinde kontrol sağlamak mümkündür. Zayıf çıktı ile zor bir iş olan derin sinir ağı kullanmak ve her filtreyi ve katmanı kendi başına tasarlamak yerine basitçe katmanlar eğitim amacıyla kolayca yeniden formüle edilebilmektedir. Zaten eğitilmiş olan bu ağ kullanılarak, algoritma istenen doğruluk için kolayca optimize edilebilir. ResNet50, ImageNet ve Caltech 101 dahil olmak üzere farklı çevrimiçi veri seti sağlayıcı siteden veri setleri almıştır. ImageNet'te ResNet50, VGG Net üzerinden 8x çözünürlükte yaklaşık 150 katman kullanır ve yine de daha az karmaşıklıkla çalışır. İnsanlar uzmanlık alanlarına ve yeteneklerine bağlı olarak ortalama %5-10 hata oranı ile görüntü sınıflandırırken ResNet 2015 yılında yapılan ILSVRC ImageNet yarışmasında yakaladığı %3,6 hata oranı ile insanlardan daha iyi bir görsel tanıma yaptığını göstermiş ve bu hata oranı ile 2015 yılında gerçekleştirilen ILSVRC ImageNet yarışmasının kazananı olmuştur. Benzer şekilde Caltech 101, ResNet50'nin ImageNet ve diğerlerine göre daha az hata vermesine neden olan optimize edilmiş veri kümesi de sağlamaktadır. Bu nedenle, bu ağ birçok görsel tanıma görevi için çok uygundur. ResNet50 mimarisi Şekil 3.5'te verilmiştir [65].



Şekil 3.5. ResNet50 mimarisi [65].

3.3. EVRİŞİMLİ SİNİR AĞI VE YÜZ TANIMA

Çok katmanlı algılayıcıların bir türü olan CNN sayesinde görüntü içinde birden fazla yüz dahi olsa tanımlanabilmektedir [2], [66]. Görüntülerdeki eğriler ve kenarlar gibi alt düzey özellikleri tespit eder ve görüntüyü katmanlarla işler. Görüntünün özellik tespiti ve görüntü ile ilgili bilgilerin tespiti için farklı katmanlardan faydalanır. Son yıllarda yapılan çalışmalar yüz tanımada hataları azaltmıştır. Ancak hala daha bazı içsel ve dışsal parametrelere duyarlılığı sürmektedir [5]. Görüntülerdeki nesneler ve yüzler farklı boyutlarda olabilir ve ayrıca farklı konumlarda yer alabilir, bu da çıktı katmanında farklılıklara neden olur. Bu nedenle, birden fazla bölge seçme problemini çözmek için standart bir CNN'de yüz tanıma problemlerini çözmek zordur. Standart bir CNN bu sorunları çözemeyebilir. Bu gibi problemleri çözmek için, bölge önerilerinin CNN ile birleştirildiği R-CNN geliştirilmiştir [42]. R-CNN, bir görüntü içindeki görüntü bölgelerini sınıflandırmak için CNN kullanan bir nesne tespit çerçevesidir [42]. Her

bölgeyi kayan bir pencere kullanarak sınıflandırmak yerine, R-CNN dedektörü yalnızca bir nesne içermesi muhtemel bölgeleri işler. Bu da bir CNN çalıştırırken ortaya çıkan hesaplama maliyetini büyük ölçüde azaltır. Belirli bir görüntü için R-CNN ile kutu şeklindeki sınırlar çizilebilir [67]. Bu doğrultuda, bir R-CNN, dikdörtgen bölge önerilerini CNN özellikleriyle birleştiren bir derin öğrenme yaklaşımı olarak ifade edilebilir. Otonom sürüş, akıllı gözetim sistemleri ve yüz tanıma gibi alanlarda R-CNN kullanılabilmektedir. Evrişimli sinir ağları ile yüz tanıma alanında son yıllarda yapılan çalışmalardan bazı örnekler için metot, veri seti ve elde edilen sonuçlara ilişkin bilgiler Çizelge 3.1'de gösterilmiştir.

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri.

Çalışma	**Algoritma**	**Mimari**	**Veri Seti**	**Sonuç**
Jiang ve ark., 2017 [68]	Faster R-CNN	VGG16	Face Detection FDDB IJB-A	Doğru Pozitif %71,8 ile %95,2 arasında
			WIDER FACE	Kolay - mAP %90,3
				Orta - mAP %83,6
				Zor - mAP %46,4
Li ve ark., 2017 [69]	Faster R-CNN	ZFNet	CLDC	mAP %82,03
		VGG16		mAP %83,12
Zhang ve ark., 2018 [70]	Faster R-CNN	ResNet101	WIDER FACE	Kolay - mAP %95,9
				Orta - mAP %94,5
				Zor - mAP %88,8
Hao ve ark., 2018 [71]	Faster R-CNN	VGG16	WIDER FACE	mAP %67,64

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri (devamı).

Çalışma	Algoritma	Mimari	Veri Seti	Sonuç
Garg ve ark., 2018 [72]	YOLOv2		FDDB	Doğruluk %92,2
Cui ve ark, 2019 [73]*	Faster R-CNN	VGG16	CASIA 4.0	Recall %96,3
		ResNet		Recall %94,8
	YOLO			Recall %47,4
Li ve ark, 2019 [74]	MTCNN		FDDB	mAP %90,2
	YOLOv3	DarkNet53		mAP %93,9
Gyawali ve ark., 2019 [75]	Faster R-CNN	VGG16	Chokepoint Employee	Doğruluk %96,0 Precision %92,0 Recall %88,5
			Nepal Telecom employee	Doğruluk %95,2 Precision %83,0 Recall %96,0
Hai ve ark., 2020 [76]	Faster R-CNN	SG-NET	WIDER FACE	AP %87,3
Lin ve ark., 2020 [77]	Mask R-CNN	ResNet101	FDDB	Doğru pozitif %88,8
			AFW	AP %95.97
			WIDER FACE	Kolay - AP %90,2
				Orta - AP %85,4
				Zor - AP %66,2

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri (devamı).

Çalışma	**Algoritma**	**Mimari**	**Veri Seti**	**Sonuç**
Chun ve ark., 2020 [78]	YOLOv3	DarkNet53	WIDER FACE	Kolay - mAP %80,5
				Orta - mAP %76,2
				Zor - mAP %55,2
Singh ve ark., 2021 [79]**	YOLOv3		MAFA, WIDER FACE ve webten karma	AP %55,0
	Faster R-CNN	ResNet		AP %62,0
Jiang ve ark., 2021 [80]	Fast R-CNN	UPL-RCNN	WIDER FACE	Kolay - AP %77,9
				Orta - AP %77,9
				Zor - AP %63,0
Loey ve ark., 2021 [81]**	YOLOv2	ResNet50	MMD+FMD karma	AP %81,0
Sanchez-Moreno ve ark., 2021 [82]	YOLOv3	DarkNet53	LFW	Doğruluk %99,7
			YTF	Doğruluk %94,7
Geroge ve ark., 2021 [83]	Fast R-CNN		bir aktörün görüntüleri ile hazırlanan veri seti	mAP %69,9
	Faster R-CNN			mAP %73,2
	YOLOv3	DarkNet53		mAP %63,4

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri (devamı).

Çalışma	Algoritma	Mimari	Veri Seti	Sonuç
Muhaimin ve ark., 2021 [84]**	YOLOv3	DarkNet53	webte bulunan görüntülerden hazırlanan veri seti	mAP %96,32
	YOLOv3 (Tiny)			mAP %94,09
	YOLOv4			mAP %97,57
	YOLOv4 (Tiny)			mAP %96,75
	YOLOv4 (Tiny-3l)			mAP %96,49
Jebraelzadeh ve ark., 2021 [85]	YOLOv3	DarkNet53 +AlexNet	AFW	mAP %99,4
			PASCAL face	AP %98,34
			FDDB	AP %90,7
Fan ve ark., 2021 [86]	YOLOv3	DarkNet53 +DenseNet	karma	Doğruluk %86,74
Chen ve ark., 2021 [87]	YOLOv3	DarkNet53	WIDER FACE	Kolay - AP %74,1
				Orta - AP %70,6
				Zor - AP %47,1
Yolcu Öztel ve ark., 2021 [88]**	Faster R-CNN	ResNet50	FMD	AP %79,0
	YOLO			AP %92,0

Çizelge 3.1. Yüz tanıma ile ilgili son yıllarda yapılan çalışma örnekleri (devamı).

Çalışma	Algoritma	Mimari	Veri Seti	Sonuç
Kale ve ark., 2022 [89]	YOLOv3	Inceptionv3	Google görüntüler	Doğruluk %93,75
Tsai ve ark., 2022 [90]	SSD	MobileNetV1	LFW	Doğruluk %70,5
		MobileNetV2		Doğruluk %75,5
		MobileNetV2, dynamic triplet		Doğruluk %92,4
		MobileNetV1	VGGFACE2	Doğruluk %93,2
		MobileNetV2		Doğruluk %96,1
		MobileNetV2, dynamic triplet		Doğruluk %94,1

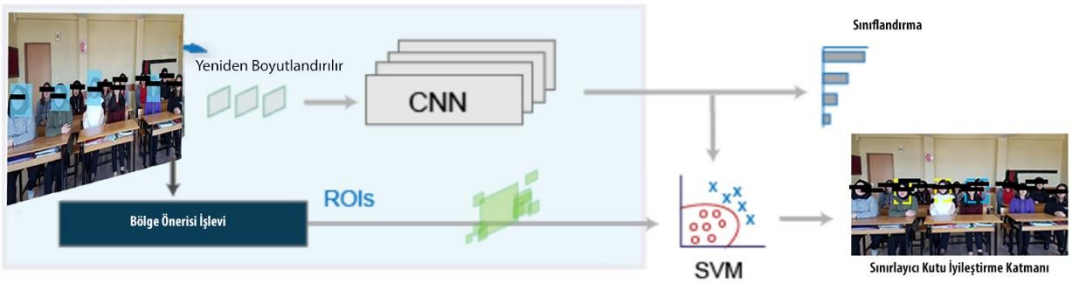
Bu çalışmalarda * göz ve ** maske kullanımı ile yüz tanıma çalışmaları yapılmıştır.

3.4. BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (REGION BASED CONVOLUTIONAL NEURAL NETWORK, R-CNN)

Bölge önerilerine dayalı bir yaklaşım olarak R-CNN 2013 yılında Girshick ve ark. tarafından önerilmiştir [42]. Daha önceki nesne tespit algoritmalarına kıyasla daha başarılı sonuçlar elde ettiği öne sürülen R-CNN diğer uygulamalardan farklı olarak özellik çıkarımı için CNN'lerden faydalanmakta ve bunu bölge teklifi önerileri ile birleştirmektedir. Görüntü sınıflandırması için genellikle CNN kullanılırken, nesne tespiti yapılıyorsa R-CNN tercih edilir [91], [92]. Nesne tanıma, bir görüntüde hangi nesnelerin

bulunduğunu tanımlayan bir işlem iken, nesne tespiti yalnızca görüntüde hangi nesnelerin bulunduğunu söylemekle kalmaz, aynı zamanda nesnelerin görüntüdeki konumunu (nesnenin görüntüde nerede olduğunu) gösterir ve sınırlarını (x, y, genişlik, yükseklik) tespit eder. Bu nedenle, nesne tespiti ve sınıflandırma algoritmaları arasındaki fark, tespit algoritmalarında görüntüdeki konumunu belirtmek için ilgilenilen nesnenin etrafına bir sınırlayıcı kutu çizmektir. Ek olarak, görüntüde ilgilenilen birçok farklı nesne olabileceğinden bir sınırlama kutusu yeterli olmayabilir ve bu nedenle görüntüdeki ilgilenilen farklı nesnelerin konumunu temsil eden birçok sınırlama kutusu çizmek gerekebilir. Üstelik bunun sayısı önceden bilinmeyebilir.

R-CNN ile nesne tespitinde görüntü önce bölge önerilerine bölünür ve ardından her bölge için CNN uygulanır. Bu doğrultuda, R-CNN, giriş görüntüsüne ek olarak ilgilenilen bölgenin (region of interest, ROI) de girdi olarak yer aldığı bir yapı olarak, daha özel bir CNN-özellikli bölgeler olarak ifade edilebilir. Bölge önerilerini CNN özellikleriyle birleştiren R-CNN, iki aşamalı bir tespit algoritmasıdır. İlk olarak, bir görüntüde nesneleri içerebilecek bölgeler tanımlanır. Ardından, her bir bölgedeki nesne sınıflandırılır. Bu nedenle, bir R-CNN algoritması kullanılarak nesne tespiti üç işlemle gerçekleştirilir; i) bir nesne içeren bölgeleri bulunması, ii) bölge önerilerinden CNN özelliklerinin çıkarılması ve iii) çıkarılan özellikler kullanılarak nesnelerin sınıflandırılması. İlk aşamada, bölge önerileri olarak adlandırılan, görüntüde bir nesneyi içerme olasılığı olan bölgelerin bir alt kümesi bulunur. İkinci aşamada, bu bölge önerilerinden CNN özellikleri çıkarılmaktadır. Son olarak, önerilen bölgeler daha sonra eğitilmiş destek vektör makinesi (support vector machine, SVM) tarafından geliştirilir ve nesneler, çıkarılan özellikler kullanılarak sınıflandırılır [42]. Bu çalışmada olduğu gibi, öğrencilerin yüzünü tanımada, görüntünün tamamına değil, öğrencilerin yüzünün bulunduğu alanlara odaklanılmaktadır. Sistemin daha etkin ve hızlı çalışmasını sağlayan R-CNN bu aşamada devreye girmektedir. R-CNN dedektörü, öncelikle kenar kutuları (edge boxes) gibi bir algoritma kullanarak bölge önerileri üretir [93]. Önerilen bölgeler kırpılır, yeniden boyutlandırılır ve sınıflandırılır. Bir R-CNN object detector eğitimi için trainRCNNObjectDetector fonksiyonu kullanılır ve bu fonksiyon bir görüntüdeki nesneleri algılayan bir rcnnObjectDetector nesnesi döndürür. R-CNN dedektör algoritmasının çalışma prensibi Şekil 3.6'da gösterilmiştir.



Şekil 3.6. R-CNN dedektör algoritmasının çalışma prensibi.

Nesne tespiti için CNN tabanlı bir yöntem olarak önerilen R-CNN bu anlamda sonraki çalışmalar için yol gösteren bir özelliğe sahiptir. Bununla birlikte ilk olmaktan kaynaklı bir takım dezavantajlar da söz konusudur. Yine Girshick ve ark. tarafından geliştirilmeye devam eden R-CNN için üç ana sorun üzerinde durulmuştur [94]. Eğitim sürecinin ayrı ayrı aşamalar halinde olması ve eğitim sürecinin zaman alıcı olması iki ana problem olarak belirtilmiştir. Bölge önerilerinin her biri için özellik çıkarımı yapılarak bu özellikler kaydedilmekte, bu süreç hem SVM hem de bölge öneri algoritması için işlemektedir. Bu da eğitim sürecinin çok uzun, hatta günlerce sürmesine ve eğitim süreci için fazla miktarda depolama alınana gereksinim duyulmasına neden olmaktadır. Üçüncü ve en önemli dezavantaj ise eğitim sürecinden sonra nesne tespit işlemi için gereken süredir. Bu aşamada bir dakikaya yakın bir süre gereksinimi ortaya çıkmaktadır. Bunun nedeni farklı bölge önerileri aynı nesneyi işaret etse de veya bölge önerileri üst üste çakışsa da bu süreçteki tüm işlemlerin aynı şekilde tüm bölge önerilerine uygulanmasıdır. R-CNN için önemli eksiklikler olarak görülen bahsedilen bu üç dezavantaj ile ilgili problemlere yönelik olarak R-CNN daha verimli sonuçlar alınabilmesi için sonraki yıllarda da geliştirilmeye devam etmiştir.

Bilgisayarla görme ve özellikle nesne tespiti için makine öğrenimi modellerinin bir ailesi olan R-CNN'de asıl amaç, alınan bir girdi görüntüsünden çıktı olarak her biri bir nesne ve bu nesnenin sınıfını da içeren bir dizi sınırlayıcı kutu üretmektir. Daha sonraki yıllarda, diğer bilgisayarla görme görevlerini de gerçekleştirebilmek için R-CNN genişletildi. 2013 yılında ilk önerilen R-CNN formunda, verilen bir girdi görüntüsü için her biri görüntüdeki bir nesnenin sınırlarını belirleyen bir dikdörtgen şeklinde olan ve sayıları iki bin kadar olabilen ROI'leri çıkarmak için seçici arama mekanizmasını uygulanmaktadır. Her bir ROI, çıktı özellikleri üretmek için bir sinir ağına beslenmektedir. Çıktı özellikleri

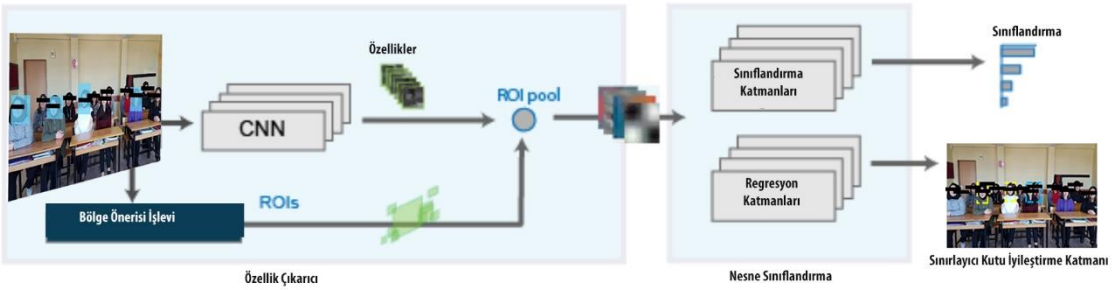
için, her bir ROI'de ne tür bir nesnenin bulunduğunu belirlemek için SVM sınıflandırıcıları kullanılmaktadır [42], [67]. Daha sonra, 2015 yılında fast R-CNN önerildi [94]. İlk R-CNN'de iki bine kadar olan her bir ROI için sinir ağı özellikleri bağımsız olarak hesaplanırken fast R-CNN'de sinir ağı tüm görüntü üzerinde bir kez çalıştırılmaktadır. Ağın sonunda, her bir ROI'yi ağın çıktı tensöründen ayıran, yeniden şekillendiren ve sınıflandıran RoI havuzlama (ROIPooling) adlı yeni bir yöntem vardır. Bölge önerileri oluşturmak için fast R-CNN'de de ilk R-CNN'de olduğu gibi seçici arama kullanılmaktadır [67], [94]. Yine 2015 yılında, fast R-CNN'den hemen sonra bölge önerilerinin üretilmesinde seçici arama yerine bu işlemin sinir ağının kendisine entegre edildiği faster R-CNN önerilmiştir. Fast R-CNN'e benzer bir şekilde, faster R-CNN'de de görüntü girdi olarak özellik haritası çıkaran bir ağa verilir. Bu ağda bölge önerileri önerilen bölge içinde görüntünün sınıflandırılması ve sınırlayıcı kutular için değerlerin tahmin edilmesi için kullanılan RoI havuzlama katmanının kullanılmasıyla yeniden şekillendirilmektedir [67], [95].

3.5. HIZLI BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (FAST REGION BASED CONVOLUTIONAL NEURAL NETWORK, FAST R-CNN)

Seçici arama ile bir nesneyi içerme olasılığı daha yüksek olan farklı 2000 bölge önerisi çıkarılan ve bu bölgeler üzerinde çalışılan R-CNN mimarisinin halen birkaç temel dezavantajı vardı. Bunlardan birisi, her bir görüntü için 2000 bölge önerisinin sınıflandırılması gerektiğinden ağın eğitiminin halen çok fazla zaman almasıydı. Ayrıca, her bir test görüntüsü için de yaklaşık olarak 47 saniye gerektiğinden gerçek zamanlı olarak da uygulanamıyordu. Son olarak, seçici arama bir sabit algoritma olduğundan bu algoritma aşamasında hiçbir öğrenmenin gerçekleşmiyor olması ve bu durumun da kötü bölge önerilerinin üretilmesine yol açabilmesiydi. Bu dezavantajlara yönelik çözüm ve daha hızlı bir nesne algılama önerisi olarak fast R-CNN yine Girshick ve arkadaşları tarafından R-CNN'in geliştirilmiş bir hali olarak 2015 yılında önerildi [94]. Geliştirmedeki temel fikir evrişim katmanında yapılan hesaplamaların her bir bölge önerisi için ayrı olarak değil görüntünün tamamı için sadece bir kez yapılması prensibine dayanmaktaydı. R-CNN algoritmasına benzer olan bu yaklaşımda, tüm bölge önerileri için bir CNN çalıştırılması yerine evrişimli bir özellik haritası oluşturulması amacıyla giriş görüntüsü için bir CNN çalıştırılmaktadır. Böylece her bir görüntü için sadece bir

kez evrişim işlemi yapılmakta ve bir evrişimli özellik haritası oluşturulmaktadır. Bu şekilde eğitim ve test süresinin kısalmasıyla fast R-CNN, R-CNN mimarisinden önemli ölçüde daha hızlı olmaktadır.

R-CNN dedektöründe olduğu gibi, fast R-CNN dedektörü de bölge önerilerinin oluşturulması için yine kenar kutuları (edge boxes) gibi bir algoritma kullanır [94]. Bölge önerilerinin kırpıldığı ve tekrar boyutlandırıldığı R-CNN dedektöründen farklı olarak, fast R-CNN dedektörü tüm görüntüyü işler. Bir R-CNN dedektörü her bir bölgeyi sınıflandırırken, fast R-CNN ise her bir bölge önerisine karşılık gelen CNN özelliklerini havuzlar. Fast R-CNN dedektöründe çakışan bölgeler için yapılan hesaplamalar paylaşıldığından, R-CNN'e göre daha verimlidir. Bir fast R-CNN object detector eğitimi için trainFastRCNNObjectDetector fonksiyonu kullanılır ve bu fonksiyon bir görüntüdeki nesneleri algılayan bir fastRCNNObjectDetector nesnesi döndürür. Fast R-CNN dedektör algoritmasının çalışma prensibi Şekil 3.7'de gösterilmiştir.



Şekil 3.7. Fast R-CNN dedektör algoritmasının çalışma prensibi.

Fast R-CNN ile uygulanan tespit işleminde nesne tespiti yapılacak olan görüntü girdi olarak verilir. Evrişim ve maksimum havuzlama katmanlarından geçirilen görüntü için bölge önerileri R-CNN'deki gbi CNN'den ayrı olarak oluşturulur. Bu şekilde elde edilen bölge önerileri ve özellikler havuzlama katmanına gider. Havuzlama katmanında her bir bölge önerisi rxc boyutunda alanlara bölünerek bu alanların her birine maksimum havuzlama işlemi kendi içinde yapılır. Böylece bölge önerileri farklı boyutlarda olsa bile

sonuç olarak elde edilen özellik matrisi her bölge önerisi için rxc boyutlu olmaktadır. Bu sayede her bölge önerisi için özellikler ayrı ayrı elde edilir. Bu şekilde elde edilen özellikler, son olarak tam bağlı katmana gönderilir. Bu katman nesne için sınıf olasılıklarının hesaplandığı softmax katmanı ve kenar kutuları (edge boxes) için koordinatların hesaplandığı regresyon katmanı olmak üzere iki adet katmandan oluşmaktadır [96]. Özellikler son olarak bu tam bağlı katmana gönderilir ve çıktı verisi elde edilmiş olur.

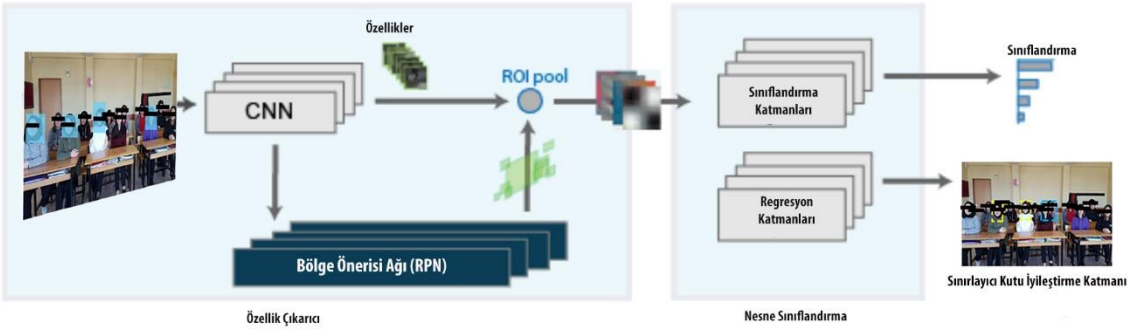
Bu noktada fast R-CNN için öne çıkan temel özellik her bölge önerisi için CNN hesaplamalarının ayrı ayrı yapılmıyor olmasıdır. Bu özellik sayesinde fast R-CNN'de eğitim süresinin R-CNN'e kıyasla 9 kat, test süresinin ise 213 kat kadar azaldığı belirtilmiştir [94]. Bununla birlikte, seçici arama algoritması ile bölge önerilerinin oluşturulduğu fast R-CNN'de bölge önerilerinin üretilmesi noktasında hız açısından bir dezavantaj ortaya çıkmaktadır [95]. Fast R-CNN'de yaklaşık 2 saniye kadar sürmekte olan bölge önerilerinin oluşturulması işlemi R-CNN'de test işlemi için geçen sürenin yaklaşık %86'sı kadardır [94]. Bunun üzerine, daha sonra önerilen faster R-CNN ile bu duruma çözüm önerisi getirilmiştir.

3.6. DAHA HIZLI BÖLGE TABANLI EVRİŞİMLİ SİNİR AĞI (FASTER REGION BASED CONVOLUTIONAL NEURAL NETWORK, FASTER R-CNN)

Hem R-CNN hem de fast R-CNN mimarisinde bölge önerilerinin bulunması için kullanılan seçici arama algoritması ağın performansını olumsuz etkileyen zaman alıcı ve yavaş bir süreçtir. Ren ve ark. tarafından seçici arama algoritmasının modelden kaldırıldığı ve ağ tarafından bölge önerilerinin öğrenilmesinin sağlandığı bir nesne tespiti algoritması tasarlanmıştır [95]. Faster R-CNN adı verilen bu algoritmada, fast R-CNN ile benzer bir şekilde, girdi olarak görüntü özellik haritası üreten bir ağa verilir. Bölge önerilerinin belirlenmesi için seçici arama algoritmasının özellik haritası üzerinde kullanılması yerine, bölge önerilerinin tahmin edilmesi için son evrişim katmanından da sonra bölge önerisi ağı (region proposal network, RPN) olarak adlandırılan ayrı bir ağ kullanılır. RPN ile getirilen yenilik, doğrudan havuzlama katmanına bağlanılmasıdır [95]. RPN de tam olarak bir CNN'dir. Bu ağ girdi olarak verilen bir görüntüdeki bölge önerileri için kullanılır ve bu bölge önerileri için doğruluk skorları (tahmin doğruluk değerleri)

hesaplanarak sınıflama katmanı aşamasına gelindiğinde fast R-CNN'de birleştirilir. Fast R-CNN'de biri softmax sınıflama katmanı ve ikincisi bölgenin tespit edilme doğruluğunu gösteren regresyon katmanı olmak üzere iki çıkış katmanı vardır [96]. Faster R-CNN'de, görüntünün üzerinde bölge önerilerinin yapılması yerine görüntü önce bir CNN katmanından geçirilerek özellik haritası çıkarılır. Ve bu özellik haritasında bölge önerileri yapılır. Bu şekilde, önerilen her bir bölgenin ayrı ayrı CNN katmanından geçirilmesi yerine, görüntü CNN ağından yalnız bir kez geçirilir. CNN ağı, en fazla işlem yüküne sahip olan katmandır. Bu şekilde, işlem yükü önemli ölçüde azaltılır. Yani faster R-CNN ile girdi olarak verilen görüntü, sınıflandırma işlemi için yalnız bir kez işlenir ve ilgilenilen nesnelerin bulunduğu bölgeleri belirleyen kenar kutularının sınırları düzeltilir. Fast R-CNN'de olduğu gibi bölge önerileri ve özellikler havuzlama, sınıflandırma ve kenar kutu katmanlarına birlikte gönderilerek çıktılar elde edilir. Böylece, faster R-CNN modelinde, özellik haritasında önerilen bölgeler tam bağlantılı katmana bağlanır. Tam bağlı katman belli boyutta bir girdi gerektirdiğinden, tespit edilmiş olan bölge boyutları tam bağlı katmanın girişine uygun olarak ayarlanır [92]. Sınıflandırma işlemi, faster R-CNN ile belirlenen aday bölgeler rastgele seçilerek gerçekleştirilir. Faster R-CNN, CNN ile sınıflandırmada görülen aynı sonuçları elde etme sorununu ortadan kaldırdığı için hesaplama süresini azaltır. Evrişim katmanlarının paylaşımlı şekilde kullanılarak bölge önerilerinin daha maliyetsiz olarak üretilmesi ve işlemlerin baştan sona tek bir ağ üzerinde yapılarak sonuçların elde edilmesi, işlemlerin de önceki versiyonlara göre çok daha hızlı bir şekilde yapılabilmesi demektir [95]. Böylece, faster R-CNN, hem R-CNN ve hem de fast R-CNN ile kıyaslandığında çok daha hızlı olup gerçek zamanlı nesne tespiti için bile kullanılabilir [67].

Faster R-CNN dedektörü, kenar kutuları (edge boxes) gibi bir harici algoritmanın kullanılması yerine, doğrudan sinir ağında bölge önerilerinin oluşturulması için bir RPN kullanır [95]. RPN, Nesne Algılama için Bağlantı Kutularını (Anchor Boxes for Object Detection) kullanır. Ağda bölge önerileri oluşturmak daha hızlıdır ve verilere göre daha iyi ayarlanmıştır. Bir faster R-CNN object detector eğitimi için trainFasterRCNNObjectDetector fonksiyonu kullanılır ve bu fonksiyon bir görüntüdeki nesneleri algılayan bir fasterRCNNObjectDetector nesnesi döndürür. Faster R-CNN dedektör algoritmasının çalışma prensibi Şekil 3.8'de gösterilmiştir.



Şekil 3.8. Faster R-CNN dedektör algoritmasının çalışma prensibi.

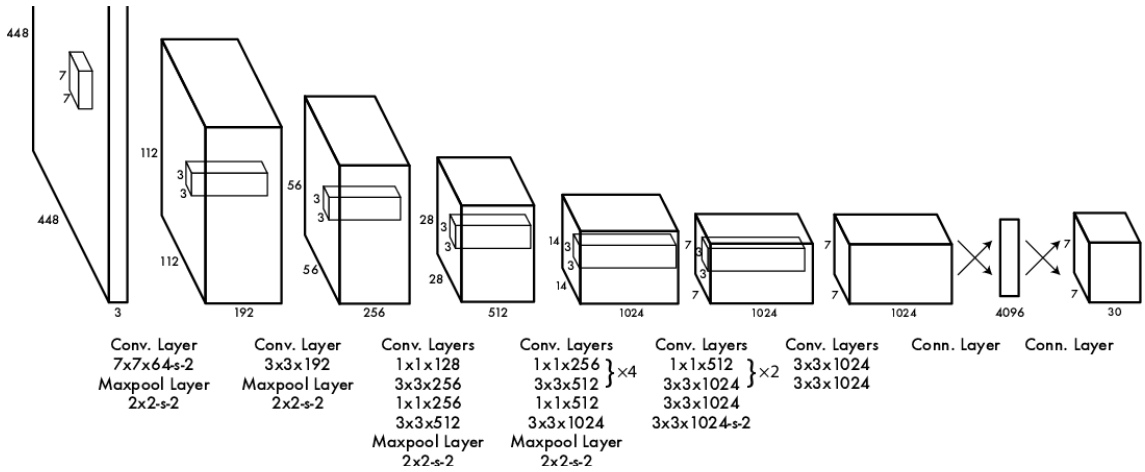
3.7. SADECE BİR KEZ BAK (YOU LOOK ONLY ONCE, YOLO)

Daha önceki tüm nesne tespit algoritmalarında, nesnenin görüntüdeki konumunu belirlemek amacıyla bölgeler kullanılmaktadır. Ağ, görüntünün tamamına bakmaz, nesnenin içerilme olasılığı yüksek olan görüntünün parçalarını kullanır. Redmon ve ark. tarafından bir görüntüde hangi nesnelerin yer aldığını ve bu nesnelerin görüntünün neresinde olduğunu tahmin etmek için görüntüye yalnızca bir kez bakarsınız (you only look once, YOLO) şeklinde ifade edilerek önerilen ve adlandırılan YOLO, yukarıda verilen bölge tabanlı diğer algoritmalardan çok farklı olan, hem nesnenin yerinin tespit edilmesi hem de sınıflandırılması için tek bir CNN kullanılan bir nesne tespit algoritmasıdır. Tek bir evrişimli ağ, tüm görüntüden tek bir değerlendirme ile sınırlayıcı kutuları (bounding boxes) ve bu kutular için sınıf olasılık değerlerini tahmin eder [97].

YOLO'nun çalışma prensibinde, bir görüntü SxS boyutlu bir ızgaraya bölünür, her bir ızgaranın içinde m adet sınırlayıcı kutu belirlenir. Eğer bir nesnenin merkezi SxS boyutlu bu matrisin bir hücresi içerisinde olursa o nesneyi algılamak ilgili matris hücresinin sorumlu olduğu bir durum olur. Her bir matris hücresi için B adet sınırlayıcı kutu belirlenir ve bu kutular için olasılık değerleri tahmin edilir. Buradaki olasılık değeri, modelin ilgili kutuda bir nesne içerildiğinden ne düzeyde emin olduğu ve o kutunun yaptığı tahminin ne düzeyde doğru olduğunu göstermektedir. Ağ, her bir sınırlayıcı kutu için sınıf olasılığını ve sınırlayıcı kutunun ofset değerlerini verir. Bir eşik değerinin üstünde sınıf olasılığı olan sınırlayıcı kutuları seçilir ve görüntü içinde nesneyi konumlandırmak üzere kullanılırlar. Olasılık değerleri, modelde tahmin edilen sınırlayıcı kutunun gerçekteki sınırlayıcı kutuya oranı ile regresyon yöntemiyle elde edilen sınıf

olasılığı değerinin çarpılmasıyla elde edilmektedir. Eğer, ilgili matris hücresi hiç bir nesne içermiyorsa olasılık değeri sıfır olmaktadır. Belirlenen sınırlayıcı kutuların her biri için beş adet değer tahmin etmektedir: i) sınırlayıcı kutunun matris hücresine göre hesaplanan merkezi (x, y); ii) sınırlayıcı kutunun tüm görüntüye göre hesaplanan en ve boy değeri (weight, height); iii) sınırlayıcı kutunun gerçekteki sınırlayıcı kutuya oranı olarak elde edilen olasılık değeri. Bir de her bir matris hücresi için bu hücrenin içerdiği parçanın hangi sınıfa ait bir görüntünün bir parçası olduğunu belirten C adet sınıf tahmini oluşturulmaktadır. YOLO, Redmon ve ark. tarafından Pascal VOC [98] veri setinde S değeri 7 ve B değeri 2 alınarak değerlendirilmiştir. Bu veri setinde 20 farklı sınıfa ait nesneler yer aldığından C değeri, yani sınıf sayısı ise 20'dir. Bu durumda sonuç tahmini için SxS x(Bx5 + C) ile 7x7x30 boyutlu bir matris ortaya çıkmıştır [97].

Ağın ilk evrişim katmanlarında, görüntüden özelliklerin çıkarımı yapılırken, tam bağlı katmanlarda ise çıktı olasılıkları ve koordinatlar tahmin edilmektedir. Ağ mimarisinde, görüntü sınıflandırma için GoogLeNet modelinden esinlenilmiştir [35]. Ağ yapısında 24 adet evrişim katmanı ve daha sonra 2 adet tam bağlı katman bulunmaktadır. GoogLeNet modelinde yer alan başlangıç modülleri yerine, Lin ve ark. [99] tarafından yapılan çalışmadakine benzer şekilde 1x1 indirgeme katmanları ve ardından 3x3 evrişim katmanları kullanılmaktadır. YOLO ağ mimarisi Şekil 3.9'da gösterilmiştir.



Şekil 3.9. YOLO ağ mimarisi [97].

Nesne tespit işlemi daha önceki algoritmalarda sınıflandırma yöntemi ile çözülmekteyken YOLO'da nesne tespit işlemi bir regresyon denklemi olarak ele yürütülmektedir. Bu regresyon denkleminde görüntü pikselleri, sınırlayıcı kutular ve sınıf olasılıkları ele alınmaktadır. Tek bir evrişimli ağ, birden çok sınırlayıcı kutu ve bu kutuların sınıf olasılıklarını aynı anda tahmin etmektedir. Bu birleşik model yapısı, daha önceki nesne tespit algoritmalarına kıyasla bazı avantajlara sahiptir. Algılama süreci bir regresyon modeli olarak ele alındığından kompleks ardışık bir düzene ihtiyaç duyulmayan YOLO'da girdi olarak görüntü algoritmaya bölge önerileri şeklinde değil bütün olarak verildiğinden, diğer nesne algılama algoritmalarından çok daha hızlıdır. Temel ağın saniyede 45 kare hızında olduğu ve hızlı bir sürümde bu hızın saniyede 150 kareden daha hızlı çalışabildiği belirtilmiştir [97]. Bu, gerçek zamanlı bir video akışının 25 milisaniyeden daha az bir gecikmeyle işlenebileceği anlamına gelmektedir. Ayrıca, YOLO diğer gerçek zamanlı sistemlerin ortalama kesinlik değerlerinin ortalamasının (mean average precision, mAP) iki katından daha fazlasını başarabilmektedir.

Tahmin yaparken YOLO görüntünün tamamı ile ilgili genel olarak tahminler yapar. YOLO, bölge önerisi tabanlı ve kayan pencere yöntemlerinin aksine, eğitim ve test süresi boyunca görüntünün tamamını dikkate alır, ve böylece sınıfların görünümleri kadar bu sınıflar ile ilgili bağlamsal ve bütünsel bilgileri de örtük olarak kodlar. En iyi tespit yöntemi olan fast R-CNN, daha geniş ve bütünsel olarak bağlamı göremediğinden bir görüntüde yer alan arka plan parçaları ile nesneleri karıştırırken, YOLO, fast R-CNN'ye kıyaslandığında yarısından da daha az arka plan hatası yapmaktadır ve arka planda yanlış pozitif tahmin yapma olasılığı da daha düşüktür.

YOLO nesnelerin genelleştirilebilir temsillerini öğrenmektedir. Doğal görüntüler ile eğitilip sanatsal eserler üzerinde test edildiğinde, deforme parça modeli (deformable parts model, DPM) ve R-CNN gibi en iyi nesne tespit yöntemlerini bile açık ara geride bırakmıştır. YOLO oldukça genelleştirilebilir bir mimari olduğundan, yeni alanlara ya da beklenmeyen girdilere uygulandığında da bozulma olasılığı düşüktür [97].

YOLO algoritmasının kısıtlılığı, görüntü içinde yer alan küçük nesneler ile mücadele etmesidir, örnek olarak bir kuş sürüsünün tespitinde YOLO zorlanabilir. Bu durum, algoritmanın sınırlayıcı kutu tahminlerine güçlü uzaysal kısıtlamalar getirmesinden kaynaklanmaktadır. Başka bir kısıtlılık ise YOLO modeli veriden sınırlayıcı kutular tahmin etmeyi öğrendiğinden, olağandışı, yeni veya beklenmeyen en ve boy oranındaki ya da konfigürasyonlardaki nesneler için genelleme yapmakta zorlanmasıdır [97].

YOLO, sınırlayıcı kutular çizme ve sınıf etiketlerini tanımlama problemini uçtan uca türevlenebilir tek bir ağda birleştiren ilk nesne tespit ağıdır. YOLO geliştirilerek gerçek zamanlı bir nesne tespit modeli olan YOLOv2 önerilmiştir [100]. Yine Redmon ve Farhadi tarafından önerilen, ImageNet ve MS COCO veri setlerinin birleştirilmesinden elde edilen verilerin kullanılmasıyla eğitilen, 9000'in üzerinde nesne sınıfını algılayabilen ve YOLO9000 olarak da adlandırılan bu model, araştırmacılar tarafından da daha iyi, daha hızlı ve daha güçlü şeklinde ifade edilerek önerilmiştir. Modelin geliştirilmesinde öncelik hızlı olma özelliğine verilirken başarı oranını artırabilmek için ağ yapısını daha derinleştirmek yerine daha basitleştirerek temsilin öğrenilmesinin kolaylaştırılması amaçlanmıştır. YOLOv2 ile YOLO üzerinde BatchNorm, daha yüksek çözünürlük ve bağlantı kutuları (anchor boxes) dahil bir dizi iyileştirme yapılmıştır. Bağlantı kutuları görüntüdeki her bir nesne için ayrı ayrı kutular oluşturmakta ve her bir nesne için ayrı ayrı güven skoru, konum (x, y) bilgisi, en ve boy (w, h) bilgisi ve göreli sınıf olasılığını hesaplamaktadır. Bu hesaplamalarda birleşme alanındaki kesişme (Intersection over Union, IoU) değeri kullanılmaktadır. Bu şekilde algoritma daha kararlı hale gelmektedir. Ancak bütün bunları yaparken gereksiz olarak birden fazla kutu elde edilmektedir. Bu karışıklığın önüne geçmek için Non Max Suppression algoritması kullanılmıştır. En başta hesaplanan güven skoru düşük olan kutular silinerek işleme başlanmaktadır. Algoritmanın görevi en yüksek güven skoruna sahip kutuyu çıktı olarak vermektir. YOLOv2'de 19 adet evrişim katmanı ve nesne takibi için eklenmiş olan ekstra 11 adet katmanı olan DarkNet-19 mimarisi kullanılmıştır. Önceki sınıflandırıcıların çoğu girdi olarak 256x256 boyutundan daha küçük görüntülerle çalışırken YOLO'da sınıflandırma ağı 224x224 boyutunda eğitilmekte ve nesne tespiti için çözünürlüğü iki katına, 448x448 boyutuna çıkarmaktadır. Girdi olarak alınan görüntünün çözünürlüğünün artırılması ortalama doğruluğu yoklaşık olarak %4 oranında artırmıştır [97]. YOLOv2'de YOLO'daki tam bağlantılı katmanlar çıkarılmış, sınırlayıcı kutuların tahmin edilmesi için bağlantı kutuları kullanılmış ve girdi görüntü boyutu da 416x416 şeklinde ayarlanarak büyük görüntüler için tek bir konumun tahmin edilebileceği bir ağ yapısı tasarlanmıştır. YOLOv2'de farklı bir özellik olarak görüntünün boyutlarının 32'nin katı şeklinde 320x320 ila 608x608 arasında her 10 döngüde bir rasgele değiştirilmesi sayesinde ağın farklı boyutlardaki görüntüler için de öğrenmeye devam etmesidir. Bu şekilde ağ daha küçük boyutlarda daha hızlı çalıştığından hız ve doğruluk açısından da bir denge sağlanmış olmaktadır. YOLOv2 sınırlayıcı kutuların koordinatlarının tahmini ve yerelleştirme özelliği yanında, görüntüdeki ortak olan nesnelerin nasıl sınıflandırılacağı

gibi nesne tespitine has durumları da öğrenmek için tasarlanmış olan bir algoritma olduğundan ağın eğitimi sürecinde hem nesne tespiti hem sınıflama ile ilgili verilerin karıştırıldığı veri setleri ile yapılmıştır. VOC 2007 ve VOC 2012 veri setleri ile eğitilen SSD ve faster R-CNN modelleri ile karşılaştırma yapıldığında, YOLOv2'nin daha hızlı olduğu ve daha doğru sonuçlar ürettiği gösterilmiştir [100].

Daha sonra, yine Redmon ve Farhadi tarafından YOLOv2 geliştirilerek ağın daha derin hale getirildiği YOLOv3 önerilmiştir [101]. Bu modelde de yine görüntü üzerinde ilk olarak sınırlayıcı kutular çizilmekte ve daha sonra bu kutu içerisinde bulunma olasılığı olan bir nesnenin hangi sınıfa ait olduğunu belirlemek üzere nesne tespit işlemi iki aşamalı olarak ele alınmaktadır. YOLOv3 ile sınırlayıcı kutu tahminine bir nesnellik puanı eklenerek önceki modeller üzerine inşa edilmiş, ağ katmanlarına bağlantılar eklenmiş ve daha küçük nesnelerdeki performansı artırmak için üç farklı ayrıntı düzeyinde tahminler yapılmıştır. YOLOv3 modelinde, daha önceki YOLOv2'de bulunmayan, artık bloklar (residual blocks), bağlantı atlama (skip connection) ve yukarı örnekleme (upsampling) özellikleri yer almaktadır. YOLOv3 modelinde, özellik çıkarmak için DarkNet53 mimarisi kullanılmaktadır. Nesne takibi için kullanılan 53 katman ile toplam 106 adet evrişim katmanı ile oluşturulan YOLOv3 içerdiği daha fazla katman nedeniyle YOLOv2'ye kıyasla daha yavaş kalmaktadır. YOLOv3'ün üç farklı ölçekte tahmin yapması, en belirgin özelliği olarak karşımıza çıkmaktadır. Ayrıca YOLOv2'deki havuzlama katmanı çıkarılarak özellik haritasının büyüklüğünün tespit edilen nesnenin büyüklüğüne göre değiştiği bir model oluşturulmuştur. Önerilen bu modelin hem RetinaNet hem SSD modellerine göre daha başarılı ve hızlı olduğu gösterilmiştir [101], [102].

2020 yılında Bochkovskiy ve ark. tarafından önceki versiyonlarına göre daha doğru ve daha hızlı sonuçlar elde edildiği belirtilen YOLOv4 önerildi [103]. YOLOv4 birkaç bileşeni olan iki aşamalı bir nesne tespit dedektörüdür. Modern bir nesne tespit dedektörü genellikle iki parçadan oluşmaktadır. Bunlar, ImageNet ile önceden eğitilmiş olan bir omurga (backbone) ve nesneleri içeren sınırlayıcı kutuları ve bu nesnelerin sınıflarını tahmin etmek için kullanılan bir kafa (head) parçasıdır. Omurga VGGNet, ResNet, ResNeXt veya DenseNet gibi mimariler olabilirken kafa parçası genellikle bir aşamalı nesne ve iki aşamalı nesne tespit dedektörleri olmak üzere iki tipe ayrılır. En sık görülen iki aşamalı nesne tespit dedektörleri fast R-CNN, faster R-CNN gibi modelleri de içeren R-CNN serisi modellerken tek aşamalı nesne tespit dedektörlerinde en sık görülen

modeller ise YOLO, SSD ve RetinaNet gibi modellerdir. Son yıllarda önerilen nesne tespit dedektörlerinde genellikle omurga ve kafa parçaları arasına, farklı aşamalardan özellik haritalarını toplamak için kullanılan katmanlar eklenmektedir. Bu katmana da boyun (neck) adı verilmiştir [103]. Bir boyun yapısı genellikle aşağıdan yukarıya doğru (bottom-up) ve yukarıdan aşağıya doğru (top-down) yollardan oluşmaktadır. Bu mekanizmanın kullanılarak oluşturulduğu ağlar özellik piramit ağı (feature pyramid network, FPN) [104], yol birleştirme ağı (path aggregation network, PAN) [105], iki yönlü özellik piramit ağı (bi-directional feature pyramid network, BiFPN) [106] ve sinirsel mimari arama - özellik piramit ağı (neural architecture search - feature pyramid network, NAS-FPN) [107] gibi ağlardır. Araştırmacılar YOLOv4 mimarisinde omurga olarak CSPDarknet53 omurgasını, boyun olarak PAN yol birleştirme boynunu, SPP ek modülünü ve kafa olarak da bağlantı tabanlı (anchor based) bir yapı olan YOLOv3 kafasını seçtiklerini belirtmişlerdir [103]. Önerilen modelin ImageNet veri setindeki sınıflandırma doğruluğu ve MS COCO veri setindeki tespit doğruluğu üzerindeki etkisini incelemek amacıyla sınıf etiketi düzleştirme, veri artırma, aktivasyon fonksiyonları değiştirme gibi farklı eğitim geliştirme tekniklerini de test etmişlerdir. Önerilen YOLOv4 modelinin yine yeni teknoloji olan diğer gerçek zamanlı nesne tespit modellerine kıyasla daha başarılı ve daha hızlı olduğu belirtilmiştir [103].

3.8. VERİ SETİ

Aynı sınıfta bulunan on iki kız öğrencinin fotoğrafları farklı zamanlarda çekilmiştir. Öğrencilerin tamamı lisedeydi ve yaşları 16 ile 18 arasındaydı. Öğrencilerin yerleri rastgele değiştirilerek farklı oturma düzenleri oluşturulmuş ve fotoğrafları tüm öğrencilerin yüzleri görünecek şekilde farklı oturma düzenleri ile farklı açılardan çekilmiştir. Veri seti 400 farklı fotoğraftan oluşmaktadır. Tüm fotoğraflara on iki öğrencinin tamamı dahil edilmiş ve tüm fotoğraflar aynı uygun mesafeden korunarak çekilmiştir. Veri setindeki görüntüler, 300/100 eğitim/test oranı kullanılarak eğitim ve test seti olmak üzere iki alt gruba ayrılmıştır. Örnek bir fotoğraf Şekil 3.10'da gösterilmiştir.

Şekil 3.10. Veri setinden örnek bir fotoğraf.

3.9. DENEYSEL ÇALIŞMALAR

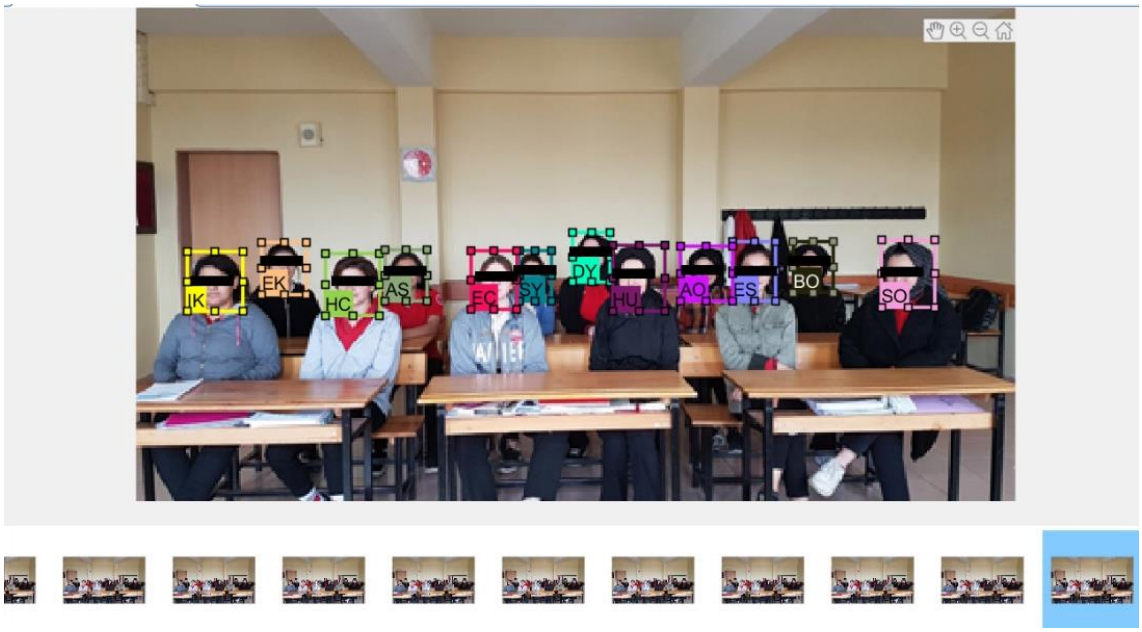
Bu çalışmada, araştırmacılar tarafından çekilen fotoğraflarla oluşturulan veri seti üzerinde öğrencilerin yüz görüntülerinden kimlik tespiti yapılmıştır. Fotoğraflar 4032*2268 boyutunda Samsung 12 MP kamera kullanılarak çekildi. İlk olarak eğitim aşaması Intel 9. nesil i7 işlemcili ve NVIDIA GeForce RTX 2070 8 GB ekran kartı hafızasına sahip bir bilgisayar ile test edildi ancak hafıza yetersiz geldi. Bu nedenle, grafik kartı, grafik kartı belleği ve işlemci hızı gibi sınırlı kaynaklar nedeniyle görüntülerin boyutu zorunlu olarak 420*236'ya indirgendi. Çalışmadaki tüm uygulamalar ve işlemler MATLAB© yazılımı ile oluşturulan uygulamalar ile gerçekleştirilmiştir. Deneysel çalışmalar için kullanılan bilgisayarın konfigürasyonu ve özellikleri Çizelge 3.2'de verilmiştir.

420*236 boyutuna indirgenen veri seti yeniden etiketlendi. On iki öğrencinin her biri, veri setindeki her bir fotoğraf için ayrı ayrı etiketlenmiştir. Etiketli görüntü için bir örnek Şekil 3.11'de görülmektedir.

Etiketler Export Labels ile gTruth nesnelerine dönüştürülmüştür. Böylece eğitim süreci için kullanılacak etiketli görseller oluşturulmuştur.

Çizelge 3.2. Deneysel çalışmalarda kullanılan bilgisayar konfigürasyonu.

Donanım	**Özellik**
İşlemci	Intel Core i7-9750H @2.6 Ghz Turbo Boost 4.5 GHz (6 CPUs)
Hafıza (RAM)	16 GB (DDR IV 8GB*2 (2666 MHz))
İşletim sistemi	Win10 Pro
Ekran kartı	NVIDIA GeForce RTX 2070 ,GDDR6 8GB
Sabit disk	512 GB M.2 PCIe SSD +1TB (SATA) 7200rpm



Şekil 3.11. Etiketleme örneği.

3.9.1. Doğrulama

Doğrulama her eğitim turu sonrası yapay sinir ağının doğruluğunu tespit etmek için kullanılmaktadır. Doğrulama sürecinde, yapay sinir ağının eğitiminin ne kadar iyi olduğunu, eğitimde hataların olup olmadığı tespit edilir. Yapay sinir ağı, eğitim setine göre her görüntüden sonra öğrenir ve kaybı azaltmak için her nöronu ayarlar. Doğrulama herhangi bir eğitim turunda genelleme hatasının bir tahminini vermek için kullanılır. Böylece birkaç farklı hiperparametre konfigürasyonunu yineleyebilir. Doğrulama seti doğruluklarına göre en iyi modeli seçtikten sonra test setine göre hata hesaplanabilir. Sınıflandırma modellerinde sistemin performansını incelemek için çapraz doğrulama yöntemi, en çok kullanılan yeniden örnekleme yöntemlerinden biridir [108] - [110]. Çapraz doğrulama, tahmine dayalı modellerde genelleme yeteneğinin değerlendirilmesi

ve aşırı uyum sorununun önlenmesi için kullanılan yeniden örnekleme yöntemidir [111]. Çapraz doğrulama, modellerin doğru tahmin hatasını hesaplayabilmek ve model parametrelerini ayarlamak için kullanılmaktadır [112]. Çapraz doğrulama, tekrarlanan rastgele alt örnekleme yöntemine benzer, ancak örnekleme, herhangi bir iki test setinin çakışmayacağı bir şekilde yapılır. K-kat çapraz doğrulamada, mevcut eğitim veri seti, her biri eşit büyüklükte olan k adet farklı alt kümeye bölünür. Burada kat, aynı zamanda elde edilen alt kümelerin sayısını ifade eder. Bu bölümleme, geri yerine koyma işlemi yapılmadan eğitim veri setinden rasgele şekilde verileri örnekleyerek gerçekleştirilir. Model, eğitim veri setini temsil eden k-1 adet alt küme kullanılarak eğitilir. Daha sonra model, doğrulama kümesi olarak gösterilen kalan bir adet alt kümeye uygulanır ve modelin performansı ölçülür. İlk katta, ilk alt küme doğrulama kümesi olarak ve geriye kalan dokuz alt küme eğitim kümesi olarak kullanılır. İkinci katta, ikinci alt küme doğrulama kümesidir ve kalan dokuz alt küme eğitim kümesidir, süreç k adet alt kümenin her biri doğrulama kümesi olarak kullanılana kadar bu şekilde tekrarlanır. İşlemler k defa gerçekleştirildikten sonra, her işlemde elde edilen test sonuçlarının ortalaması alınır ve bu şekilde nihai performans hesaplanır. k adet doğrulama setindeki k adet performans ölçümünün ortalaması, çapraz doğrulanmış performanstır [112]. Eğitim veri seti, ilgilenilen popülasyonun bir örneğini temsil etmektedir, bu nedenle eğitim veri setindeki sınıf oranı, popülasyondaki sınıf oranı için en iyi tahmindir. Önyargılı bir değerlendirmeden kaçınmak için, modeli değerlendirmek için kullanılacak olan veri alt kümeleri de bu sınıf oranını yansıtmalıdır. k değerinin seçimi için net bir kural olmamakla birlikte genellikle 5 veya 10 olmaktadır. k büyüdükçe, eğitim veri seti ile yeniden örnekleme alt kümeleri arasındaki boyut farkı küçülür. Bu fark azaldıkça da tekniğin yanlılığı küçülür [113]. Uygulamalı makine öğrenimi alanında ise 10 değeri önerilmektedir [114] ve çok yaygın olarak kullanılmaktadır [115] - [118]. Kohavi, gerçek dünya veri setleri için tabakalı 10-kat çapraz doğrulamayı tavsiye etmektedir [119]. Ayrıca, farklı veri yeniden örnekleme metotlarının karşılaştırıldığı çalışmalarda 10-kat çapraz doğrulamanın en küçük yanlılık değerine sahip yöntemlerden biri olduğu [120] ve 10-kat çapraz doğrulamada tahmin hatası hesaplamalarının neredeyse tamamen yansız olduğu [121] belirtilmiştir.

Bu çalışmada sınıflandırma sonuçlarının güvenilirliğini artırmak için 10-kat çapraz doğrulama kullanılmış, her bir aşamada doğrulama için 300 görüntüden oluşan eğitim setinin %90'ı eğitim ve %10'u doğrulama olmak üzere rasgele şekilde ayrılmıştır.

Ayrıca, eğitimde kullanılacak veri sayısının daha fazla olması durumunda eğitim başarısının nasıl etkileneceğini gözlemlemek ve eğitilen ağın genel başarısının düşmesine sebep olan sınıflardaki etkisini incelemek için eğitim ve test setlerinin oranları değiştirilerek veri seti 350 eğitim ve 50 test olacak şekilde rasgele ayrılmıştır ve aynı süreçler uygulanmıştır.

3.9.2. Eğitimde Kullanılan Hiperparametreler

Çok katmanlı ağ tasarımlarında modelin veriden en etkin şekilde öğrenebilmesini sağlamak amacıyla modele uygun şekilde belirlenmesi gereken model parametreleri bulunmaktadır. Bunlar ağ yapısında kullanılan katman sayısı, filtre boyutu, filtre sayısı, öğrenme hızı, aktivasyon fonksiyonu, mini yığın boyutu ve eğitim turu sayısı gibi model başarısını etkileyen ve hiperparametre olarak adlandırılan parametrelerdir [122] - [126].

3.9.2.1. *Filtre Boyutu*

CNN mimarisinde girdi katmanındaki görüntü genişlik (w) ve yükseklik (h) olarak daha küçük parçalara bölünebilir özellikte olmalıdır. Sinir ağının nesneyi tanıması için piksel sayısının fazla olması gereken durumlarda büyük filtreler kullanılabilir. Nesnelerin birbirinden ayıran küçük detaylarsa daha küçük filtreler tercih edilebilir. Genellikle 32, 64, 224 veya 512 gibi değerler kullanılmaktadır. CNN görüntüde nesne tanıma için denetimli bir öğrenme yöntemidir. Eğitim için etiketli verilere ihtiyaç duyulur. Evrişim filtrenin ağırlıklarını optimize etmeye, hatayı en aza indirmek için filtreleri şekillendirmeye imkan verir. Eğitimde kullanılan filtreler rasgele bir değerle başlatılabilir. Genel olarak, evrişim katmanındaki filtre boyutu 3x3 veya 5x5 şeklinde tercih edilmektedir. Bu noktada evrişim katmanının sahip olduğu en önemli özellik girdi verisi olan görüntünün gerçek uzamsal boyutunu değiştirmemesidir [127], [128].

3.9.2.2. *Öğrenme Hızı (Initial Learn Rate)*

Parametrelerin güncellenmesi geri yayılım işlemi ile yapılır. Modelin her geri yayılım sonucunda ağırlık değerleri güncellenme hızı, öğrenme hızı olarak tanımlanır. Geriye doğru yayılım sonucunda türev alınarak fark bulunur, fark öğrenme hızı parametre değeri ile çarpılır, elde edilen değer ağırlık değerlerinden çıkarılır, böylece yeni ağırlık değeri bulunur. Öğrenme hızı, modeldeki mevcut ağırlık değerlerine kayıp eğiminin ne oranda uygulanacağının belirlendiği parametredir. Öğrenme hızı sabit bir değer olarak ayarlanabilir. Adım adım artan bir değer verilebileceği gibi momentum değerine göre adaptif algoritmalar tarafından da belirlenebilir [123], [129]. Öğrenme hızının başlangıçta

düşük bir değer seçilmesi optimum eğim noktasına gidiş süresinin uzamasına neden olurken yüksek bir değer seçilmesi ise yeni ağırlık değerlerinin daha hızlı bir şekilde değişmesine neden olur. Başlangıçtaki öğrenme hızı için derin ağ modellerinde genellikle 0,1 ile 0,001 arasında bir değer seçilmektedir [130].

3.9.2.3. *Optimizasyon Algoritmaları*

Derin öğrenmede çok katmanlı ağ mimarisi ile temel amaç eğitilen modelin iyi bir genelleme yapabilmesi ve ardından doğrulama verisi ve test seti verilerine uygulandığında başarısının değerlendirilmesidir. Eğitim sürecinde olası hataları en aza indirebilmek amacıyla optimizasyon algoritmaları kullanılmaktadır. Uygun değerlerin tespiti için doğrusal olmayan problemlerin çözümünde optimizasyon yöntemleri kullanılır. Derin öğrenme uygulamalarında genel olarak momentumlu stokastik gradyan inişi (stochastic gradient descent), Adam, AdaGrad, RMSprop, Adadelta, Adamax gibi algoritmalar kullanılmaktadır [123], [131]. Bu algoritmalar arasında hız ve başarı farklılıkları bulunmaktadır. Mini yığın boyutu, momentum ve beta seçimi de optimizasyon algoritmalarıyla ilgili hiperparametrelerdir [124]. Optimizasyon algoritması, ağda her bir ileri yayılım ve geri yayılımda daha iyi sonuç üretebilecek olan parametre değerleri arar. Öğrenme hızının da etkili olduğu optimizasyon süreci algoritmanın tekrarlı şekilde devam ettiği aşama aşama ilerleyen bir arama sürecidir [131].

3.9.2.4. *Mini Yığın Boyutu (Mini Batch Size)*

Her öğrenme iterasyonunda geriye dönük olarak hesaplama yapılır ve ağırlıklar güncellenir. Bu süreçte tüm eğitim veri seti üzerinde çalışmak hem zaman alıcı hem de hafıza açısından maliyetli olacağından, belirli boyutlardaki veri setleri üzerinde çalışılarak parametre güncellemeleri daha hızlı yapılabilir. Veri seti küçük gruplara ayrılarak öğrenme işlemi bu gruplar üzerinde yapılır. Veri setinin parçalar halinde işlenmesi mini yığın olarak adlandırılır. Kullanılan her eğitim örneği için ağ parametrelerini güncellemek zaman alır. Belli sayıda güncelleme yapmak süre maliyetinden tasarruf sağlar. Mini yığın boyutu parametre güncellemesi gerçekleşen ağa verilen örnek sayısıdır. Modelin aynı anda kaç veriyi işleyeceği mini yığın boyutu ile belirlenebilir. Mini yığın boyutu 1 ile tüm veri setinin sayısı aralığında bir değer olarak belirlenebilir. Eğer 1 olarak belirlenirse her iterasyonda sadece bir adet veri üzerinde işlem yapılacaktır. CNN yığın değerine karşı hassas olduğundan yığın değerinde yapılacak küçük bir değişiklik bile performansa büyük etki yapabilir [132] - [134].

3.9.2.5. *Eğitim Turu (Epoch) Sayısı*

Modelin eğitim sürecinde tüm veriler aynı anda eğitime dahil olmaz. Bunun yerine belirli sayıda veriden oluşan parçalar halinde eğitime katılırlar. İlk parça eğitildikten sonra model başarısı test edilir ve elde edilen başarıya göre geri yayılım ile ağırlık değerleri güncellenir. Ardından yeni bir parça ile model tekrar eğitilir ve aynı şekilde tekrar ağırlık değerleri güncellenir. Model için en uygun olan ağırlık değerleri hesaplanana kadar bu şekilde adım adım işleme devam edilir. Modelin bütün eğitim veri seti ile doğrulama veri seti kullanılarak bir defa ileri yayılım ve geriye yayılım yapmasına bir eğitim turu adı verilmektedir. Eğitim setindeki veriler kullanılarak ileri yayılım yapan model elde edilen hataları geri yayılım ile göndürdüğünde eğitim turu tamamlanır. Modelde mümkün olan en yüksek başarının elde edilebilmesi için modelin belirli bir eğitim turu sayısı ile eğitilmesi gerekir. Eğitim turu sayısı, bir ile sonsuz arasında bir tamsayı değeri olarak ayarlanabilir. Ağırlıklar adım adım hesaplandığından ilk eğitim turlarında modelin başarısı düşükken eğitim turu sayısı arttıkça modelin başarısı da belirgin olarak artacaktır. Ancak belirli bir eğitim turu sayısına ulaştıktan sonra modelin öğrenmesine katkısı daha az olacak ve modelin başarısı daha küçük miktarlarda etkilenecektir [132] - [134].

3.9.3. Transfer Öğrenme (Transfer Learning)

Genel manada öğrenmenin aktarılması, öğrenilmiş olan bir bilgi, beceri veya stratejinin yeni karşılaşılan bir duruma uygulanması veya yeni bir probleme çözüm üretirken önceki bilgilerden yararlanılması şeklinde ifade edilebilir. Bu noktada aktarım işlemi, farklı veya yeni bir aktivite değil, öğrenme sürecinin bir bileşenidir [135]. Derin öğrenme alanında da eğitim alanındaki öğrenme aktarımının bu genel anlamına benzer şekilde, daha önceden eğitilmiş olan bir ağdan bilgi aktarımının yapılarak yeni uygulamada kullanılması transfer öğrenme olarak tanımlanmaktadır. Transfer öğrenme, daha önce bir veri setiyle eğitilmiş olan bir ağdaki öğrenilmiş ağırlıkların farklı başka bir veri seti için kullanılmasıdır [136], [137]. Önceden eğitilmiş olan bir ağ, yeni bir görevi öğrenmek için bir başlangıç aşaması olarak kullanılabilir. Transfer öğrenme genellikle önceden eğitilmiş olan modelin eğitiminde kullanılmış olan orijinal veri setinden daha küçük olan bir yeni veri seti olduğunda uygulanır [138], [139]. Transfer öğrenme ile bir ağda ince ayar yapmak, bir ağı rastgele ağırlıklarla başlatarak sıfırdan eğitmekten çok daha hızlı ve kolaydır. Daha az sayıda eğitim veri seti kullanılarak daha önceki öğrenilen özellikler yeni bir görev için hızla yeni ağa aktarılabilir [140]. Geleneksel öğrenme modelleri, tüm

hesaplama adımlarının baştan yapıldığı ve iyi bir başarı performansı elde edebilmek için büyük miktarda veri gerektiren ve uzun süren bir sıfırdan eğitim süreci gerektirirken transfer öğrenme hesaplama süreci açısından daha verimli olan ve küçük bir veri seti kullanarak da iyi sonuçlar elde edilmesine imkan veren bir yöntemdir. Geleneksel öğrenmede modelin önceki bilgilere bağlı olmadan bağımsız olarak belirli bir amaç için baştan eğitildiği izole bir eğitim sözkonusuyken transfer öğrenmede yeni model daha öncekine benzer olan yeni bir farklı görevi yapmak için önceden eğitilmiş olan modelde elde edilen bilgiyi kullanır. Burada önemli olan, bu yeni farklı görevin tamamen ilgisiz bir görev olmaması, önceki modelde öğrenilen bilgiler ile ilişkili özellikler içermesidir. Örneğin araba tanıma için eğitilmiş bir modelin, yine taşıt tanıma için kullanılması gibi. Transfer öğrenme modelleri ile geleneksel öğrenme modellerinden daha hızlı bir şekilde optimum başarı performansı elde edilebilir. Bunun nedeni, önceden eğitilmiş modellerden elde edilen özellikler, ağırlıklar, gibi bilgilerden yararlanan modelin, özellikleri zaten hazır olarak öğrenmiş olmasıdır. Böylece transfer öğrenme sıfırdan eğitim yapılmasına kıyasla sinir ağlarının daha hızlı olmasını ve daha başarılı sonuçlar elde edilmesine imkan sağlamaktadır [138], [141] - [143].

Transfer öğrenme yaklaşımı, derin öğrenme modelleri arasında çoğunlukla nesne tespiti, görüntü ve ses tanıma uygulamaları için yaygın olarak kullanılır. Transfer öğrenmenin temelinde daha önce oluşturulmuş ve binlerce, hatta milyonlarca örnek eğitim görüntüsü ile eğitilmiş derin öğrenme modelleri bulunur. Önceden eğitilmiş birçok model mevcuttur ve yeni oluşturulan model için seçilmek istendiğinde her birinin dikkate alınması gereken yeni model için avantaj veya dezavantaj oluşturabilecek farklı özellikleri vardır. Bunlar modelin boyutu, doğruluğu ve hızıdır. Yeni modelin çalıştırılacağı donanım, ortam ve bellek özellikleri açısından seçilecek önceden eğitilmiş modelin boyutu önemli bir özellik olacaktır. Yine önceden eğitilmiş modelin ne oranda başarılığı sınıflama yaptığı da başka bir önemli özelliktir. Modelin düşük başarı performansı olması daha sonraki görevlerde de performansının düşük olacağı anlamına gelmese de örneğin bir milyon görüntü ve bin görüntü sınıfı içeren ve yaygın olarak kullanılan bir veri seti olan ImageNet için iyi bir başarı gösteren bir model, muhtemelen yeni, benzer görevlerde de iyi bir performans gösterecektir. Modelin hızı, yeni görev için kullanılan donanım, mini yığın boyutu vb. gibi özelliklerden etkilenmekle birlikte seçilecek önceden eğitilmiş modelin mimarisi ve boyutu gibi özellikleri de hız üzerinde etkili olacaktır. Başlangıç için yaygın olarak kullanılan mimariler, AlexNet, GoogLeNet, VGG-16 ve VGG-19 gibi modellerdir.

Transfer öğrenme akış sürecinde, ilk olarak önceden eğitilmiş bir model seçilir. Yeni bir görüntü veri seti ve sınıf bilgisi ile sınıflandırma yapmak üzere ağı yeniden eğitmek için bu mimarideki son katmanlar değiştirilir. Yeni görevdeki sınıf sayısı ile aynı sayıda çıktı oluşturmak üzere son tam bağlı katman ve softmax katmanı tarafından sınıf olasılıklarının hesaplandığı çıktıları vermek üzere sınıflandırma katmanı değiştirilir. Katmanlar değiştirildikten sonra, son tam bağlı katman ağın öğreneceği yeni sınıf sayısını belirleyecek ve sınıflandırma katmanı da bu yeni çıktı kategorilerinden çıktıları belirleyecektir. Alternatif olarak, özellikle yeni görevdeki görüntülerin özellikleri ile önceden eğitilmiş modelde öğrenilen özelliklerin farklılıklarının fazla olduğu durumlarda, sadece son katmanlar değiştirilmekle kalmayıp önceki katmanların bazılarında da yeni veri seti kullanılarak geri yayılım ile ağırlıkların güncellenmesi söz konusudur. Bir ağ yapısında, ilk katmanlar kullanılan veri seti ile ilgili daha genel ve temel özelliklerin çıkarıldığı ve öğrenildiği katmanlar olup daha derinlere gidildikçe gelen sıradaki katmanlar ise daha detay özelliklerin çıkarıldığı ve öğrenildiği katmanlardır. Yeni veri setinde özelliklerin farklı olduğu durumlarda, bu daha derindeki katmanların da yeni veri seti ile eğitim sürecine dahil edilmesi ve ağırlıkların güncellemesi yapılmaktadır. Bu şekilde bir değişiklik yapıldığında, ağırlıkları değiştirilmeyecek olan baştaki katmanlar için dondurma (freeze) işlemi söz konusu olmaktadır. Bu işlemdeki erken aşamadaki katmanlardaki öğrenme hızı sıfır olarak ayarlanıp bu katmanlardaki ağırlıklar güncellenmez, yani dondurulmuş olur. Eğitim aşamasında, dondurulmuş katmanlardaki parametreler güncellenmeyeceğinden dondurma işlemi eğitim süresinin daha hızlı olmasını sağlayacaktır. Özellikle yeni veri seti küçük olduğunda, ağırlıkların dondurulması, ağın yeni veri setine aşırı uyum sorununu da engelleyecektir. Bu gibi yeni veri setindeki özelliklerin önceden öğrenilmiş özelliklere göre biraz daha farklı olması gibi durumlar nedeniyle opsiyonel olarak ağırlıkları dondurma (freeze) işlemi uygulanabilmektedir. Katmanlarla ilgili bu işlemler uygun şekilde tamamlandıktan sonra, model yeni veri seti ile yeniden eğitilir. Bu yeni eğitim ile model yeni görüntüler ve bu görüntüdeki sınıflarla ilgili özellikleri öğrenmek için ağı günceller. Bu yeni eğitim süreci, çoğunlukla bir modeli sıfırdan eğitmeye kıyasla daha az veriye ihtiyaç duyar. Eğitim süreci de tamamlandıktan sonra ağın doğruluğunu ve başarı performansını incelemek üzere yeni görüntülerle test işlemi yapılır. Böylece transfer öğrenmenin kullanılması ile daha kısa sürede daha yüksek model doğruluğu elde etmek mümkün olur [139], [141], [144] - [146].

3.9.4. Eğitim

Bu çalışmadaki yüz tanıma sürecinde R-CNN, faster R-CNN ve YOLOv4 modelleri eğitilmiş ve test edilmiştir. AlexNet, GoogleNet ve ResNet50 mimarileri ile oluşturulan R-CNN modellerinde ve ResNet50 mimarisi ile oluşturulan faster R-CNN ve YOLOv4 modellerinde elde edilen sonuçlar karşılaştırılmıştır.

R-CNN ile AlexNet, GoogleNet ve ResNet50 ağlarında eğitim için kullanılan hiperparametreler Çizelge 3.3'te gösterilmiştir. Tüm ağlarda filtre sayısı 32 ve eğitim turu sayısı 10 olarak dikkate alınmıştır. AlexNet ve GoogleNet ağlarında mini yığın boyutu 16 olarak alınırken ResNet50 ağında ise ağ yapısındaki katman sayısının çokluğu nedeniyle daha fazla ekran kartı hafızası gerektireceğinden mini yığın boyutu 4 şeklinde ayarlanmıştır. imageInputLayer parametresinde boyut değeri resimlerin boyutuna göre 420*236 şeklinde değiştirilmiştir. On iki öğrenci olduğundan fullyconnectedLayer parametresi 12 farklı etiketi temsil etmektedir.

Eğitim aşamasında kullanılan trainRCNNObjectDetector hiperparametrelerinden öğrenme hızı tüm ağlarda 1e-3 şeklinde ayarlanmıştır.

İlk uygulama Şekil 3.12'de verilen AlexNet sinir ağı mimarisi ile yürütülmüştür. Aktivasyon fonksiyonu olarak ReLU fonksiyonu kullanılmıştır. Çalışmadaki AlexNet mimarisinde kullanılan katmanlar ve bu katmanların özellikleri Çizelge 3.4'te gösterilmiştir.

Çizelge 3.3. R-CNN Object Detector eğitim seçenekleri.

Ağ	**AlexNet**	**GoogleNet**	**ResNet50**
Mini Yığın Boyutu	16	16	4
Eğitim Turu Sayısı	10	10	10
Öğrenme Hızı	1e-3	1e-3	1e-3
Optimizasyon Algoritması	SGDM	Adam	SGDM
Eğitim Süresi	40'	90'	342'

Çizelge 3.4. AlexNet mimarisinde kullanılan katmanlar.

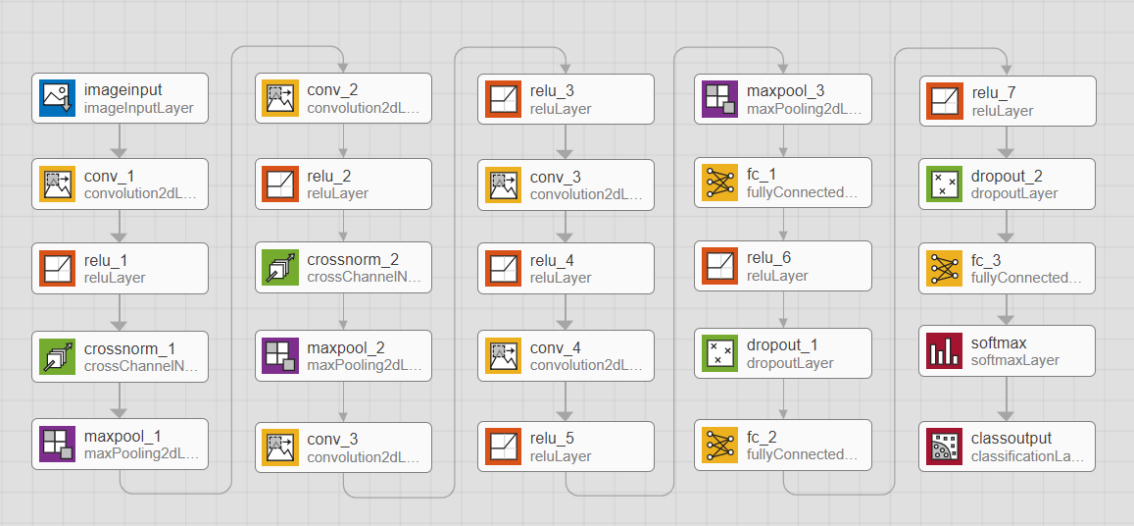
Katmanlar ve Özellikleri	**Açıklamalar**
[Input Layer] imageInputLayer([420 236 3]	Giriş katmanı boyutu 420x236 olan renkli görüntüler alır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [11 11] ve Filtre Sayısı [96] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] crossChannelNormalizationLayer(5,"K",1)	Kanallar arası normalizasyon için kullanılır. Pencere boyutu 5 K hiperparametre 1 olarak ayarlanmıştır.
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [5,5] ve Filtre Sayısı [128] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] crossChannelNormalizationLayer(5,"K",1)	Kanallar arası normalizasyon için kullanılır. Pencere boyutu 5 K hiperparametre 1 olarak ayarlanmıştır.
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [3,3] ve Filtre Sayısı [384] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman

Çizelge 3.4. AlexNet mimarisinde kullanılan katmanlar (devamı).

Katmanlar ve Özellikleri	**Açıklamalar**
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [3,3] ve Filtre Sayısı [192] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [3,3] ve Filtre Sayısı [128] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.
[FINAL LAYERS] fullyConnectedLayer(12)	Sinir ağında kaç nesne sınıflandırılacaksa o değer yazılır. 12 farklı kişi olduğu için 12 olarak ayarlandı.
[FINAL LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[FINAL LAYERS] dropout	Verilen girdi öğelerini rasgele sıfıra ayarlar.
[FINAL LAYERS] fullyConnectedLayer(12)	Sinir ağında kaç nesne sınıflandırılacaksa o değer yazılır. 12 farklı kişi olduğu için 12 olarak ayarlandı.
[FINAL LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[FINAL LAYERS] dropout	Verilen girdi öğelerini rasgele sıfıra ayarlar.

Çizelge 3.4. AlexNet mimarisinde kullanılan katmanlar (devamı).

Katmanlar ve Özellikleri	**Açıklamalar**
[FINAL LAYERS] rcnnfullyConnectedLayer(12)	Sinir ağında kaç nesne sınıflandırılacaksa o değer yazılır. 12 farklı kişi olduğu için 12 olarak ayarlandı.
[FINAL LAYERS] rcnnSoftMax()	Softmax fonksiyonunun uygulandığı katman
[FINAL LAYERS] rcnnClassificationLayers()	Sınıflandırma işleminin yapıldığı katmandır.



Şekil 3.12. Çalışmada kullanılan AlexNet mimarisi.

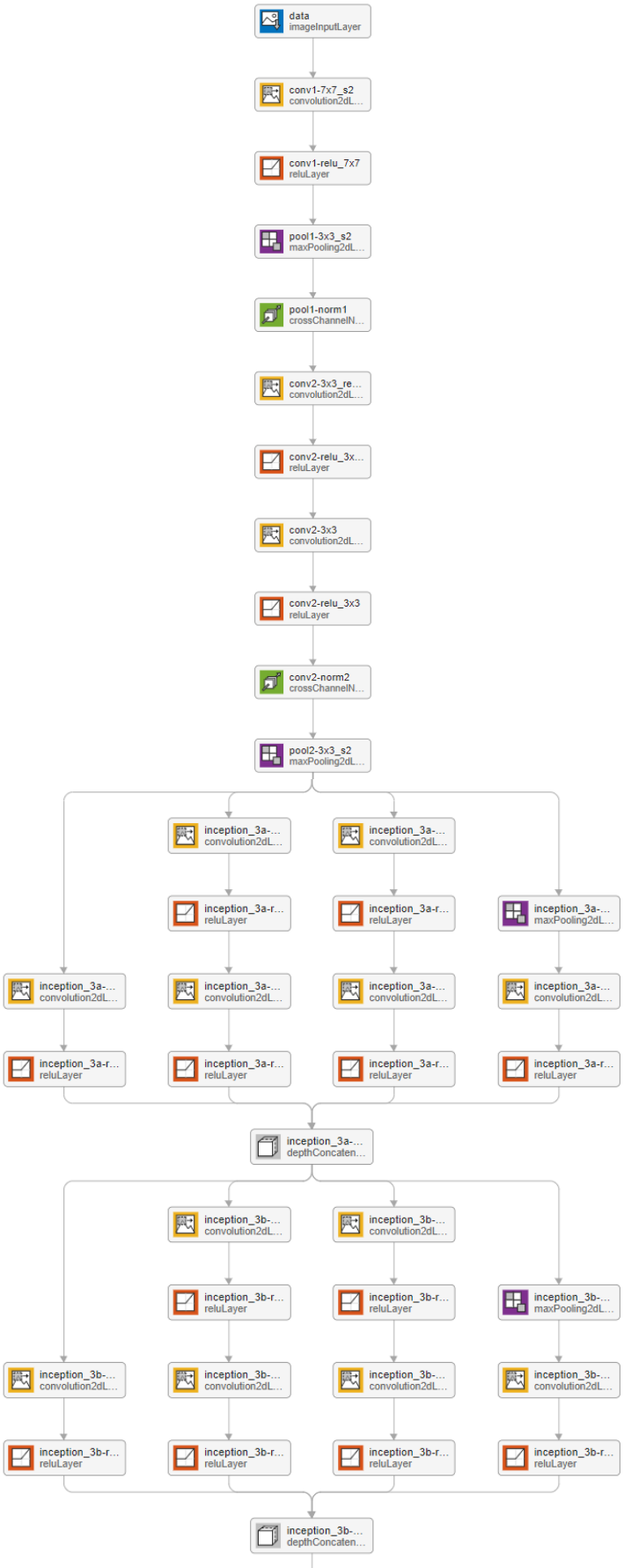
Daha sonra GoogleNet ve ResNet50 mimarilerinin kullanıldığı R-CNN uygulamaları da yapılmıştır. GoogleNet ile yürütülen uygulamanın sinir ağı mimarisi Şekil 3.13'te ve ResNet50 ile yürütülen uygulamanın sinir ağı mimarisi ise Şekil 3.14'te gösterilmiştir. Çalışmada kullanılan katmanlar ve bu katmanların özellikleri GoogleNet mimarisi için Çizelge 3.5'te ve ResNet50 mimarisi için Çizelge 3.6'da verilmiştir.

Çizelge 3.5. GoogleNet mimarisinde kullanılan katmanlar.

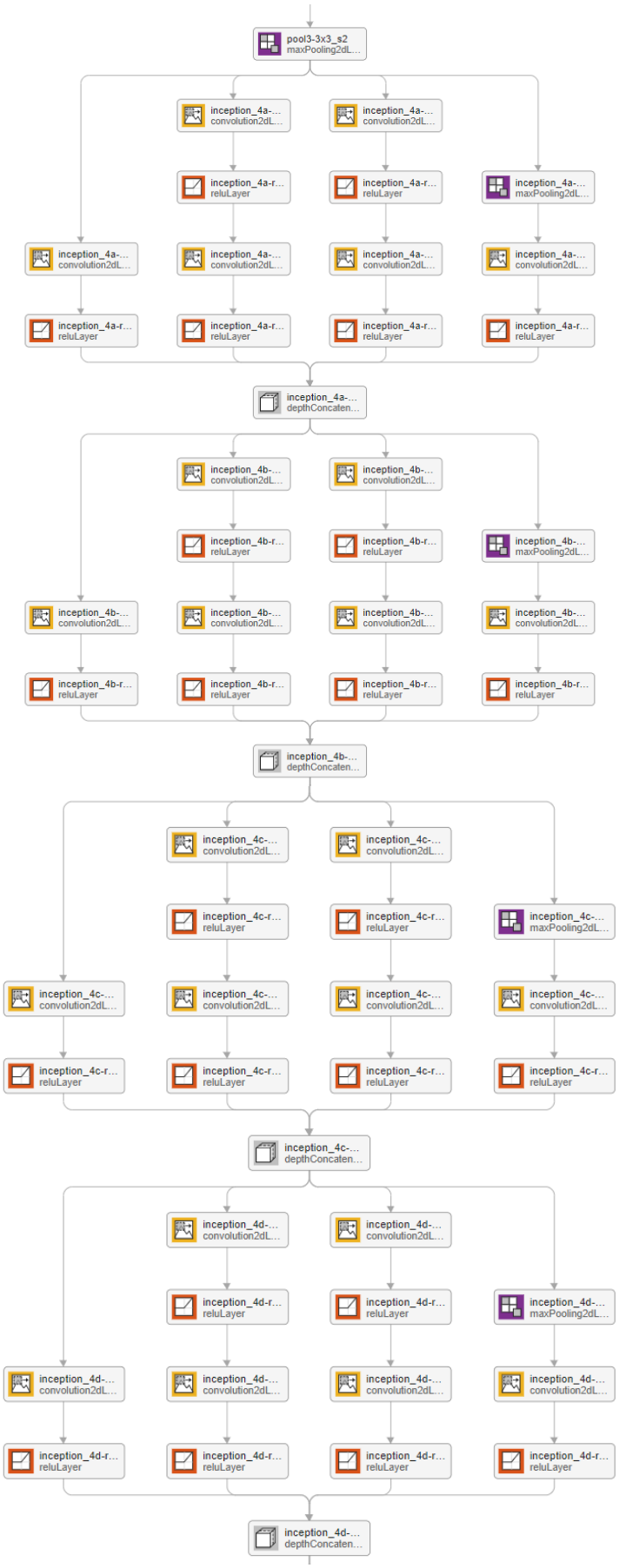
Katmanlar ve Özellikleri	**Açıklamalar**
[Input Layer] imageInputLayer([420 236 3]	Giriş katmanı boyutu 420x236 olan renkli görüntüler alır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [7,7] ve Filtre Sayısı [64] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.
[MIDDLE LAYERS] crossChannelNormalization	Kanal bazında normalleştirmeyi gerçekleştirir.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [1,1] ve Filtre Sayısı [64] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [3,3] ve Filtre Sayısı [192] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] crossChannelNormalization	Kanal bazında normalleştirmeyi gerçekleştirir.
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.

Çizelge 3.5. GoogleNet mimarisinde kullanılan katmanlar (devamı).

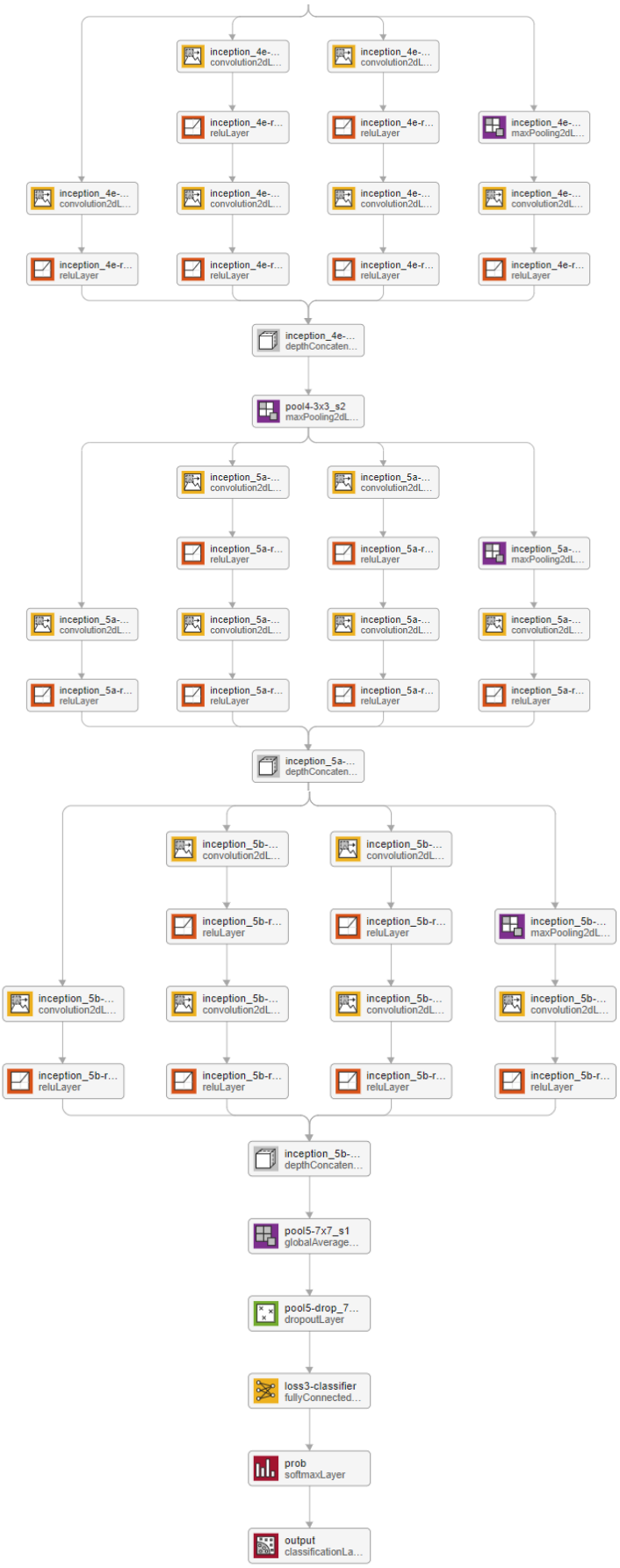
Katmanlar ve Özellikleri	**Açıklamalar**
. . .	. . .
[FINAL LAYERS] rcnnfullyConnectedLayer(12)	Sinir ağında kaç nesne sınıflandırılacaksa o değer yazılır. 12 farklı kişi olduğu için 12 olarak ayarlandı.
[FINAL LAYERS] rcnnSoftMax()	Softmax fonksiyonunun uygulandığı katman
[FINAL LAYERS] rcnnClassificationLayers()	Sınıflandırma işleminin yapıldığı katmandır.



Şekil 3.13. Çalışmada kullanılan GoogleNet mimarisi.



Şekil 3.13. Çalışmada kullanılan GoogleNet mimarisi (devamı).



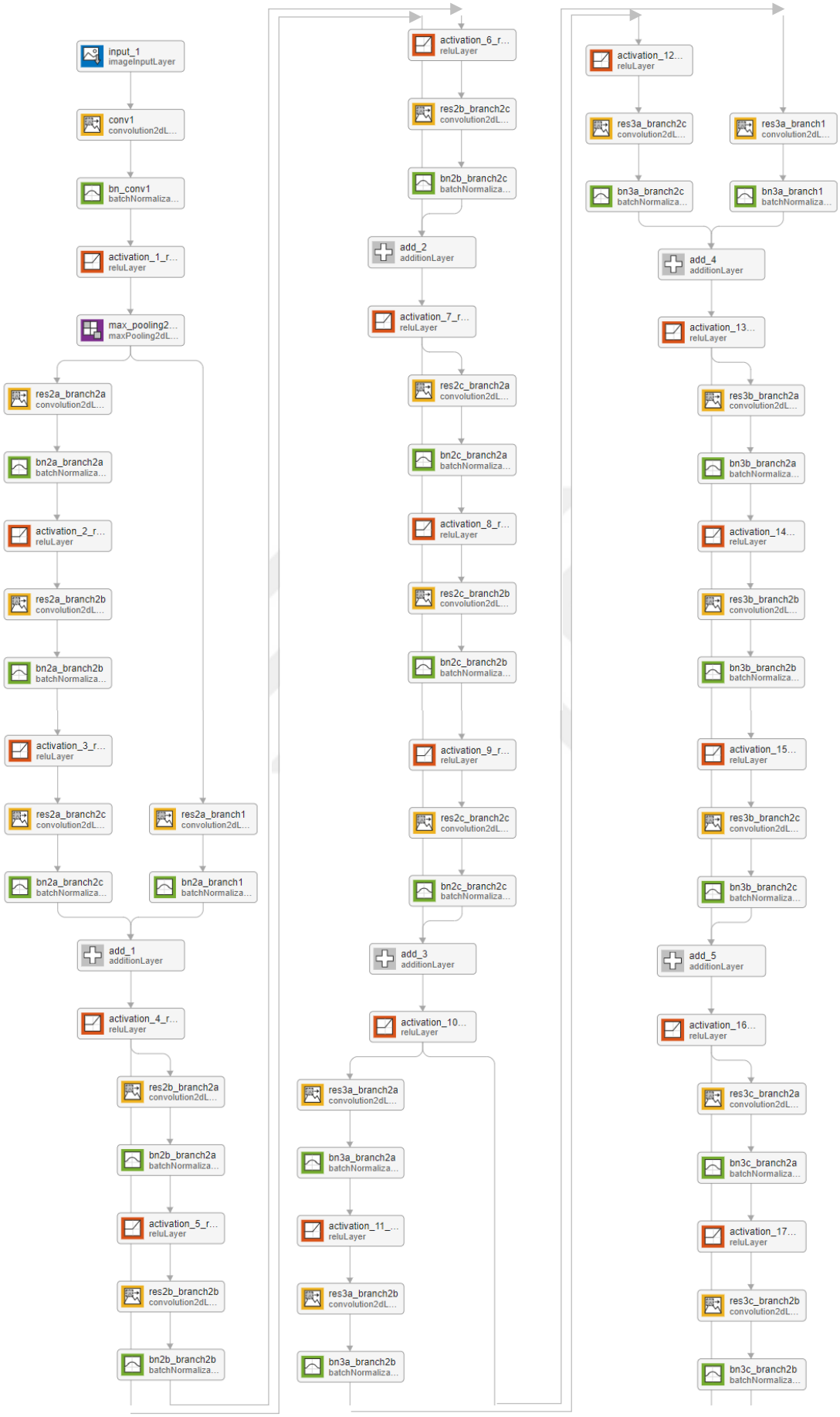
Şekil 3.13. Çalışmada kullanılan GoogleNet mimarisi (devamı).

Çizelge 3.6. ResNet50 mimarisinde kullanılan katmanlar.

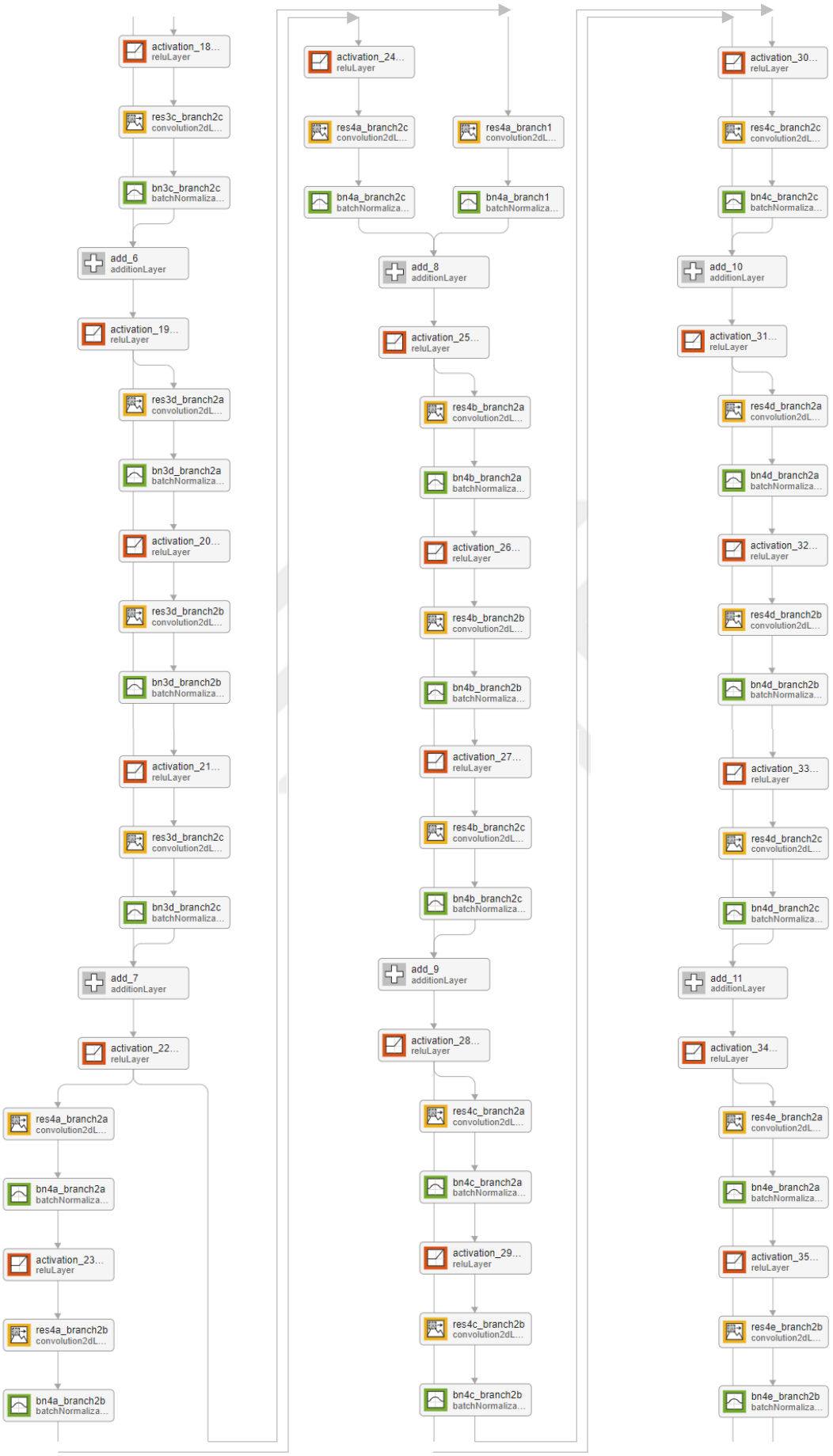
Katmanlar ve Özellikleri	**Açıklamalar**
[Input Layer] imageInputLayer([420 236 3]	Giriş katmanı boyutu 420x236 olan renkli görüntüler alır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [7 7] ve Filtre Sayısı [64] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] batchNormalizationLayer	Her kanal için, bağımsız olarak tüm gözlemlerde mini bir veri grubunu normalleştirir.
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] maxPooling2dLayer(3,'Stride',2)	Maksimum havuzlama katmanı oluşturur. Adım değeri 2 olarak ayarlanmıştır.
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [1 1] ve Filtre Sayısı [64] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] batchNormalizationLayer	Her kanal için, bağımsız olarak tüm gözlemlerde mini bir veri grubunu normalleştirir.
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [3 3] ve Filtre Sayısı [64] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] batchNormalizationLayer	Her kanal için, bağımsız olarak tüm gözlemlerde mini bir veri grubunu normalleştirir.
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman

Çizelge 3.6. ResNet50 mimarisinde kullanılan katmanlar (devamı).

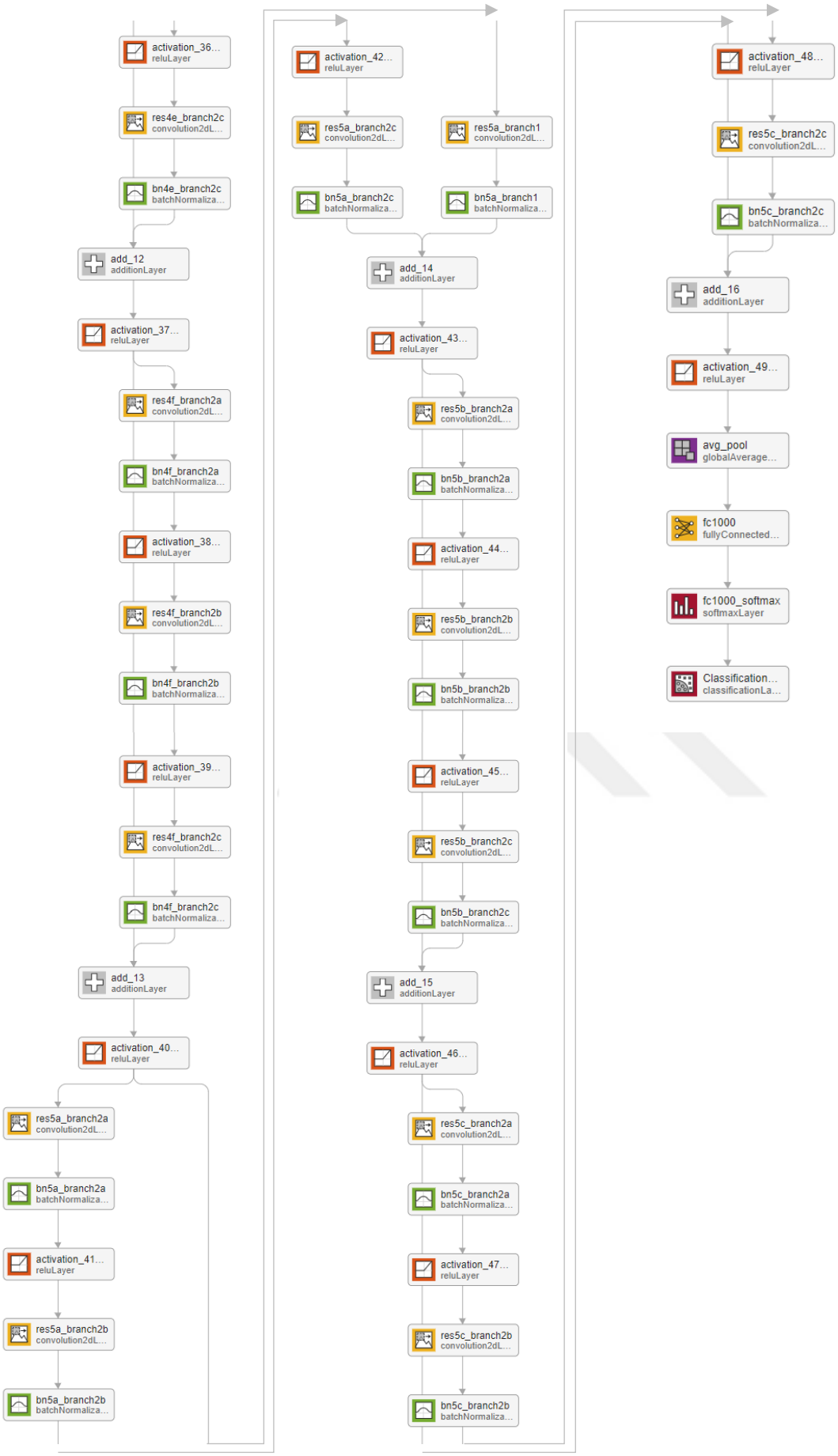
Katmanlar ve Özellikleri	**Açıklamalar**
[MIDDLE LAYERS] conv2dLayer(FilterSize,NumFilter,'Padding,2')	Filtre Boyutu [1 1] ve Filtre Sayısı [256] parametreleri ile padding işlemini gerçekleştiren evrişim katmanı
[MIDDLE LAYERS] batchNormalizationLayer	Her kanal için, bağımsız olarak tüm gözlemlerde mini bir veri grubunu normalleştirir.
[MIDDLE LAYERS] additionLayer()	Birden çok sinir ağı katmanından girdiler ekler.
[MIDDLE LAYERS] reluLayer()	ReLu aktivasyon fonksiyonunun uygulandığı katman
. . .	. . .
[FINAL LAYERS] rcnnfullyConnectedLayer(12)	Sinir ağında kaç nesne sınıflandırılacaksa o değer yazılır. 12 farklı kişi olduğu için 12 olarak ayarlandı.
[FINAL LAYERS] rcnnSoftMax()	Softmax fonksiyonunun uygulandığı katman
[FINAL LAYERS] rcnnClassificationLayers()	Sınıflandırma işleminin yapıldığı katmandır.



Şekil 3.14. Çalışmada kullanılan ResNet50 mimarisi.



Şekil 3.14. Çalışmada kullanılan ResNet50 mimarisi (devamı).



Şekil 3.14. Çalışmada kullanılan ResNet50 mimarisi (devamı).

Eğitim aşaması, AlexNet ile 40 dakika, GoogleNet ile 1 saat 30 dakika ve ResNet50 ile 5 saat 42 dakika sürmüştür. Veri setinin 350 eğitim ve 50 test olarak ayrıldığı uygulamada ise mini yığın boyutu 32, eğitim turu sayısı 10 ve öğrenme hızı 1e-3 şeklinde alınarak AlexNet ile eğitim aşaması 1 saat 35 dakika sürmüştür.

Bu uygulamalardan GoogleNet mimarisinin kullanıldığı R-CNN uygulaması eğitim hiperparametrelerinde uyarlanabilir moment tahmini (adaptive moment estimation, Adam) algoritması yerine momentumlu stokastik gradyan inişi (stochastic gradient descent with momentum, SGDM) algoritması kullanılarak tekrar test edilmiştir.

AlexNet mimarisinde doğruluk oranının etkilenip etkilenmeyeceğini incelemek için ImageAugmenter işlemi ile arttırma uygulanarak ve ayrıca hiperparametrelerde farklı eğitim turu sayısı ve mini yığın boyutu değerleri kullanılarak dört farklı uygulamada test edilmiştir. Bu uygulamalardan birincisinde eğitim turu sayısı 10 ve mini yığın boyutu 25 şeklinde; ikincisinde eğitim turu sayısı 10 ve mini yığın boyutu 32 şeklinde; üçüncüsünde eğitim turu sayısı 12 ve mini yığın boyutu 32 şeklinde ve dördüncüsünde eğitim turu sayısı 15 ve mini yığın boyutu 32 şeklinde ayarlanmıştır. Arttırma işlemi ayna görüntüleri oluşturma ve pixel değerlerinde öteleme kullanarak gerçekleştirilmiştir.

Daha sonra ResNet50 mimarisi kullanılarak faster R-CNN ile yeni bir uygulama daha yapılmıştır. Bu ağda eğitim için kullanılan hiperparametreler Çizelge 3.7'de gösterilmiştir. Bağlantı (anchor) sayısı 7, mini yığın boyutu 1 ve eğitim turu sayısı 10 olarak dikkate alınmıştır. Öğrenme hızı bu eğitimde de 1e-3 şeklindedir. ResNet50 mimarisi kullanılarak yapılan faster R-CNN uygulamasında eğitim aşaması 4 saat 10 dakika sürmüştür.

Çizelge 3.7. Faster R-CNN Object Detector eğitim seçenekleri.

Ağ	**ResNet50**
Bağlantı (Anchor) sayısı	7
Mini Yığın Boyutu	1
Eğitim Tur Sayısı	10
Öğrenme Hızı	1e-3
Optimizasyon Algoritması	SGDM
Eğitim Süresi	250'

Son olarak yine ResNet50 mimarisi kullanılarak YOLOv4 ile veri setinin 300 eğitim ve 100 test, 350 eğitim ve 50 test olmak üzere ayrıldığı iki uygulama daha yapılmıştır. Bu ağda eğitim için kullanılan hiperparametreler Çizelge 3.8'de gösterilmiştir. Mini yığın boyutu 16 ve eğitim turu sayısı 40 olarak dikkate alınmıştır. Öğrenme hızı bu eğitimde 1e-4 şeklindedir. ResNet50 mimarisi kullanılarak yapılan YOLOv4 uygulamasında eğitim aşaması 40 dakika sürmüştür.

Çizelge 3.8. YOLOv4 Object Detector eğitim seçenekleri.

Ağ	**ResNet50**
Mini Yığın Boyutu	16
Eğitim Tur Sayısı	40
Öğrenme Hızı	1e-4
Optimizasyon Algoritması	SGDM
Eğitim Süresi	40'

3.9.5. Test

Doğru sınıflamayı tespit edebilmek için, test için ayrılan 100 resim etiketlenerek farklı bir gTruth nesnesi hazırlanmıştır. Tüm uygulamalarda eğitilmiş olan R-CNN detector nesnesi test için ayrılan 100 resim ile test edilmiştir. Eğitim için 350 resmin kullanıldığı uygulamada test işlemi 50 resim ile yapılmıştır. Testteki eşik değer 0,7 olarak ayarlanıp bu eşiğe ulaşan sınıflar için sınırlama kutuları ve etiketleri gösterilmiştir. Ayrıca mini yığın boyutu 1 olarak alınmıştır. Sonuçların değerlendirilmesi için kullanılacak değerlerin elde edilmesi için evaluateDetectionPrecision fonksiyonu kullanılmıştır. Böylece 12 öğrenci için test görüntülerindeki doğru sınıflama oranları tespit edilmiştir.

3.10. SONUÇLARIN DEĞERLENDİRİLMESİ

Derin öğrenme alanında yaygın olarak kullanılan performans değerlendirme ölçütleri doğruluk (accuracy), kesinlik (precision), hatırlama (recall) değerleri ile F1 skoru

ölçütleridir [147]. Bu ölçütlerin hesaplanabilmesi için test sonucu doğru pozitif (DP) ve doğru negatif (DN) ile yanlış pozitif (YP) ve yanlış negatif (YN) sayılarının elde edilmesi gerekmektedir. Bu değerler karışıklık matrisi (confusion matrix) ile belirlenebilmektedir. DP bir sınıfta yer aldığı doğru olarak tahmin edilen sayıyı gösterirken YP ise başka bir sınıfta olduğu halde bu sınıfta tahmin edilen sayıyı belirtmektedir. Benzer şekilde, DN başka bir sınıfta yer alan ve bu sınıfta olmadığı doğru olarak tahmin edilen sayıyı belirtirken YN ise bu sınıfta yer almasına rağmen başka sınıfta tahmin edilen sayıyı göstermektedir. Bu sayılar belirlenerek karışıklık matrisi elde edildikten sonra aşağıdaki formüller yardımıyla test için doğruluk (accuracy), kesinlik (precision), hatırlama (recall) ve F1 skoru hesaplanır.

$$Doğruluk\ (Accuracy) = \frac{DP + DN}{DP + DN + YP + YN}$$

$$Kesinlik\ (Precision) = \frac{DP}{DP + YP}$$

$$Hatırlama\ (Recall) = \frac{DP}{DP + YN}$$

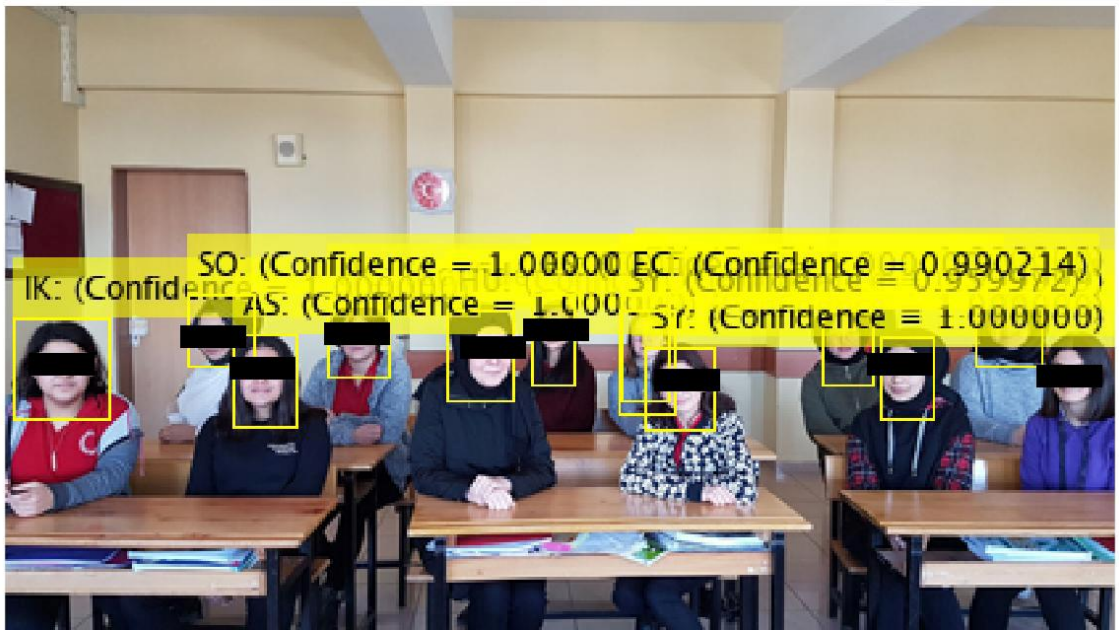
$$F1\ Skor = 2x\frac{Kesinlik\ x\ Hatırlama}{Kesinlik + Hatırlama}$$

Kesinlik (precision), doğru yapılan tespit sayısının doğru ve yanlış olarak yapılan toplam tespit sayısına oranlanmasıyla elde edilir. Yani kesinlik değeri, tahmin edilen sınırlayıcı kutuların doğru olma oranını göstermektedir ve bire yakın olması yüksek oranda doğru olduğu anlamına gelmektedir. Hatırlama (recall) ise doğru yapılan tahmin sayısının tahmin edilmesi gereken toplam sayıya oranlanmasıyla hesaplanır. Benzer şekilde, hatırlama (recall) değeri, tespit edilmesi gereken nesnelerin ne oranda doğru şekilde tespit edilebildiğinin bir ölçütüdür ve bire yaklaşması tespit edilmesi gereken nesnelerin yüksek oranda doğru şekilde tespit edilebildiği anlamına gelmektedir. Yanlış tespitlerin, yani YP

sayısının artması durumunda kesinlik (precision) azalırken, tespit edilmesi gerekip tespit edilemeyenlerin, yani YN sayısının artması durumunda da hatırlama (recall) azalmaktadır. Nesne tespiti işlemlerinde sonuçların değerlendirilmesinde doğru yapılan tespitlerin yanında yanlış yapılan tespitlerin de dikkate alınması gerekmektedir. Bu doğrultuda, ortalama kesinlik (average precision, AP) değeri önerilmiştir [148]. AP değeri hesaplanırken modelin tespitleri için kesinlik (precision) - hatırlama (recall) grafiği çizilir. Bu grafikte, dikey eksende kesinlik (precision) değerleri ve yatay eksende hatırlama (recall) değerleri yer alır. AP değeri, grafikte her bir hatırlama (recall) değerine karşılık gelen kesinlik (precision) değerlerinin ortalaması alınarak hesaplanır. Nesne tespit uygulamalarında, veri setindeki tüm sınıflar için AP değeri hesaplanarak bu değerlerin de ortalamasının alınmasıyla elde edilen ortalama kesinlik değerlerinin ortalaması (mean average precision, mAP), nesne tespit uygulaması başarısının değerlendirilmesinde standart bir değerlendirme kriteri olarak kabul edilmektedir [149].

4. BULGULAR

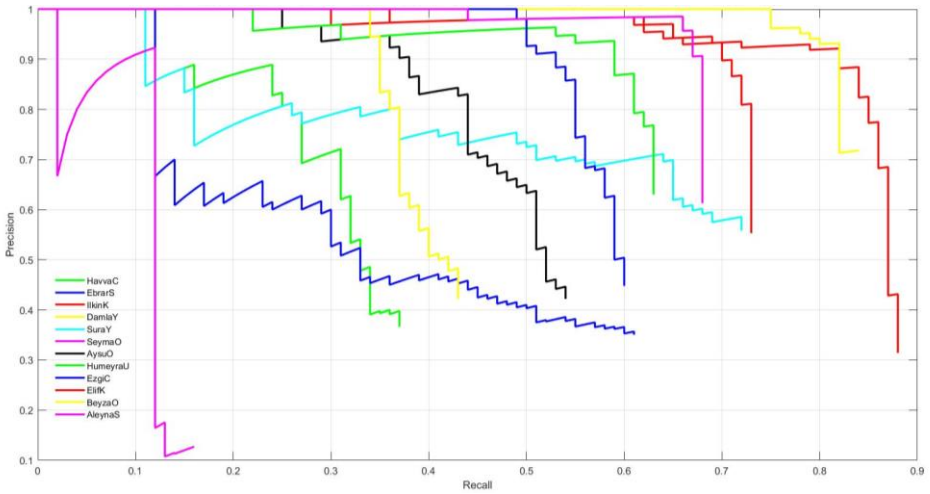
Bu çalışmada öğrencilerin fotoğraflarından yüz tanıma için R-CNN, faster R-CNN ve YOLOv4 modelleri eğitilmiş ve test edilmiştir. Öğrencilerin test setindeki görsellerdeki tanıma sonuçlarına ilişkin bir örnek Şekil 4.1'de gösterilmiştir. Farklı CNN mimarilerinin doğru sınıflama oranlarını karşılaştırmak için AlexNet, GoogleNet ve ResNet50 mimarileri ile oluşturulan R-CNN modellerinde ve faster R-CNN ile YOLOv4 modelinde elde edilen sonuçlar karşılaştırılmıştır. Ayrıca, oluşturulan modeller kullanılarak test işlemlerinin yapılması için bir kullanıcı arayüzü oluşturulmuştur.



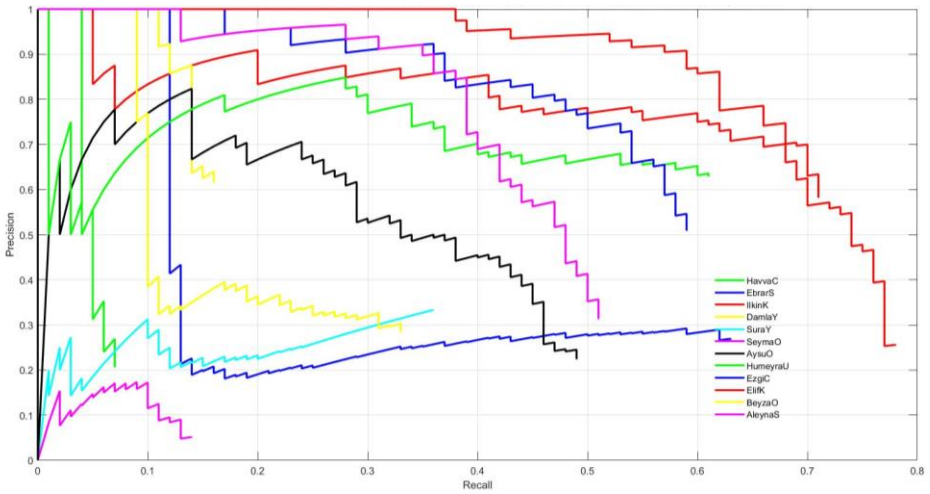
Şekil 4.1. Test veri setinde başarılı tanınan öğrencilerin görüntü ve etiketleri.

4.1. R-CNN İLE ALEXNET, GOOGLENET VE RESNET50 SONUÇLARI

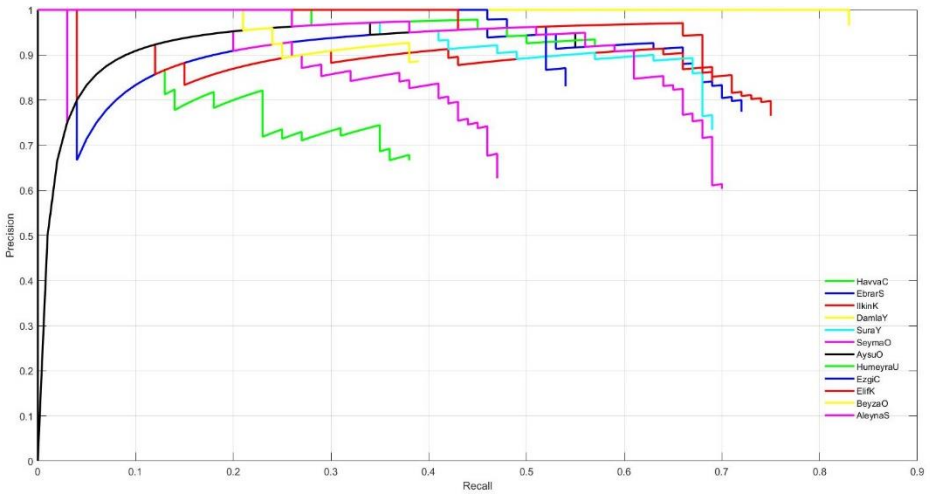
AlexNet mimarisinin kullanıldığı sinir ağı ile elde edilen ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafiği Şekil 4.2'de verilmiştir. Şekil 4.3'te GoogleNet ve Şekil 4.4'te ise ResNet50 mimarisinin kullanıldığı sinir ağı ile elde edilen ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafikler görülmektedir.



Şekil 4.2. AlexNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği.



Şekil 4.3. GoogleNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği.



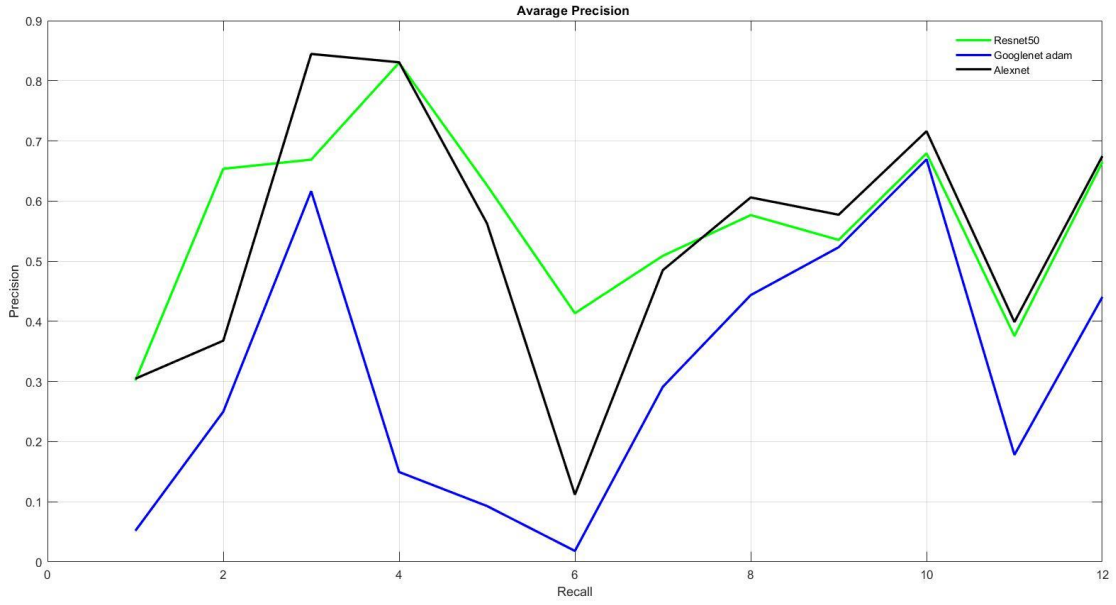
Şekil 4.4. ResNet mimarisi ile elde edilen ortalama kesinlik (AP) grafiği.

Çizelge 4.1. AlexNet, GoogleNet ve ResNet50 mimarileri ile elde edilen ortalama kesinlik (AP) değerleri.

	AlexNet	**GoogleNet**	**ResNet50**
HC	0,30	0,05	0,30
ES	0,37	0,25	0,65
IK	0,84	0,62	0,67
DY	0,83	0,15	0,83
SY	0,56	0,09	0,83
SO	0,11	0,02	0,41
AO	0,49	0,29	0,51
HU	0,61	0,44	0,58
EC	0,58	0,52	0,54
EK	0,72	0,67	0,68
BO	0,40	0,18	0,38
AS	0,67	0,44	0,66
mAP	**0,5400**	**0,3100**	**0,5867**

AlexNet, GoogleNet ve ResNet50 mimarilerinin kullanıldığı sinir ağları ile elde edilen ortalama kesinlik (AP) değerleri Çizelge 4.1'de verilmiştir. 0,30 ile 0,83 arasında değişen ortalama kesinlik (AP) değerlerinin görüldüğü ResNet50 mimarisinin kullanıldığı ağın ortalama kesinlik değerlerinin ortalaması (mAP) 0,59 olarak elde edilirken 0,11 ile 0,84 arasında değişen ortalama kesinlik (AP) değerlerinin görüldüğü AlexNet mimarisinin kullanıldığı ağın ortalama kesinlik değerlerinin ortalaması (mAP) ise 0,54 olarak elde edilmiştir. Bu uygulamadaki en düşük başarı oranı olarak, 0,02 ile 0,67 arasında değişen

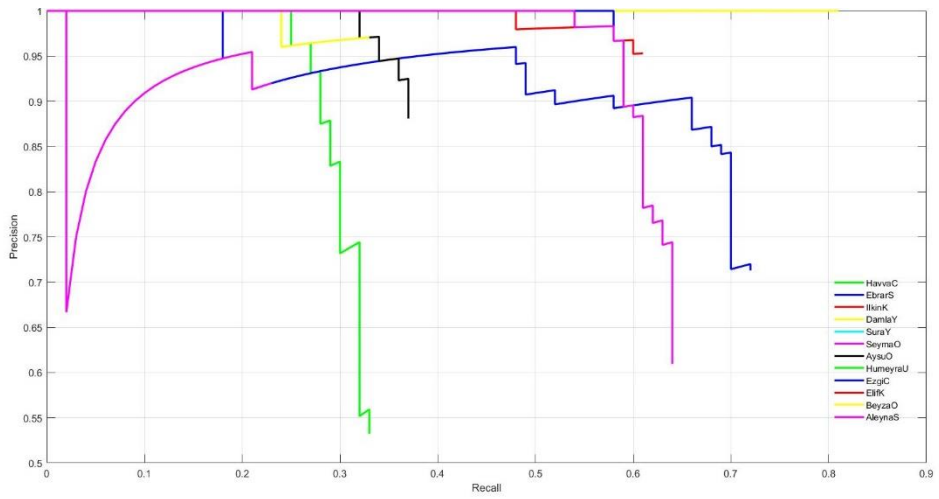
ortalama kesinlik (AP) değerlerinin görüldüğü GoogleNet mimarisinin kullanıldığı ağın ortalama kesinlik değerlerinin ortalaması (mAP) 0,31 olarak elde edilmiştir. Şekil 4.5'te görülen ortalama kesinlik (AP) - hatırlama grafiğinde AlexNet, GoogleNet ve ResNet50 mimarilerinin kullanıldığı farklı derinliklere sahip derin sinir ağlarının aynı hiperparametrelerle aynı veri setindeki başarı oranları görülmektedir. AlexNet 25 katmana sahipken, GoogleNet 144 katmana ve ResNet50 177 katmana sahip olup aralarında en derin ağ yapısına sahip olan ResNet50 mimarisidir. En yüksek başarı oranının en iyi ortalama kesinlik (AP) değerine sahip olan ResNet50 mimarisi ile elde edildiği görülmektedir. Ancak bu durum sinir ağlarının başarısının derinlikleri ile doğru orantılı olduğu şeklinde yorumlanmamalıdır. Başarı oranlarının farklı olmasının nedeni, ağın derinliğinden değil farklı sinir ağlarının kullandığı farklı aktivasyon fonksiyonları ve katmanlar içerisinde yer alan farklı filtre sayılarıdır. Nitekim, ağın derinliği açısından bakıldığında GoogleNet mimarisine göre daha az katmana sahip olan AlexNet mimarisinin GoogleNet mimarisine kıyasla daha yüksek bir başarı gösterdiği görülmektedir. Bu durum aynı zamanda hiperparametrelerin önemini işaret etmektedir.



Şekil 4.5. AlexNet, GoogleNet ve ResNet50 mimarilerinin karşılaştırılması.

4.2. GOOGLENET İLE ADAM VE SGDM ALGORİTMALARININ SONUÇLARI

GoogleNet mimarisinin kullanıldığı sinir ağında SGDM algoritması ile elde edilen ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafiği Şekil 4.6'da verilmiştir.



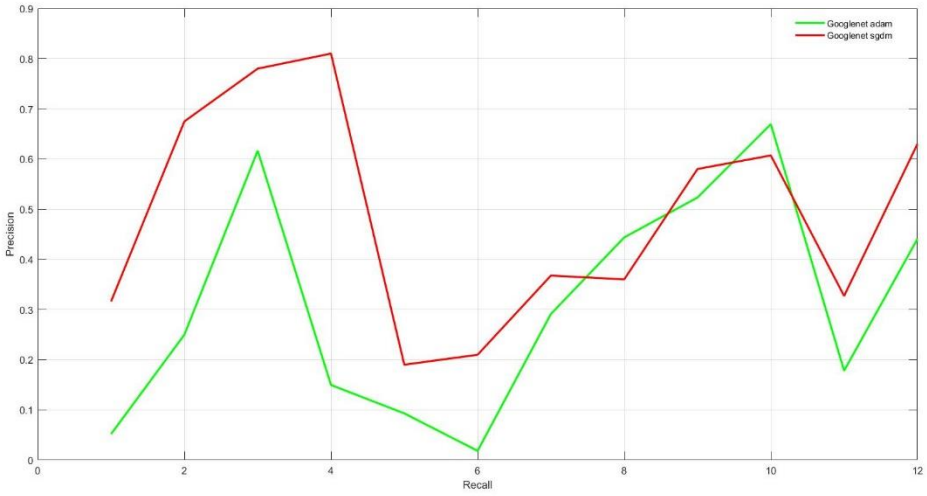
Şekil 4.6. GoogleNet SGDM algoritması ile elde edilen ortalama kesinlik (AP) grafiği.

Orijinal veri seti ile sinir ağı eğitiminde veri setinin büyük olması, fazla örnek içermesi doğruluğu yüksek bir eğitim gerçekleşmesini etkileyen önemli parametrelerdendir. Ancak diğer önemli bir konu da eğitim sürecinde kullanılan hiperparametreler ve algoritmalardır. Öğrenme hızındaki değişim sinir ağının testindeki sonucu doğrudan değiştirebilmektedir. GoogleNet mimarisi bir önceki uygulamada en düşük ortalama kesinlik değerlerinin ortalamasına (mAP) sahip olduğundan hiperparametrelerde ince ayar yapılarak başarı oranına etkisini incelemek amacıyla GoogleNet mimarisinde ADAM ve SGDM algoritmalarının kullanılmasıyla yapılan uygulamalara ait ortalama kesinlik (AP) değerleri Çizelge 4.2'de verilmiş olup ortalama kesinlik (AP) - hatırlama grafiği ise Şekil 4.7'de görülmektedir. ADAM algoritmasının kullanıldığı sinir ağı ile elde edilen ortalama kesinlik (AP) değerleri 0,02 ile 0,67 arasında değişmekte olup ortalama kesinlik değerlerinin ortalaması (mAP) 0,31 olarak elde edilmişken 0,21 ile 0,81 arasında değişen ortalama kesinlik (AP) değerlerinin görüldüğü SGDM algoritmasının kullanıldığı sinir ağının ortalama kesinlik değerlerinin ortalaması (mAP) 0,48 olarak elde edilmiştir. Optimize edici bir öğrenme hızı bilgisi olan ADAM ağ parametrelerini güncellemek için hareketli ortalamaları kullanmaktadır. Hata oranını azaltmak amacıyla

sık kullanılan bir başka optimizasyon algoritması olan SGDM ise bu güncelleme için momentum yöntemi kullanmaktadır. SGDM ve ADAM algoritmalarının kullanıldığı GoogleNet mimarileri karşılaştırıldığında SGDM ile daha yüksek bir başarı oranının sağlandığı görülmektedir.

Çizelge 4.2. GoogleNet mimarisinde ADAM ve SGDM algoritmaları ile elde edilen ortalama kesinlik (AP) değerleri.

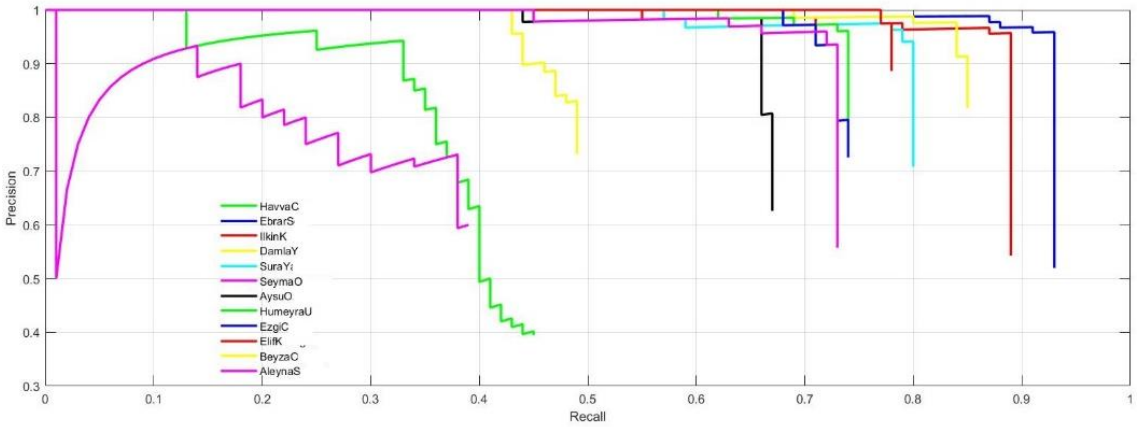
	ADAM	**SGDM**
HC	0,05	0,32
ES	0,25	0,67
IK	0,62	0,78
DY	0,15	0,81
SY	0,09	0,19
SO	0,02	0,21
AO	0,29	0,37
HU	0,44	0,36
EC	0,52	0,58
EK	0,67	0,61
BO	0,18	0,33
AS	0,44	0,63
mAP	**0,3100**	**0,4883**



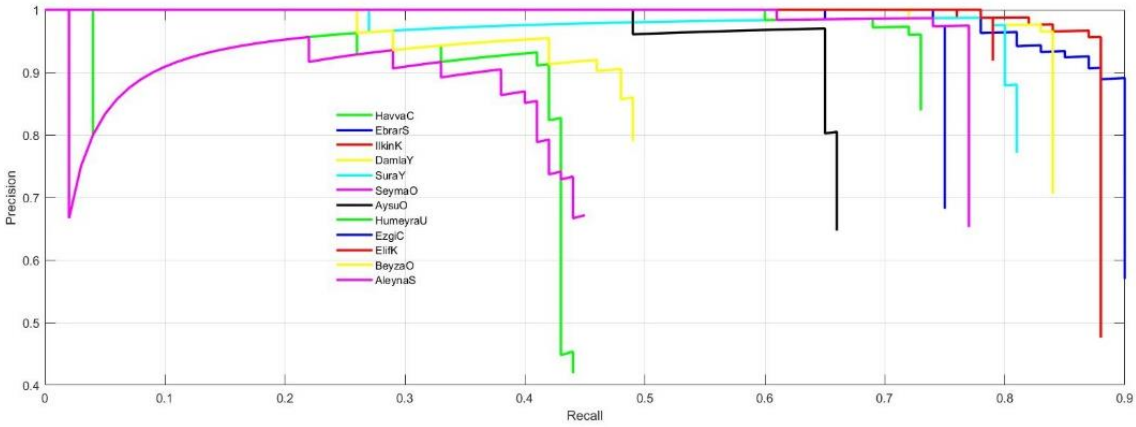
Şekil 4.7. GoogleNet mimarisinde ADAM ve SGDM algoritmalarının karşılaştırılması.

4.3. ALEXNET İLE ARTTIRMA SONUÇLARI

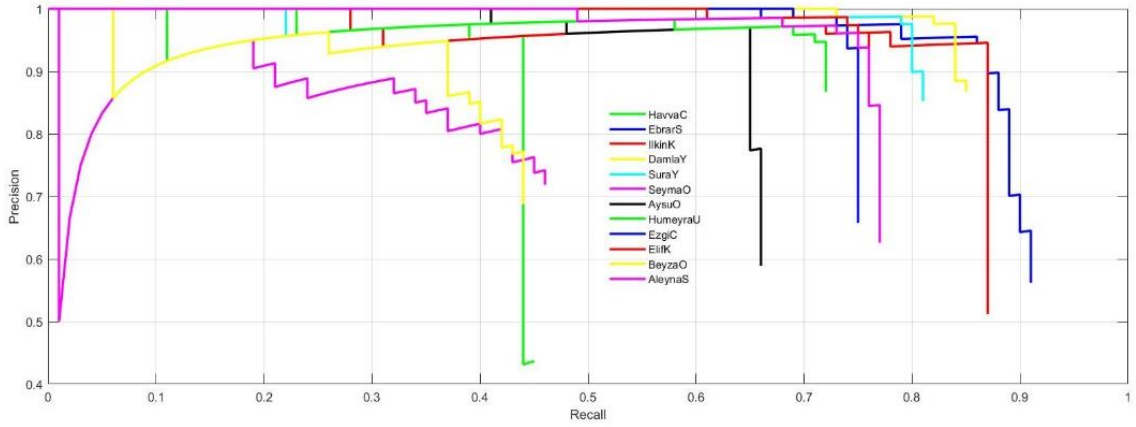
AlexNet mimarisinin kullanıldığı sinir ağında artırma işlemi ve ayrıca hiperparametrelerde farklı filtre boyutu, eğitim turu sayısı ve mini yığın boyutu değerleri kullanılarak yapılan uygulamaların sonucunda elde edilen ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafikleri sırasıyla Şekil 4.8, Şekil 4.9, Şekil 4.10 ve Şekil 4.11'de verilmiştir.



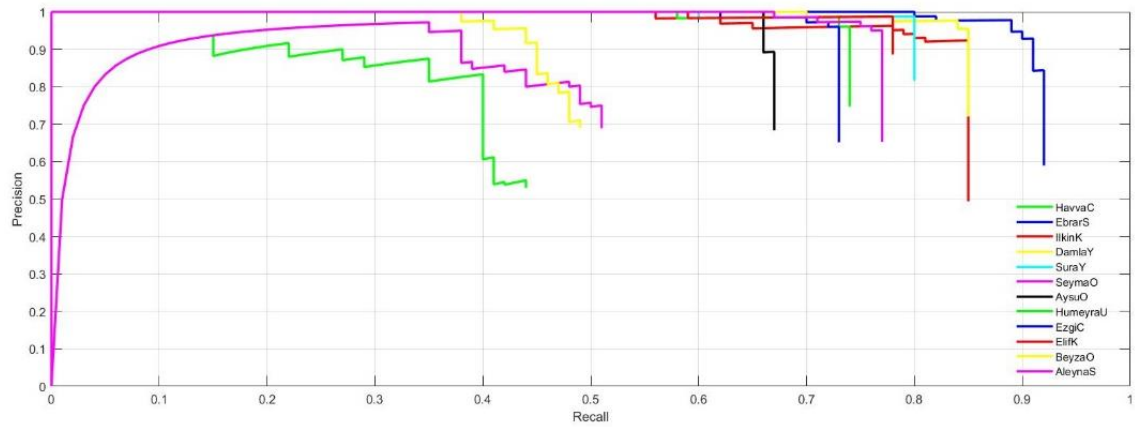
Şekil 4.8. AlexNet arttırma 1 ile elde edilen ortalama kesinlik (AP) grafiği.



Şekil 4.9. AlexNet arttırma 2 ile elde edilen ortalama kesinlik (AP) grafiği.



Şekil 4.10. AlexNet arttırma 3 ile elde edilen ortalama kesinlik (AP) grafiği.



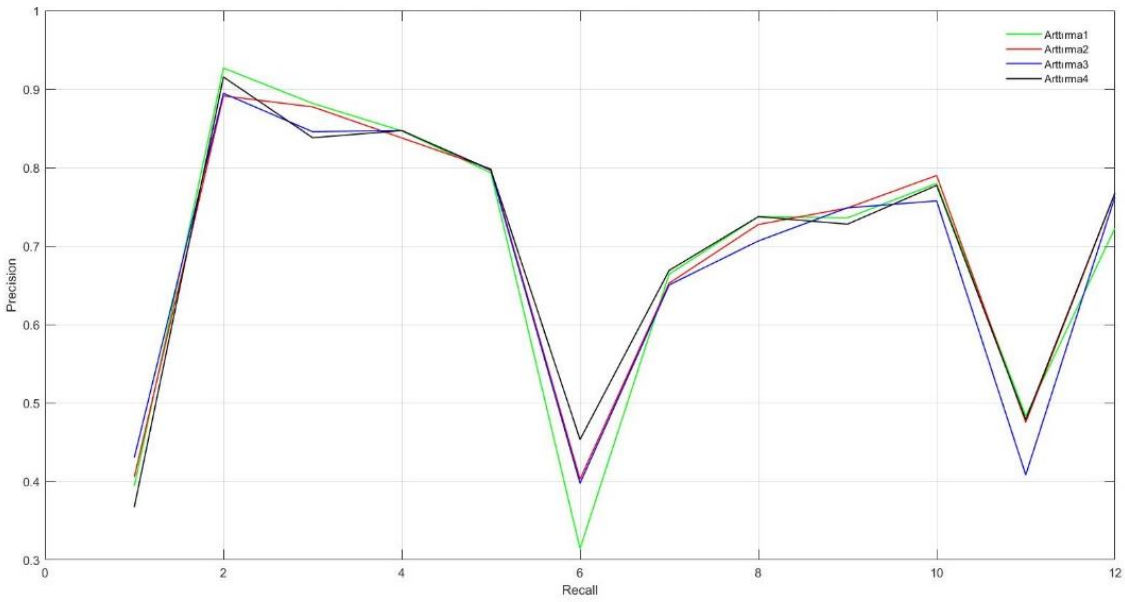
Şekil 4.11. AlexNet arttırma 4 ile elde edilen ortalama kesinlik (AP) grafiği.

Çizelge 4.3. AlexNet mimarisinde arttırma uygulama ve farklı hiperparametre değerleri ile elde edilen ortalama kesinlik (AP) değerleri.

	Arttırma 1	**Arttırma 2**	**Arttırma 3**	**Arttırma 4**
HC	0,39	0,41	0,43	0,37
ES	0,93	0,89	0,89	0,92
IK	0,88	0,88	0,85	0,84
DY	0,85	0,84	0,85	0,85
SY	0,79	0,80	0,80	0,80
SO	0,31	0,40	0,40	0,45
AO	0,66	0,65	0,65	0,67
HU	0,74	0,73	0,71	0,74
EC	0,74	0,75	0,75	0,73
EK	0,78	0,79	0,76	0,78
BO	0,48	0,48	0,41	0,48
AS	0,72	0,77	0,76	0,77
mAP	**0,6892**	**0,6992**	**0,6883**	**0,7000**

Orijinal veri seti ile bir eğitim yapılacağı zaman oldukça büyük veri setlerinin kullanımı doğruluğu ve başarı oranını artırır. Ancak her zaman uygun ve yeterli sayıda veriden oluşan bir veri setinin temin edilmesi mümkün olamamaktadır. Özellikle az veri ile sinir ağlarının eğitiminde ImageAugmenter işlemi sıklıkla başvurulan bir fonksiyondur. Resimlerin ayna görüntüleri oluşturularak, resimler belirli açılarla döndürülerek veya piksel değerleri verilen aralıklarda değiştirilerek veri seti büyütülebilir. Bu doğrultuda, AlexNet mimarisinde artırma işlemi uygulanması ve ayrıca hiperparametrelerde farklı

eğitim turu sayısı ve mini yığın boyutu değerleri kullanılmasının ağın başarısına etkisini incelemek için yapılan uygulamalarda elde edilen ortalama kesinlik (AP) değerleri Çizelge 4.3'de verilmiş ve bu uygulamaların birbirileriyle karşılaştırılmasına ilişkin ortalama kesinlik (AP) - hatırlama grafiği de Şekil 4.12'de gösterilmiştir. Genel olarak arttırma işlemi ağın genel başarısını oldukça küçük bir miktarda etkilerken farklı hiperparametre değerleri kullanılmasının da bu başarı üzerine bir etkisi gözlenmemiştir. Eğitim turu sayısı 10 ve mini yığın boyutu 25 olarak uygulanan birinci arttırmada ortalama kesinlik (AP) değerleri 0,31 ile 0,93 arasında değişmekte olup ağın ortalama kesinlik değerlerinin ortalaması (mAP) 0,69 olarak elde edilirken, eğitim turu sayısı 10 ve mini yığın boyutu 32 olan ikinci arttırmada ortalama kesinlik (AP) değerleri 0,40 ile 0,89 arasında değişmiş ve ağın ortalama kesinlik değerlerinin ortalaması (mAP) 0,70 olarak elde edilmiştir. Eğitim turu sayısı 12 ve mini yığın boyutu 32 olan üçüncü arttırmada ortalama kesinlik (AP) değerleri 0,40 ile 0,89 arasında olup ağın ortalama kesinlik değerlerinin ortalaması (mAP) 0,69 iken, eğitim turu sayısı 15 ve mini yığın boyutu 32 olan dördüncü arttırma uygulamasında ise ortalama kesinlik (AP) değerleri 0,37 ile 0,92 arasında ve ağın ortalama kesinlik değerlerinin ortalaması (mAP) ise 0,70 olarak elde edilmiştir.

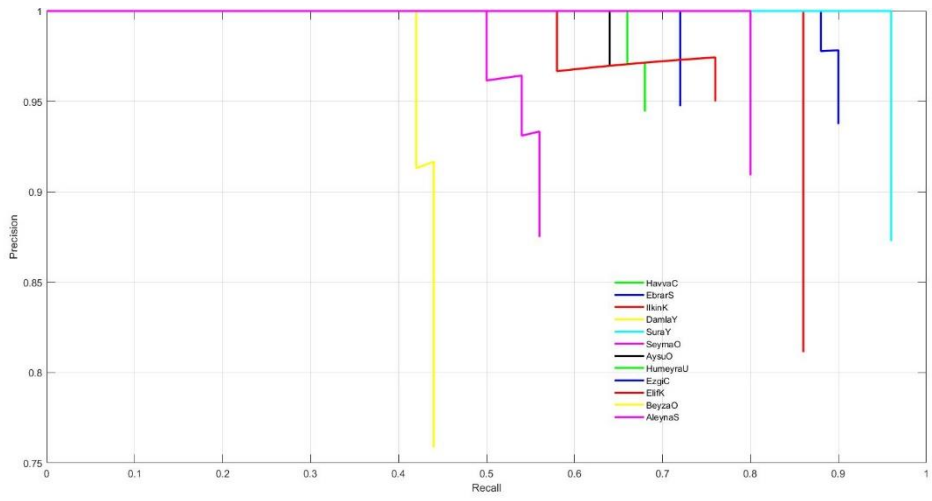


Şekil 4.12. AlexNet arttırma uygulamalarının karşılaştırılması.

4.4. ALEXNET İLE 350/50 VERİ SETİ SONUÇLARI

Veri setinin 350 eğitim ve 50 test verisi şeklinde ayrılarak R-CNN ile AlexNet mimarisinin kullanıldığı ağdan elde edilen ortalama kesinlik (AP) değerleri Çizelge 4.4'te ve ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafiği Şekil 4.13'te verilmiştir.

Eğitim için ayrılan veri setindeki görüntü sayısı artırılarak uygulanan yeni eğitim genel ağ başarısını 0,72 düzeyine çıkarırken, tüm sınıfların doğruluk başarısında da bir miktar iyileşmeler tespit edilmiştir. Eğitim için kullanılan görüntü sayısının 300 ve test için kullanılan görüntü sayısının 100 olduğu ve R-CNN ile AlexNet mimarisinin kullanıldığı uygulama sonuçları ile kıyaslandığında tüm sınıflarda görülen bu artışın özellikle genel başarısının düşük olmasına neden olan sınıflarda daha yüksek miktarlarda gerçekleştiği, artış miktarlarının 0,07 ile 0,53 arasında değiştiği görülmüştür. Eğitim için kullanılan görüntü sayısının 350 ve test için kullanılan görüntü sayısının 50 olduğu bu ağ ile yapılan eğitim neticesinde sınıflardaki ortalama kesinlik (AP) değerlerinin 0,42 ile 0,96 arasında değişim gösterdiği ve ağın genel başarısının ise 0,72 olarak elde edilebildiği görülmektedir.



Şekil 4.13. AlexNet 350/50 veri seti ile elde edilen ortalama kesinlik (AP) grafiği.

Çizelge 4.4. AlexNet 350/50 veri seti ile elde edilen ortalama kesinlik (AP) değerleri.

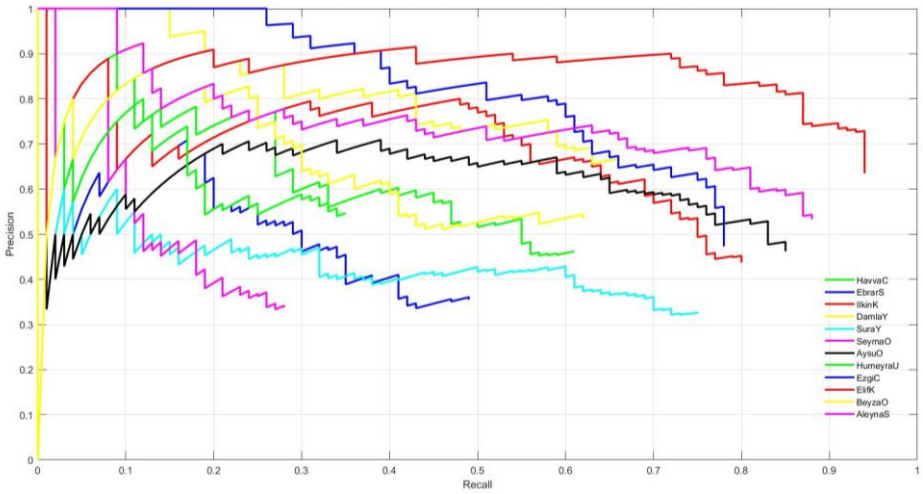
	350/50
HC	0,42
ES	0,90
IK	0,86
DY	0,96
SY	0,96
SO	0,56
AO	0,64
HU	0,68
EC	0,72
EK	0,75
BO	0,44
AS	0,80
mAP	**0,7242**

4.5. FASTER R-CNN SONUÇLARI

ResNet50 mimarisinin kullanıldığı faster R-CNN modeli kullanılarak elde edilen ortalama kesinlik (AP) değerleri Çizelge 4.5'te ve ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafiği Şekil 4.14'te verilmiştir. Bu ağda, ortalama kesinlik (AP) değerleri 0,16 ile 0,81 arasında değişmekte olup ağın ortalama kesinlik değerlerinin ortalaması (mAP) ise 0,47 olarak elde edilmiştir.

Çizelge 4.5. Faster R-CNN ile elde edilen ortalama kesinlik (AP) değerleri.

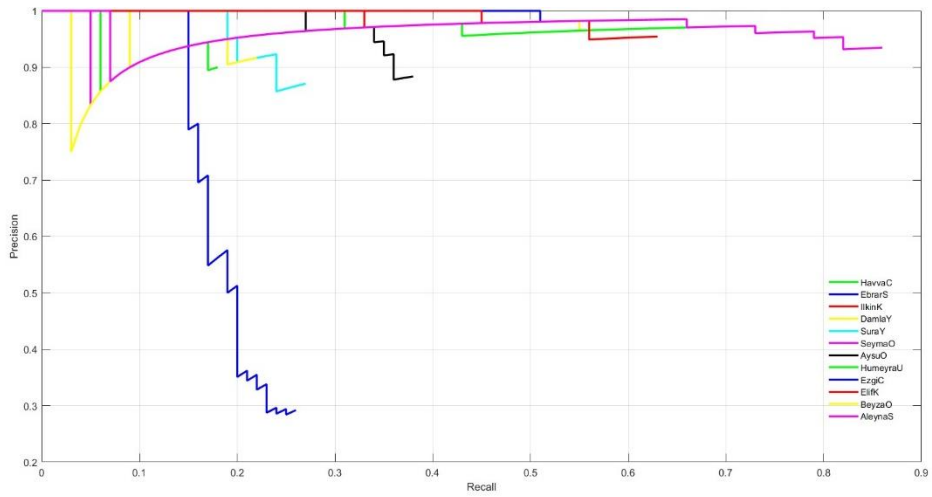
	Faster R-CNN
HC	0,26
ES	0,27
IK	0,56
DY	0,55
SY	0,33
SO	0,16
AO	0,54
HU	0,36
EC	0,67
EK	0,81
BO	0,41
AS	0,67
mAP	**0,4658**



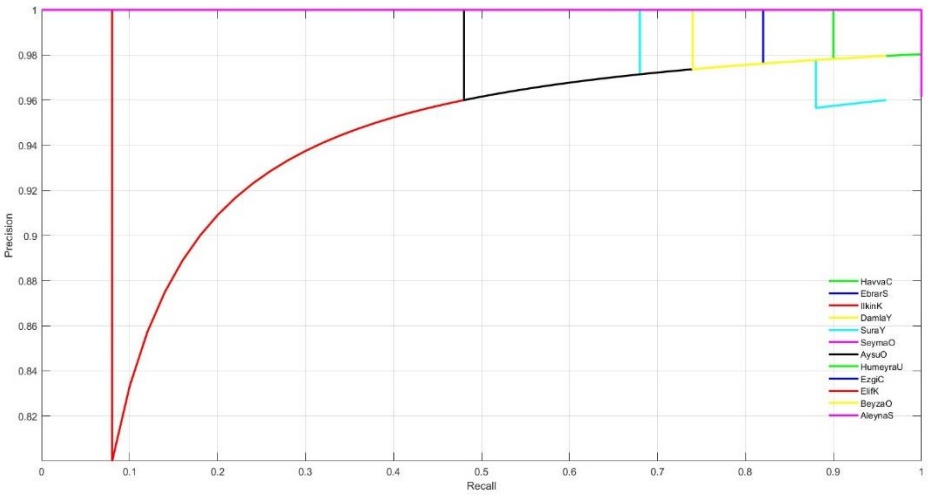
Şekil 4.14. Faster R-CNN ile elde edilen ortalama kesinlik (AP) grafiği.

4.6. YOLOv4 SONUÇLARI

ResNet50 mimarisinin kullanıldığı YOLOv4 modeli ile elde edilen sonuçlara ilişkin ortalama kesinlik (AP) değerleri Çizelge 4.6'da ve ortalama kesinlik (AP) değerlerine ait kesinlik - hatırlama grafiği ise 300 eğitim ve 100 test verisi için Şekil 4.15'te ve 350 eğitim ve 50 test verisi için Şekil 4.16'da verilmiştir. Eğitim için kullanılan görüntü sayısının 300 ve test için kullanılan görüntü sayısının 100 olduğu uygulamada ortalama kesinlik (AP) değerleri 0,17 ile 0,83 arasında değişmekte olup bu ağda ortalama kesinlik değerlerinin ortalaması (mAP) 0,42 olarak elde edilmiştir. Eğitim için ayrılan görüntü sayısı 350 olarak alınarak artırma uygulandığında genel ağ başarısı 0,98 düzeyine çıkarken, tüm sınıfların doğruluk başarısında ciddi düzeyde artışlar olduğu görülmüştür. Ortalama kesinlik (AP) değerlerinin 0,93 ile 1 arasında değiştiği bu uygulamada altı sınıfta başarı oranının 1 olduğu tespit edilmiştir. Eğitim için kullanılan veri setindeki görüntü sayısının 300 ve test için kullanılan görüntü sayısının 100 olduğu uygulama sonuçları ile kıyaslandığında artış miktarlarının 0,17 ile 0,81 arasında değiştiği ve özellikle genel ağ başarısının düşük olmasına neden olan sınıflarda daha yüksek miktarlarda artış gerçekleştiği görülmüştür.



Şekil 4.15. YOLOv4 ile elde edilen ortalama kesinlik (AP) grafiği.



Şekil 4.16. YOLOv4 350/50 veri seti ile elde edilen ortalama kesinlik (AP) grafiği.

Çizelge 4.6. YOLOv4 ile elde edilen ortalama kesinlik (AP) değerleri.

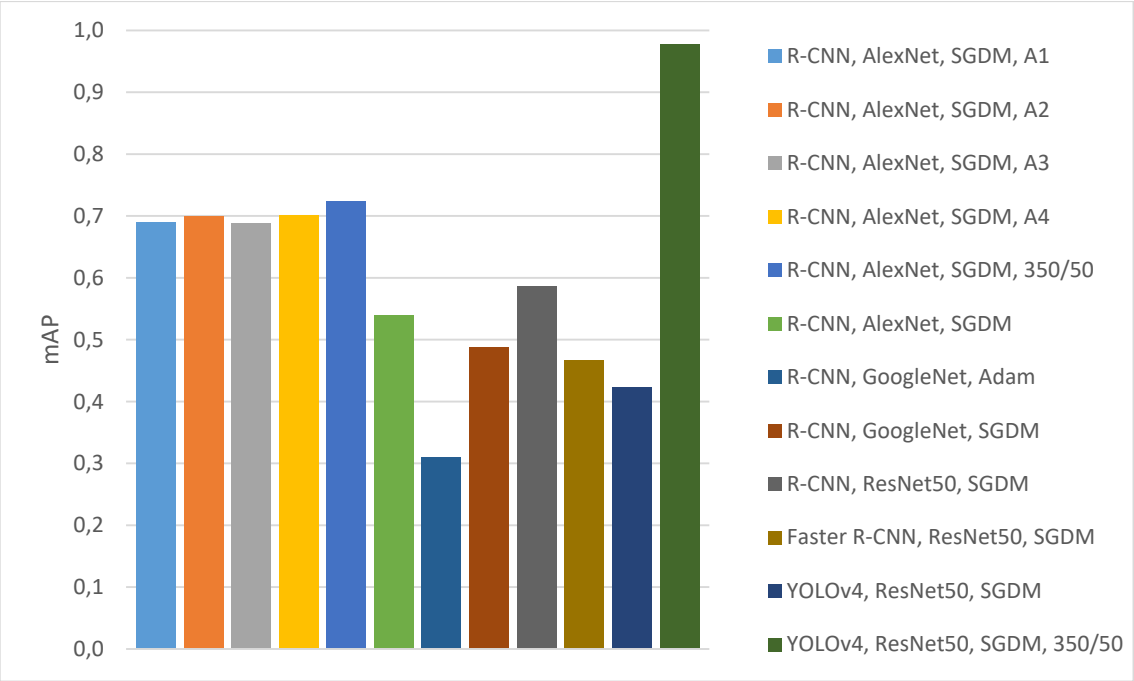
	300/100	**350/50**
HC	0,17	0,98
ES	0,20	1
IK	0,40	0,94
DY	0,54	1
SY	0,26	0,95
SO	0,19	1
AO	0,37	0,93
HU	0,67	1
EC	0,63	1
EK	0,62	0,98
BO	0,20	0,95
AS	0,83	1
mAP	**0,4233**	**0,9775**

4.7. TÜM UYGULAMA SONUÇLARININ KARŞILAŞTIRILMASI

Bu çalışmada yapılan tüm uygulamaların sonucunda elde edilen ortalama kesinlik değerlerinin ortalaması (mAP) kullanılan ağ mimarisi, algoritma, veri setinin boyutu ve artırma uygulanmasına göre Çizelge 4.7'de verilmiş ve bu uygulamalar sonucunda elde edilen ortalama kesinlik değerlerinin ortalaması (mAP) Şekil 4.17'de toplu olarak gösterilmiştir.

Çizelge 4.7. Farklı uygulamalar ile elde edilen ortalama kesinlik (mAP) değerleri.

<table>
<tr><th colspan="2">Ağ</th><th>Veri Seti</th><th>Algoritma</th><th>mAP</th></tr>
<tr><td rowspan="8">R-CNN</td><td rowspan="5">AlexNet</td><td>Artırma1</td><td rowspan="6">SGDM</td><td>0,6892</td></tr>
<tr><td>Artırma2</td><td>0,6992</td></tr>
<tr><td>Artırma3</td><td>0,6883</td></tr>
<tr><td>Artırma4</td><td>0,7000</td></tr>
<tr><td>350/50</td><td>0,7242</td></tr>
<tr><td rowspan="4">300/100</td><td>0,5400</td></tr>
<tr><td rowspan="2">GoogleNet</td><td>Adam</td><td>0,3100</td></tr>
<tr><td rowspan="4">SGDM</td><td>0,4883</td></tr>
<tr><td rowspan="3">ResNet50</td><td>0,5867</td></tr>
<tr><td>Faster R-CNN</td><td>0,4658</td></tr>
<tr><td rowspan="2">YOLOv4</td><td>0,4233</td></tr>
<tr><td>350/50 Artırma</td><td>0,9775</td></tr>
</table>

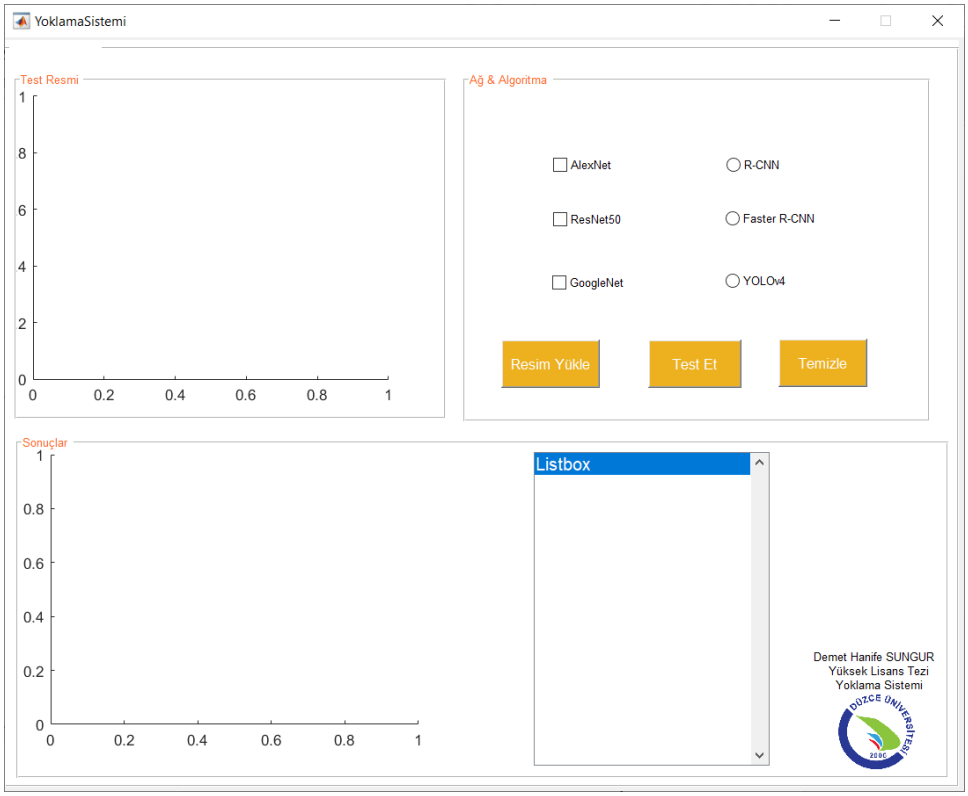


Şekil 4.17. Farklı uygulamalar ile elde edilen ortalama kesinlik (mAP) değerleri.

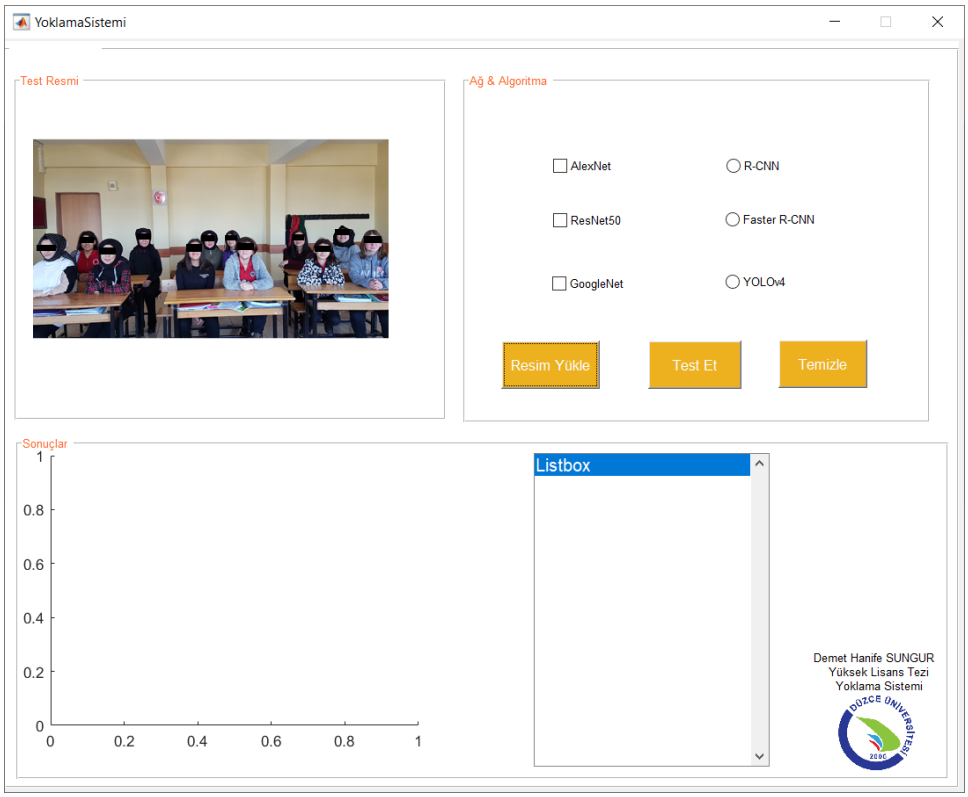
4.8. KULLANICI ARAYÜZÜ

Bu çalışma kapsamında yapılan yüz tanıma işlemlerinin kolaylıkla test edilebilmesi ve sonuçların görüntülenebilmesi amacıyla bir kullanıcı arayüzü oluşturulmuştur. Oluşturulan arayüzün ilk açıldığındaki ekran görüntüsü Şekil 4.18'de gösterilmiştir. Kullanıcı arayüzünün üst kısmında bulunan iki adet alandan soldaki alanda test edilmek istenen görüntünün önizlemesi kullanıcıya gösterilmekte, sağdaki alanda ise test işlemi için kullanılabilecek ağ ve mimarilerin seçilebilmesi için seçim kutuları ile test edilecek görüntünün arayüze aktarılması için "Resim Yükle", görüntü yüklendikten sonra test işleminin yapılabilmesi için "Test Et" ve test işlemi tamamlandıktan sonra başka bir test yapmak üzere arayüzün başlangıç haline getirilebilmesi için "Temizle" butonları yer almaktadır. Arayüzün alt kısmında bulunan iki adet alandan soldaki alanda test işlemi tamamlandıktan sonra görüntü üzerinde algılanan yüzler işaretlenerek o sınıfa ait olma olasılıkları gösterilmektedir. Bu alanın sağ tarafında ise algılanan yüzler için bir liste oluşturularak o sınıfa ait olma olasılıkları görünürlüğü ve okunabilirliği kolaylaştırmak için liste halinde verilmektedir.

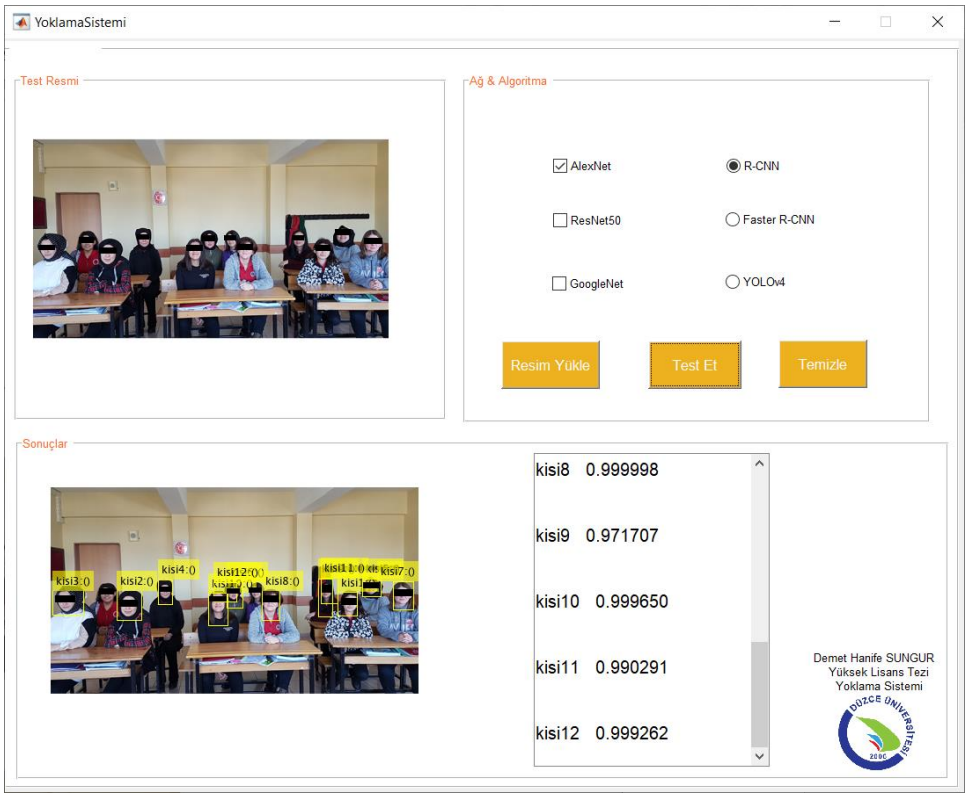
Oluşturulan bu kullanıcı arayüzü ile test işlemleri iki aşamada kolaylıkla yapılabilmektedir. Test edilecek görüntü "Resim Yükle" butonu ile seçilerek yüklendikten sonra sol üst alanda yüklenen görüntünün önizlemesi gösterilmekte, ardından test için kullanılacak ağ ve mimarinin seçilmesinden sonra "Test Et" butonuyla test işlemi başlamaktadır. Test işlemi tamamlandıktan sonra sonuçlar attaki alanda yine görüntü üzerinde işaretlenerek ve hemen yanında liste halinde gösterilmektedir. Arayüze kullanıcı tarafından resim yüklendikten sonraki ekran görüntüsü Şekil 4.19'de ve test işlemi sonrası sonuç ekranı ise Şekil 4.20'da gösterilmektedir.



Şekil 4.18. Kullanıcı arayüzü giriş ekranı.



Şekil 4.19. Kullanıcı arayüzü görüntü yükleme.



Şekil 4.20. Kullanıcı arayüzü sonuç ekranı.

5. SONUÇLAR VE ÖNERİLER

Bu çalışmada, öğrencilerin sınıf ortamında ve toplu halde bulundukları fotoğraflardaki yüz görüntülerinden yola çıkarak öğrencilerin tanınması ve görüntü üzerinde tespiti için R-CNN derin öğrenme modeline dayalı olan bir yaklaşım önerilmiştir.

Eğitimler, araştırmacılar tarafından hazırlanan özgün ve bir sinir ağının yeniden eğitimi için küçük sayılacak miktarda görüntüden oluşan bir veri seti ile R-CNN modeli kullanılarak gerçekleştirilmiştir. Eğitim sürecinde farklı derinliklere sahip ağ yapıları ve mimarileri test edilmiştir. Hiperparametrelerdeki değişikliklerin eğitim başarısına etkileri de karşılaştırılarak incelenmiştir. Bu doğrultuda, sinir ağı eğitiminin başarısının ağın derinliğinden daha çok katmanlardaki özellik çıkarma, katmanlardaki filtre boyutları, kullanılan aktivasyon fonksiyonları ile ilgili olduğu gözlenmiştir. Ayrıca veri arttırma gibi uygulamalarla ağın eğitim başarısının iyileştirildiği ümit verici sonuçlar bulunmuştur.

Maliyet performans açısından değerlendirildiğinde daha derin ağların eğitim süresinin daha uzun olması ile birlikte bu ağlarda arttırma uygulanmaksızın doğruluğun da daha yüksek olduğu görülmüştür. Ancak AlexNet gibi derinliği daha az olmasına rağmen arttırma kullanılarak daha kısa sürede daha yüksek başarı oranlarının gözlendiği umut verici sonuçlar da elde edilmiştir.

Test veri seti ile yapılan deneysel çalışmaların sonucunda doğru sınıflandırma başarısı açısından en yüksek değer olarak YOLOv4 uygulaması ile %98 ortalama kesinlik değerlerinin ortalaması (mAP) değerine ulaşılmıştır. ImageAugmenter uygulanmadığı durumlarda tahminlerdeki başarı oranı %31 ile %59 arasında olsa bile modelin doğru sınıflandırma yaptığı görülmüştür. Artırma uygulandığında ise bu oranın %70 düzeyine çıkabildiği görülmüştür. Eğitim için kullanılan veri setindeki görüntü sayısının daha fazla olduğunda, 350 görüntü ile eğitim yapıldığında ise bu oranların AlexNet mimarisinin kullanıldığı R-CNN ile %72 ve ResNet50 mimarisinin kullanıldığı YOLOv4 ile %98 düzeyinde oladuğu tespit edilmiştir.

Bu çalışmada kullanılan veri setinde 12 farklı sınıf bulunmaktadır. Bazı sınıfların başarı oranı tüm testlerde %50'nin altında kalırken bazı sınıflarda da %100 oranına kadar çıkan başarılı sonuçlar elde edilebilmiştir. Özel bir ortam ışıklandırması, düzenlenmesi veya

öğrencilerin görünümü ile ilgili herhangi bir ön hazırlık veya başka bir müdahale olmadan ve profesyonel olmayan 12 MP bir kamera kullanılarak ön işleme olmadan 4032*2268 boyutunda çekilen ve daha sonra eğitim sürecinin yürütülmesi sırasında, ekran kartı ve hafıza yetersizliği nedeniyle zorunlu olarak boyutları 420*236'ya indirgenen fotoğrafların kullanıldığı düşünüldüğünde, bu çalışmada elde edilen doğru sınıflama başarısının yeterli düzeyde olduğu söylenebilir. Dolayısıyla bu çalışmada önerilen bu bilgisayar destekli yaklaşımın uzmanlar tarafından farklı amaçlarla öğrencilerin yüzlerini tanımada ikincil bir araç olarak kullanılabileceği öngörülmektedir.

Sonuç olarak öğrencilerin biyometrik özelliklerini belirlemek ve sınıfta geleneksel olarak isim okuyarak yoklama alma sürecine kıyasla çok daha kısa sürede yoklama alınarak derse daha kısa sürede devam edebilmek için uygun derin öğrenme modellerinin kullanılabileceği de gösterilmiş olmuştur.

Derin öğrenme kütüphanelerinin kullanılmasıyla oluşturulan bu model ile yüzlerin kolayca tanınmasının yanı sıra yeterli sayıda görsel içeren bir veri seti ile çalışılması durumunda öğrencilerin duygu durumlarının tanınması da sağlanabilir. Bu, bireylerin stresli olup olmadığının çok daha kısa sürede belirlenebilmesini sağlayacaktır. Bu amaçla, yüz görüntülerinden sınıfların ortam koşullarını ve bireylerin o andaki eğitime hazır olup olmama durumlarını belirlemek için bir veri tabanı oluşturulabilir ve derin öğrenme ağlarına uygulanabilir. Böylece öğrenme ortamının iyileştirilmesi ve öğrencilerin hazır bulunuşluklarının değerlendirilmesi mümkün olabilir. Bu şekilde eğitimin etkisinin artması sağlanabilir. Ayrıca bireylerin farklı biyometrik özelliklerini tanımak için derin öğrenme modelleri kullanılabilir. Daha hızlı bir model isteniyorsa daha fazla performans modeli eğitilebilir.

Bu çalışmanın en önemli kısıtlılıklarından biri veri setindeki görüntü sayısıdır. Ayrıca bir sınıftaki öğrenci sayısı değişebilir veya bazı sınıflarda çok kalabalık ortamlarda tanıma yapılması gerekebilir. Bu çalışmada kullanılan yöntemde, tanımlama modeline yeni öğrenci görüntülerini eklemek, modelin yeniden eğitilmesini gerektirdiğinden zaman alıcı bir süreçtir. Bu eksiklik ileriki çalışmalarda giderilebilir.

Yüz tanıma nesne tanımaya göre zorluklar barındırmaktadır. Bazı sınıflarda başarı oranının diğer sınıflara göre daha düşük olmasında görüntülerde daha geride olması, ışık açısı, çözünürlüğün düşük olması gibi parametreler etkili olabilir. Yüksek çözünürlükte verilerle tekrar test edilerek başarı karşılaştırması yapılabilir. Yazılım teknolojisindeki

son gelişmelerle bulut tabanlı sistemler üzerinde de eğitim ve testler yapılabilmektedir. Daha sonraki çalışmalarda özgün veri seti yüksek çözünürlükte etiketlenerek çözünürlüğün farklı ağlardaki özellik çıkarma, sınıflandırma başarısı performansına etkisi incelenebilir ve ayrıca sınıflar arasındaki başarı oranlarının test edilmesi ve karşılaştırması da yapılabilir.

Sonuç olarak, bu çalışmada fotoğraflardaki görüntülerden yüz tespiti ve tanıma işlemlerinde deneysel çalışmalarda en az sayıda veri içeren eğitim seti kullanıldığında %59 düzeyinde olan başarı oranının, eğitim setinde daha fazla sayıda görüntü kullanılması ve artırma işlemi uygulanması durumunda daha da artırılabildiği ve %98 doğrulukla sınıflandırma düzeyine kadar yükselebildiği görülmüştür.

6. KAYNAKLAR

[1] U. Jayaraman, P. Gupta, S. Gupta, G. Arora ve K. Tiwari, "Recent development in face recognition", *Neurocomputing*, c. 408, ss. 231-245, 2020.

[2] M. Z. Khan, S. Harous, S. U. Hassan, M. U. Ghani Khan, R. Iqbal ve S. Mumtaz, "Deep unified model for face recognition based on convolution neural network and edge computing," *IEEE Access*, c. 7, ss. 72622-72633, 2019.

[3] J. X. Mi, D. S. Huang, B. Wang, X. Zhu, "The nearest-farthest subspace classification for face recognition", *Neurocomputing*, c. 113, ss. 241-250, 2013.

[4] B. Li, C. H. Zheng, D. S. Huang, "Locally linear discriminant embedding: An efficient method for face recognition", *Pattern Recognit*, c. 41, sayı 12, 3813-3821, 2008.

[5] P. J. Thilaga, B. A. Khan, A. A. Jones ve N. K. Kumar, "Modern face recognition with deep learning," *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, India, 2018, ss. 1947-1951.

[6] L. Zuo, P. He, C. Zhang ve Z. Zhang, "A robust approach to reading recognition of pointer meters based on improved mask-RCNN", *Neurocomputing*, c. 388, ss. 90-101, 2020.

[7] K. Simonyan ve A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *The 3rd International Conference on Learning Representations (ICLR2015)*, CA, USA, 2015, ss. 1-14.

[8] A. Krizhevsky, I. Sutskever ve G. E. Hinton, "ImageNet classification with deep convolutional neural networks", *25th International Conference on Neural Information Processing Systems (NIPS 2012)*, Nevada, USA, 2012, ss. 1097-1105.

[9] S. Zhou, Q. Chen ve X. Wang, "Convolutional deep networks for visual data classification", *Neural Processing Letters*, c. 38, ss. 17-27, 2013.

[10] A. I. Kukharenko ve A. S. Konushin, "Simultaneous classification of several features of a person's appearance using a deep convolutional neural network", *Pattern Recognition and Image Analysis*, c. 25, ss. 461-465, 2015.

[11] Y. Hu, H. Chang, F. Nian, Y. Wang ve T. Li, "Dense crowd counting from still image with convolutional neural networks", *Journal of Visual Communication and Image Representation*, c. 38, ss. 530-539, 2016.

[12] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella ve J. Schmidhuber J, "Flexible, high performance convolutional neural networks for image classification", *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, Spain, 2011, ss. 1237-1242.

[13] M. Sarıgül, B. M. Ozyildirim ve M. Avci, "Differential convolutional neural network", *Neural Networks*, c. 116, ss. 279–287, 2019.

[14] B. B. Traorea, B. Kamsu-Foguema ve F. Tangara, "Deep convolution neural network for image recognition", *Ecological Informatics*, c. 48, ss. 257-268, 2018.

[15] Y. Seo, K. S. Shin, "Hierarchical convolutional neural networks for fashion image classification", *Expert Systems with Applications*, c.116, ss. 328–339, 2019.

[16] E. Cetinic, T. Lipic, ve S. Grgic, "Fine-tuning convolutional neural networks for fine art classification", *Expert Systems with Applications*, c. 114, 107–118, 2018.

[17] W. S. McCulloch ve W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, c. 5, sayı 4, ss. 115-133, 1943.

[18] G. Piccinini, "The first computational theory of mind and brain: a close look at Mcculloch and Pitts's "logical calculus of ideas immanent in nervous activity"", *Synthese*, c. 141, sayı 2, ss. 175-215, 2004.

[19] D. O. Hebb, "The organization of behavior; a neuropsychological theory", New York: Wiley, 1949.

[20] M. R. Aydın, "Yapay sinir ağları ile talep tahmini: Perakende sektöründe bir uygulama", *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, c. 18, sayı 35, ss. 43-55, 2019.

[21] S. Sabak ve Ö. Deniz Başar, "Enflasyonu etkileyen faktörlerin ülkeler bazında yapay sinir ağları analizi ile incelenmesi," *İstanbul Ticaret Üniversitesi Working Paper Series*, c. 1, sayı 1, ss. 27-38, 2020.

[22] A. Krenker, J. Bešter ve A. Kos, "Introduction to the artificial neural networks", içinde *Artificial Neural Networks - Methodological Advances and Biomedical Applications*, London: Intech Open, ss. 3-18, 2011.

[23] N. N. Urgan ve M. Tamgöz, "Yapay sinir ağları ile aktif kullanıcı sayısı tahmini üzerine bir uygulama", *European Journal of Engineering and Applied Sciences*, c. 3, sayı 2, ss. 8-14, 2020.

[24] R. Lippmann, "An introduction to computing with neural nets", *IEEE ASSP Magazine*, c. 4, sayı 2, ss. 4-22, 1987.

[25] M. D. Zeiler ve R. Fergus, "Visualizing and understanding convolutional networks", *European Conference on Computer Vision*, Switzerland, 2014, ss. 818-833.

[26] A. K. Jain, J. Mao ve K. M. Mohiuddin, "Artificial neural networks: a tutorial", *Computer*, c. 29, sayı 3, ss. 31-44, 1996.

[27] R. Eberhart ve J. Kennedy, "A new optimizer using particle swarm theory", *Proceedings of the Sixth International Symposium on Micro Machine and Human Science (MHS'95)*, Japan, 1995, ss. 39-43

[28] Z. Pan, A. G. Rustve H. Bolouri, "Image redundancy reduction for neural network classification using discrete cosine transforms", *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New, Challenges and Perspectives for the New Millennium*, Italy, 2000, ss. 149-154.

[29] A. Şentürk, Z. Karapınar Şentürk, "Yapay sinir ağları ile göğüs kanseri tahmini", *El-Cezerî Fen ve Mühendislik Dergisi*, c. 3, sayı 2, ss. 345-350, 2016.

[30] A. V. Savchenko, "Probabilistic neural network with complex exponential activation functions in image recognition", *IEEE Transactions on Neural Networks and Learning Systems*, c. 31, sayı 2, ss. 651-660, 2020.

[31] F. Agostinelli, M. Hoffman, P. Sadowski ve P. Baldi, "Learning activation functions to improve deep neural networks", *International Conference on Learning Representation*, CA, USA, 2015, ss. 1-9.

[32] P. Ramachandran, B. Zoph ve Q. V. Le, "Searching for activation functions", *International Conference on Learning Representation*, Canada, 2018, ss. 1-13.

[33] G. E. Hinton ve R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", *Science*, c. 313, sayı 5786, ss. 504-507, 2006.

[34] M. D. Zeiler ve R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks", *Computing Research Repository (CoRR)*, 2013, ss. 1-9.

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke ve A. Rabinovich, "Going deeper with convolutions", *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, MA, USA, 2015, ss. 1-9.

[36] Y. LeCun, Y. Bengio ve G. Hinton, "Deep learning", *Nature*, c. 521, sayı 7553, ss. 436-444, 2015.

[37] M. A. Shipp, K. N. Ross, P. Tamayo, A. P. Weng, J. L. Kutok, R. C. T. Aguiar, M. Gaasenbeek, M. Angelo, M. Reich, G. S. Pinkus, T. S. Ray, M. A. Koval, K. W. Last, A. Norton, T. A. Lister, J. Mesirov, D. S. Neuberg, E. S. Lander, J. C. Aster, T. R. Golub, "Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning", *Nature Medicine*, c. 8, sayı 1, ss. 68-74, 2002.

[38] T. Hastie, R. Tibshirani ve J. Friedman, "Unsupervised learning", içinde *The elements of statistical learning*, New York, NY: Springer, ss. 485-585, 2009.

[39] A. Şeker, B. Diri ve H. H. Balık. "Derin öğrenme yöntemleri ve uygulamaları hakkında bir inceleme", *Gazi Mühendislik Bilimleri Dergisi*, c. 3, sayı 3, ss. 47-64, 2017.

[40] F. Doğan ve İ. Türkoğlu, "Derin öğrenme modelleri ve uygulama alanlarına ilişkin bir derleme", *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, c. 10, sayı 2, ss. 409-445, 2019.

[41] N. Andrew, "Unsupervised feature learning and deep learning tutorial", [Online], 30 Mart 2021, Erişim: http://deeplearning.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/.

[42] R. Girshick, J. Donahue, T. Darrell ve J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", *2014 IEEE Conference on Computer Vision and Pattern Recognition*, OH, USA, 2014, ss. 580-587.

[43] S. Branson, G. V. Horn, S. Belongie ve P. Perona, "Bird species categorization using pose normalized deep convolutional nets", *Proceedings of the British Machine Vision Conference (BMVA) 2014*, University of Nottingham, UK, 2014, ss. 101.

[44] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition", *Proceedings of the 31st International Conference in Machine Learning (ICML)*, Beijing, China, 2014, ss. 647-655.

[45] B. Hariharan, P. Arbeláez, R. Girshick, ve J. Malik, "Simultaneous detection and segmentation" içinde *Computer Vision-ECCV 2014, Lecture Notes in Computer Science*, Springer, Cham, 2014, ss. 297-312.

[46] J. Tompson, A. Jain, Y. LeCun, ve C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation", *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*, Montreal, Canada, 2014, ss. 1799-1807.

[47] R. Girshick, F. Iandola, T. Darrell, J. Malik, "Deformable part models are convolutional neural networks", *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR) 2015*, Boston, MA, USA, 2015, ss. 437-446.

[48] D. Ravi, C. Wong, F. Deligianni, M. Berthelot, J. Andreu-Perez, B. Lo, G. Z. Yang, "Deep learning for health ınformatics", *IEEE Journal of Biomedical and Health Informatics*, c.21, sayı 1, ss. 4-21, 2017.

[49] K. Fukushima, ve S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition" içinde *Competition and Cooperation in Neural Nets Lecture Notes in Biomathematics*, vol 45, Berlin: Springer, ss. 267-285, 1982.

[50] K. Fukushima, "A neural network model for selective attention in visual pattern recognition", *Biological Cybernetics*, c. 55, sayı 1, ss.5-15, 1986.

[51] P. Y. Simard, D. Steinkraus ve J. C. Platt, "Best practices for convolutional neural networks applied to visual document analysis" *Seventh International Conference on Document Analysis and Recognition*, Scotland, 2003, ss. 958-963.

[52] D. H. Hubel ve T. N. Wiesel, "Receptive fields and functional architecture of monkey striate cortex", *The Journal of Physiology*, c. 195, sayı 1, ss.215-243, 1968.

[53] H. S. Koppula ve A. Saxena, "Anticipating human activities using object affordances for reactive robotic response", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, c. 38, sayı 1, ss. 14-29, 2016.

[54] MathWorks, "Convolutional neural network", [Online], 30 Mart 2021, Erişim: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html.

[55] Y. Lu, S. Yi, N. Zeng, Y. Liu ve Y. Zhang, "Identification of rice diseases using deep convolutional neural networks", *Neurocomputing*, c. 267, ss. 378-384, 2017.

[56] A. Krizhevsky, I. Sutskever ve G. E. Hinton, "Imagenet classification with deep convolutional neural networks", *Communications of the ACM*, c. 60, sayı 6, ss. 84-90, 2017.

[57] I. Goodfellow, Y. Bengio ve A. Courville, "Deep learning", Massachusetts, USA: MIT Press, 2016.

[58] D. Maturana ve S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition", *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Germany, 2015, ss. 922-928.

[59] F. Celesti, A. Celesti, J. Wan ve M. Villari, "Why deep learning is changing the way to approach NGS data processing: A review", *IEEE Reviews in Biomedical Engineering*, c. 11, ss. 68-76, 2018.

[60] S. A. Shah, H. Tan, G. Tinkhauser ve P. Brown, "Towards real-time, continuous decoding of gripping force from deep brain local field potentials", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, c. 26, sayı 7, ss. 1460-1468, 2018.

[61] G. -W. Kim ve D. -H. Lee, "Intelligent health diagnosis technique exploiting automatic ontology generation and web-based personal health record services", *IEEE Access*, c. 7, ss. 9419-9444, 2019.

[62] G. E. Hinton, S. Osindero, ve Y. W. Teh, "A fast learning algorithm for deep belief nets", *Neural Computation*, c. 18, sayı 7, ss. 1527-1554, 2006.

[63] K. He, X. Zhang, S. Ren ve J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, Nevada, 2016, ss. 770-778.

[64] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg ve L. Fei-Fei, "ImageNet large scale visual recognition challenge", *International Journal of Computer Vision*, c. 115, sayı 3, ss. 211-252, 2015.

[65] Q. Ji, J. Huang, W. He ve Y. Sun, "Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images", *Algorithms*, c. 12, sayı 3, ss. 51, 2019.

[66] E. Cengil ve A. Çınar, "A new approach for image classification: convolutional neural network", *European Journal of Technic*, c. 6, sayı 2, ss. 96-103, 2016.

[67] R. Gandhi, "R-CNN, fast R-CNN, faster R-CNN, YOLO - object detection algorithms", [Online], 30 Mart 2021, Erişim: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

[68] H. Jiang ve E. Learned-Miller, "Face detection with the faster R-CNN", *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, Washington, DC, USA, 2017, ss. 650-657.

[69] J. Li, D. Zhang, J. Zhang, J. Zhang, T. Li, Y. Xia, Q. Yan ve L. Xun, "Facial expression recognition with faster R-CNN", *Procedia Computer Science*, c. 107, sayı 2017, ss. 135-140, 2017.

[70] C. Zhang, X. Xu ve D. Tu, "Face detection using improved faster R-CNN", arXiv:1802.02142, doi: 10.48550/arXiv.1802.02142, 2018.

[71] L. Hao ve F. Jiang, "A new facial detection model based on the faster R-CNN", *IOP Conference Series: Materials Science and Engineering*, c. 439, sayı 3, ss. 032117, 2018.

[72] D. Garg, P. Goel, S. Pandya, A. Ganatra ve K. Kotecha, "A deep learning approach for face detection using YOLO", *1st International Conference on Data Science & Analytics Conference Proceedings (2018 IEEE Punecon)*, Pune, India, 2018, ss. 1-4.

[73] J. Cui, F. Chen, D. Shi ve L. Liu1 "Eye detection with faster R-CNN", içinde *Proceedings of the International Conference on Advances in Computer Technology, Information Science and Communications (CTISC 2019)*, Xiamen, China, 2019, ss. 111-116.

[74] C. Li, R. Wang, J. Li ve L. Fei, "Face detection based on YOLOv3", içinde V. Jain, S. Patnaik, F. Popențiu Vlădicescu ve I. K. Sethi (editörler), *Recent Trends in Intelligent Computing, Communication and Devices: Proceedings of ICCD 2018*, Singapore: Springer, c. 1031, ss. 277-284, 2020.

[75] R. Gyawali ve D. Raj Pant, "An approach for the employee face recognition by RPN and faster R-CNN techniques", *Proceedings of IOE Graduate Conference, 2019-Summer*, c. 6, sayı 5, 231-237, 2019.

[76] L. Hai ve H. Z. Gou, “Face detection with improved face R-CNN training method”, *3rd International Conference on Control and Computer Vision (ICCCV'20)*, Macau, China, 2020, ss. 22-25.

[77] K. Lin, H. Zhao, J. Lv, C. Li, X. Liu, R. Chen ve R. Zhao, “Face detection and segmentation based on improved mask R-CNN”, *Discrete Dynamics in Nature and Society*, c. 2020, ss. 1-11, 2020.

[78] L. Z. Chun, L. Dian, J. Y. Zhi, W. Jing ve C. Zhang, “YOLOv3: Face detection in complex environments”, *International Journal of Computational Intelligence Systems*, c. 13, sayı 1, ss. 1153-1160, 2020.

[79] S. Singh, U. Ahuja, M. Kumar, K. Kumar ve M. Sachdeva, “Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment”, *Multimedia Tools and Applications*, c. 80, sayı 13, ss. 19753-19768, 2021.

[80] L. Jiang, J. Chen, H. Todo, Z. Tang, S. Liu ve Y. Li, “Application of a fast R-CNN based on upper and lower layers in face recognition”, *Computational Intelligence and Neuroscience*, c. 2021, ss. 1-12, 2021.

[81] M. Loey, G. Manogaran, M. H. N. Taha, N. E. M. Khalifa, “Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection”, *Sustainable Cities and Society*, c. 65, sayı 2021, ss. 1-11, 2021.

[82] A. S. Sanchez-Moreno, J. Olivares-Mercado, A. Hernandez-Suarez, K. Toscano-Medina, G. Sanchez-Perez ve G. Benitez-Garcia, “Efficient face recognition system for operating in unconstrained environments”, *Journal of Imaging*, c. 7, sayı 9, ss. 161, 2021.

[83] S. M. M, A. Geroge, A. N ve J. James, “Custom face recognition using YOLO.V3”, *3rd International Conference on Signal Processing and Communication (ICPSC 2021)*, Coimbatore, Tamil Nadu, India, 2021, ss. 454-458.

[84] M. Muhaimin ve W. S. Tjong, “Real-time detection of face masked & face shield using YOLO algorithm with pre-trained model and DarkNet”, *Indonesian Journal of Artificial Intelligence and Data Mining*, c. 4, sayı 2, ss. 97-107, 2021.

[85] P. Jebraelzadeh, A. Charmin, H. Vahdati ve M. Ebadpour, “Providing a hybrid method for face detection and gender recognition using deep learning to improve accuracy”, *Journal of Xi'an University of Architecture & Technology*, c. 13, sayı 4, ss. 453-462, 2021.

[86] Y. Fan, Y. Luo ve X. Chen, “Research on face recognition technology based on improved YOLO deep convolution neural network”, *Journal of Physics: Conference Series*, c. 1982, 2nd International Conference on Artificial Intelligence and Information Systems (ICAIIS 2021), Chongqing, China, 2021, ss. 1-10.

[87] W. Chen, H. Huang, S. Peng, C. Zhou ve C. Zhang, “YOLO-face: a real-time face detector”, *The Visual Computer: International Journal of Computer Graphics*, c.37, sayı 4, ss. 805-813, 2021.

[88] G. Yolcu Öztel ve İ. Öztel, “A multi-task deep learning system for face detection and age group classification for masked faces”, *Sakarya University Journal of Science*, c. 25, sayı 6, ss. 1394-1407, 2021.

[89] Y. V. Kale, A. U. Shetty, Y. A. Patil, R. A. Patil ve D. V. Medhane, “Object detection and face recognition using yolo and inception model”, içinde I. Woungang, S. K. Dhurandher, K. K. Pattanaik, A. Verma, P., Verma (editörler), *Advanced Network Technologies and Intelligent Computing (ANTIC 2021), Communications in Computer and Information Science*, c. 1534, ss. 274-287, 2022.

[90] T. H. Tsai, ve P. T. Chi, “A single-stage face detection and face recognition deep neural network based on feature pyramid and triplet loss”, *IET Image Processing*, c. 16, sayı 8, ss. 2148-2156, 2022.

[91] A. Arı ve D. Hanbay, "Tumor detection in MR images of regional convolutional neural networks", *Journal of the Faculty of Engineering and Architecture of Gazi University*, c. 34, sayı 3, ss. 1395-1408, 2019.

[92] Ö. İnik ve E. Ülker, "Derin öğrenme ve görüntü analizinde kullanılan derin öğrenme modelleri", *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, c. 6, sayı 3, ss. 85-104, 2017.

[93] C. L. Zitnick, ve P. Dollar, "Edge boxes: locating object proposals from edges" içinde *Computer Vision-ECCV 2014, Lecture Notes in Computer Science*, Springer, Cham, ss. 391-405, 2014.

[94] R. Girshick, "Fast R-CNN," *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2015, ss. 1440-1448.

[95] Yang, S., Fang, B., Tang, W., Wu, X., Qian, J., & Yang, W. (2017). Faster R-CNN based microscopic cell detection, In Security, Pattern Analysis, and Cybernetics (SPAC), Shenzhen, China, 345-350.

[96] S. Q. Ren, K. M. He, R. Girshick ve J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks", *IEEE Transactions on Pattern Analysis & Machine Intelligence*, c. 39, sayı 6, ss. 1137-1149, 2017.

[97] J. Redmon, S. Divvala, R. Girshick ve A. Farhadi, "You only look once: unified, real-time object detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, Nevada, 2016, ss. 779-788.

[98] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn ve A. Zisserman, "The pascal visual object classes challenge: A retrospective", *International Journal of Computer Vision*, c. 111, s. 1, ss. 98-136, 2015.

[99] M. Lin, Q. Chen ve S. Yan, "Network in network", *International Conference on Learning Representations (ICLR) 2014*, Banff, Canada, 2014, ss. 1-10.

[100] J. Redmon ve A. Farhadi, "YOLO9000: Better, faster, stronger", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, ss. 6517-6525.

[101] J. Redmon ve A. Farhadi, "YOLOv3: An incremental improvement", arXiv:1804.02767, doi: 10.48550/arXiv.1804.02767, 2018.

[102] R. Geng, Y. Ma ve W. Huang, "An improved helmet detection method for YOLOv3 on an unbalanced dataset", *2021 3rd International Conference on Advances in Computer Technology, Information Science and Communication (CTISC)*, Shanghai, China, 2021, ss. 328-332.

[103] A. Bochkovskiy, C. -Y. Wang ve H. -Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection", arXiv:2004.10934, doi: 10.48550/arXiv.2004.10934, 2020.

[104] T. -Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan ve S. Belongie, "Feature pyramid networks for object detection", *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, ss. 936-944.

[105] S. Liu, L. Qi, H. Qin, J. Shi ve J. Jia, "Path aggregation network for instance segmentation", *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2018, ss. 8759-8768.

[106] M. Tan, R. Pang ve Q. V. Le, "EfficientDet: Scalable and efficient object detection", *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 2020, ss. 10778-10787.

[107] G. Ghiasi, T. -Y. Lin ve Q. V. Le, "NAS-FPN: Learning scalable feature pyramid architecture for object detection", *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 7029-7038.

[108] V. H. Kamble ve M. P. Dale, "Machine learning approach for longitudinal face recognition of children", içinde P. P. Sarangi, M. Panda, S. Mishra, B. S. P. Mishra, B. Majhi (editörler), *Machine Learning for Biometrics: Concepts, Algorithms and Applications, A volume in Cognitive Data Science in Sustainable Computing*, India: Elsevier, Academic Press, ss. 1-27, 2022.

[109] J. Han, M. Kamber ve J. Pei, "Classification: Basic concepts, içinde *Data Mining: Concepts and Techniques (Third Edition)* In The Morgan Kaufmann Series in Data Management Systems, USA: Elsevier, Morgan Kaufmann, ss. 327-391, 2012.

[110] J. D. Domingo, R. M. Aparicio ve L. M. G. Rodrigo, "Cross validation voting for improving CNN classification in grocery products", *IEEE Access*, c. 10, ss. 20913-20925, 2022.

[111] T. Hastie, R. Tibshirani ve J. Friedman, "Cross-validation", içinde *The elements of statistical learning*, New York, NY: Springer, ss. 241-248, 2009.

[112] D. Berrar, "Cross-validation", *Encyclopedia of Bioinformatics and Computational Biology*, c. 1, sayı 1, ss. 542-545, 2019.

[113] M. Kuhn ve K. Johnson, "Applied predictive modeling", New York: Springer, 2013.

[114] G. James, D. Witten, T. Hastie ve R. Tibshirani, "An introduction to statistical learning", New York: Springer, 2013.

[115] X. Li ve J. Sun, "Facial emotion recognition via stationary wavelet entropy and particle swarm optimization", içinde Y. -D. Zhang, A. K. Sangaiah (editörler) *Cognitive Systems and Signal Processing in Image Processing, A volume in Cognitive Data Science in Sustainable Computing*, India: Elsevier, Academic Press, ss. 145-162, 2022.

[116] A. Degerli, M. Yamac, M. Ahishali, S. Kiranyaz ve M. Gabbouj, "Medical image analysis", içinde A. Iosifidis, A. Tefas (editörler) *Deep Learning for Robot Perception and Cognition*, India: Elsevier, Academic Press, ss. 541-577, 2022.

[117] M. Kumar ve S.K. Rath, "Feature selection and classification of microarray data using machine learning techniques", içinde Q. N. Tran, H. R. Arabnia (editörler) *Emerging Trends in Applications and Infrastructures for Computational Biology, Bioinformatics, and Systems Biology: Systems and Applications, A volume in Emerging Trends in Computer Science and Applied Computing*, USA: Elsevier, Morgan Kaufmann, ss. 213-242, 2016.

[118] S. K. Zhou, A. Wimmer ve B.M. Kelm, "Spine disk and RIB centerline parsing", içinde *Medical Image Recognition, Segmentation and Parsing: Machine Learning and Multiple Object Approaches, A volume in The Elsevier and MICCAI Society Book Series*, USA: Elsevier, Academic Press, ss. 405-424, 2016.

[119] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection", içinde *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, c. 2, Morgan Kaufmann Publishers Inc., ss. 1137-1143, 1995.

[120] A. M. Molinaro, R. Simon ve R. M. Pfeiffer, "Prediction error estimation: a comparison of resampling methods", *Bioinformatics*, c. 21, sayı 15, ss. 3301-3307, 2005.

[121] R. Simon, "Resampling strategies for model assessment and selection", içinde W. Dubitzky, M. Granzow, D. Berrar (editörler), *Fundamentals of Data Mining in Genomics and Proteomics*, Boston, MA: Springer, ss. 173-186, 2007.

[122] P. Radhakrishnan, "What are hyperparameters? and How to tune the hyperparameters in a deep neural network?", [Online], 30 Mayıs 2022, Erişim: https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a

[123] S. Altun, M. F. Talu, "Derin sinir ağları için hiperparametre metodlarının ve kitlerinin incelenmesi", *DÜMF Mühendislik Dergisi*, c. 12, sayı 2, ss. 187-199, 2021.

[124] T. Yu, ve H. Zhu, "Hyper-parameter optimization: a review of algorithms and applications", arXiv:2003.05689, doi: 10.48550/arXiv.2003.05689, 2020.

[125] M. Feurer, ve F. Hutter, "Hyperparameter optimization", içinde F. Hutter, L. Kotthoff, J. Vanschoren (editörler) *Automated Machine Learning: Methods, Systems, Challenges*, Cham, Switzerland: Springer, ss. 3-33, 2019.

[126] J. S. Bergstra, R. Bardenet, Y. Bengio, ve B. Kégl, "Algorithms for hyper-parameter optimization", içinde J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira ve K. Q. Weinberger (editörler) *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, ss. 2546-2554, 2011.

[127] I. Goodfellow, Y. Bengio ve A. Courville, "Optimization for training deep models", içinde *Deep Learning*, MIT Press, ss. 271-325, 2016.

[128] I. Goodfellow, Y. Bengio ve A. Courville, "Convolutional networks", içinde *Deep Learning*, MIT Press, ss. 326-366, 2016.

[129] L. N. Smith, "Cyclical learning rates for training neural networks", *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Santa Rosa, CA, 2017, ss. 464-472.

[130] H. Zulkifli, "Understanding learning rates and how it improves performance in deep learning", [Online], 30 Mayıs 2022, Erişim: https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10.

[131] E. Seyyarer, F. Ayata, T. Uçkan ve A. Karcı, "Derin öğrenmede kullanılan optimizasyon algoritmalarının uygulanması ve kıyaslanması", *Anatolian Journal of Computer Sciences*, c. 5, sayı 2, ss. 90-98, 2020.

[132] S. Sharma, "Epoch vs batch size vs iterations", [Online], 30 Mayıs 2022, Erişim: https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9.

[133] F. C. Soon, H. Y. Khaw, J. H. Chuah ve J. Kanesan, "Hyper-parameters optimisation of deep CNN architecture for vehicle logo recognition", *IET Intelligent Transport Systems*, c. 12, sayı 8, ss. 939-946, 2018.

[134] L. Yang ve A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice", *Neurocomputing*, c. 415, ss. 295-316, 2020.

[135] S. Hajian, "Transfer of learning and teaching: a review of transfer theories and effective instructional practices", *IAFOR Journal of Education*, c. 7, sayı 1, ss. 93-111, 2019.

[136] K. Fırıldak, M. F. Talu, “Investigation of transfer learning approaches used convolution neural networks”, *Anatolian Journal of Computer Science*, c. 4, sayı 2, ss. 88-95, 2019.

[137] MathWorks, "Get started with transfer learning", [Online], 30 Mayıs 2022, Erişim: https://www.mathworks.com/help/deeplearning/gs/get-started-with-transfer-learning.html.

[138] M. Hussain, J. J. Bird ve D.R. Faria, “A study on CNN transfer learning for image classification”, içinde A. Lotfi, H. Bouchachia, C. Langensiepen ve M. McGinnity (editörler), *UK Workshop on Computational Intelligence (UKCI 2018) Advances in Computational Intelligence Systems*, c. 840, ss. 191-202, 2018.

[139] R. Levin, V. Cherepanova, A. Schwarzschild, A. Bansal, C. B. Bruss, T. Goldstein, A. G. Wilson ve M. Goldblum, “Transfer learning with deep tabular models”, arXiv:2206.15306, doi: 10.48550/arXiv.2206.15306, 2022.

[140] J. Yosinski, J. Clune, Y. Bengio ve H. Lipson, “How transferable are features in deep neural networks?”, *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*, c. 2, ss. 3320-3328, 2014.

[141] MathWorks, "Transfer learning for training deep learning models", [Online], 30 Mayıs 2022, Erişim: https://www.mathworks.com/discovery/transfer-learning.html.

[142] S. Akcay, M. E. Kundegorski, M. Devereux ve T. P. Breckon, “Transfer learning using convolutional neural networks for object classification within X-ray baggage security imagery”, *2016 IEEE International Conference on Image Processing (ICIP)*, Arizona, USA, 2016, ss. 1057-1061.

[143] Z. Huang, Z. Pan ve B. Lei, “Transfer learning with deep convolutional neural network for sar target classification with limited labeled data”, *Remote Sensing*, c. 9, sayı 9, ss. 907, 2017.

[144] D. Sarkar, “A comprehensive hands-on guide to transfer learning with real-world applications in deep learning”, [Online], 30 Mayıs 2022, Erişim: https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a.

[145] P. Marcelino, “Transfer learning from pre-trained models”, [Online], 30 Mayıs 2022, Erişim: https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751.

[146] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong ve Q. He, “A comprehensive survey on transfer learning”, arXiv:1911.02685, doi: 10.48550/arXiv.1911.02685, 2019.

[147] C. Goutte ve E. Gaussier, "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation" içinde *Advances in Information Retrieval. ECIR 2005. Lecture Notes in Computer Science*, Berlin: Springer, ss. 345-359, 2005.

[148] M. Zhu, "Recall, precision and average precision", Department of Statistics and Actuarial Science Technical Report, University of Waterloo, Waterloo, 2004.

[149] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn ve A. Zisserman, "The PASCAL visual object classes (VOC) challenge", *International Journal of Computer Vision*, c. 88, sayı 2, ss. 303-338, 2010.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Demet Hanife SUNGUR

Yabancı Dili : İngilizce

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Lisans	Elektronik ve Bilgisayar Eğitimi Bölümü	Fırat Üniversitesi/Teknik Eğitim Fakültesi	2008
Lise	Bilgisayar	Zonguldak Anadolu Kız Teknik ve Meslek Lisesi	2003

YAYINLAR

1. D. H. Sungur and P. Erdoğmuş, "Face recognition with region based convolutional neural network", *1st International Congress on Artificial Intelligence and Data Science Proceeding Book*, ss. 81-86, 2021.