KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

DERİN ÖĞRENME YÖNTEMİ İLE ARAÇ VE PLAKA TANIMA

SERKAN KIRCA

KOCAELİ 2021

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

DERİN ÖĞRENME YÖNTEMİ İLE ARAÇ VE PLAKA TANIMA

SERKAN KIRCA

Doç.Dr. Hikmet Hakan GÜREL		
Danışman,	**Kocaeli Üniv.**	**.....**
Doç.Dr. Serdar SOLAK		
Jüri Üyesi,	**Kocaeli Üniv.**	**.....**
Dr.Öğr. Üyesi Süha TUNA		
Jüri Üyesi,	**Fatih Sultan Mehmet Vakıf Üniv.**	**.....**

Tezin Savunulduğu Tarih: 14.06.2021

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışmasında, görüntü üzerinde araç ve plaka tanıma çalışması yapılmıştır. Yapılan çalışmada kullanılan veriler sahadan gerçek veri toplanarak yapılmıştır. Derin öğrenme algoritması ile görüntü üzerinde plaka ve araç modeli tespiti yapılmıştır.

Tez çalışmamda desteğini esirgemeyen, çalışmalarıma yön veren, bana güvenen ve yüreklendiren danışmanım Doç. Dr. Hikmet Hakan GÜREL ’e sonsuz teşekkürlerimi sunarım.

Yüksek lisans öğrenimim boyunca, birçok aşamada beni destekleyen ve desteğini esirgemeyen, ailem olarak gördüğüm Anadolu Anonim Türk Sigorta Şirketi’ne teşekkürlerimi sunarım.

Akademik çalışmalarım sırasında, birçok aşamada beni destekleyen ve desteğini esirgemeyen sevgili yöneticim Vedat GÜNEŞ ’e teşekkürlerimi sunarım.

Fen Bilimleri Enstitüsü’ndeki çalışma hayatım boyunca, üzerimdeki emekleri için minnettar olduğum Fen Bilimleri Enstitüsü çalışanlarına teşekkürü borç bilirim.

Hayatım boyunca bana güç veren en büyük destekçim, her aşamada sıkıntılarımı ve mutluluklarımı paylaşan sevgili eşim Elif AYDIN KIRCA ’ya teşekkürlerimi sunarım.

Mayıs – 2021 Serkan KIRCA

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

AUC : Area Under Curve (Eğri Altında Kalan Alan)
CNN : Convolutional Neural Network (Evrişimli Sinir Ağı)
Fast R-CNN : Fast Region Based Convolutional Neural Network (Hızlı Bölge Bazlı Evrişimli Sinir Ağı)
Faster R-CNN : Faster Region Based Convolutional Neural Network (Daha Hızlı Bölge Evrişimli Sinir Ağı)
OCR : Optical Character Recognition (Optik Karakter Tanıma)
R-CNN : Region Based Convolutional Neural Network (Bölge Bazlı Evrişimli Sinir Ağı)
ROC : Receiver Operator Characteristics (Alıcı Operatör Özellikleri)
SSD : Single Shut Multiple Detector (Çoklu Kutu Algılama)
YOLO : You Only Look Ones (Sadece Bir Kez Bak)

DERİN ÖĞRENME YÖNTEMİ İLE ARAÇ VE PLAKA TANIMA

ÖZET

Yapılan çalışmada, görüntü üzerinden araç ve plaka tanıma çalışması yapılmıştır. Bu çalışmada kullanılan veriler sahadan toplanan gerçek veriler olup, model eğitim ve test işlemi bu veriler üzerinden gerçekleşmiştir.

Öncelikle, piyasada mevcut birçok araç ve plaka tanıma çalışması yapılmıştır ve yapılan çalışmalarda özellikle araç tanıma kısmında etiketlenen resimler aracın tamamını kapsamaktadır.

Literatürde yapılan çalışmalarda, araç tanıma için her marka ve model araçlar veri setinde bulunmak zorundadır. Örneğin son model bir araç ile aynı markaya ait on yaşında bir araç veri setinde ayrı ayrı bulunmak zorundadır. Yine herhangi bir model için binek (hususi oto) ve binek olmayan (otobüs, truck, kamyon... vb.) markaların her birinin ayrı ayrı veri setinde bulunmak zorundadır. Eğer herhangi biri veri setinde bulunmaz ise o modeli tespit edemez. Yine literatürde yapılan çalışmada, gelecek sene çıkacak yeni modellerde kasa değişimi olacağı için oluşturulan nesne tespit modeli bu araçları da görüntü üzerinden tespit edemeyecektir. Bundan dolayıdır ki yeni modeller piyasaya çıktığında bu modellerle tekrardan bir eğitim aşaması söz konusu olup nesne tespit modelinin güncellenmesi gerekecektir.

Bu çalışmada ise yukarıda bahsettiğimiz handikapların hiçbiri bulunmamaktadır. Bu çalışmada araç tanıma için yapılan yaklaşımda araçların amblemleri üzerinden araç tanıma yapılmaktadır. Bunun nedeni trafik halinde iken araç donanımında bulunan kameralar sayesinde diğer araçların marka ve plakalarını acil durumda tespit edilmesini sağlayarak, ilgili aracın bilgilerine ivedi bir şekilde ulaşılması amaçlanmaktadır.

Yapılan çalışmada sahadan araç verileri toplanmış olup bu veriler üzerinde etiketleme çalışması yapılarak görüntüde bulunan araçlar üzerindeki amblemler ve araçların plakaları üzerinden bir araç ve plaka tanıma çalışması yapılmıştır.

Anahtar Kelimeler: Araç Tanıma, Derin Öğrenme, Görüntü İşleme, Plaka Tanıma, YOLO.

VEHICLE AND LICENCE PLATE RECOGNITION

ABSTRACT

In the study, vehicle and license plate recognition studies were carried out on the image. The data used in this study are real data collected from the field, and the model training and testing process was carried out on these data.

First of all, many vehicle and license plate recognition studies available in the market have been carried out, and the pictures tagged especially in the vehicle recognition section cover the entire vehicle.

Studies in the literature, every brand and model of vehicles must be included in the data set for vehicle identification. For example, a latest model vehicle and a ten year old vehicle of the same brand must be included in the dataset separately. Again, for any model, each of the passenger (private car) and non-passenger (bus, truck, truck ... etc.) brands must be in the data set separately. If any of them are not found in the data set, it cannot detect that model. Again, in the study conducted in the literature, the object detection model created will not be able to detect these vehicles over the image, since there will be a case change in the new models to be released next year. Therefore, when new models are released, there is a training phase with these models and the object detection model will need to be updated.

In this study, however, none of the above mentioned handicaps are present. In this study, in the approach made for vehicle identification, vehicle identification is done through the emblems of the vehicles. The reason for this is to ensure that the brands and license plates of other vehicles are detected in an emergency, thanks to the cameras in the vehicle equipment while in traffic, and it is aimed to reach the information of the relevant vehicle immediately.

In the study, vehicle data were collected from the field, and a vehicle and license plate recognition study was carried out on the emblems and license plates of the vehicles in the image by tagging work on these data.

Keywords: Vehicle Recognition, Deep Learning, Image Processing, License Plate Recognition, YOLO.

GİRİŞ

Teknolojinin çok hızlı gelişmesi sayesinde günümüzde artık insanların yerini insanlar gibi düşünme yeteneği kazandırılan yapay zekâ destekli makineler almaktadır. İnsanın doğumundan başlayan ve hayatı boyunca devam eden bir öğrenme dönemi vardır. Bu öğrenme dönemi boyunca insan, çevresindeki olayları gözlemler ve analiz eder. Kısaca çevresindeki durumlara karşı nasıl aksiyon alacağını öğrenmiş olur. Artık insanlara özgü öğrenme işini makinelere de yaptırıyoruz. Makinelere bu tarz öğretme yaptırabilmenin neticesinde ortaya yapay zekâ diye adlandırdığımız içerisinde alt kümeleri olan bir küme ortaya çıkmıştır [1]. Aslında yapay zekâ günümüzde ortaya çıkan bir kavram değildir. Bu kavram 19. YY ortalarında İngiliz matematikçi Alan Mathison Turing tarafından geliştirilmiş olan turing test makinaları tarafından, bilgisayarlara düşünme yetisini kazandırmayı amaçlamış ve II. Dünya Savaşı sırasında Alman şifrelerinin kırılmasını sağlayarak bilgisayarlara (Enigma) düşünme yeteneğini kazandırmada ilk adımı atmıştır [2].

Yapay zekâ kendi içerisinde; uzman sistemler, makine öğrenmesi ve derin öğrenme olarak adlandırılan ve sırasıyla iç içe geçmiş, 3 alt kümeden oluşmaktadır.

Bu alt kümelerden biri olan derin öğrenme, genellikle görüntü üzerinde sınıflandırma problemleri için kullanılan algoritmadır. Derin öğrenme makine öğrenmesinin bir alt dalı olup, özellikleri yönünden makine öğrenmesinden farklıdır [3]. En basit anlamda makine öğrenmesi, belli algoritmalardan oluşur ve bu algoritmaların düzgün çalışabilmesi için algoritmaya verilen girdi verilerinin öncelikle öznitelik mühendisliği olarak adlandırılan ön işlemden geçirilmesi gerekir ki, burada algoritma belirlenen özellikler üzerinden bir öğrenme yapmış olsun. Ama derin öğrenmede herhangi bir öznitelik mühendisliğine gerek yoktur. Derin öğrenme verilen özelliklerden hangisini seçeceğine kendisi karar verir ve öğrenme işlemini o özellikler üzerinden yapar.

Diğer bir taraftan bakıldığında, derin öğrenme kullanımında özellik mühendisliğine gerek duyulmaması zaman açısından büyük fayda sağlamaktadır.

Derin öğrenme yöntemleri kullanılarak farklı konular üzerinde literatürde çok sayıda çalışmalar yapılmıştır. Görüntü işleme [4], ses işleme [5], metin işleme [6] yapılan bu çalışmalardan en popülerleridir. Derin öğrenme ile görüntü işleme yöntemi kullanılarak insansız hava araçlarından silah tespiti [7], ses işleme ile alakalı olarak ses tanıma [8], metin işleme ile alakalı olarak ise yazılan eserin kime ait olduğu ile alakalı [9] çalışmalar yapılmış olup, bu çalışmalara ek literatürde farklı çalışmalarda mevcuttur [9-10].

Bu tez kapsamında ise derin öğrenme teknikleri ile görüntü üzerinden araç ve plaka tanıma çalışması yapılmaktadır. Literatürde bu tarz çalışmalar bulunmakla birlikte farklı uygulama alanları da vardır [11]. Yapılan çalışmalarda özellikle araç tanıma kısmında, aracın tüm resmi etiketlenerek eğitim verisi oluşturulmuş ve aracın tüm resmi üzerinden araç tanıma yapılmıştır [12].

Tüm resim üzerinde aracın tam fotoğrafı ile araç sınıflandırma yönteminde, en büyük dezavantajlardan biri arabaların tüm modellerinin geçmiş ve gelecekte tespit edecek bir veri seti oluşturup oluşturulan sınıflandırma modelini tekrar tekrar eğitime sokmaktır. Ayrıca aracın önden ve arkadan görüntüleri farklı olduğundan her bir marka ve modelin hem önden hem arkadan resimleri ile birlikte veri seti oluşturulmaktadır.

Günümüzde araba firmaları her sene olmasa bile sıklıkla marka ve modelin kasasını değiştirmektedirler. Bu sebepten dolayı, ortaya çıkan her yeni kasa için literatürdeki modeli optimize etme problemi sürekli olarak karşılaşılan en önemli sorunlardandır.

Ayrıca bu tez çalışmasında ilgili, nesne tespiti yapılacak olan aracın trafikte seyir halinde olduğu düşünülmüş olup, seyir halinde hareket eden bir aracın görüntüsü üzerinden araç ve plaka tanıma işlemi yapılacaktır. Aksi halde araç markasının ve plakasının duvara asılı olduğu herhangi bir yerde de model sanki orada araç varmış gibi davranıp görüntü üzerinde yine araç ve plaka tanıma işlemini gerçekleştirecektir. Bu tarz dezavantajları önleme adına, aracın farklı yerlerden görüntülerinin olduğu resimlerde eğitim setine ilave edilebilir. Proje kapsamında araç marka ambleminin ve plakanın araç üzerinde olduğu varsayılarak proje çalışması yapılmıştır.

Yapılan çalışmalarda bahsedilen dezavantajların dışında aracın tüm resmi eğitim veri setinde kullanıldığından ayrıca aracın rengi ile alakalı etiketleme de yaparak görüntü üzerinde aracın hem marka hem hangi renk olduğu bilgisi tespit edilebilir.

Bu tez kapsamında yapılacak olan çalışmada ise mevcut çalışmalarda kullanılan sınıflandırma yönteminde araç tanıma için farklı bir yöntem önerilmiş ve ilk defa kullanılmış olup, eğitim ve test verileri ile denemiş ve istenilen performans yakalanmıştır. Yapılan çalışmada aracın tüm resmi üzerinden sınıflandırma yapmak yerine araç marka amblemi üzerinden bir sınıflandırma işlemi yapılarak, diğer çalışmalardaki dezavantaj ortadan giderilmiştir. Böylelikle ilgili markada firma üst düzey yöneticileri çok radikal bir karar almadığı sürece (araç markasının ambleminin değişmesi gibi) ilgili resim üzerinde aracın hangi model hangi kasa olduğuna bakılmaksızın sınıflandırma işlemi başarılı bir şekilde yapılmaktadır.

Günümüzde plaka tanıma yöntemlerinde ise farklı görüntü işleme teknikleri kullanılmaktadır. Bu tekniklere ek olarak artık görüntü verilerinin fazla olması sayesinde derin öğrenme yöntemleri kullanılarak da başarılı plaka tanıma yöntemleri geliştirilebilmektedir.

Herhangi bir resim üzerinde nesne tespiti yapılması gibi, resim veya video üzerinde bulunan plakalarında aynı mantık ve algoritma çerçevesinde tespitinin yapılması hem kolay hem de hızlı olmaktadır.

Yapılan çalışmada, araç plakası da sınıf olarak veri setine eklenmiş ve görüntü üzerinde hem araç amblemi kullanarak araç ve plaka tanıma işlemi aynı anda geçekleştirilmektedir.

1. DERİN ÖĞRENME

Giriş kısmında bahsedildiği üzere yapay zekânın alt kümelerinden biri olan derin öğrenme günümüzde popüler olmuş bir konu olmakla birlikte, bilgisayarlardaki performansların artması, grafik işlemci birimlerinin (GPU) işlemlerde kullanılarak yüksek ve hızlı performans kazanımı sayesinde [13] ve çok fazla verinin bulunmasından kaynaklı 21.YY'a damgasını vurmuştur.

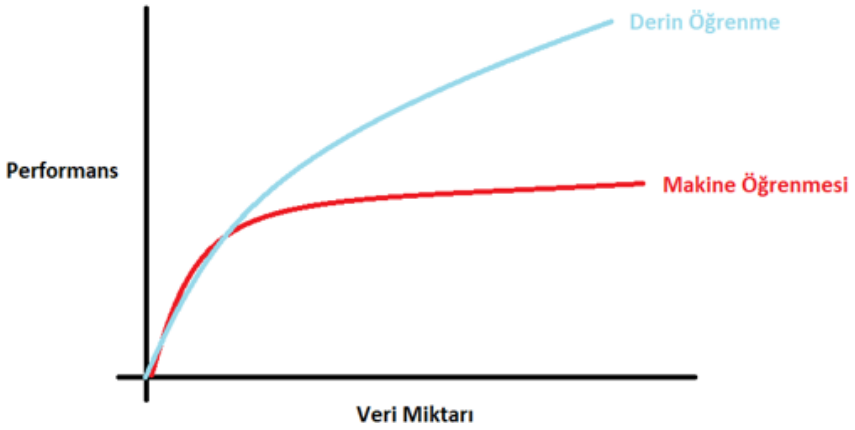
Makine öğrenmesi kendi içerisindeki algoritmalardan yola çıkarak farklı problemlerde farklı değerlendirme yapacak algoritmalara sahip olmaktadır. Makine öğrenmesi algoritmaları, ilgili algoritmaya değişkenlerin verilmeden önce ön işlemden geçirilerek, farklı analizler sonucunda tahmin edilecek olan bağımlı değişkenin hangi bağımsız değişkenler üzerinden etkilendiği tespit edilip diğer bağımsız değişkenler veriden atılarak, geriye kalan bağımsız değişkenler üzerinden, bağımlı değişkeni tahmin etme gibi bir yaklaşıma sahiptir.

Makine öğrenmesi alt kümesi olan derin öğrenmede ise ilgili ön işlem yani bağımsız değişken seçimi gibi bir işlem bulunmamaktadır. Derin öğrenme tamamen insan beyninden ilham alınarak tasarlanmış olup, insan vücudundaki iletim işlemleri nasılsa o temel alınarak adeta insan vücudu taklit edilmiştir.

Derin öğrenme algoritmaları için en önemli kaynak verinin fazla olmasıdır. Veri fazlalığı makine öğrenmesi yöntemlerinde belli bir miktardan sonra eğitim ve modelleme kısmında başarı değerleri için pek fazla bir değişikliğe neden olmamaktadır [14]. Bundan dolayıdır ki derin öğrenme çok fazla veri kullanarak yüksek başarı değerleri ortaya çıkarmaktadır.

1.1. Derin Öğrenme ile Makine Öğrenmesi Arasındaki Fark

Derin öğrenme makine öğrenmesinin bir alt dalıdır ama derin öğrenme özellikleri doğrudan veriden öğrenir. Yani veriden hiçbir öznitelik çıkarımı yapmadan model kurulur. Özellikle son yıllarda veri sayımızın artmasıyla birlikte, klasik makine öğrenmesi modelleri performans açısından iyi sonuçlar vermemeye başladı (Şekil 1.1).



Şekil 1.1. Derin Öğrenme İle Makine Öğrenmesi Arasındaki Fark

Şekil 1.1 yorumlanacak olursa; belli miktarda veri sayısına kadar derin öğrenme ile makine öğrenmesi hemen hemen aynı performansı sergiliyor denilebilir. Ama veri sayısı çok fazla arttığında artık makine öğrenmesi teknikleri performans açısından tatminkâr sonuçlar vermiyor (düşük accuracy gibi). Bundan dolayı veri sayısının çok fazla olduğu yerlerde derin öğrenme kullanıldığında algoritma güzel sonuçlar veriyor demektedir. (yüksek accuracy gibi).

Diğer bir taraftan her zaman verinin çok olması iyi modeller yapılacak olduğu anlamına gelmemektedir. Asıl önemli olan veri setinin fazlalığının yanında, verinin temiz ve düzenli olmasıdır. Veriler düzenli ve ya temiz olmadıktan sonra eğitim yapılacak olan modelin başarımı, veri sayısı fazla olsa bile artış göstermeyecektir.

Bu sebeplerden dolayı verinin toplanması temizlenmesi ve model için hazır hale getirilmesi yani veri manipülasyonu süreçleri, çok fazla zahmet ve emek gerektirmekle birlikte, bu verilerin hazır hale getirilmesi ise uzun zamanlar almaktadır. Model oluşturulurken özellikle görüntü üzerinde farklı zamanda ve kirli olan görüntü verileride eğitim setinde bulunmalı ve model bu görüntü kalitesi düşük resimler üzerinde de nesne tanıma yapabilmelidir.

Derin öğrenmenin makine öğrenmesinden farkı özet bir şekilde tablo 1.1 ’de gösterildiği gibi olup, bu tablo daha da zenginleştirilebilir [15]:

Tablo 1.1. Makine Öğrenmesi ile Derin Öğrenme Arasındaki Fark

	Makine Öğrenmesi	**Derin Öğrenme**
Veri Sayısı	Küçük miktarda veri kullanılabilir.	Büyük miktarlarda veri kullanmak gereklidir
Donanım	Düşük kaliteli makinelerde çalışabilir. Büyük miktarda hesaplama gücüne gerek yoktur	Yüksek kaliteli makinelere bağlıdır. Çok sayıda matris çarpma işlemi yapar. GPU, bu işlemleri verimli bir şekilde iyileştirebilir
Öznitelik Seçimi	Özelliklerin doğru şekilde tanımlanması ve oluşturulması gerekir.	Verilerden yüksek düzey özellikleri öğrenir ve kendi kendine yeni özellikler oluşturur
Öğrenme Yaklaşımı	Öğrenme işlemini daha küçük adımlara böler. Ardından, her adımın sonuçlarını tek bir çıktıda birleştirir	Sorunu, bir uçtan uca temelinde çözerek öğrenme süreci boyunca ilerleyerek.
Eğitim Süresi	Birkaç saniye ile birkaç saat arasında uyum sağlamak için karşılaştırmayı daha az zaman alır	Derin bir öğrenme algoritması birçok katman içerdiğinden, genellikle eğitmek katman sayısına bağlı olarak uzun sürer.
Çıktılar	Çıktı genellikle bir puan veya sınıflandırma gibi sayısal bir değerdir.	Çıktıda metin, puan veya ses gibi birden çok biçim olabilir

1.2. Ne Zaman Derin Öğrenme Kullanmalıyız

Geleneksel yöntemlerle işlenmesine olanak olmayan büyük hacimli verinin çok fazla olduğu durumlarda derin öğrenme kullanılmaktadır [26]. Fazla verinin olduğu yerlerde makine öğrenmesi yetersiz kalmaktadır. Bu sebepten yeterli miktarda veri varsa derin öğrenme yöntemi kullanılmalıdır.

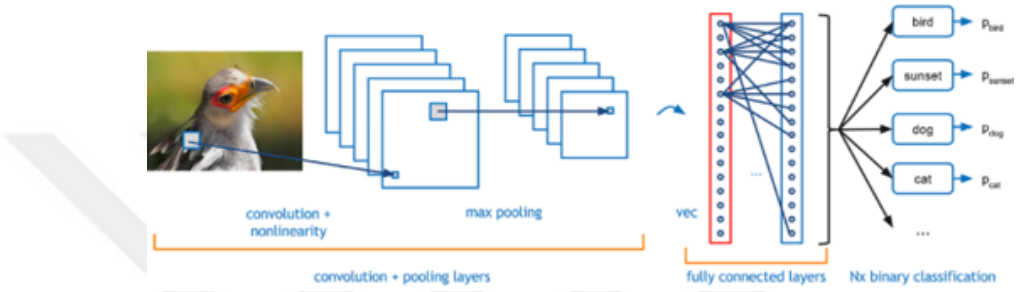
Bu özel çalışmalar;

- ✓ Ses Tanıma
- ✓ Görüntü İşleme
- ✓ Tavsiye Sistemleri
- ✓ Doğal Dil İşleme

gibi alanlarda da derin öğrenme sıklıkla kullanılmaktadır.

2. EVRİŞİMLİ SİNİR AĞLARI

Convolution Neural Network olarak adlandırılan evrişimli sinir ağları [16]; görüntü üzerinde sınıflandırma, nesne tespiti ve takibi problemlerini çözmek için özelleşmiş ağlardır. Evrişimli sinir ağlarında öncelikle girdi olarak resim alınır.



Şekil 2.1. Evrişimli Sinir Ağı Yapısı [27]

Şekil 2.1 'de örnek bir evrişimli sinir ağı görülmektedir. Bir resmi anlamlandırmak için resimle alakalı belli başlı özelliklerin (resimdeki kenarlar, yuvarlaklar, resimde bulunan kedinin kuyruğu, kulağı, gözü vb.) çıkartılması gereklidir. Sonrasında giriş resmine dair çıkartılan özellikleri kullanarak sınıflandırma yapmak gerekir. Öznitelik çıkarımı denilen kısımdan sinir ağına giriş olarak adlandırılan kısma kadar öz nitelik çıkarımı yapılır. Öznitelik çıkartma işlemi bittiğinde seyreltme dediğimiz yapay sinir ağının girişlerine özniteliği çıkartılmış olan değerler verilir. Bu sinir ağının sonucunda da öznitelikleri kullanarak bir çıktı üretilir. Çıktı sonucunda da ilgili resimde sınıflandırma yapılacaksa hangi sınıf olduğu bilgisi, nesne tespiti yapılacaksa ilgili nesnenin tespit edilip edilememesi gibi ilgili probleme göre (sınıflandırma, nesne tespiti, nesne takibi gibi) sonuç gerçekleşir.

Öz nitelik diye adlandırılan şey aslında bir resmi ifade etmek için kullanılan özelliklerdir [17]. Görüntü işlemede öznitelik denilen kavramlar;

- ✓ Kenar
- ✓ Köşe
- ✓ Kenar ile köşenin birleşmesiyle oluşan başka bir şekil
- ✓ Düz çizgi

- ✓ Yan çizgi vb.

gibi kavramlar öz nitelik diye adlandırılır. Evrişim katmanı diye adlandırılan katman, resim üzerinden yukarıda bahsedilen şekilde öznitelikler çıkartır ve özellik haritalarını oluşturulur [18]. Öznitelik çıkartılırken özelliklerin boyutu azaldığından dolayı, resmin boyutunun orijinal boyuna çevirmek için piksel arttırma işlemi (same padding)

gerçekleştirilir. Sonra tekrar evrişim katmanı eklenir ki bunun nedeni yüksek seviye olarak adlandırılan evrişim işleminde daha karmaşık şekiller öznitelik olarak çıkartılır. İlk evrişim işlemi alçak seviye olarak adlandırıldığından ve burada öznitelik olarak resimlerle alakalı temel özellikleri (kenar, köşe, çizgi vb. gibi) tespit edildiğinden dolayı, evrişim katmanı eklenerek yüksek seviyelere doğru çıkılır. Bunun amacı resimlerle alakalı ayırt edici özelliği çıkartmaktır. Kısaca evrişim katmanı eklendikçe resimlerden elde edilen özelliklerin derinliği artmış olur.

Öz nitelikler ortaya çıktıktan sonra, elde edilen öz nitelikler yapay sinir ağının inputuna giriş olarak verilir. Dikkat edilmesi gereken husus sinir ağının inputuna verilen öz nitelikler tam bağlantılı (fully connected) olarak verilmektedir. Yani açıkta herhangi bir input bekleyecek nöron bırakılmamalıdır.

2.1. Evrişimsel Operasyon Nedir?

Evrişimli operasyon;

- ✓ Özellik algılayıcı, resmin niteliğini belirleyen özellikleri algılar. Örneğin, görüntü bir kedi ise; kedinin kuyruğu, gözü ve ya kulağı gibi özellikleri algılar
- ✓ Özellik haritası = Evrişim. (Matrislerin elemanlarının çarpımı)
- ✓ Belirli kurallara göre görüntü üzerinde gezinme gerçekleştirilir
- ✓ Yapılan işlemler sonucunda orijinal görüntünün boyutu azalmış oluyor (bu oluşturulan modelin hızlı çalışması açısından önemli)
- ✓ Birden çok özellik haritası oluşturulur. Bunun nedeni; birden çok özellik algılayıcı (filtre) kullanılır

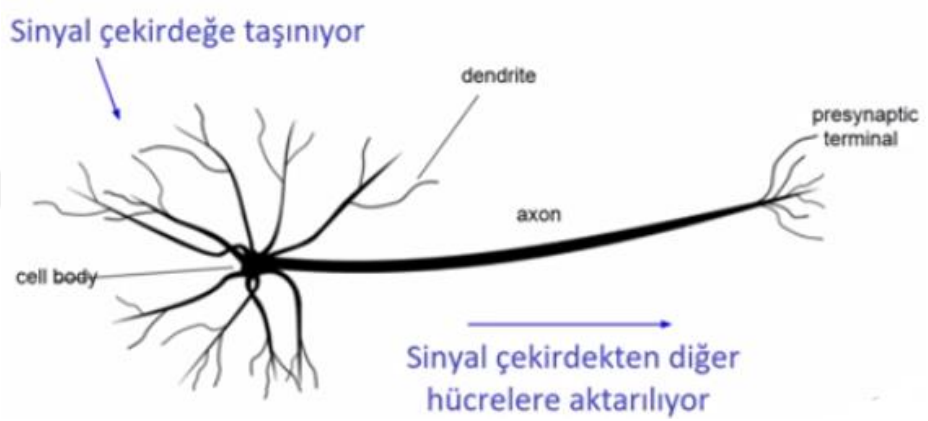
Yukarıda oluşturulan maddeler açıklanmaya çalışılırsa, ilk olarak özellik algılayıcı filtre; görüntü üzerinde herhangi bir kenar tespit edilmek isteniliyorsa, görüntüde kenarı tespit eden filtre, bir köpeğin kuyruğunu tespit etmek isteniliyorsa; köpeğin kuyruğunu tespit eden filtrelerdir. Bu filtreler evrişimli sinir ağlarında öğrenilir.

Filtreler resmi sınıflandırmak için evrişimli sinir ağlarından öğrenilen parametrelerdir. Derin öğrenme yöntemi ile en iyi filtreler belirlenmiş olmaktadır. Filtreler belirlendikten sonra bir özellik haritası ortaya çıkmaktadır.

Oluşturulan filtre görüntü üzerinde gezdirilerek; input resmi, boyutu azaltılmış bir resme dönüştürülür. Görüntü üzerindeki piksel değerleri ile filtre içerisindeki değerler tek tek çarpılıp sonrasında toplanarak tek bir değer elde edilir [19]. Filtre, bir kenarı, kuyruğu ya da herhangi bir cismin şeklini tespit ediyor olabilir. Kısaca kullanılan resim ne ise o resmi ayırt etmek için kullanılan bir filtredir.

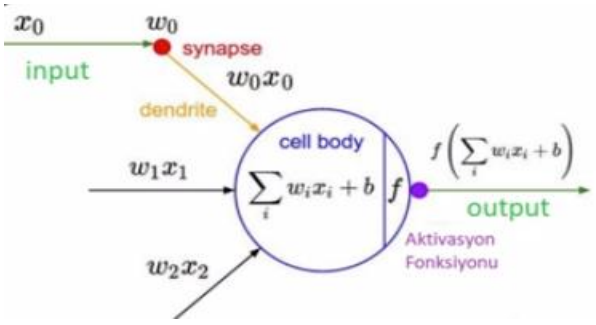
2.1.1. Yapay sinir ağları

Yapay sinir ağlarını anlamak için sinir ağları ve nöronları iyi anlamak gereklidir. Yapay sinir ağlarında bu ikiliden sıklıkla bahsedilir.



Şekil 2.2. Biyolojik Nöron Yapısı

Şekil 2.2 ’de bir insanın sinir sisteminde yer alan bir nöronun yapısı gösterilmektedir. Bu yapılardan insanların sinir sisteminde milyonlarca hatta milyarlarca adet bulunmaktadır.



Şekil 2.3. Yapay Nöronun Matematiksel Yapısı

Şekil 2.3 ‘de ise matematiksel olarak oluşturulan yapay bir nöronun yapısı bulunmaktadır. Nöronlar uçlarında bulunan ağaç köküne benzeyen dendirit yardımıyla sinyalleri alır ve bir çıktı üretir. Üretilen çıktı ağaç dallarına benzeyen aksonlar yardımıyla, sinaps denilen iletişim noktalarına taşınarak, sinyal diğer nöronlara iletilir. Nöronlar arasındaki iletişim böylelikle elektriksel sinyallerle gerçekleştirilir [20].

Yapay nöronlarda benzer mantıkta çalışmaktadır [21]. Yapay nöronlar biyolojik nöronlardan ilham alınarak tasarlanmıştır. Ancak yapay nöronlar biyolojik nöronlar gibi karmaşık bir yapıya sahip değildir.

Nöronlar birbirine denklem (2.1)’deki gibi bağlanmaktadır.

$$N = X * W + b \tag{2.1}$$

Bu denklemde sırasıyla;

- ✓ X = İnput, yani bir resmi oluşturan piksel değerleri
- ✓ W = Weight diye adlandırılan ve öğrenme kısmında modelin öğrendiği parametre olan ağırlık
- ✓ b = bias denilen katsayı, bu katsayı da eğitim sırasında güncellenir

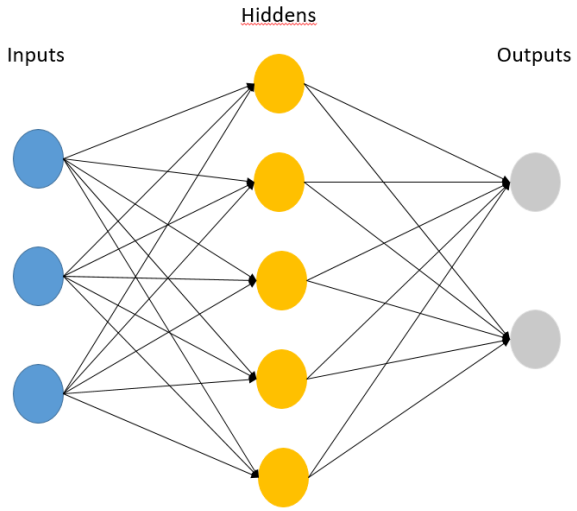
ifade etmektedir.

2.1.2. Katmanlar (Layers)

Layers olarak bahsettiğimiz katmanlar [22] sinir ağlarında farklı şekillerde çalışmaktadır. Aşağıda bazı katman türleri gösterilmektedir.

- ✓ Fully connected (dense) layers
- ✓ Convolutional layers
- ✓ Poooling layers
- ✓ Recurrent layers
- ✓ Normalization layers

Fully connected layer diye adlandırılan tam bağlanım katmanları, bir sinir ağında standart olarak kullanılan katmanlardır. Şekil 2.4 ‘te bir tam bağlanım katmanı gösterilmektedir.



Şekil 2.4. Tam Bağlanımlı Nöron Gösterimi

Convolutional layers denilen evrişimli katmanlar, resimler ile çok iyi sonuçlar alınan evrişimli sinir ağları içerisinde kullanılırlar.

2.1.3. Aktivasyon fonksiyonları

Yapay sinir ağları eğitilirken aktivasyon fonksiyonları sıklıkla kullanılır [23]. Aktivasyon fonksiyonlarının amacı ağırlık (W) ve bias (b) değerlerini ayarlamaktır.

Denklem (2.1)'deki işlemi yaptıktan sonra elde edilen değer aktivasyon fonksiyonundan geçirilir. Bu fonksiyonlar doğrusal olmayan operasyonlardır. En sık kullanılan aktivasyon fonksiyonları denklemleri ile birlikte;

Sigmoid: $\sigma(x) = 1/(1 + e^{-x})$ (2.2)

Tanh: tanh(x) (2.3)

ReLu: max(0, x) (2.4)

Leaky ReLu: max(0.1x, x) (2.5)

Maxout: $\max(\omega_1^T x + b_1, \omega_2^T x + b_2)$ (2.6)

gibidir. Bu aktivasyon fonksiyonları nöronların tetiklenmesi olarak düşünülebilir. Yüksek bir değer geldiğinde nöron daha yüksek, düşük bir değer geldiğinde ise nöron düşük bir sinyal üretecektir.

İnsan vücuduyla alakalı olarak, herhangi bir şekilde ele iğne batması sırasında meydana gelen acı ile parmağın kesilmesi ile meydana gelen acı arasında yüksek seviyede fark vardır. Burada parmağın kesilmesinde ortaya çıkan acı yüksek sinyal sonucu olmuş olup, iğne batması sonucunda ortaya çıkan acı düşük sinyal olarak nitelendirilebilir.

Sigmoid aktivasyon fonksiyonu; aldığı değerleri [0, 1] arasına sıkıştırır. Yüksek bir değer geldiğinde 1' e, düşük bir değer geldiğinde ise 0'a yakın değer üretir. Kısaca 1'i geçen ve 0'dan düşük değerler olmayacaktır.

Tanh aktivasyon fonksiyonu; sigmoid fonksiyonuna benzeyen bir aktivasyon fonksiyonudur. Sigmoid fonksiyonundan farkı, tanh aktivasyon fonksiyonu gelen değerleri [-1, 1] arasına sıkıştırır. Yüksek bir değer geldiğinde 1'e yakınken, düşük bir değer geldiğinde -1'e yakındır. Tanh aktivasyon fonksiyonu ile sigmoid aktivasyon fonksiyonuna göre daha iyi sonuçlar alınmaktadır.

ReLu aktivasyon fonksiyonu; günümüzde oldukça popüler olan aktivasyon fonksiyonudur. Fonksiyon gelen değerlerin pozitif veya negatif olup olmadığına göre sonuç üretir. Gelen değer 0'dan küçükse işlem değeri olarak 0 verilirken, 0'dan büyük değerler için herhangi bir sıkıştırma uygulamaz ve filtreden olduğu gibi geçer. Sigmoid ve tanh aktivasyon fonksiyonunda karmaşık hesaplamalar yapılırken; ReLu aktivasyon fonksiyonunda sadece negatif veya pozitif durum değerlendirmesi yapıldığından dolayı oldukça etkili bir yöntemdir. Bu da bilgisayarın diğer aktivasyon fonksiyonlarına göre daha hızlı işlem yapması demektir. Ayrıca ReLu aktivasyon fonksiyonu sinir sistemimizde bulunan biyolojik nöronlara daha yakın bir yapısı vardır. Biyolojik nöronlar belli seviyenin altında gelen sinyalleri görmezden gelir ve bu seviyeye biyolojide eşik şiddeti denilir. Gelen sinyal eşik şiddetini geçemezse yok sayılır.

Leaky ReLu aktivasyon fonksiyonu; ReLu aktivasyon fonksiyonundan farklı olarak, negatif bir değer geldiğinde çok küçük negatif bir sayı döndürür. Bu sayede nöronların ölmesinin önüne geçilir.

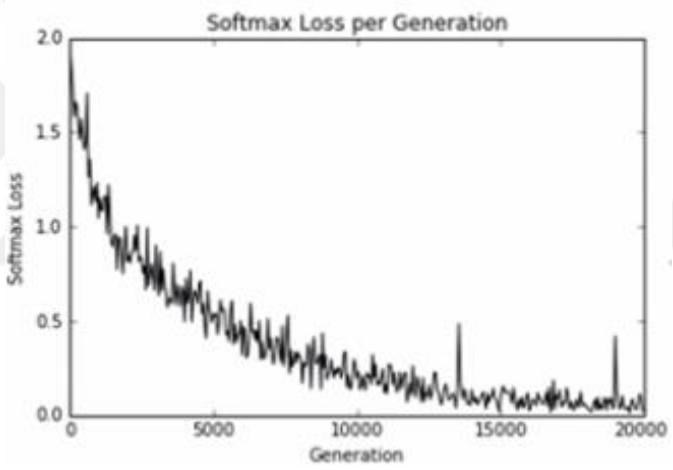
Maxout aktivasyon fonksiyonu ise; ReLu ve Leaky ReLu aktivasyon fonksiyonlarını genelleştirir ve yine Leaky ReLu aktivasyon fonksiyonundaki gibi ölü nöronların

önüne geçilir. Maxout aktivasyon fonksiyonunda parametreler ikiye katlandığından dolayı, ReLu ve Leaky ReLu aktivasyon fonksiyonuna göre daha yavaş çalışır.

Sonuç olarak eldeki problemlere göre sinir ağlarında yukarıda bahsedilen aktivasyon fonksiyonları kullanılır. En popüler olanı ReLu aktivasyon fonksiyonudur. Sigmoid aktivasyon fonksiyonu ise artık günümüzde kullanımını yitirmiştir.

2.1.4. Loss (Cost) fonksiyonları

Loss fonksiyonu makine öğrenmesi için çok önemli bir fonksiyondur. Loss fonksiyonu herhangi bir modelde tahmin edilen değerin gerçek değerden ne kadar uzak olduğunu hesaplar ve bize hatanın büyüklüğünü gösterir [24]. Şekil 2.5 'te loss fonksiyonunun grafiği gösterilmektedir.



Şekil 2.5. Loss Fonksiyonu Grafiği [28]

Şekil 2.5 'teki grafiği incelersek, loss grafiğinin gittikçe düştüğünü görebiliriz. Bu modelin öğrenmesi açısından iyi bir performans ve istenilen bir çıktıdır. Amaç loss fonksiyonunu sıfır yapmak, sıfır yapamasak bile en azından sıfıra yaklaştırmaktır. Model eğitilirken input için, olasılığa benzer değerler verecektir. Verilen bu olasılıklardan en yüksek değere bakılarak, tahmin yürütülür.

Loss fonksiyonu da bu tahminin ne kadar doğru olduğunun matematiksel olarak hesaplanması yapılır. Kısaca loss fonksiyonu ile model hatalarını görür ve optimizasyon ile hataları azaltmaya yani daha iyi olmaya çalışır.

2.1.5. Gradyan Azalma (Gradient descent)

Sinir ağını özetleyecek olursak; oluşturulan model bir giriş bilgisi alacak ve bu giriş bilgisini sinir ağından geçirip sonuç olarak bir olasılık değeri verecektir. Verilen bu olasılığın ne kadar doğru olduğu hesaplanıp, modele hatanın büyüklüğü verilecektir. Model verilen bu hataya bakarak; tekrardan öğrendiği ağırlık (weight) ve bias değerlerini güncelleyecektir [30]. Eğitim esnasında model bu işlemi veri setine bağlı olarak (veri setinin sayısı burada önemlidir) binlerce kez gerçekleştirecektir. Bu işlem esnasında ise model öğretilmek istenen her ne ise onu öğrenecektir. Loss değeri, modelin; gerçek değerden ne kadar uzak olduğu bilgisini verir. Loss değerini modelin azaltması gerektiği artık anlaşılmaktadır. Bu azaltma işlemini yapmak için amaç en alçak noktayı yani loss fonksiyonunun en küçük olduğu noktayı bulmak olacağı için; belirli bir adım büyüklüğüne göre sürekli gezinerek en düşük nokta bulunulabilir. Adım büyüklüğü kullanıcıya bağımlı, kullanıcı tarafından belirlenen bir parametredir. Bu adım büyüklüğünün literatürde karşılığı learning rate yani öğrenme oranıdır.

Öğrenme oranı modele atanacak en önemli parametrelerdendir. Bu oran için çok küçük sayılar atanacaktır. Doğru öğrenme oranı model başarısını direkt olarak etkileyen bir parametredir.

2.1.6. Optimizasyon algoritmaları

Gradient descent algoritmasında, belirlediğimiz öğrenme oranı kadar ilerleyerek minimum noktasına ulaşılıyordu. Bu minimum noktasına ulaşmada kullanılan farklı optimizasyon algoritmaları;

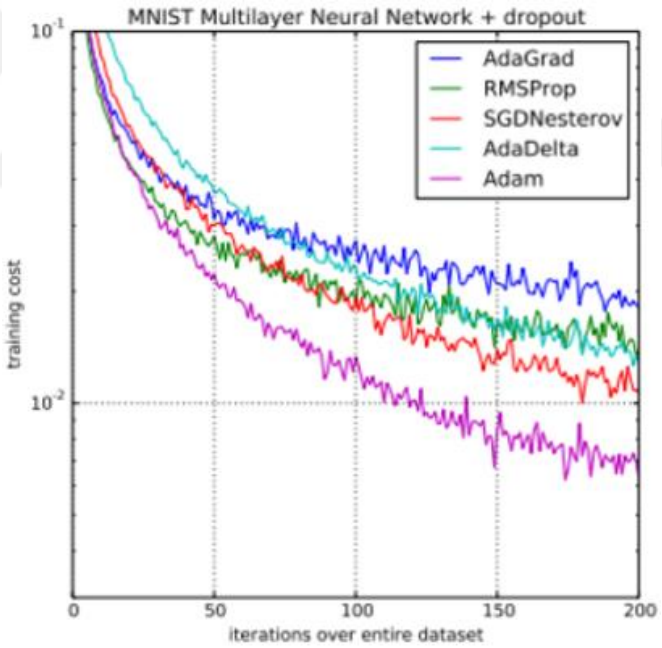
- ✓ Stochastic Gradient Descent (SGD)
- ✓ SGD + Momentum
- ✓ RMSProp
- ✓ Adam
- ✓ Adamax
- ✓ Adadelta
- ✓ Adagrad

gibi optimizasyon algoritmaları bulunmaktadır. Derin öğrenmede kullanılan optimizasyon algoritmaları incelenecek olursa;

Stochastic Gradient Descent (SGD); Bu optimizasyon algoritmasında kademeli olarak düşüş yapılarak optimum değer bulmaya çalışılır [31]. En büyük dezavantajı algoritma değer optimizasyon algoritmalarına göre yavaş çalışmaktadır. Bu yavaşlığı azaltmak için algoritma momentum ile birlikte kullanılmıştır. Resim tanımada çok fazla yavaşlamaya neden olduğundan dolayı bu algoritma resim tanıma probleminin çözümü için uygun değildir.

Adam; Parametrelerin her birinin öğrenme oranları ile birlikte momentum değişikliklerinin de önbellekte tutulmasından dolayı algoritma SGD' ye göre hızlı çalışmaktadır.

AdaGrad; Doğal dil işleme ve resim tanıma gibi seyrek olan veriler için uygun bir optimizasyon algoritmasıdır. Her parametrenin kendi öğrenme hızı vardır. Öğrenme oranının belli bir noktada tamamen durması bu algoritmanın en büyük dezavantajıdır.



Şekil 2.6. Farklı Optimizasyon Algoritmalarının Verim Grafiği [29]

Şekil 2.6 'da ki verim grafiği incelendiğinde en iyi optimizasyon algoritmasının Adagrad olduğu görülmektedir. Ancak bu optimizasyon algoritmalara veri setine ve uğraşılan probleme göre değişkenlik gösterdiğinden dolayı her zaman Adagrad optimizasyon algoritmasının veriminin yüksek olduğu yanılgısına düşülmemelidir.

2.1.7. Geriye yayılım algoritması (Backpropagation)

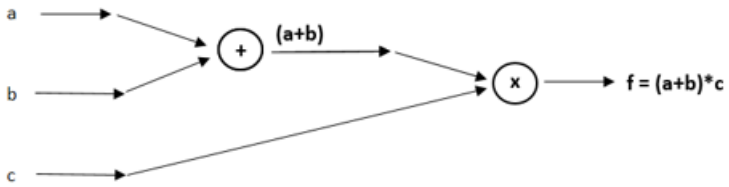
Evrişimli sinir ağında, bir input değeri (evrişimli sinir ağı kullandığımız için bu input değeri bir resim) alınır ve sinir ağında ileriye doğru gidilerek bir çıkış (output) üretilir. Üretilen bu değere göre oluşturulan model işlem bittiğinde, ortaya çıkan hata değerine göre ağırlıklar (weight) ve bias değerleri güncellenmektedir [32]. Bunun nedeni oluşturulan modelin daha iyi sonuç vermesi içindir. Model bahsedilen bu işlemleri yapması için geriye yayılım algoritmasını kullanır.

Yapay sinir ağlarında her nöronun hataya ne kadar katkısı olduğunu hesaplamak için backpropagation yöntemi kullanılmaktadır. Yapay sinir ağında modelin öğrenmesi istenilen parametreler ağırlıklar ve biaslardı. Bunlar ilk başta rast gele verilerek sonrasında model tarafından optimize edilmektedir. Backpropagation yöntemi yardımıyla, model geri giderek ağırlıkları ve bias değerini daha isabetli olması için güncelleyecektir. Güncelleme işlemi, bilindiği üzere matematikte kullanılan türev işlemindeki zincir kuralı ile yapılmaktadır.

Denklem (2.7) ‘deki gibi bir fonksiyon olsun.

$$f(a,b,c) = (\mathrm{a} + \mathrm{b}) * \mathrm{c} \tag{2.7}$$

Yukarıda tanımlanan f fonksiyonunu incelendiğinde, a ile b toplamı ve bu toplam sonucunun c ile çarpımı f fonksiyonunu verir.



Şekil 2.7. Denklem (2.7) ‘deki Fonksiyonun Gösterimi

Şekil 2.7 incelendiğinde a ile b toplanarak + düğümüne gelinir. Bu toplamın sonucu ile c çarpılarak x düğümüne geliyor ve f fonksiyonunu oluşturulur.

Fonksiyondaki değerlere sayılar verecek olursak;

- ✓ a = 3
- ✓ b = 6
- ✓ c = -5 olsun;

Bu değerleri denklem (2.7)'de yerine koyarsak,

$f(3{,}6, -5) = (3 + 6) * (-5)$

$f(3{,}6, -5) = -45$

Sonucuna ulaşılmaktadır. Şekil 2.7' de de görüldüğü gibi ileriye giderek sonucu elde edilir. Şimdi yine şekil 2.7 kullanarak, backpropagation ile geriye gidilecektir. Geriye girmek için matematikte sıkça kullanılan türev işlemlerinden zincir kuralı (chain rule) uygulanacaktır. Her bir düğüm noktasını bir değere eşitlemek adına,

x = a + b olsun. Bu işlem sonucunda, f = x*c olur.

$$x = a + b \quad \frac{\partial x}{\partial a} = 1\,,\ \frac{\partial x}{\partial b} = 1 \tag{2.8}$$

$$f = x * c \quad \frac{\partial f}{\partial x} = \mathrm{c}\,,\ \frac{\partial f}{\partial c} = x \tag{2.9}$$

olarak ifade edilir. O halde f fonksiyonunu ifade eden şekil 2.7 incelendiğinde backpropagation yapılabilmesi için denklem (2.10) 'da ki değerler gereklidir.

$$\frac{\partial f}{\partial a}\,,\ \frac{\partial f}{\partial b}, \frac{\partial f}{\partial c} \tag{2.10}$$

Yukarıda belirtilen denklemlerdeki hesaplamalar hiçbir şekilde manuel olarak yapılmaz. Kullanılan yazılım programlarının kütüphanelerini kullanarak işlemler gerçekleştirilir. Temel olarak sinir ağlarında nöron olarak bahsedilen düğümler bulunmaktadır. Bu düğümler sadece düğüme gelen ve düğümden çıkan kısımlar tarafından etkilenmektedir. Böylelikle her bir nöron yani düğüm kendisine giren ve çıkan kısımlardan sorumlu olmasından dolayı, basit bir şekilde hesaplanabilmektedir.

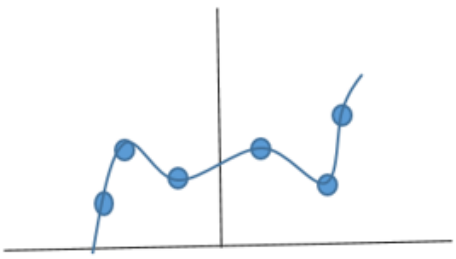
2.1.8. Overfitting (Aşırı öğrenme) ve düzenlileştirme (regularization)

Makine öğrenmesinde karşılaşılan en sık problemlerden birisi de aşırı öğrenme (overfitting) diye adlandırılan problemdir. Aşırı öğrenme temel olarak; veriler kullanarak ortaya çıkarılan model, eğitim setindeki verileri tanımada oldukça iyi olduğu halde, hiç görmediği bir test setindeki verileri tanımada kötü sonuçlar veriyorsa, bu olay aşırı öğrenme olarak adlandırılır [33].

Aşırı öğrenmenin daha iyi anlaşılması için, aşırı öğrenme ezber yapan bir öğrenciye ezberletilebilir. Sınav öncesi sınavla alakalı 20 soru verilmiş olsun. Öğrenci bu 20 soruyu öğrenmek yerine ezberlerse;

- ✓ Eğer şanslıysa, dersin hocası sınavda aynı soruları sorduğunda öğrenci soruları kolaylıkla yapacaktır,
- ✓ Eğer şanslı değilse, yani dersin hocası verdiği soruları biraz değiştirip sorduğunda öğrenci ezberlediği için değişik gelen bu soruları cevaplayamayacak ve sonucunda kötü not alacaktır.

Öğrencinin kötü not almasının nedeni; soruları öğrenmek yerine ezberlemiş olmasıdır. Eğer ezberleme yerine öğrenmiş olsaydı öğrenci iyi not alacaktı. Şimdi bu örnekten asıl problem geçildiğinde, öğretmenin sınav için verdiği sorular eğitim kümesini, sınavda sorduğu sorular ise test kümesini oluşturuyor. Oluşturulan modelin iyi performans vermesini sağlamak için modelden eğitim verisindeki verileri genellemesi beklenmektedir. Bu genelleme sonucunda model hiç görmediği bir test kümesindeki verisinde iyi sonuçlar vermiş olacaktır. Makine öğrenmesi sonucu oluşturulan modellerin performansı eğitim setindeki verilerin performansından değil, hiç görmediği test setindeki verilerin performansının başarısından ölçülmektedir.



Şekil 2.8. Aşırı Öğrenme Grafiği

Şekil 2.8 incelendiğinde, mavi yuvarlaklar gerçek değerleri, mavi çizgiler ise makine öğrenmesi modelindeki eğitim verisinin sonuçlarını göstermektedir. Görüldüğü üzere model eğitim setindeki verileri mükemmel bir şekilde sınıflandırmak için eğip büktü ve nerede ise hiç hata yapmadan %100 başarı ile şekil 2.8 ‘deki gibi bir grafik oluşturdu. Yeni bir veri geldiğinde ve bu veriler eğitim sırasında çizilen çizgi üzerinde olmadığında model eğitim verisinde elde ettiği başarıyı gösteremeyecektir.

Kısaca oluşturulan makine öğrenmesi modellerinin genelleştirme yapabilmesi için olabildiğince basit olması beklenmektedir. Model karmaşıklığı arttığında, artık model eğitim kümesindeki verileri genelleyemeyecek ve ezberleyecektir.

Aşırı öğrenme problemini çözmek için yapılması gereken işlemler düzenlileştirme (regularization) olarak adlandırılır. En iyi çözüm,

- ✓ Mümkünse veri setine daha fazla veri eklenmelidir. Gerçek hayatta veri genel olarak sınırlı olduğundan dolayı sıklıkla diğer yöntemlere başvurulmaktadır
- ✓ Data augmentation, bu yöntemde elde bulunan veri, eğitim esnasında genelleştirilerek modelin daha iyi performans vermesi sağlanmaktadır
- ✓ Dropout yöntemi, sinir ağlarında oldukça yaygın olarak kullanılmaktadır. Bu yöntemde sinir ağı üzerinden her ileriye doğru gidişte, her katmanda (layer) bulunan nöronlardan bazıları sıfırlanır [34]. Dolayısıyla bu nöronlar aktif olamazlar. Böylelikle model tek bir noktaya odaklanmadan daha iyi genelleştirme yapabilir hale getirilir. Eğitim bittiği anda iste tüm nöronlar tekrardan aktif hale getirilir

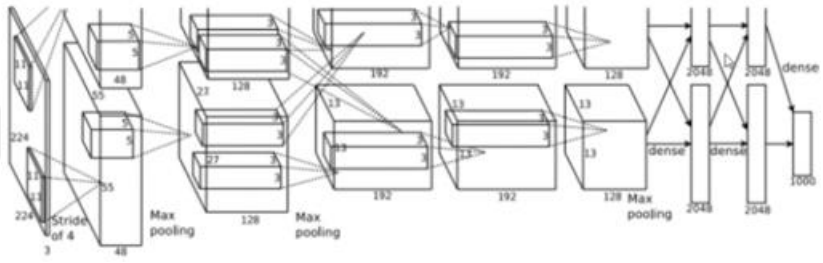
Yukarıda bahsedilen özellikler temel özellikler olup liste literatürün getirdiği katkılar ile uzatılabilmektedir.

2.2. Klasik Evrişimli Sinir Ağları

Günümüzde optimizasyon algoritmalarının gelişmesi ve bilgisayarların işlem gücünün giderek artmasından kaynaklı farklı mimarilere dayalı evrişimli sinir ağları oluşturulmaktadır [35]. Literatür daha detaylı incelendiğinde 1998 ile 2012 yılları arasında evrişimli sinir ağları üzerine geliştirme yapılamamıştır. Geliştirme yapılamamasının nedeni, bilgisayarlar o dönemde evrişim işlemlerindeki hesaplama güçlüğünden dolayı yeterli hesaplama yapacak performansa sahip olmamasından kaynaklanmaktadır. Evrişim işlemlerinde parametre sayıları oluşturulan model ile orantılı olarak arttığından dolayı, 2012 öncesi bilgisayarlar henüz artan parametre sayılarını işleyecek güçte değillerdi. Bundan dolayı o örneklemelerde rakam tanıma gibi evrişim işleminde aşırı miktarda içerisinde parametre bulundurmayan hesaplamalar yapılabiliyordu [36].

2.2.1. AlexNet

Mimariden bağımsız olarak herhangi bir evrişimli sinir ağı modelinin başarılı olması fazla miktarda veri ile alakalıdır. Bilgisayarların gelişmesine paralel olarak internet hızlarının gelişmesi, buna bağlı olarak artık hemen hemen her evde bilgisayar bulunması ve bu bilgisayarların internete açık olmalarından dolayı, günümüzde görüntü verileri üretilmekte ve bu verilere ulaşım kolaylaşmaktadır. Evrişimli sinir ağı mimarilerinde üretilen bu görüntüler kullanarak, hızlı bir şekilde projelere (yüz tanıma, parmak izi tanıma, otonom araç, plaka tanıma, araç tanıma vb.) dönüşmektedir. AlexNet mimarisinin oluşmasında çok büyük bir veri seti kullanılmıştır. AlexNet, 2012 yılında yapılan ImageNet yarışmasını kazanan mimari olmasından dolayı, derin öğrenmenin tanınmasına ve bilinmesine önemli bir katkı sağlamış, derin öğrenmeyi dünyaya resmen duyurmuştur [37].



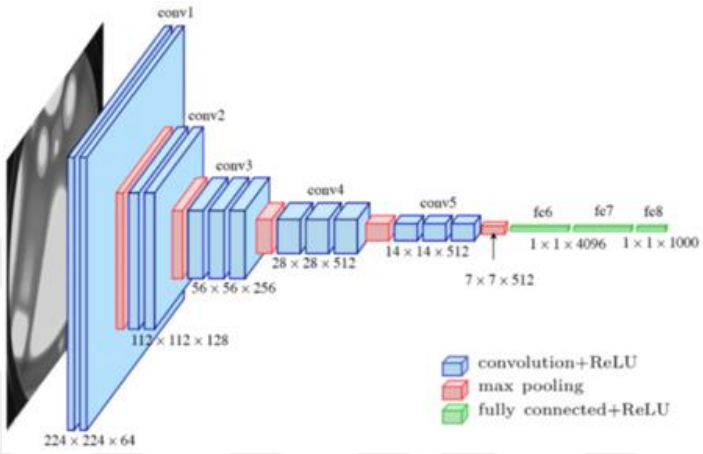
Şekil 2.9. AlexNet Evrişimli Sinir Ağı Mimarisi [38]

AlexNet mimarisini inceleyecek olursak, AlexNet mimarisini diğer mimarilerden ayıran en önemli özelliği; Şekil 2.9 ‘dan anlaşılacağı üzere giriş görüntüsünü iki paralel evrişimli ağda eğitiyor olmasıdır. Birbirine paralel ilerleyen iki evrişimli model en sonunda birleştiriliyor. Şekil 2.9 ‘a dikkat edilirse giriş görüntüsünde renkli bir görüntü (3 kanallı) kullanılmıştır.

2.2.2. VGG

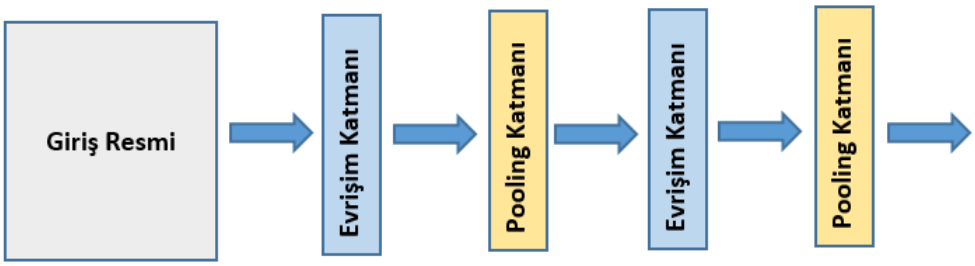
Bir başka evrişimli sinir ağı mimarisi VGG ’dir. VGG, oxford üniversitesinde kurulan ve bir araştırma grubu olan Visual Geometry Group ‘un kısaltmasıdır [39]. 1000 sınıfa ait 14 milyondan fazla veri kullanarak yapılan sınıflandırmada yaklaşık 92% başarı oranı ile başarısını ispatlamıştır. VGG-16 ve VGG-19 olarak kendi içerisinde iki farklı mimarisi bulunmaktadır. 16 ve 19 rakamları mimarideki gizli katman sayısını ifade etmektedir.

Genel olarak bir sinir ağında başarımın arttırılması için sinir ağının daha da derine inmesi (gizli katman sayısının arttırılması) tekniğine dayanır. Sinir ağının derinliği arttıkça model karmaşıklığı artar. Model karmaşıklığının artması ise hem aşırı öğrenmeye, hem de model ilk kısımlarda öğrendiği özelliklerin kaybolmasına neden olmaktadır. Bu sebeplerden dolayı mimarilere farklı özellikler eklenerek bu gibi handikapların önüne geçilmiştir.



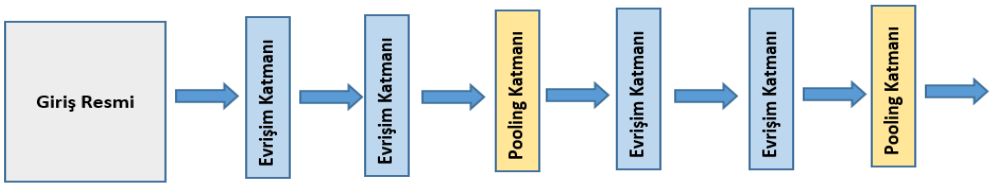
Şekil 2.10. VGG-16 Evrişimli Sinir Ağı Mimarisi [40]

VGG mimarisi AlexNet ‘ten farklı olarak oldukça küçük denilebilecek katmandan (Şekil 2.10, 7x7x512 katman) işlem yapmaktadır. Bu mimarinin diğer mimarilerden en önemli farklı ise yine şekil 2.10 ‘da görüleceği üzere, ilk katmanda 64 kanallı iki sonuç elde edilir. Genelde diğer mimarilerde giriş görüntüsü verildikten sonra, şekil 2.11 ‘deki sıralama ile devam edilir.



Şekil 2.11. Genel Evrişimli Sinir Ağı Mimarisinde Katmanların Sıralaması

VGG mimarisi ise, sadece giriş katmanından sonraki ilk katmanda art arda 64 kanallı iki evrişim kanalından sonra, devamında yine ara katmanlarda da, bir ortalama (pooling katmanı) işleminden sonra birden çok evrişim işlemi gerçekleşiyor.



Şekil 2.12. VGG Evrişimli Sinir Ağı Mimarisinde Katmanların Sıralaması

Şekil 2.12 göz önünde bulundurulduğunda, VGG mimarisinde hesaplanan parametre sayısı ağ mimarisinden dolayı diğer mimarilerden fazladır.

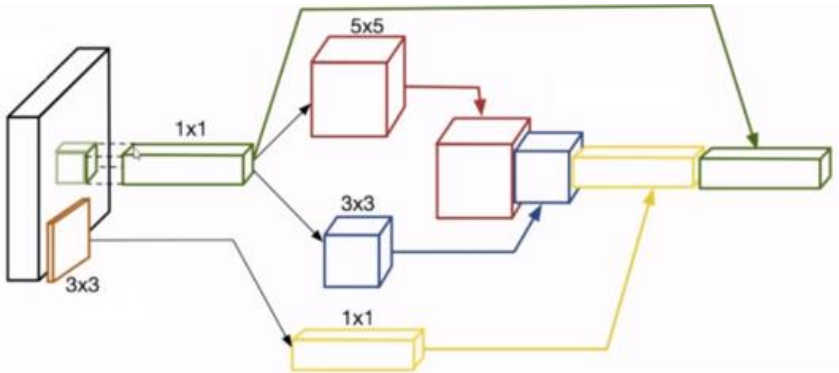
2.2.3. GoogleNet

GoogleNet mimarisi adını 2014 yılında yapılan ImageNet yarışmasında birinci olarak duyurmuştur. Yapısında Inception modülü bulundurmasından dolayı karmaşık bir mimaridir. Diğer mimarilerde genellikle, evrişim ve havuzlama katmanları ardışık olarak birbirini takip etmektedir. GoogleNet mimarisi bu ardışıklığı kullanmayan ilk evrişimli sinir ağı modellerindendir [31].

Inception modülü, matristen ziyade tensör özelinde işlem yaptığı için modeldeki parametre sayısı üzerinde etkilidir.

$$\text{Yeni Parametre Sayısı} = \frac{\text{Model Parametre Sayısı}}{10} \tag{2.11}$$

Denklem (2.11) incelendiğinde inception modülünde oluşan parametre sayısı geleneksel mimariye göre 10 kat daha az olup, parametre sayısının az olması modelin eğitim işlemi sırasında daha kısa sürede eğitimi tamamlamasına ve sınıflandırma yaparken de daha hızlı sınıflandırma yapmasını sağlayacaktır.



Şekil 2.13. Inception Sinir Ağı Mimarisinde Katman Yapısı [41]

Inception sinir ağı mimarisindeki katman yapısı incelendiğinde, giriş görüntüsü üzerine 3x3 ‘lük max pooling katmanı uygulanır. Giriş görüntüsü üzerine ayrıca, 1x1 ‘lik, 5x5 ‘lik ve 3x3 ‘lük evrişim işlemi uygulanır. Şekil 2.13 ‘te görüldüğü üzere en son ayrı ayrı uygulanan bu evrişim işlemi birbirine eklenir.

Bu işlemleri yapmadaki amaç; 28x28x192 kanallı bir tensöre, 5x5 ‘lik 32 adet filtre uygulandığında 28x28x32 kanallı bir tensor elde edilir. Yalnızca bu işlem için ortaya çıkacak parametre sayısı hesaplanırsa; (28x28x32)x(5x5x192) = 120 milyon parametre hesaplanması gereklidir.

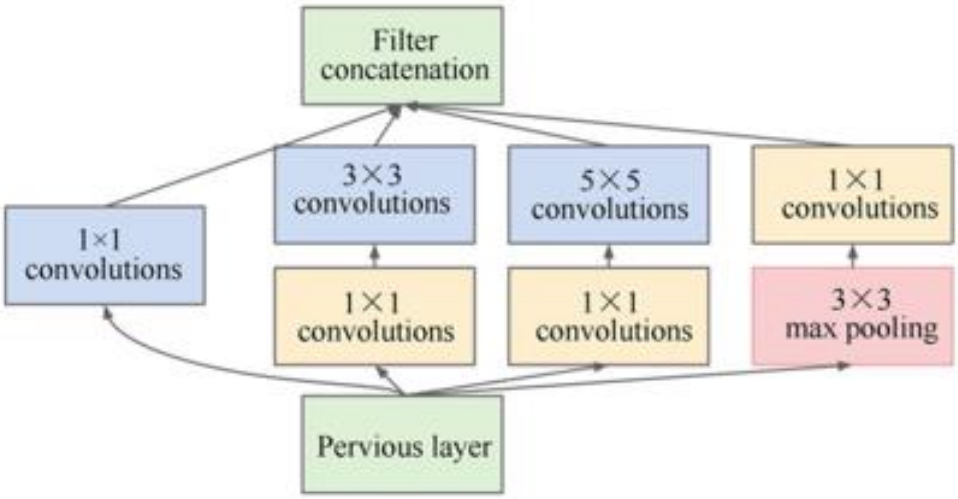
Aynı tensör üzerinde (28x28x192 kanallı) 1x1x192 ‘lik 16 adet filtre kullanıldığında ise; 28x28x16 kanallı bir tensor elde edilir. Bu işlemin sonrasında ise, 5x5 ‘lik 32 adet filtre uygulandığında 28x28x32 kanallı bir tensor elde edilir. Sonuç olarak yukarıda elde edilen tensör ile aynı sayıda tensör elde edildi.

Bu işlem için ortaya çıkacak parametre sayısı hesaplandığında, (28x28x16)x(1x1x192) = 2.4 milyon parametreye ek olarak, (28x28x32)x(5x5x16) = 10 milyon parametre, toplamda ise 12.4 milyon parametre hesaplanması gerekir.

Bu hesaplamadan yola çıkarak ilk duruma göre yaklaşık 10 kat daha az parametre hesabı yapılacak olup, bu hem ortaya çıkacak olan modelin hızını arttıracak, hem de fazla parametre model karmaşıklığını arttırıp, aşırı öğrenmeye neden olacağı için, ortaya çıkacak olan model aşırı öğrenmeden kaçınılmış olunacaktır.

Inception modülü GoogleNet mimarisinde kullanılmasından dolayı, katman sayısının fazla olması yani derin bir ağ yapısına sahip olmasına rağmen, beklenenden daha az parametre hesaplanması yapılır.

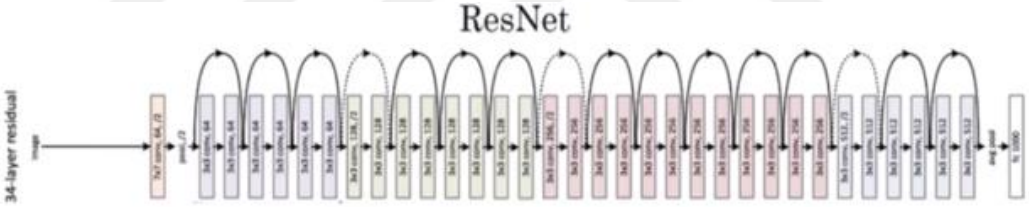
Parametre hesaplamasının az olması ise uygulanan mimarinin üzerinde oluşturulan modelin daha hızlı çalışması yani görüntü veya video üzerinde nesne tespitinin daha hızlı yapılmasını sağlamaktadır. Nesne tespit işlemlerinde yüksek başarımın yanında artık gerçek zamanlı olarak nesne tespiti yapılmalıdır. Mimariler üzerinde hesaplanacak parametre sayısının azlığı, hız kazandırmanın yanında, aşırı öğrenmenin önüne geçilmesini sağlamaktadır.



Şekil 2.14. Inception Modülü [42]

GoogleNet mimarisinde şekil 2.14 ile gösterilen inception modülü 9 tane ard arda kullanılır.

2.2.4. ResNet

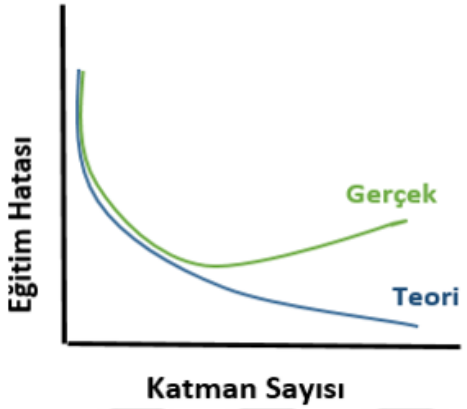


Şekil 2.15. ResNet Mimarisi [43]

ResNet mimarisi incelendiğinde, bir evrişim ve ortalama katmanlarından oluşan 34 katmanlı düz bir ağ katmanı oluşturulan zamana göre, bir katman atlayarak oluşturulan bağlantılarla model eğitildiğinde iki model arasında fark oluşmaktadır.

Özellikle katmanların arasına yapılan şekil 2.15 ‘deki gibi aktarma işlemine ResNet mimarisi denilmektedir. ResNet mimarisinde katman sayısı arttırıldığında model iyi performans vermektedir. Katmanlar arasında yapılan aktarma işlemine odaklanıldığında, eğer aradaki katmandaki değerler 0 çıkmış olsa yani aradaki katmanlarda öğrenme olmadığında bile, bir önceki katmandan gelen bilgi ile devam edilmektedir. Bu yöntem, modeli garantiye alınması olarak düşünülebilir. Kısaca bu mimaride, eski katmanda herhangi bir şey öğrenilmemiş olsa bile eski katmandan gelen bilgi yani iki önceki katmandan gelen bilgi, en kötü ihtimalle yeni katmanda

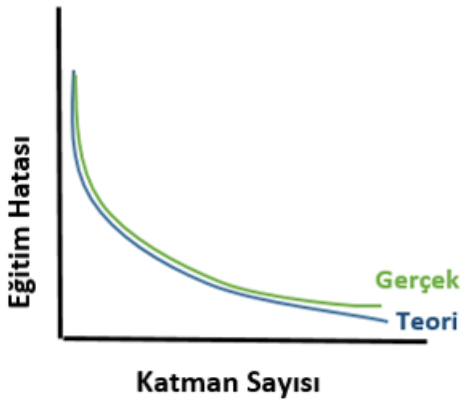
devam ettirilerek model güçlü bir hale getirilir. Böylelikle katmanlarda meydana gelebilecek herhangi bir bilgi kaybı minimuma indirilmiş olunur.



Şekil 2.16. Eğitim Hatası - Katman Sayısı Grafiği (a)

Normal sinir ağı mimarisinde, şekil 2.16 'daki gibi teorik olarak katman sayısı arttıkça eğitim hatasının azalması beklenmektedir. Ancak gerçek bir sistemde katman sayısı arttıkça eğitim hatası bir süre sonra artmaktadır ki bu istenilmeyen bir durumdur. Bunun nedenleri incelendiğinde;

- ✓ Katman sayısı arttıkça model ilk katmanlarda öğrendiği basit şekilleri artık öğrenememektedir
- ✓ Katman sayısının artması, modeldeki parametre sayısının artması anlamına geldiğinden, parametre sayısının fazla olması modelin karmaşıklığını arttırarak aşırı öğrenmeye neden olmaktadır



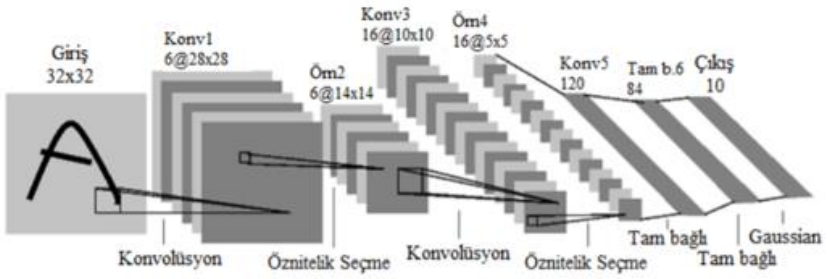
Şekil 2.17. Eğitim Hatası - Katman Sayısı Grafiği (b)

ResNet mimarisinde yapılan aktarmalar sayesinde; katman sayısı arttıkça eğitim hatası şekil 2.17 ‘de gösterildiği üzere, teoride olması gerektiği gibi azalmaktadır. Bu sebepten dolayı ResNet mimarisinde farklı katmanlı versiyonlarda (ResNet-50, ResNet-152 v.b.) bulunmaktadır. Bu versiyonlar özellikle görüntü sınıflandırma ve tanıma problemlerinde kullanılmaktadır [44].

2.2.5. LeNet5

LeNet mimarisi; neredeyse 1980 ’lerden, 1998 ‘lere kadar üzerinde çalışılan bir konuydu. LeNet sinir ağı mimarisi ile o yıllarda çek senet tanıma, sokak tabelalarının tanınması ve el yazısı tanıma gibi çalışmalar bulunmaktaydı.

Günümüze kıyasla o yıllarda yapılan bu çalışma devrim niteliğinde olmakla birlikte, artık LeNet5 mimarisi günümüzde yaygın olarak kullanılmamaktadır.



Şekil 2.18. LeNet5 Mimarisi [15]

LeNet5 evrişimli sinir ağı mimarisinde giriş resmi olarak 32x32x1 ‘lik tek kanallı yani siyah – beyaz resim kullanılmıştır. 5 katmanlı bir sinir ağıdır. Elde edilen sonuçlarla, bilgisayar ekranında insan tarafından yazılan el yazılarının tanınmasında kullanılmıştır. Kullanıldığı yıl incelendiğinde o yıllarda böyle bir çalışmanın yapılıyor olması oldukça keyifli ve insanlık için heyecan verici olmaktadır.

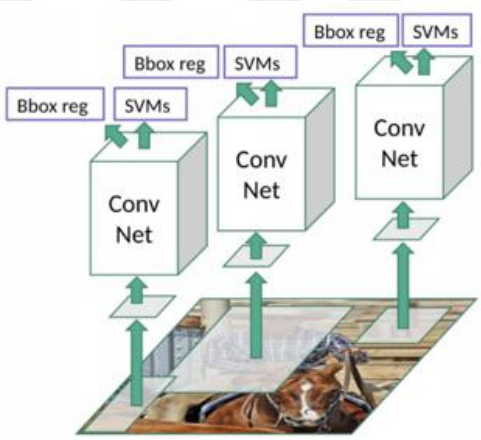
Giriş resmine evrişim işlemi yapılıyor ve evrişim katmanında 28x28 boyutunda 6 adet filter uygulanıyor. Böylelikle 28x28x6 kanallık tensör elde edilir. Sonrasında max pooling katmanıyla ortaklama işlemi yapılarak ikinci evrişim katmanın geçilmektedir. Bu işlem sonucunda 14x14x6 kanallık tensör kalır. Sonuç olarak şekil 2.18 ‘deki gibi bir yapı oluşturulmuş olunur.

2.2.6. R-CNN

Evrişimli sinir ağı mimarisinde ağa giriş olarak resim alındığı incelenmişti. Genel olarak giriş resmi üzerinde filtreler gezdirilerek özellik haritaları çıkarılıp, bu özellik haritalarına bakılarak, sınıflandırma veya tanıma gibi işlemler yapılmaktadır. Giriş resminin boyutu arttırıldığında tüm resmin her yerinde farklı boyutlarda pencereler oluşturup üzerinde filtre ile gezinmek, bilgisayarlar için oldukça zor olmaktadır.

Bilgisayarlar için hesaplaması güç olan bu zorlukla baş edilebilmesi için, resim üzerinde yaklaşık 1000-2000 kadar içerisinde nesne olabilecek alanlar belirlenir. Böylelikle tüm resmi pencerelerle gezmek yerine, bölge önerisi ile belirlenen 1000-2000 bölge içerisinde nesne aramak çok daha kolay olmaktadır.

Region-based Convolution Neural Network (R-CNN) mimarisi de bahsedilen yapıya dayanır. Şekil 2.19 incelendiğinde, tüm resimde nesne aramak yerine bölge önerisi ile nesne olma ihtimali yüksek bölgelerde nesne aranır.



Şekil 2.19. R-CNN Mimarisi [45]

Mimari incelendiğinde, resim üzerinde yaklaşık 2000 bölge seçilir. Seçilen bölgelerin farklı boyutları olmasından dolayı, evrişimli sinir ağından geçirebilmek için bölgelerin boyutları eşitlenir. Boyutları eşitlenen her bir bölge evrişimli sinir ağına input olarak verilerek sinir ağından geçirilir. Ağın sonucunda seçilen bölgelerde sınıflandırma yapmak için destek vektör makineleri (support vector machine) kullanılır. Destek vektör makineleri gözetimli öğrenmede kullanılan bir makine öğrenme yöntemidir. Lineer regresyon yöntemi ile sınıflandırma yapılan bölgedeki nesnenin sınırları belirlenir. Lineer regresyon bağımsız değişkenler ile başka bir bağımlı değişken

arasındaki bağlantıyı modellemek için kullanılan makine öğrenmesi yöntemidir. Bu şekilde sınıflandırma yapılan nesnenin bulunduğu bölgeler üzerinde dikdörtgenler çizilerek görüntü üzerinde bulunan nesne tespit edilmiş olur.

Nesne olma ihtimali bulunan herhangi bir bölge alındığında, eğer alınan bölgede nesne varsa genel tabiri ile resim üzerinde bulunan nesnenin konumu belirlenmiş olunur. Lineer regresyon ile nesnenin bulunduğu alan tam olarak tespit edilerek, nesnenin etrafında çizilecek olan dikdörtgen nesneyi tam kapsayacak şekilde ayarlanır. Resim üzerinde belirlenen bölgelerde nesne bulunursa ve bu nesnenin bir parçası seçilen bölge alanının dışında ise, çizilen dikdörtgen belirlenen bölgenin dışına çıkabilir.

Sonuç olarak R-CNN mimarisinde giriş görüntüsü üzerinde bölgeler seçilerek o bölgeler üzerinde nesne aranır. Eğer seçilen bölge üzerinde nesne bulursa o bölgenin sınıfını döndürür. Aynı zamanda nesnenin resim üzerinde konumunu vermekle birlikte, 4 tane nokta verir. Bu noktaları resim üzerinde birleştirip dikdörtgen elde edildiğinde, ilgili nesne çizilen dikdörtgenin içerisinde kalır.

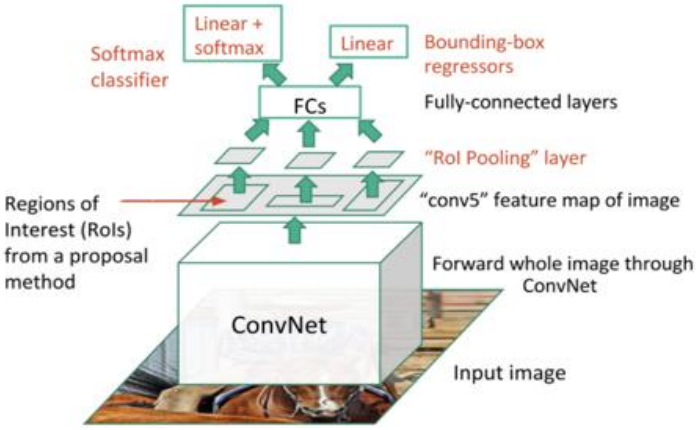
R-CNN ‘nin karşılaştığı bazı problemler bulunmaktadır. Önerilen bölgelerin alınması ve hepsinin ayrı ayrı evrişimli sinir ağından geçirilmesinden dolayı oldukça yavaş çalışmaktadır. Bu yavaşlıktan dolayı gerçek zamanlı olarak nesne tanıma yapması neredeyse mümkün değildir.

2.2.7. Fast R-CNN

R-CNN mimarisinin en büyük problem oldukça yavaş çalışması diye belirtilmişti. Bu yavaşlığı ortadan kaldırabilmek için Fast R-CNN mimarisi geliştirildi.

Fast R-CNN, R-CNN ile oldukça benzer bir yapıdadır. Sadece hızlı olması için farklı teknikler kullanılmıştır. Yinede bu hız gerçek zamanlı işlem yapmak için yeterli değildir.

Fast R-CNN, R-CNN ‘e göre hızlı olmasının yanında içerisindeki mimari sayesinde nesne tespit işlemleri başarımı da, R-CNN ‘e göre yüksektir. Her iki mimari temelde aynı olup, farklılık katmanlar arasından kaynaklanmaktadır. Faster R-CNN mimarisinde, bölge önerisi rastgele yapılmak yerine, şekil 2.20 ‘deki gibi sinir ağı ile birlikte yapılmaktadır.



Şekil 2.20. Fast R-CNN Mimarisi [46]

Fast R-CNN ‘de giriş resmi alınır, bu resim üzerinde ayrı ayrı bölge önerisi yapmak yerine; resim olduğu gibi evrişimli sinir ağından geçirilir. Daha sonra orijinal giriş resmine benzeyen yüksek çözünürlüklü bir özellik haritası çıkartılarak, özellik haritası üzerinde bölge önerisi çıkartılır. Kısaca orijinal resimde yaklaşık 2000 adet bölge önerisi oluşturmak yerine, aynı işlem özellik haritasına uygulanır. Böylelikle her bölge ayrı ayrı evrişimli sinir ağından geçirilmemektedir. Resmi ilk başta tek bir seferde evrişimli sinir ağından geçirmek yeterli olmaktadır.

Evrişimli sinir ağı ciddi zaman alan kısım. Tüm resmi tek bir seferde sinir ağına verilmesinden dolayı zaman kazanılmaktadır. Daha sonra önerilen bölgeler alınıp boyutları eşitlenerek sinir ağına bağlanır. Sinir ağına bağlandıktan sonra, sınıflandırma yapılarak ilgili nesnenin sınırları belirlenir. R-CNN ‘de sınıflandırma yapmak için SVM, Fast R-CNN ‘de ise softmax layer kullanılır. Sınırları belirlemek için ise, aynı R-CNN ‘de olduğu gibi lineer regresyon kullanılır. Sonuç olarak; Fast R-CNN, R-CNN ‘e göre oldukça hızlıdır.

Tablo 2.1. R-CNN ile Fast R-CNN Karşılaştırması

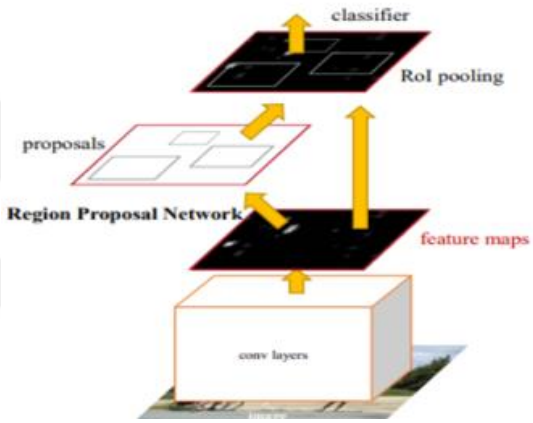
	R-CNN	**Fast R-CNN**
Eğitim Süresi	84 Saat	9 Saat
Eğitim Hızı	1x	8.8x
Test Resmi Üzerindeki Sınıflandırma Süresi	47 Saniye	0.32 Saniye
Test Hızı	1x	146x

Fast R-CNN, R-CNN ‘e göre hızlı olmasına rağmen tablo 2.1 ‘deki gibi, Fast R-CNN test aşamasında zamanın çoğunu bölge önerisi yapmakla harcamaktadır.

2.2.8. Faster R-CNN

Faster R-CNN mimarisinde bölge önerisi almak yerine, bölge önerilerini ağ içerisinde yaparak hız kazanılır.

Faster R-CNN ‘de giriş resmi alınır ve evrişimli sinir ağından geçirilerek özellik haritası çıkartılır. Bu aşamada bölge önerisi almak yerine; şekil 2.21 ‘deki gibi ayrı bir bölge önerisi ağı oluşturulur. Artık bölge önerileri oluşturulan bu ağ üzerinde yapılır. Ağ bölgeleri belirledikten sonra geri kalan işlem Fast R-CNN ile aynıdır. Faster R-CNN de eğitilmesi gereken dört farklı parametre ortaya çıkmaktadır. Hem bölge önerisi veren ağ eğitilir, hem de normal evrişim işlemi yapılan ağ eğitilmektedir.



Şekil 2.21. Faster R-CNN Mimarisi [47]

Bölge önerisinin iki farklı görevi bulunmaktadır;

- ✓ Her önerilen bölgede nesnenin olup olmadığına karar vermek
- ✓ Aynı zamanda önerilerin pencere büyüklüğünü belirlemek

Asıl ağ mimarisine geçildiğinde ise yine yapılacak iki farklı görev bulunmaktadır;

- ✓ Sinir ağı bölge içerisinde nesnenin olup olmadığını tespit etmek
- ✓ Bulduğu nesnenin sınırlarını belirlemektir

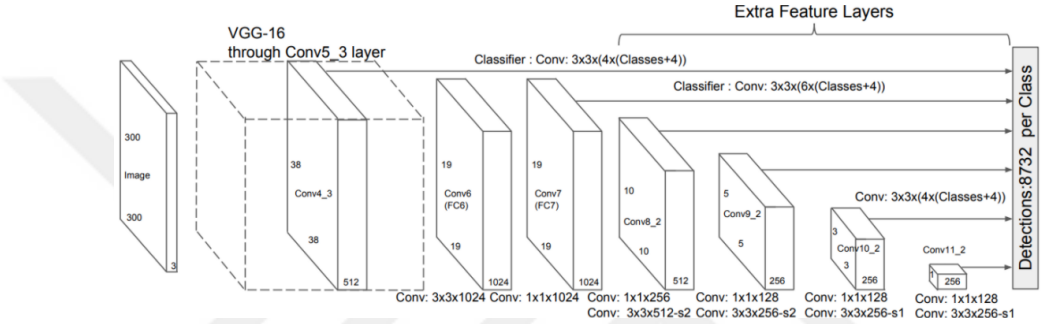
Tablo 2.2. Fast R-CNN ile Faster R-CNN Karşılaştırması

	Fast R-CNN	**Faster R-CNN**
Test Resmi Üzerindeki Sınıflandırma Süresi	0.32 Saniye	0.032 Saniye
Test Hızı	1x	10x

Tablo 2.2 incelendiğinde, Fast R-CNN ile Faster R-CNN arasında hız ve sınıflandırma süresi açısından aynı donanım üzerinde nerede ise 10 kat hız farkı vardır.

2.2.9. SSD

Faster R-CNN ‘den daha hızlı mimarilerde bulunmaktadır. Single shot multibox detector kısaca SSD bu mimarilerden biridir. İsminden de anlaşıldığı gibi SSD ile tek seferde nesne tanıma yapılmaktadır. R-CNN ‘de bölge önerisi ve nesne sınıflandırma iki farklı aşamada yapılmaktaydı.



Şekil 2.22. SSD Mimarisi [48]

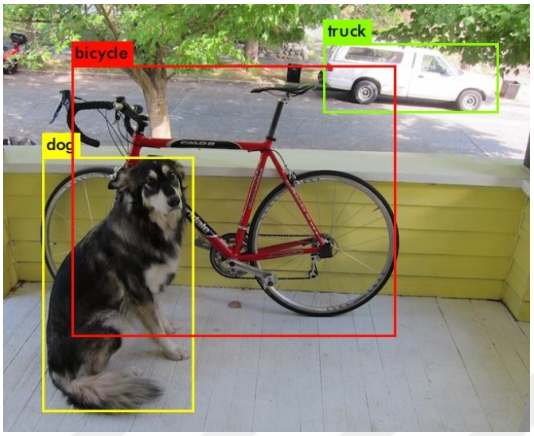
Giriş olarak her zamanki gibi bir resim alınır. Resim sırayla evrişimli sinir ağlarından geçirilmesinden dolayı (Şekil 2.22) farklı boyutlarda özellik haritaları çıkmaktadır. Tüm özellik haritalarında 3x3 evrişimli filtre yardımıyla, sınırlayıcı dikdörtgenler elde edilmektedir. Elde edilen her bir dikdörtgen için hem sınıflar hem de sınırlandırmalar belirlenmektedir. Bu dikdörtgenler her aktivasyon haritasında olduğundan dolayı hem küçük hem de büyük nesneler tanınmaktadır. Eğitim esnasında giriş resmi üzerinde doğru olan sınırlar ile tahmin edilen sınırlar karşılaştırılarak, belirli bir eşiğin üzerinde kalan dikdörtgenler pozitif olarak etiketlenmektedir [49].

Kısaca her bölge için farklı işlemler yapmak yerine, bütün tahminler tek seferde evrişimli sinir ağı içerisinde yapılır. Faster R-CNN ile SSD kıyaslandığında, eğer elde edilecek olan modelde, hız önemli ise SSD, isabet önemli ise Faster R-CNN mimarisi tercih edilmelidir.

2.2.10. You Only Look Once (Yolo)

Açılımı sadece bir kez bak (You Only Look Once) anlamına gelen Yolo mimarisi, evrişimli sinir ağlarını kullanarak nesne tespiti yapan algoritmalardan bir tanesi ve en

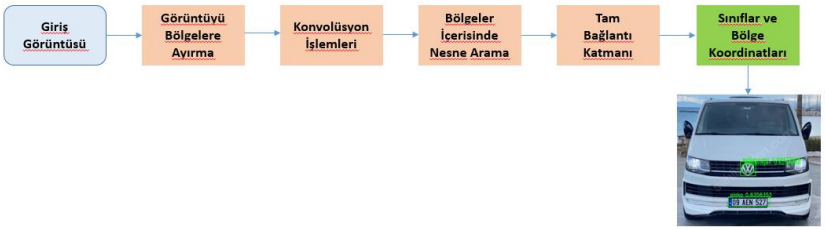
popüleridir. Algoritma nesne tespitini oldukça hızlı bir şekilde ve tek seferde yapabilmektedir. Algoritma çalışmaya başladığı anda, görüntü veya videolardaki nesneleri ve nesnelerin koordinatlarını aynı anda tespit edebilmektedir.



Şekil 2.23. YOLO Nesne Tanıma [50]

Şekil 2.23 ‘te görüldüğü üzere, resimde nesneleri çevreleyen dikdörtgenler bulunmaktadır. Dikdörtgenlerin sol üst köşesinde bulunan kısımda ise bulunan nesnenin ne olduğu bilgisi yazmaktadır. Nesneleri çevreleyen dikdörtgenler bounding box olarak adlandırılırken, bulunan nesnenin ne olduğunun bilgisinin yazılı olduğu kısma ise etiket (label) olarak adlandırılmaktadır. Şekil 2.23 dikkatli incelendiğinde, bir fark daha dikkat çekmektedir. Her bir nesne farklı bir renkle gösterilmiştir.

Kısaca, bir görüntü ya da video Yolo mimarisine verildiğinde, o görüntü veya videodaki nesneler üzerinde bounding boxlar oluşturulur ve bu bounding boxların üzerinde de ilgili nesnenin ne olduğu bilgisi mevcuttur, her bir nesne farklı bir renkle ifade edilir [51]. Yolo algoritması, giriş görüntüsünü öncelikle belli boyutlarda bölgelere (grid) ayırmaktadır (ızgara gibi düşünülebilir). Sonra her bir bölgedeki nesneleri belirleyen bounding boxlar bulunur. Bir nesne birden fazla bounding box ile ifade edilebildiğinden dolayı bir güven skoru hesaplanmaktadır. Hesaplanan güven skoruna göre, güven skoru en yüksek olan bounding box yani bir nesneyi ifade eden en iyi bounding box seçilerek, diğer bounding boxlar görüntü üzerinden çıkartılmaktadır. Şekil 2.24 ‘te algoritmanın mimari yapısı gösterilmektedir.



Şekil 2.24. YOLO Algoritması Mimari Yapısı

3. NESNE TESPİTİ

Derin öğrenme mimarisinden önceki çalışmalarda nesne tespiti yapmak oldukça zahmetli bir iş olmakla birlikte, görüntü üzerinde nesne bulunan bir bölgenin geleneksel görüntü işleme yöntemleri ile tespit edilmesi uzun sürmekteydi. Bunun yanında nesne tespiti yapılan bölge çok büyük bir dikdörtgenle ifade edilebildiği gibi, algoritmanın doğruluk değeri de derin öğrenme yöntemlerine kıyasla çok yüksek değildi. Derin öğrenme mimarilerinin gelişmesi ve bilgisayarların işlem gücünün artması sayesinde nesne tespiti artık günümüzde çok önemli bir hale gelmiştir [52].

3.1. Nesne Tespiti Nedir?

Nesne tespiti, görüntü veya video üzerindeki belirli bir sınıftaki (eşya, bardak, bilgisayar, hayvan, araba gibi) anlamsal nesnelerin benzerlerini veya kendilerini algılamakla ilgilenen, görüntü işleme ile alakalı bir bilimdir.

Nesne tespitinde görüntü üzerinde bulunan nesnelerin koordinatlarının, genişlik ve yükseklik bilgilerinin bulunması amaçlanmaktadır. Nesne tespiti denildiğinde, görüntü üzerindeki nesnelerin, görüntü üzerinde bulunduğu konum kastedilmektedir. Genellikle nesnelerin ne olduğunun anlaşılması ile karıştırılmaktadır. Nesnelerin ne olduğunun anlaşılması konusu ise nesne sınıflandırma konusudur [53]. Bu iki kavram bazen karıştırılabilmektedir.

Nesne tespitinde aşağıdaki gibi bir sonuç çıkarılmaktadır;

- ✓ Bir görüntü üzerinde x, y koordinatlarında, genişliği a, yüksekliği b olan bir tane nesne vardır diye sonuç çıkmaktadır ve bu nesnenin sınıfının ne olduğu ise bilinememektedir

Nesne tespiti görüntü işlemenin ilk aşamasıdır.

Görüntü Tesbit Edilir ⟶ Görüntü Sınıflandırılır

Şekil 3.1. Görüntü İşleme İşlem Basamağı

Sırasıyla, şekil 3.1 ‘deki gibi, önce görüntü üzerinde nesne tespit edilir, görüntü üzerinde nesne tespit edildikten sonra sınıflandırılır.

3.1.1. Kenar algılama

Kenar algılama, bir resim üzerindeki nesnelerin kenarlarını algılanmasını sağlayan, sınırlarının çizilmesini sağlayan bir yöntemdir. Kenar algılama yöntemi, görüntü üzerindeki parlaklığın keskin bir biçimde değiştiği noktaları tanımlamayı amaçlar [54].

Kenar algılama yönteminde görüntü parlaklığının yani görüntü üzerindeki piksel değerinin keskin bir şekilde değiştiği noktalar tespit edilmektedir.

3.1.2. Köşe algılama

Görüntü işlemede, bir görüntü üzerinde genellikle bir resmin ya da videonun farklı çerçeveleri arasında eşleşen noktalar bulunmalıdır. Çünkü iki görüntünün birbiriyle nasıl ilişkili olduğu bilinebilirse, her iki görüntüden de bilgi alınabilir.

Eşleştirme noktaları, genel anlamda sahnedeki kolayca tanınabilecek özelliklere atıfta bulunulur. Bahsedilen özellikler resim içerisindeki nesnelerin tanınabilir olmalarını sağlayan özelliklerdir. Bir nesneyi tanımak için;

- ✓ Kenarlar
- ✓ Köşeler

gibi özellikler bilinebilirse, nesnenin ne olduğunun bilinmesine gerek kalmadan çıkarımda bulunulabilir.

Köşe algılamada, köşeler iki kenarın kesişimi olduğundan dolayı bu iki kenarın yönlerinin değiştiği bir nokta temsil edilir. Köşeler, resimdeki renk geçişindeki bir varyasyonu temsil ettiğinden, görüntü üzerinde bu varyasyon aranır.

Özetle, pencerenin şu an bulunduğu konumla bir sonraki bulunduğu konum arasındaki yoğunluk farklı fazla ise, bir köşeden geçilmiştir denilebilir. Yoğunluk farkı bulunan değerler görüntü üzerinde belirlenerek ilgili görüntü üzerinde yoğunluk farkından dolayı şekil algılama işlemi yapılmış olmaktadır.

3.1.3. Kontur algılama

Kontur algılama, aynı renk ve yoğunluğa sahip tüm kesintisiz noktaları (pikselleri) sınırları ile birlikte birleştirmeyi amaçlayan yöntemdir. Konturlar şekil analizi ve nesne algılama ve nesne tanıma için kullanılmaktadır.

3.1.4. Havza algoritması

Havza algoritması, segmentasyon için yani bir görüntüdeki farklı nesneleri tespit etmek için kullanılan klasik bir algoritmadır. Segmentasyon işlemi dijital bir görüntüyü birden çok bölüme ayırma işlemidir.

Siyah – beyaz olarak oluşturulan herhangi bir görüntü, yoğunluğu yüksek yerlerdeki zirve ve tepe noktaları, yoğunluğun düşük olduğu vadi gibi yerleri ifade ettiği topoğrafik yüzey olarak görülmektedir. Havza dönüşüm yüksekliğini temsil eden her bir noktanın parlaklık, bir topoğrafik haritası gibi davranır ve çıkıntıların üst çizgileri boyunca uzanan çizgiler bulur.

3.2. Görüntü İşlemede Temel Konular

Görüntü işlemede kullanılan farklı teknik ve yöntemler bulunmaktadır. Görüntü üzerinde yapılan matematiksel işlemler ile ilgili görüntü veya video üzerinde istenilen şekilde esneklik sağlanılabilir.

3.2.1. Eşik değer alma

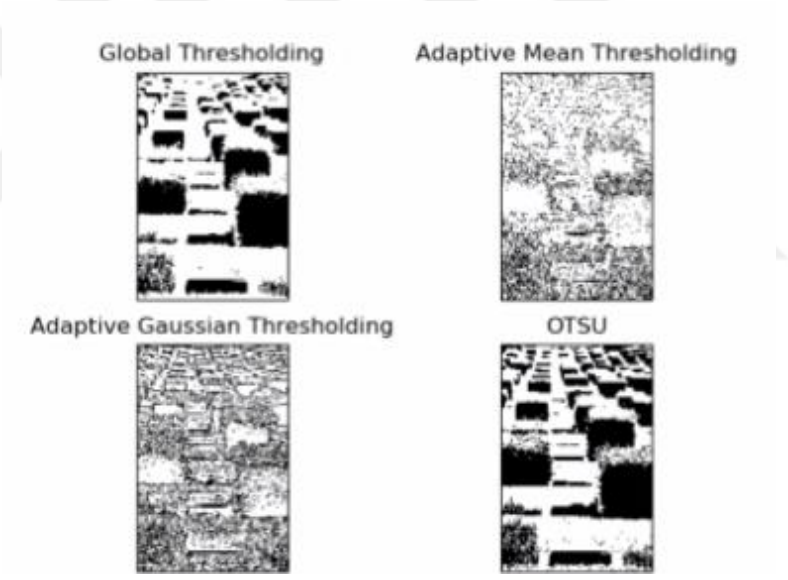
Herhangi bir görüntü üzerinde ilk aşama olarak görüntü eğer renkli ise ki (Şekil 3.2), renkli görüntünün literatürdeki diğer adı üç kanallı, bu görüntüyü siyah-beyaz (literatürde tek kanallı görüntü) çevirmek gerekmektedir [55]. Amaç görüntü üzerindeki her bir piksel değerini [0-255] aralığına getirmektir.

Eşik değer alırken, görüntü üzerinde farklı yöntemler kullanılmaktadır. Kullanılan yöntemler,

- ✓ Global Thresholding
- ✓ Adaptive Mean Thresholding
- ✓ Adaptive Gaussian Thresholding
- ✓ OTSU

olmak üzere 4 farklı yöntem bulunmaktadır (Şekil 3.3).

Şekil 3.2. Orijinal Görüntü



Şekil 3.3. Orijinal Görüntüye Uygulanan Eşik Değer Alma Yöntemleri [59]

Global thresholding yönteminde, 0-255 arasında bir değer seçilerek, seçilen değerin üzerinde kalan pikselleri 255, altında kalan piksel değerlerini 0 yapılacak şekilde bir eşik değeri belirlenir. Örneğin; eşik değeri 50 olarak belirlendiğinde, 50 ‘nin üzerindeki piksel değerleri (50 ile 255 kadar) 255, 50 ‘den küçük (0 ile 50 arasında) piksel değerleri 0 olacak şekilde görüntünün piksel değerleri değiştirilir. Global thresholding, uygulanması ve basit mantığı sayesinde, genelde tüm görüntü işleme yöntemlerinde uygulanabilir bir yöntemdir.

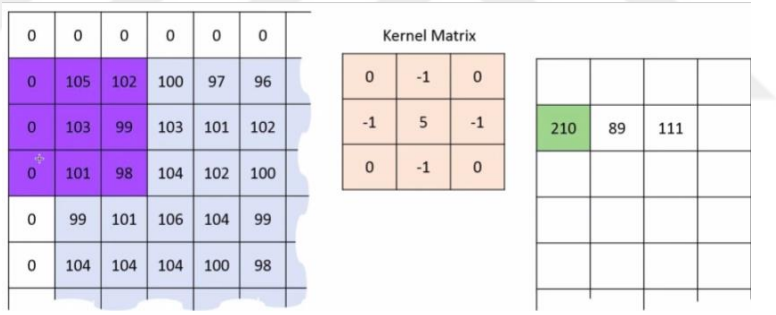
Adaptive mean thresholding yönteminde, önce bir komşuluk alanı belirlenir. Verilen bu alan için algoritma kendisi bir eşik değeri belirleyerek, komşuluk alanına göre algoritmanın belirlediği eşik değeri görüntüye uygulanmaktadır.

Adaptive Gaussian Thresholding, yine bir komşuluk alanı belirlenir ama adaptive mean thresholding yönteminden farklı olarak, gauss dağılımından yola çıkarak [56] merkeze yakın olan piksel değerlerinin ağırlığı daha fazla olacaktır.

Otsu yöntemi ise, diğer 3 yöntemden daha gelişmiş bir yöntemdir [57]. Otsu yöntemi görüntünün histogramını bularak kendisi bir eşik değer hesaplar. Görüntünün histogramı çıkarıldıktan sonra ani değişimlerin olduğu yerde pik noktaları belirlenir. Belirlenen pik noktalarına göre yine kendisi bir eşik değer belirler.

3.2.2. Görüntü keskinleştirme ve kenar tespiti

Görüntü keskinleştirmede, görüntü üzerinde özel bir filtre gezdirilerek çarpma işlemi yapılır.

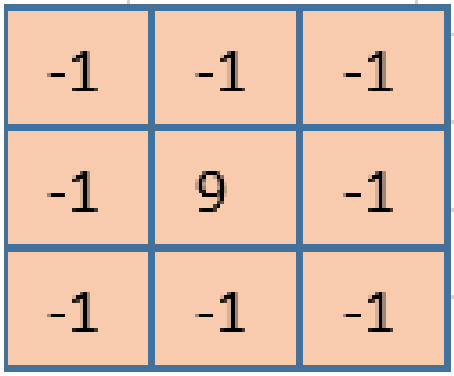


Şekil 3.4. Görüntü Üzerine Filtre İşlemi Uygulanması [58]

Şekil 3.4 ‘te soldan sağa doğru, ilk matris görüntü matrisini, ikinci matris görüntüye uygulanacak filtreyi, son matris ise; görüntüye filtre uygulandıktan sonra ki işlemi göstermektedir.

Aslında hem derin öğrenme de, hem de görüntü işlemede yapılan bu işleme evrişim işlemi denilmektedir. Burada görüntüye uygulanacak olan filtredeki sayılar değiştirilerek özel görüntüler elde edilmektedir.

Görüntü keskinleştirme, evrişim işlemine benzemekle birlikte bu işlemden farkı filtre içerisindeki sayılardır. Bu işlemde görüntüye özel bir filtre uygulanmaktadır.



Şekil 3.5. Özel Filtre

Özel filtrenin merkezindeki değer 9 iken diğer değerler -1 ‘dir. Şekil 3.5 ‘te gösterilen bu filtre görüntü üzerine uygulanarak görüntünün keskinleştirmesi sağlanır. Bu yöntemde, başka bir ifade ile görüntüdeki keskin kenarlar belirginleştirilmiş olmaktadır.

3.2.3. Morfolojik işlemler

Temel olarak siyah-beyaz görüntü üzerinde uygulanan temel işlemlerdir [59]. Görüntünün şekli üzerinde oynama yapılmasına imkân sağlar ve çeşitli görüntü işleme projelerinde ön işlem olarak yaygın bir şekilde kullanılır. Görüntü üzerinde kaydırılacak ve hesaplama yapacak bir filtre oluşturduktan sonra, yine görüntü üzerinde erosion ve dilation olmak üzere iki temel işlem yapılır.

Erosion, adından da anlaşılacağı üzere görüntüdeki kenarları erozyona uğratır yani aşındırır. Böylelikle kalın ve silik kenarlar daha keskin olur. Yalnız aşındırmadan dolayı kenar kalınlığı biraz incelir.

Dilation işlemi ise; erosion işleminin tam tersi bir işlemdir. Görüntü üzerindeki kenarları kalınlaştırır. Böylece görüntüdeki silik ve ince kenarlar kalınlaşarak daha da belirginleşir.

3.3. Derin Öğrenme Algoritmaları ile Nesne Tespiti

Herhangi bir görüntü üzerinde nesne tespiti yapılabilmesi için görüntü işleme yöntemlerinin yetersiz kaldığı durumlar incelendiğinde, bilgisayarların ve teknolojinin gelişmesi sayesinde derin öğrenme yöntemleri ile nesne tespiti tercih edilmektedir [60]. Bu tercihin nedenleri;

- ✓ Resimler üzerindeki nesnenin doğru ve hızlı tespit edilmesi
- ✓ Nesne tespiti yapılan bölgenin doğru bir şekilde bulunması ve gösterimi
- ✓ Geleneksel görüntü işleme yöntemlerinin birçok problemlerde yetersiz kalması
- ✓ Hızla artan veriler üzerinden anlamlı öznitelikler çıkartan derin öğrenme mimarilerinin geliştirilmesi

olarak söylenebilir. Ayrıca derin öğrenme algoritmaları problemlere özgün olup problem üzerinden farklı sonuçlar verebilmekle beraber, her problem özelinde farklı derin öğrenme mimarileri de kullanılabilmektedir.

3.3.1. ResNet evrişimli sinir ağları sınıflandırıcısı ile nesne tespiti

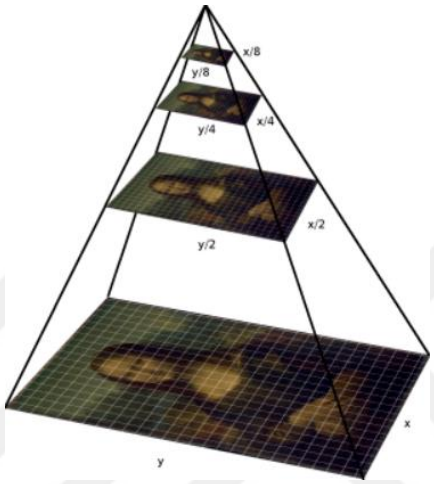
Herhangi bir resim üzerinde (resim olmak zorunda değil videoda olabilir) nesne tespiti yapılmak istenildiğinde; bu resim üzerinde resmin boyutundan küçük resimler seçilmektedir. Bir dikdörtgen olduğunu varsayılırsa, bu dikdörtgen resim üzerinde kaydırılarak dolaştırılır. Bu dikdörtgeni resim üzerinde kaydırılma işlemi sonucunda ortaya, aynı boyutlarda kesilmiş küçük resimler oluşmaktadır (Şekil 3.6). Bu küçük resimlerin birbirine eklenmiş hali orijinal resmi ifade etmektedir.

Şekil 3.6. Orijinal ve Bölgelere Ayrılmış Resim [61]

Eşit boyutlarda bölgelere ayrılmış resim üzerinden sınıflandırma yapılarak her bir küçük resim içerisinde nesne tespiti yapılır ve bu nesne tespitinin yapıldığı küçük resimlerin orijinal resim üzerindeki konumları bilinmektedir. Sonuç olarak orijinal resimden meydana gelen küçük resimlerin konumları (x,y), yükseklikleri (w,h) ve son olarak ilgili küçük resim içerisinde aranılan nesnenin olup olmadığının bilgisi (1,0) elde edilerek nesne tespiti yapılmaktadır.

3.3.2. Piramit gösterimi

Piramit gösterimi yönteminde, görüntünün çok ölçekli temsili gerçekleştirilir. Bu yöntemin kullanılması, herhangi bir görüntünün farklı ölçeklerinde oluşturulan görüntüler üzerinde nesne tespiti yapılmasını sağlar. Görüntü piramidinin altında, orijinal boyutunda görüntüler bulunur. Sonraki her katmanda görüntü yeniden boyutlandırılır ve uygulanmak istenilen yönteme göre düzleştirilir.



Şekil 3.7. Piramit Gösterimi [62]

Yeniden boyutlandırma işlemi herhangi bir ölçeğe göre olabilmektedir. Şekil 3.7 ‘de görüldüğü üzere ölçek her bir katman için X/2, Y/2 olarak belirlenmiştir.

Genel olarak piramit gösterimi algoritmasında, görüntünün şekli küçültülerek, nesnenin daha kolay ve hızlı bir şekilde tespit edilmesi amaçlanmaktadır. Diğer bir taraftan farklı ölçülerde resimler oluştuğundan dolayı görüntü çoğaltma işlemi olarak da düşünülebilir.

3.3.3. Kayan pencere

Genişliği ve yüksekliği sabit bir dikdörtgen, görüntü üzerinde kaydırılmaktadır. Kaydırılan bu dikdörtgen içerisinde kalan bölgede nesne tespiti yapılmakta olup, ilgili dikdörtgen içerisinde kalan bölgede sınıflandırma yapılmaktadır.

Sabit görüntü üzerinde bir dikdörtgen olduğu düşünülürse, bu dikdörtgen sabit olan görüntü üzerinde birer piksel atlayarak dolaşmaktadır. Burada kaç piksel atladığı sabit değildir. Yani dikdörtgen ikişer ya da beşer pikselde atlayabilir. Sonuçta dikdörtgen içerisinde kalan bölgede ilgili nesnenin olup olmamasına bakılarak, nesne tespiti

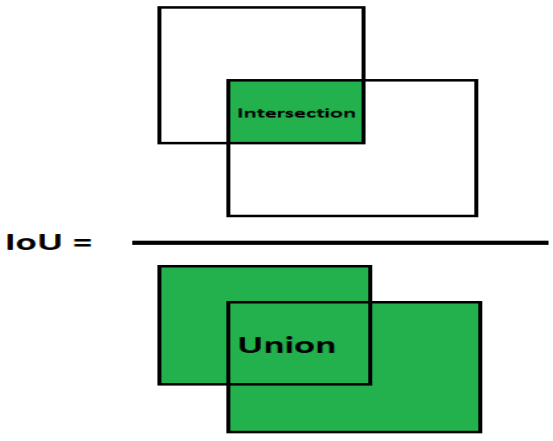
yapılmaktadır. Görüntü üzerinde kayan dikdörtgen içerisinde ilgili nesne bulunduğunda ise, nesneye ait koordinatlar elde edilmektedir. Oluşturulan dikdörtgenin genişliği ve yüksekliği bilindiğinden dolayı, nesneye ait koordinat bilgileriyle birleştirilerek sabit görüntü üzerinde aranan nesne var ise bu nesne bulunduğu bölgede kolayca gösterilmektedir.

Veri sayısı arttıkça, yani evrişimli sinir ağını eğitmek için kullanacağımız resim sayısı arttıkça, kayan pencere ve piramit gösterim yöntemi çok fazla yavaş kaldığından dolayı, fazla fayda sağlamamaktadırlar.

3.3.4. Maksimum olmayan bastırma

Görüntü üzerinde nesne tespiti yapıldığında ilgili nesne birden fazla dikdörtgen çizilerek ifade edilebilmektedir. Maksimum olmayan bastırma yönteminde, nesnenin etrafında çizilen birden çok dikdörtgen içinde küçük olan dikdörtgenleri silerek en büyük olan dikdörtgeni nesne üzerinde bırakarak, ilgili görüntü üzerinde nesne tespiti yapılır.

İlgili nesne tespitinde çizilen birden fazla dikdörtgenler arasında birlik üzerinden kesişme (intersection over union) değeri hesaplanır. Hesaplanan değer için bir eşik değeri belirlenir ve bu eşik altında kalan dikdörtgenlerin oluşturduğu alanlar elenir.



Şekil 3.8. Birlik Üzerinden Kesişme (IoU)

İki dikdörtgen arasında kalan alan (intersection) hesaplanır. Daha sonra bu iki dikdörtgenin birleşimine (union) bakılarak, bu iki alan şekil 3.8 ‘de gösterildiği gibi

birbirine bölünür. Böylelikle, eşik değer altında ya da üzerinde kalacak olan alan hesaplanmaktadır.

Şekil 3.9. Maksimum Olmayan Bastırma Yöntemi [63]

Şekil 3.9 'da kayan pencere yöntemi kullanılarak, görüntü üzerinde yüz tespiti yapılmaktadır. Kırmızı renkle belirlenen dikdörtgenler görüntü üzerinde kayan pencere yöntemi kullanarak nesne tespiti yapmaktadır. Şekilde kırmızı renkle çizilen üç farklı dikdörtgen içerisinde yüz bulunmaktadır. Ama ana görüntüde bir tane yüz bulunmaktadır. Dolayısı ile ana görüntü üzerinde yeşil dikdörtgen ile çizilen görüntüde gösterildiği gibi, görüntü üzerinde bir tek yüz olduğundan dolayı, bir tane dikdörtgen çizilerek bu dikdörtgen içerisinde nesne tespit edilmelidir.

Maksimum olmayan bastırma yöntemi ile görüntü üzerindeki üç kırmızı dikdörtgen elenerek, bu dikdörtgenler içerisinden en uygun olan dikdörtgen seçilerek, tek bir yeşil dikdörtgen elde edilir.

3.3.5. Nesne tespiti için seçmeli arama

Kayan pencere ve piramit gösterimi yöntemlerindeki yavaşlığı ortadan kaldırmak için kullanılan bir yöntemdir. Seçmeli arama (selective search) yöntemi, süper piksel algoritması [64] kullanarak görüntüyü bölümlere ayırma yöntemi olarak da açıklanmaktadır. Bu yönteme segmentasyon yöntemi de denilebilmektedir.

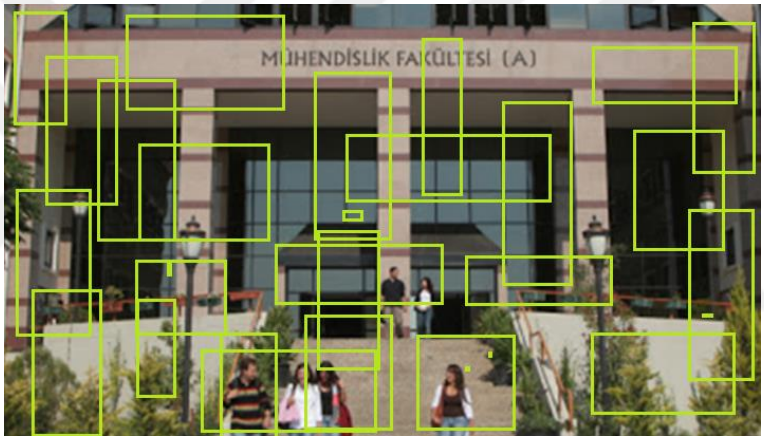
Süper piksel algoritması, görüntü üzerinde piksel yoğunluğu gibi ortak özellikleri bulunan bölgeleri paylaşan bir piksel grubu olarak tanımlanabilir. Seçmeli arama yöntemi, beş temel benzerlik ölçüsüne dayalı olarak süper pikselleri hiyerarşik bir şekilde birleştirir.

- ✓ Renk benzerliği

- ✓ Doku benzerliği
- ✓ Boyut benzerliği
- ✓ Şekil benzerliği
- ✓ Bu dört benzerliğin doğrusal kombinasyonu

Seçmeli arama yöntemi, sınıf etiketleri değil, görüntü üzerinde bölgeler oluşturur. Bu yöntem bir nesne tespiti yöntemi olmadığından, nesne tespiti yöntemi ile karıştırılmamalıdır. Yine seçmeli arama yöntemi kayan pencere ve piramit gösterimi algoritmalarına alternatif bir algoritmadır.

Seçmeli arama yönteminde görüntü üzerinde belli bölgelerde nesneler olabilir diye pencereler oluşturulmaktadır.



Şekil 3.10. Seçmeli Arama Yöntemi [65]

Kayan pencere yönteminde, belirli boyuttaki dikdörtgen veya kare olan pencereler görüntü üzerinde dolaştırılmaktadır. Seçmeli arama yönteminde ise şekil 3.10 ‘da görüldüğü üzere pencere kaydırma olmaksızın, görüntü üzerinde rastgele pencereler oluşturularak, bu pencerelerde nesne tespiti yapılmaktadır.

3.4. Nesne Tespitinde Kullanılan Kütüphaneler

Günümüzde nesne tespit yöntemlerinde, geleneksel yöntemlerden çok evrişimli sinir ağı yöntemleri tercih edilmektedir. Günlük hayatta üretilen verilerin çokluğu ve bilgisayarların işlem kapasitesinin arttırılmış olması evrişimli sinir ağlarının tercih edilmesindeki en önemli etkenlerdendir.

Evrişimli sinir ağı ile nesne tespiti yapabilmek için farklı algoritmalar kullanıldığından dolayı bu algoritmalar için kütüphaneler oluşturulmuş olup, herhangi bir görüntü üzerinde nesne tespiti yapılabilmesi için bu kütüphaneler kolaylıkla yüklenerek görüntü üzerinde nesne tanıma işlemleri yapılabilmektedir.

3.4.1. Tensorflow

Derin öğrenme için kullanılan, Google tarafından geliştirilmiş olan açık kaynaklı kütüphanelerden bir tanesidir. Tensorflow, CPU ve GPU olmak üzere iki farklı şekilde kurulabilmektedir.

Tensorflow CPU kurulumu işlemci için kurulmaktadır. Kullanacak olduğumuz algoritma üzerinde tensorflow işlemci üzerinde koşacaksa tensorflow CPU kurulumu yapılmalıdır.

Eğer tensorflow işlemci değil de ekran kartı üzerinde koşturulacaksa da tensorflow GPU kurulumu yapılmaktadır. Eğer kullanılacak olan bilgisayarda ekran kartı uyumlu ise kesinlikle tensorflow GPU kullanılmalıdır. Ekran kartları CPU ‘ya göre içerisinde çok fazla çekirdek bulundurduğundan dolayı, paralel işlemleri CPU ya göre oldukça hızlı bir şekilde gerçekleştirmektedir.

3.4.2. Keras

Tensorflow makina öğreniminde model oluşturmayı oldukça kolaylaştırmaktadır. Ancak tensorflow kullanmak için bazen aynı kodları tekrar tekrar yazmaktadır ve bunun sonucu olarak yazılan kodlar karmaşıklaşabilmektedir. Bu kod karmaşıklığını basitleştirmek için bazı uygulama programlama ara yüzü (API ‘ler) bulunmaktadır. Bu API ‘ler sayesinde basit bir kod yazılması için tekrar tekrar kod yazmaya gerek bulunmamaktadır. Keras, tensorflow üzerine kurulmuş yüksek seviye ve oldukça yaygın olarak kullanılan ve geliştirilen API ‘lerden bir tanesidir.

3.4.3. Theano

Çok boyutlu diziler dâhil, matematiksel ifadeleri etkili bir biçimde tanımlayan, en iyilemeyi ve değerlendirmeyi sağlayan bir kütüphanedir. Theno kütüphanesi yaklaşık olarak 2017 yılına kadar geliştirilmeye devam edilmiş olsa da, bu tarihten itibaren bu kütüphane için geliştirme yapılmamaktadır. Bu kütüphanenin öne çıkan özellikleri;

- ✓ Numpy ile sıkı entegrasyon
- ✓ GPU ile kullanılabilmesi
- ✓ Sembolik türev alma kabiliyeti
- ✓ Dinamik olarak C kodu üretmesi

gibi öne çıkan özellikleri bulunmaktadır.

3.4.4. Pytorch

Pytorch ‘da keras ve tensorflow gibi, derin öğrenmede matematiksel işlemleri yapmamızı kolaylaştıran kütüphanelerdendir. Facebook ‘un araştırma laboratuvarında geliştirilmiştir [66]. Bu kütüphane açık kaynaklı makina öğrenmesi kütüphanesidir. Günümüzde Yapay zekâ, derin öğrenme gibi algoritmaların içerisinde paytorch kütüphanesi kullanılabilmektedir.

Pytorch kütüphanesi, keras ve tensorflow gibi özel bir kütüphane olmasının yanında, bu kütüphanenin avantajları ve dezavantajları bulunmaktadır. Bilindiği üzere matematiksel işlemler numpy kütüphanesi ilede yapılmaktadır ancak pytorch GPU power kullanabilmesinden dolayı numpy ’dan daha gelişmiştir ve matematiksel işlemleri daha hızlı yapabilmektedir. Pytorch, GPU kullanmasından kaynaklı diğer derin öğrenme kütüphanelerine kıyasla esneklik ve hız sağlar.

Pytorch kütüphanesinin avantajları;

- ✓ Debuging işlemi kolaydı
- ✓ Dinamik grafiklerde iyi bir desteğe sahiptir
- ✓ Facebook tarafından geliştirilmektedir
- ✓ Yüksek ve düşük seviye API ‘lerin karışımı sonucunda ortaya çıkmaktadır

Bu avantajlarının yanında dezavantajları da;

- ✓ Keras ve tesnorflow kütüphanelerine göre henüz tam istenilen seviyede değildir
- ✓ Kaynak ve dökümantasyon sayısı keras ve tensorflowa göre daha azdır

4. NESNE TESPİTİNE İHTİYAÇ DUYULAN YERLER

Gerek giderek artan görüntü ve video verileri, gerekse özellikle ekran kartlarının gelişmesine nazaran günümüzde geçmiş yıllara göre daha hızlı ve işlem kapasitesi yüksek bilgisayarlara [13] erişim günden güne kolaylaşmaktadır. Bilgisayarların gelişmesine paralellik gösteren yapay zekâ algoritmaları da, gittikçe artan görüntü ve resim verilerinden nesne tespiti yapılmasına olanak sağlamaktadır [31].

Nesne tespitine ihtiyaç duyulan uygulama ve alanların en başında askeri uygulamalar, güvenlik uygulamalarından, kişi ve nesne tanıma, yüz tanıma, parmak izi tanıma, hareket tanıma, sağlık alanında MR görüntüsünden kanser tespiti gibi alanlar ve uygulamalarda sıklıkla kullanılmaktadır.

Bahsedilen bu başlıca alanlarda kullanılmasının sebebi, maliyetin düşürülmesi ve insan gözünden kaçan önemli bilgilerin tespit edilebilmesidir.

Askeri alanda, günümüzde artık genellikle video üzerinden nesne tespiti yapılarak ilgili nesnenin takip edilmesi ya da eğer riskli bir bölgede ya da durumda ise ilgili nesnenin etkisiz hale getirilmesinin yanında, yüksek seviyeli giriş gerektiren yerlerde görüntü üzerinden yüz ve parmak izi tanıma yöntemleri kullanılarak, güvenlikle alakalı önlemler alınabilmektedir [26].

Sağlık alanında ise, röntgen veya MR görüntüsünden doktorun gözünden kaçabilecek ya da herhangi bir dikkatsizliğinden dolayı görüntüler üzerinde fark etmeyecek olduğu önemli bilgileri, nesne tespiti yöntemi kullanarak ve bahsedilen riski en aza indirerek, kullanıcılara yardımcı olacak bir yaklaşım oluşturulmaktadır.

Sosyal yaşamda ise, herhangi bir sosyal platform üzerinden fotoğraf ya da herhangi bir mekân paylaşıldığında o görüntüdeki varsa kişiler ve nesnelerin otomatik tanılandırılması yaptırılarak, otomatik olarak etiketleme işlemi gerçekleştirilmektedir.

4.1. Günümüzde Uygulanan Araç Tanıma Yöntemleri

Günümüzde kullanılan araç tanıma yöntemlerinde genellikle birbirine benzer nesne tespit algoritmaları kullanılmaktadır [21]. Araç tanıma yapılırken, görüntü üzerinde

aracı ifade eden kısmın (yani aracın tamamının) etrafında dikdörtgen çizilerek görüntü üzerinde araç tanıma yapılmış olup, ayrıca bu aracın görüntü üzerindeki konum bilgileri de evrişimli sinir ağları sayesinde bilinmektedir [51].

Araç tanıma yönteminde önce orijinal görüntü üzerindeki araçlar eğitim verisi için tek tek belirlenerek görüntü üzerindeki,

- ✓ Xmin
- ✓ Ymin
- ✓ Xmax
- ✓ Ymax

ilgili nesnelerin bilgileri ile birlikte evrişimli sinir ağı algoritmalarının herhangi birinde eğitilebilinmektedir. Kısaca, günümüzde uygulanan araç tanıma yöntemlerinde, görüntü üzerinde bulunan aracın tüm resmi üzerinden bir sınıflandırma yapılmaktadır.

Bu tez konusunda yapılan çalışmada ise, görüntü üzerinde aracın tüm resminin etiketlenmesi yerine, ilgili araç markasının bulunduğu marka amblemi üzerinden bir nesne tespiti yöntemi yapılmış ve geliştirilmektedir. Bu yaklaşım ve yöntem büyük kolaylık ve hız sağlamaktadır.

Resim üzerinde araca ait komple alanı taramak yerine, tarayacak olduğu alandan küçük bir bölgeyi taraması ve bulması ilk duruma göre daha kısa sürecektir. İlk durumda ilgili aracın her pozisyonu için (önden veya arkadan çekilen görüntü) eğitim setinde fotoğraf bulunmalıdır. Herhangi bir X markası aracı düşünüldüğünde, X markasına ait aracın hem ön taraftan hem arka taraftan hem de geçmiş yıllardan günümüze gelene kadarki görüntüleri bulunmaktadır. Eğer eğitim setinde bahsettiğimiz bu veya bunlar gibi başka durumlara ait görüntüler olmazsa elde edilen model başka durumlardaki gibi olan nesneleri görüntü üzerinde tespit edemez.

Çalışmada uygulanan yöntemde ise; araç marka amblemi etiketlenerek aracın markasını aracın amblemi üzerinden tanınması gerçekleştirilmiştir. Böylece aracın marka ve modeli ne olursa olsun amblemi, kuruluşundan beri araç firması çok köklü bir karar almadığı sürece değişmediğinden dolayı, aracın amblemi üzerinden aracın marka bilgisi kolaylıkla ve çok fazla veriye ihtiyaç duymadan bir yapay zekâ modeli

oluşturulabilmektedir. İkinci bir durum ise aracın marka amblemi araç üzerinde tüm resimden daha küçük bir yerde olduğundan araç tespiti daha hızlı olacaktır.

4.2. Günümüzde Uygulanan Plaka Tanıma Yöntemleri

Plaka tanıma yöntemlerinde uygulanan çalışmalar ilk başlarda geleneksel yöntemlerle yapılırken, günümüzde artık derin öğrenme yöntemleri ile yapılmaktadır [67]. Geleneksel yöntemlerden derin öğrenme yöntemlerine geçilmesinin nedeni;

- ✓ Görüntü üzerinde plaka tanıma süresinin uzun olması
- ✓ Plaka tespit başarı oranının derin öğrenme yöntemlerinde fazla veri miktarı ile doğru orantılı olarak artması

ile plaka tanımada geleneksel görüntü işleme yöntemlerinden ayrılarak derin öğrenme yöntemlerine doğru evirilmektedir.

Bu çalışmada görüntü üzerinde hem araç hem de plaka tanıma yapılmıştır. Araç markaları sınıfına ek plakada bir sınıf olarak görüntü üzerinde etiketlenecek olup, eğitim ve test veri setinde ayrı ayrı bulunarak, bir görüntü üzerinde hem plaka hem de marka tespiti yapılmıştır.

4.3. Optik Karakter Tanıma

Kısa adı ile OCR olarakta bilinen optik karakter tanıma (Optical Chracter Recognation) işlemi, tarayıcıdan geçirilmiş herhangi bir belgenin (pdf, jpg, png v.b) içerisindeki verileri okuyabilen veya düzenleyebilen verilere dönüşmesini sağlayan bir çalışmadır [18]. Bu çalışmada ilgili belge içerisindeki veriler okunarak, analiz edilebilir veya başka bir dökümana dönüştürülebilmektedir.

OCR teknolojisi plaka tanımada sıklıkla kullanılan yöntemdir. Plaka tanıma işleminde görüntü üzerinde plaka bulunan alan OCR yöntemi ile okunarak ilgili görüntü üzerinde plaka tanıma işlemi yapılmaktadır.

4.4. Sınıflandırma Problemlerinde Model Oluşturma

Sınıflandırma problemlerinde amaç, tahmin edilecek kısmın kategorik veri diye adlandırılan, daha önceden etiketlenmiş verileri sınıflandırmaktır. Kısaca sayısal olmayan verilerin tahmin edilmesine sınıflandırma da denilebilmektedir.

Sınıflandırma problemlerinde birden fazla makine öğrenme algoritmalarının bulunmasının yanında, bu algoritmaların yetersiz olduğu durumda derin öğrenme yöntemlerine başvurulmaktadır. Bu tezin konusu olan çalışmada derin öğrenme yöntemleri kullanılarak araç ve plaka tanıma (sınıflandırma) yapılmıştır [14].

Görüntü üzerinde, derin öğrenme yöntemleri kullanarak sınıflandırma yapmak için; görüntü üzerinde etiketlenmiş veriler bulunmalıdır. Bu etiketlenen verilerin ne olduğu ve görüntü üzerindeki konum bilgisi model oluşturmada bilinmelidir. Model oluşturulurken elde edilen görüntü ve etiketler üzerinde önce eğitim işleminden sonra validasyon yani doğrulama işlemi ve en sonunda da elde edilen modeli kullanarak modelin hiç görmediği veriler ki bu çalışmada görüntüler üzerinde test işlemi gerçekleştirilmelidir. Özetle;

- ✓ Görüntüler, eğitim ve doğrulama verisi olarak ikiye ayrılır [32] (genelde literatüre göre eğer veri çoksa %80 eğitim, %20 doğrulama verisi olacak şekilde)
- ✓ Eğitim işlemi sonrasında oluşturulacak olan model, doğrulama verisi ile değerlendirilmektedir
- ✓ En son olarak ise doğrulama sonucunda kaydedilen model, hiç görmediği test verileri kullanılarak, modelin asıl başarısı ölçümlenmektedir

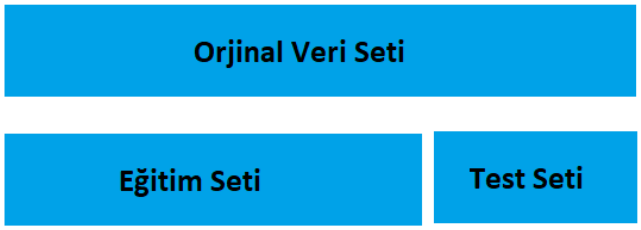
Bu adımlardan sonra, oluşturulan model kaydedilerek, ilgili probleme özgü olacak şekilde kullanılabilir.

4.4.1. Model doğrulama yöntemleri

Modellerin ürettiği sonuçların, doğru değerlendirilmesi çalışmalarıdır. Diğer bir deyişle, sınıflandırma problemlerinde eğitim verisi üzerinden model kurulacak olup, kurulan bu model üzerinden sınıflandırma yapılacaktır. Eğitim verisi üzerinden elde edilen sonuçların doğruluğunun değerlendirilmesi gerekmektedir [60]. Bununda amacı model sonucunun daha doğru değerlendirilmesini sağlamaktır. Doğrulama yöntemi olarak birden fazla model doğrulama yöntemi bulunmaktadır.

4.4.1.1. Holdout yöntemi

Holdout yönteminde orijinal veri seti eğitim ve test seti olarak ikiye bölünür ve Eğitim seti kullanarak oluşturulan modelin performansı test seti ile ölçülmektedir.

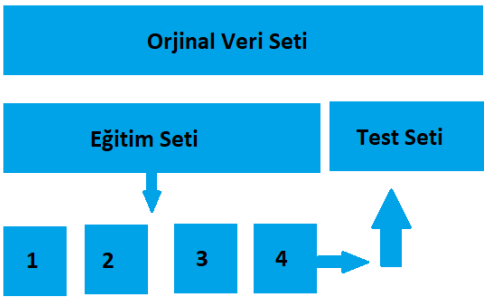


Şekil 4.1. Holdout Yöntemi

Genellikle şekil 4.1 ‘deki gibi, orijinal veri setinin 2/3 eğitim, 1/3 test seti olacak şekilde ayrılır. Orijinal veri setinin az olduğu durumda ve orijinal verideki verilerin hangilerinin eğitim, hangilerinin test seti olarak ayrılacağı probleminden dolayı holdout yöntemi yetersiz kalmaktadır. Bu yetersizliği gidermek için başka model doğrulama yöntemleri geliştirilmiştir.

4.4.1.2. K-katlı çapraz doğrulama yöntemi

Bu yöntemde de yine orijinal veri seti eğitim ve test verisi olarak iki parçaya ayrılmaktadır. Holdout yönteminden farklı olarak eğitim seti k adet alt parçalara ayrılmaktadır.



Şekil 4.2. Çapraz Doğrulama Yöntemi

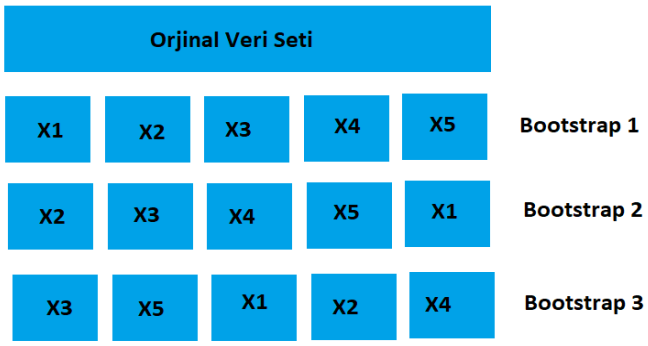
Şekil 4.2 ‘de görüleceği üzere, eğitim seti k (k burada 4 olmaktadır) adet alt parçalara ayrılarak, belirlenen alt parçalardan bir tanesi dışarıda bırakılarak, elde kalan diğer alt parçalarla oluşturulan model, dışarıda bırakılan alt parça ile test edilir. Bu dört parça kendi içerisinde eğitim ve test seti yaklaşımı yapmaktadır. Bu işlem alt parçalardaki her bir parça için yapılana kadar devam etmektedir. En son parça içinde bahsedilen işlem gerçekleştiğinde elde edilen hataların ortalaması alınarak, eğitim hatası hesaplanmaktadır. Hesaplanan bu eğitim hatasına da doğrulama hatası denilmektedir. Daha sonra ise, test verisi üzerinden de test hatası hesaplanarak bu iki hata üzerinden yorumlamalar yapılmaktadır.

4.4.1.3. Leave one out yöntemi

Bu yöntem k katlı çapraz doğrulama yönteminin özel bir halidir. Bu yöntemde eğitim setindeki örnek sayısı kadar doğrulama yapılmaktadır. Diğer bir deyişle veri seti, her bir gözlem sayısı kadar alt parçalara ayrılmaktadır. Her bir alt parça için yine doğrulama hatası hesaplanmaktadır. Her bir gözlem sayısı kadar alt parça olduğundan dolayı kullanışlı bir model doğrulama yöntemi değildir.

4.4.1.4. Bootstrap yeniden örnekleme yöntemi

Elde edilen orijinal veri setinden tekrar yerine koymak koşulu ile veriler çekilmektedir. Her bir veri çekme sonucunda elde edilen hatalar, en son çekilen verideki hata ile toplanıp ortalaması alınarak test hatası hesaplanmaktadır (Şekil 4.3).



Şekil 4.3. Bootstrap Yöntemi

4.4.2. Model başarı değerlendirme yöntemleri

Kurulan makina öğrenmesi modellerinin tahmin işlemlerinin değerlendirilmesi yöntemleridir. Karşılaşılan probleme göre herhangi bir modelin başarısını değerlendirme yaklaşımı değişebilmektedir. Problemlere bağlı olarak model başarı değerlendirme yöntemi değişmektedir.

4.4.2.1. Karmaşıklık matrisi (Confusion matrix)

Karmaşıklık matrisi (Confusion matrix), sınıflandırma problemlerinde model başarısını ölçmek için kullanılan bir başarı ölçüm metriğidir. Burada gerçek değerler ve tahmin edilen değerler karşılaştırılmaktadır. Bu karşılaştırma sonucunda yapılan model başarı hesaplamalarıyla modelin başarı değerlendirmesi yapılmaktadır.

Tablo 4.1. Karmaşıklık Matrisi

		Tahmin Edilen Sınıf	
		1	0
Gerçek	1	a	b
Sınıf	0	c	d

Gerçek değerlerde bulunan 1 ve 0 değerleri ile tahmin edilen kısımda bulunan 1 ve 0 değerleri tablo 4.1 'deki gibi karşılaştırılmaktadır.

4.4.2.2. False pozitif ve false negative kavramları

Tablo 4.1 'den yola çıkılarak, a, b, c ve d değerleri açıklanacak olursa; bir sınıf değerinin gerçekte 1 olması durumunda, Tahmin edilen kısımda da bu değer 1 olarak tahmin edildiğinde, aslında doğru olan bir işlem yapılmış olmaktadır. Bu işleme true pozitif denilmektedir. Yine aynı tablo incelendiğinde, gerçek değeri 1 olan bir sınıfın, tahmin edilen kısmının 0 olarak tahmin edildiğinde ise, yanlış bir işlem yapılmış olmakla birlikte yapılan bu yanlış işleme false negative denilmektedir. Buradan yola çıkarak;

- ✓ a: True Pozitif (TP)
- ✓ b: False Negatif(FN)
- ✓ c: False Pozitif(FP)
- ✓ d: True Negatif(TN)

şeklinde ifade edilmektedir.

4.4.2.3. Netlik/Doğruluk (accuracy) paradoksu

Herhangi bir makina öğrenmesi algoritmasında, eldeki verilerin başarısını ölçmek için her zaman doğruluk (accuracy) değerine bakmak doğru değildir. Eğer veri setindeki tahmin edilecek değer unbalance ise, yani homojen değilse, bu tip problemlerde doğruluk değerine bakmak son derece yanlıştır.

Veri setinde tahmin edilecek iki sınıf bulunduğu varsayılırsa ve bu sınıfların veri setindeki oranı % 95, %5 ise, elde edilen makina öğrenmesi modeli ağırlığı yüksek sınıfı (%95 'lik kısım) daha çok öğreneceği için, modeldeki %5 'lik kısmın ağırlığı daha az olacak ve model bu az veriyi öğrenemeyecektir. Doğruluk değerine bakıldığında veride çok olan %95 'lik kısmın hepsini doğru tahmin edildiği

varsayılırsa, %95 ‘lik bir doğruluk değeri var algısı yaratacaktır. Diğer taraftan ise model %5 öğrenmesi gereken kısmı hiç öğrenememiştir. Bu tip problemler için doğruluk değerinden ziyade confision matrix sonucunda elde edilen değerlere bakılmalıdır.

Doğruluk, confusion matrixteki doğru bilinen sınıfların tüm sınıfa oranı şeklinde tarif edilebilir.

Hata oranı, yanlış bilinen sınıfların, tüm sınıf sayısına oranı şeklinde ifade edilmektedir.

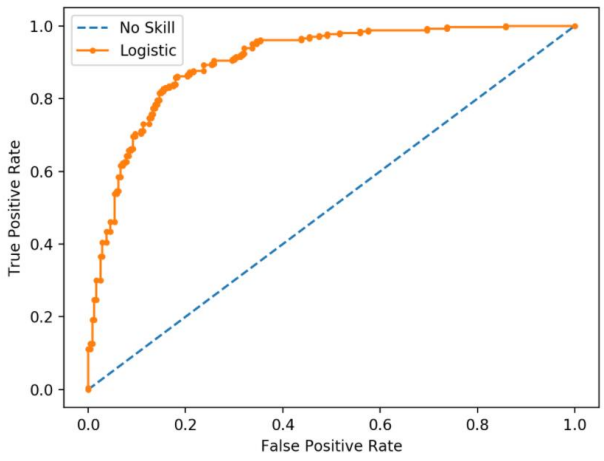
Kesinlik, doğru bilinen pozitif sınıfın, tahmin edilen pozitif sınıf ile negatif olup pozitif tahmin edilen sınıflara oranı ile tespit edilmektedir.

Anma ise, yine doğru bilinen pozitif sınıfın, tahmin edilen pozitif sınıf ile gerçekte pozitif olan sınıfın negatif olarak tahmin edilen sınıflara oranı ile tespit edilmektedir.

Sınıflandırma problemlerinde hassas çalışmalar yapılıyorsa, anma ve kesinlik değerlerine odaklanılmalıdır.

4.4.2.4. ROC eğrisi

Bir veya birden fazla sınıflandırma işleminin başarısını ölçmek için kullanılan tekniklerden biridir. Gerçek pozitif ile yanlış pozitif oranlarının üzerinde kurularak oluşturulmuş istatistiksel bir karşılaştırma tekniğidir.



Şekil 4.4. ROC Eğrisi [68]

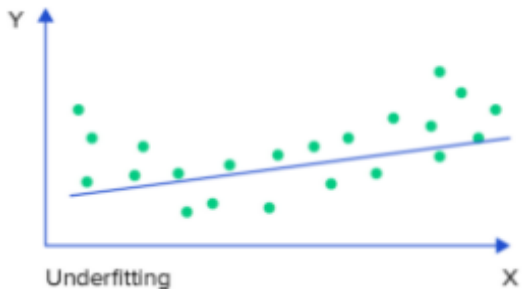
Şekil 4.4 ‘te görülen grafikte, mavi renkli ve kesikli çizgi ile ifade edilen kısım referans noktası olarak kabul edilmektedir. Kurulan makina öğrenmesi modelleri için oluşturulan eğri kesikli mavi çizginin üst bölgelerinde yer almalıdır. Bu eğrinin altında kalan alanın mümkün olduğunca geniş olması durumunda ortaya çıkan alanın büyüklüğü o modelin başarısına karşılık gelmektedir.

4.4.3. Yanlılık varyans

Makine öğrenmesi modelinde kurulan modelin, Eğitim hatası ile test hatası arasındaki durum gözlenmektedir. Eğitim hatası, model kurmada gözlemleri eğitmek için kurulan modelin hatası, test hatası ise model kurmada kullanılmayan yeni gözlemler için elde edilen hatadır. Dolayısı ile Eğitim hatası ile test hatası arasında bir bağımlılık söz konusu olmaktadır. Esneklik kavramı ise verinin fonksiyonel yapısını uygun şekilde yorumlamak, değerlendirmektir.

Verinin fonksiyonel yapısını yorumlama ve değerlendirme kısmını makine öğrenmesi algoritması yapmaktadır. Bu algoritmanın bu işi yapma esnekliği, yanlılık ve varyans kavramını ortaya çıkarmaktadır.

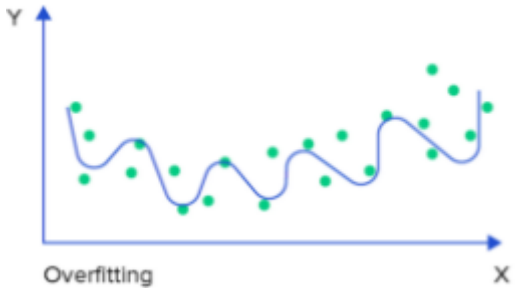
Yanlılık, gerçek değerler ile tahmin edilen değerler arasındaki mesafeyi ifade etmektedir.



Şekil 4.5. Underfitting [69]

Şekil 4.5 ‘te gerçek değerler yeşil noktalar ile, mavi çizgi ile gösterilen bir tahmin fonksiyonu bulunmaktadır. Bu şekilde çizilen doğru herhangi bir şekilde gerçek değerleri ifade eden yeşil noktaları temsil edememektedir. Kısaca oluşturulan fonksiyon, gerçek değerleri tahmin etme konusunda başarılı değildir. Burada eksik öğrenme diğer bir deyişle az öğrenme (underfitting) bulunmaktadır. Bu eksik öğrenmenin temel sebebi ise oluşturulan tahmin fonksiyonunun yüksek yanlı

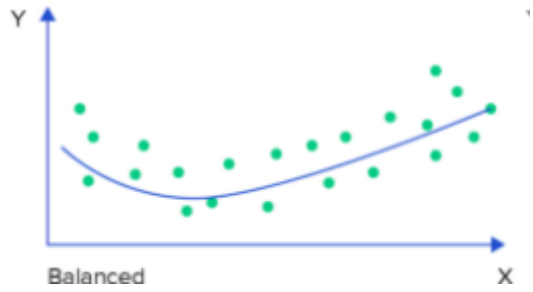
olmasından kaynaklanmaktadır. Grafikte belirli gözlemlerin lehine yüksek yanlılık olduğu görülmektedir.



Şekil 4.6. Overfitting [69]

Varyans kavramı kısaca, değişkenlik olarakta ifade edilebilmektedir. Bir makine öğrenmesi modelinin, esnekliği ya da hassaslığı da denilebilmektedir. Varyansın yüksek olması, veri seti içerisindeki yapının esnek bir şekilde temsil edilmesi olarak ta düşünülebilmektedir. Varyansın yani esnekliğin yüksek olması model oluşturma aşamasında istenmeyen bir durumdur. Bunun nedeni; genellikle veri setindeki içerisindeki veriyi temsil etme kabiliyeti arttıkça, varyans artmakta ve yanlılık azalmaktadır. Şekil 4.6 incelendiğinde, bahsedilen bu durum gözükmektedir.

Varyansın yüksek olması yani yanlılığın çok düşük olması oluşturulan model için istenmeyen bir durumdur. Burada model artık özel bir model olma yolunda ilerlemektedir ki, bu da oluşturulan modelin genelleştirilememesi sorununu ortaya çıkarmaktadır. Bu soruna aşırı öğrenme (overfitting) problemi denilmektedir. Her zaman oluşturulan makina öğrenmesi modellerinin, genelleşmesi istenilmektedir. Oluşturulan modelden, veri setinde olmayan bir veri ile karşılaştığında, düzgün ve mantıklı değerler üretebilmesi beklenmektedir.



Şekil 4.7. Balanced (Doğru Model) [69]

Doğru modelin ise yanlılık ve varyansının düşük olması beklenmektedir. Böyle bir durumda elde edilen makine öğrenmesi modeli veriyi genelleyebileceği için eldeki veriye en uygun fonksiyonu elde edecek olup, veri setinde olmayan bir veri bu fonksiyona verildiğinde fonksiyon, düzgün ve mantıklı değerler üretebilecektir. Şekil 4.7 'deki grafik aslında elde edilen modelin uygun ve kullanılabilir olduğunun göstergesidir.

Unutulmaması gereken en önemli nokta, elde edilen eğitim hatası sonuçlarında her zaman en düşük eğitim hatası veren model seçilmemelidir. Model hakkında yorum test hatası üzerinden yapılmalıdır.

4.4.4. Model parametre optimizasyonları

Burada amaç elde edilen makine öğrenmesi modelinin başarısını daha da arttırmaktır. Parametre, elde edilen makine öğrenmesi modelinin; kullanıcıya sunduğu içsel parametrelerdir. Bu parametreleri model kendisi sunmaktadır.

Hiperparametre ise; makine öğrenmesi modeline verilen ve modelin daha iyi çalışmasını sağlayan dışsal parametrelerdir. Bu parametreler dışarıdan modele verilmektedir. Kısaca modelin optimize edilmesi için kullanılan parametrelerdir.

Parametre tuning işlemi, modelin kendi içerisindeki parametreleri ya da modele verilecek dışsal parametrelerin ayarlaması, tune edilmesi işlemidir. Burada model kendi içerisindeki parametreleri en uygun olacak şekilde ayarlar ya da modele dışsal olarak en uygun parametreler verilerek model bu parametrelerin içerisinden kendisi için en uygun olanını seçer.

4.5. Araç ve Plaka Tanımanın Önemi

Günümüzde artık araç marka ve modellerinin artması, araç üretimi yapan fabrikaların endüstri 4.0 devrimi ile hızlı üretime geçmesi [70] sonucu ile birlikte Dünya nüfusunun da artmasından kaynaklı nedenlerle, trafikte kullanılan araç sayıları artmaktadır.

Bu artış trafik yoğunluğunun yanında, araç ile girilebilecek otopark ve alışveriş merkezleri (AVM) gibi yerlerde yoğunluk olmasından kaynaklı, araç takibinin zorlaşması problemlerine neden olmaktadır.

Teknolojinin ilerlemesi ile birlikte hayatımıza giren teknolojik cihazlar olmasının yanında, bu teknoloji araçlarda da kullanılmaktadır. Önceden manuel olan araçlar artık otomatik olmaya bunun yanında hemen hemen yeni model her araçlarda artık geri görüş kameraları kullanılmaktadır.

Araçların donanımında bulunan bu kameralar sayesinde, başka bir ek donanıma ihtiyaç duyulmadan araçlardaki kameralar üzerinden araç ve plaka tanıma kolaylıkla yapılabilir.

Bahsedilen problemlerin yanında, araçlarda bulunan kişilerin herhangi bir şekilde takip edilip edilmediğinin anlaşılabilmesi içinde araç ve plaka tanıma giderek önem kazanmaktadır.

Bu tezin konusu kapsamında da derin öğrenme ve görüntü işleme algoritmaları ile araç ve plaka tanıma yapılmıştır. Uygulama yönteminde de trafikte kullanılan araçlarda herhangi bir tehlike ile karşılaşılması durumunda araç donanımında bulunan kameralara eklenen yazılım sayesinde, kişilerin herhangi bir olay anında aracı ve plakayı hatırlayamama veya yanlış hatırlama gibi problemlere maruz kalmadan tespit edilmiştir. Böylelikle sürücünün olası bir tehlike karşısında her seferinde dikkatinin daüolmasına gerek kalmadan, yapılan çalışma sürüş güvenliği açısından da önemli olmaktadır.

Yukarıdaki paragrafa ek olarak trafikte kişi takip edildiğini anladığı anda kendisini tedirgin hissetmektedir. Kişideki bu tedirginlik kişinin, dikkatli ve güvenli araç kullanamamasına neden olmaktadır.

Bundan dolayı trafikte seyir halinde bulunan diğer araçlardaki kişilerinde trafikte güvenli bir şekilde seyahat edememesine neden olur. Bu tedirginliği ortadan kaldırmak için araç üzerlerinde bulunan kameralar yardımı ile belli bir süre sonunda ilgili araç hale takipte ise kişiye o plaka ve araç bilgisi, gideceğinden dolayı en azından o ana kadar kişiye trafikte güvenli bir seyahat imkânı sağlanmış olunacaktır.

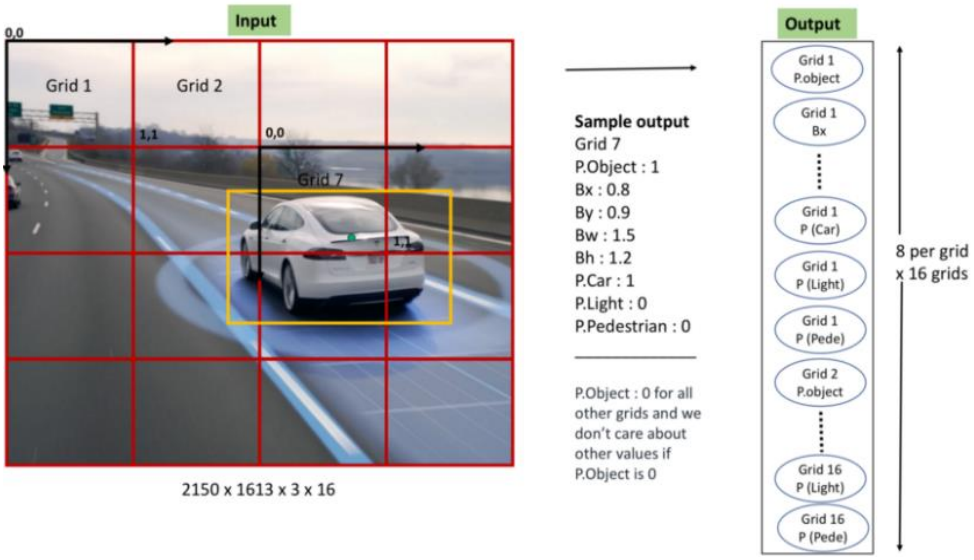
Yakın gelecekte araç donanımlarında bulunan kameraların sadece park halinde geri görüş sağlamasının yanında tez konusunda bahsedilen projelere ek daha birçok projelerde kullanılacak ve araçlarda ekstra konfor ve güvenliğin sağlanmasına yardımcı olacaktır.

5. UYGULAMA VE DEĞERLENDİRME

Bu tez konusu kapsamında derin öğrenme yöntemi ile görüntü üzerinde araç ve plaka tanıma işlemi gerçekleştirilecektir. Evrişimli sinir ağı mimarisi kullanılmış olunup, bu mimariler içerisinden de Yolo – v2 mimarisi kullanılmış ve sonuçlar test veri seti ile test edilmiştir.

Yolo algoritması; hız açısından, nesne tanımada en iyi algoritmalardan birisidir. Bu çalışmada Yolo – v2 mimarisi kullanılmış olup literatürde, Yolo – v3, Yolo – v4 ve hatta Yolo – v5 mimarileri mevcut olup her bir yeni çalışmada bir önceki çalışmada bulunan dezavantajlar giderilmeye çalışılmıştır.

Yolo – v2 algoritmasının nasıl çalıştığının anlaşılabilmesi için öncelikle Yolo – v1 algoritmasının nasıl çalıştığının bilinmesi gereklidir. Yolo – v1 algoritmasının hızlı olmasının sebebi tüm resmi tek seferde nöral ağdan geçirerek, resimde bulunan tüm nesnelerin sınıfını ve koordinatlarını bulabiliyor olmasıdır. Bu işlemi yapabilmek için, ilk önce görüntüyü S x S 'lik ızgaralara (Buradaki S değeri kullanıcıya bağlı bir hyper parametredir) bölüyor. Bölme işlemi gerçekleştikten sonra, nöral ağından geçtikten sonra şekil 5.1 'deki gibi bir vektör elde edilmektedir.



Şekil 5.1. YOLO Algoritması Çıktısı [75]

Şekil 5.1 ‘de gösterilen görüntü üzerindeki her bir ızgara, kendi içerisinde nesnenin olup olmadığını, eğer nesne varsa bu nesnenin orta noktasının bu grid içerisinde olup olmadığını, nesnenin orta noktası bu grid içerisindeyse, nesnenin uzunluğunu, yüksekliğini ve hangi sınıftan olduğunu bulmakla sorumludur. Şekil 5.1 yeniden incelendiğinde arabanın orta noktası 7. ızgaraya denk geldiğinden dolayı, arabanın tespit edilmesi ve etrafına kutucuk çizilmesine kadar o ızgara (7.) sorumludur. Bundan dolayı Yolo mimarisinde her ızgara için bir tahmin vektörü oluşturulmaktadır. Bu tahmin vektörünün her bir ızgara için (Bx, By, Bw, Bh, P(object), class) değerlerini üretir.

Şekil 5.1 ‘de her bir ızgara hücresi sadece bir tane nesne tanımlayabilmektedir. Eğer bir ızgara içerisinde birden fazla nesne varsa, ya da yine bir ızgara içerisinde iki farklı nesnenin ortak noktası varsa, Yolo mimarisinde bu bir sorun teşkil etmektedir.

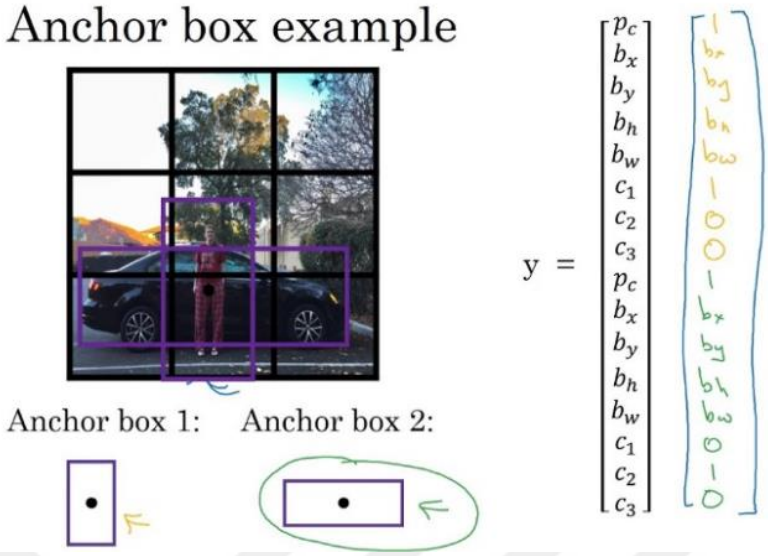
YOLO – v1 ‘de;

- ✓ Her ızgara (grid) içerisinde sadece iki adet bounding box önerilebildiğinden dolayı, aynı ızgara içerisinde bulunan birden fazla nesneyi tespit edememektedir
- ✓ Algoritma, küçük nesneleri tespit etmekte oldukça zorlanmaktadır
- ✓ Faster R-CNN kıyasla fazla miktarda yerelleştirme hatası (algoritmanın tespit ettiği nesnenin, hangi sınıfa ait olduğunu tespit edip, bounding box çizme konusunda diğer modellere göre daha az başarılı olması durumu) olmaktadır

belirtilen sıkıntılardan dolayı Yolo – v2 mimarisi geliştirilmiştir [74]. Yolo – v2 ağı, Yolo – v2 mimarisinin blok diagramı şekil 5.3 ‘de gösterilmektedir.

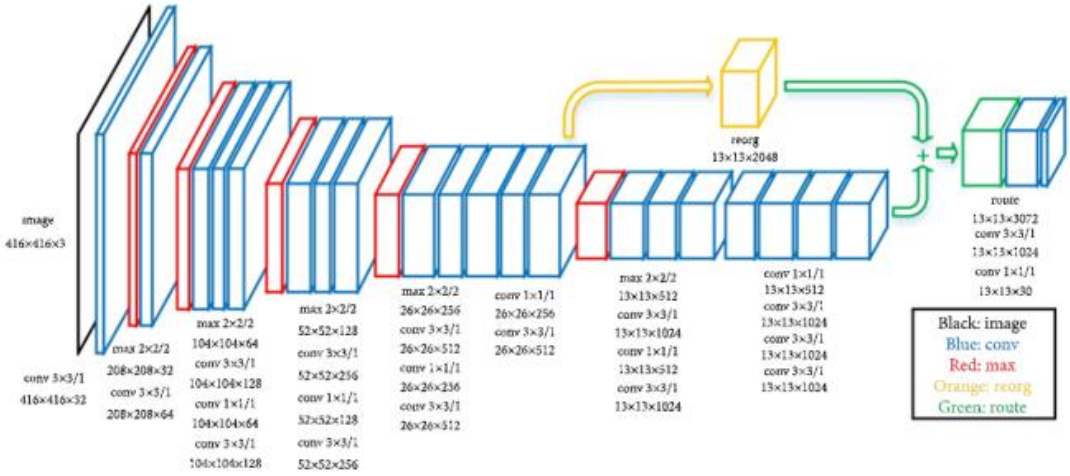
Bir ızgara içerisinde birden fazla nesnenin tespit edilmesi Yolo – v2 ‘de anchor box ‘lar ile çözülmektedir.

Bu sayede her bir ızgara için belirlenen anchor box sayısı kadar tahmin yapılabilmektedir. Anchor box sayısının fazla olması nesne tespitininde fazla miktarda yapılabilmesi Yolo – v2 mimarisinin en önemli özelliği olmakla birlikte, Yolo – v1 mimarisinin en büyük dezavantajını ortadan kaldırmıştır. Yine diğer bir avantajıda Yolo –v1 algoritmasında tespit edilemeyen çok küçük nesnelerin tespit edilmesine olanak sağlamakla birlikte, aynı nesnenin büyük resimlerininde bu mimaride tespit edilmesi Yolo – v1 mimarisinde bulunan handikaplar bu mimaride giderilmiştir.



Şekil 5.2. Andrew Ng ‘nin C4W3L08 Numaralı Dersinden

Yolo – v2 algoritmasına anchor boxların entegre edilmesi ile her bir ızgara için anchor boxların sayısı kadar x,y,w,h ve P(obje) ve diğer sınıflar için olasılık hesaplanacaktır. Şekil 5.2 ‘de orta noktası aynı ızgaraya denk gelmiş iki nesne ve bu nesnelere ait anchor box bulunmaktadır. İnsan şekli anchor box 1’e, araba şekli ise anchor box 2 ’ye denk gelmiştir. Şekil 5.2 ‘deki y vektörü incelendiğinde sarı kısım birinci anchor box, yeşil kısım ise ikinci anchor box ‘a denk gelmektedir. Direkt olarak nesne kutucuğunun en çok benzediği anchor box ‘ı bulup, o kutucuğun yükseklik ve genişlik çıktıları tahmin edilir.



Şekil 5.3. YOLO – v2 Mimari Yapısı [76]

Şekil 5.3 ‘deki Yolo - v2 ağ mimarisi incelendiğinde, giriş olarak 416 x 416 x 3 görüntü alınarak, sırasıyla evrişim işleminden geçirildikten sonra max pooling işleminden geçirilerek evrişim işlemi tekrarlanır. Burada dikkat çeken kısım son kısımda artık fully connected layer olarak adlandırılan tam bağlantılı katmanın bu mimaride çıkartılmış olmasıdır. Yolo – v2 önceki versiyonuna göre, çok daha fazla doğruluk ve küçük nesneleri de tespit edebilmede büyük bir gelişme gösterdi.

5.1. Eğitim ve Test Sonuçlarının Değerlendirilmesi

Yapılacak olan çalışmada öncelikle, toplam veri seti eğitim veri seti (%80) ve test veri seti (%20) olacak şekilde ayrılmış olup Eğitim veri setinden öğrenilen ağırlıklar ve filtreler ile test veri setindeki veriler test edilerek model performans ve başarımı hesaplanmıştır.

5.1.1. Nesne tespiti için tensorflow ve anaconda environment kurulumu

Görüntü işleme ve derin öğrenme ile uğraşıldığında ilk akla gelen sorulardan birincisi, hangi geliştirilme ortamının kurulacağıdır. Teknolojinin gelişmesi ile birlikte piyasada artık birçok geliştirme ortamına erişim oldukça kolaydır. Bu geliştirme ortamlarından bazıları;

- ✓ Python Shell
- ✓ Spyder
- ✓ Pycharm
- ✓ Anaconda

gibi günümüzde birden fazla geliştirme ortamları bulunmaktadır. Bu geliştirme ortamları incelendiğinde;

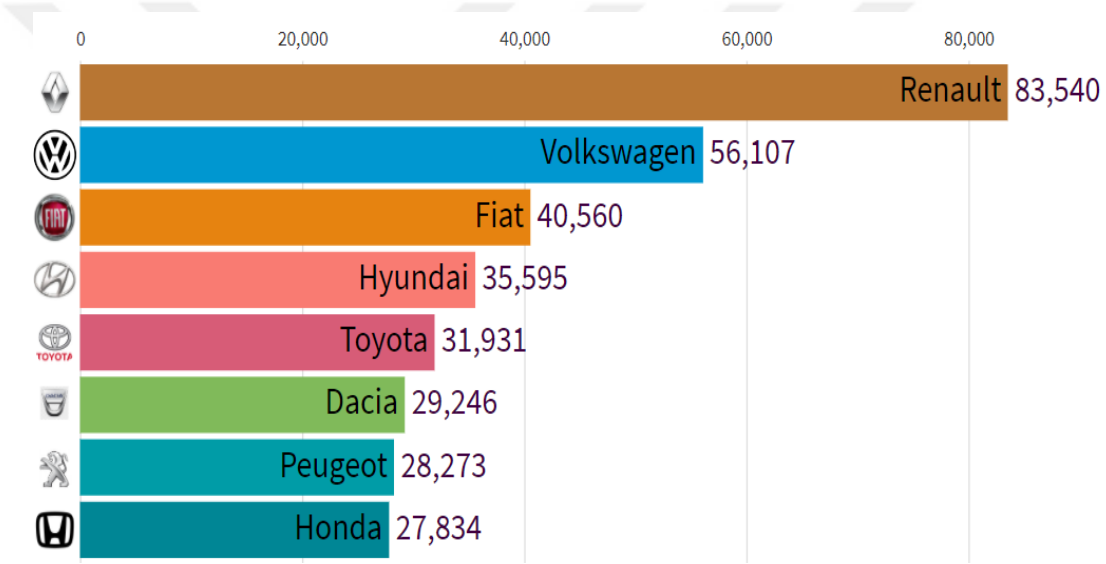
Anaconda, python programlama dili dağıtımı geliştirici denilebilir. İçerisinde birçok kütüphane bulundurmaktadır. Bu kütüphanelere erişim oldukça kolaydır. Anaconda kullanarak farklı versiyondaki kütüphaneleri yüklerken alınacak olan hataların önüne geçilmiştir. Anaconda kısaca birçok kütüphaneyi içerisinde barındıran bir yazılımdır. Böylece her türlü kütüphaneye erişim anaconda yazılımı sayesinde oldukça kolaydır.

Ayrıca anaconda yazılımı içerisinde sadece kütüphane bulundurmamaktadır. Spyder, Jupiyter Notebook gibi geliştirme ortamlarını da içerisinde bulundurmaktadır.

Derin öğrenme mimarisi kullanacağımız için bu mimarinin kütüphanesi olan tensorflow kurulumu da son derece önemlidir. Tensorflow, google tarafından derin öğrenme uygulamaları için geliştirilmiş açık kaynaklı bir kütüphanedir. Tensorflow yardımı ile kolay bir şekilde derin öğrenme uygulamaları yazılabilmektedir. Bilgisayarda bulunan ekran kartının uyumluluğuna göre tensorflow GPU veya CPU kurulumları yapılmaktadır.

5.1.2. Veri setinin oluşturulması ve etiketlenmesi

Yapılan çalışmada veri seti olarak ülkemizde, trafikte en çok bulunan ilk 5 araç şekil 5.4 ‘de gösterilmektedir.



Şekil 5.4. Ülkemizde Trafikte Kullanılan 2019 Yılına Ait Araçların Sıralaması [71]

Şekil 5.4 ‘den yola çıkılarak bu çalışmada ilk 5 otomobil seçilmiş olup, seçilen bu otomobil fotoğrafları üzerinden etiketleme çalışması yapılmıştır.

Grafikte 5. Sırada bulunan Toyota araç markası Ford araç markası ile değiştirilmiştir. Bunun nedeni sahadan veri toplamada bulunulan lokasyona bağlı olarak Ford marka aracın daha fazla bulunmasıdır. Sahadan toplanan veriler üzerinde model eğitim işlemleri gerçekleştirildiği için, bu şekilde bir değişime gidilmeye gerek duyulmmuştur.

Örnek bir çalışma şekil 5.5 ‘de gösterilmektedir.

Şekil 5.5. Araç Plaka ve Amblem Etiketlemesi

Etiketleme işlemini labelImg programı üzerinden yapılmıştır. Yeşil kutular ile belirlenen bölgeler sırasıyla Ford (amblemden) ve Plaka olarak etiketlenerek, görüntü üzerinde bu etiketlerin bulunduğu konumlar xml dosyasına yazılarak her fotoğraf için xml dosyası oluşturulmuştur. Bu xml dosyalarının içerisinde görüntü üzerindeki aracın marka ve plaka bilgilerinin olduğu konum bilgileri bulunmaktadır.

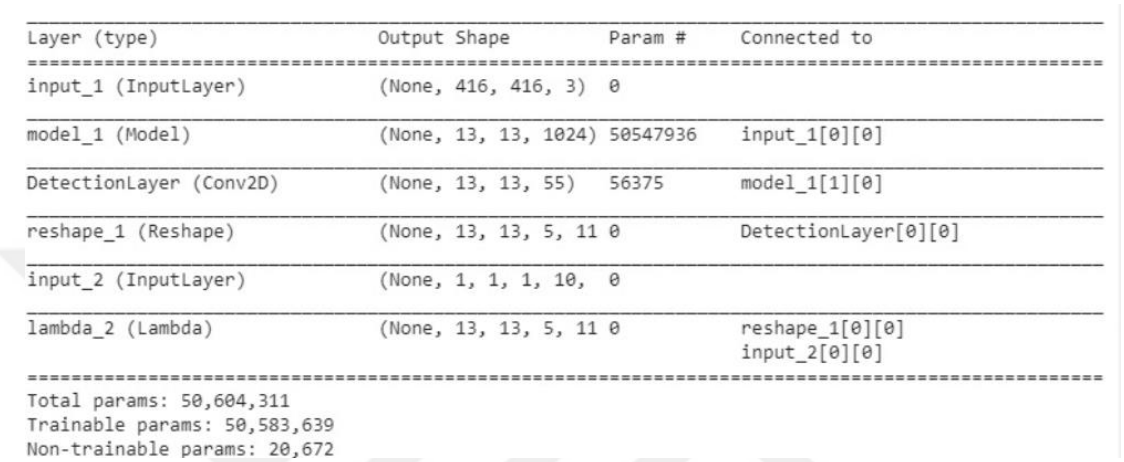
5.1.3. Model oluşturma için eğitme

Eğitim esnasında, eğitim işlemlerinin sağlıklı yapılabilmesi için güçlü ekran kartına sahip bilgisayarlar bulunmalıdır. Yalnız bilgisayarda güçlü ekran kartı bulunsa dahi, eğitim işlemi sırasında bilgisayar sadece eğitim işlemini gerçekleştirecek ve işlem veri setine göre uzun süreceğinden dolayı, bilgisayarlarda herhangi bir işlem yapılamayacaktır. Bundan dolayı günümüzde artık google, derin öğrenme ile uğraşan insanlara ücretsiz olarak sunmuş olduğu google colab [72] üzerinden eğitim işlemleri kolaylıkla yapılabilmektedir. Tek yapılması gereken Google drive ‘ın google colab ortamına tanıtılmasıdır.

Google colab ortamı google ‘ın kullanıcılara ücretsiz olarak sınduğu içerisinde ubuntu işletim sistemi olan ve Nvidia Tesla K80 GPU kartı bulundurmaktadır. Google drive ‘ınızda yeterli kadar yer olduktan sonra verilerinizi drive taşıyarak goole colab ortamında python jpyter notebok gibi kodlarınızı yazarak eğitim işlemini

gerçekleştirebilir ve modelinizi kaydedebilirsiniz. Bu çalışmada etiketlenmiş veriler google drive yüklenerek google colab üzerinde eğitim işlemleri gerçekleştirilmiştir.

Öncelikle kullanılan yolo mimarisi incelendiğinde sistem parametrelerinin neler olduğu ve sistemde kaç parametre olduğu ki bu parametreler filtrelerin ağırlıkları ve sinir ağının ağırlıkları, bu ağırlıklardan oluşan model özeti şekil 5.6 'da gösterilmiştir.



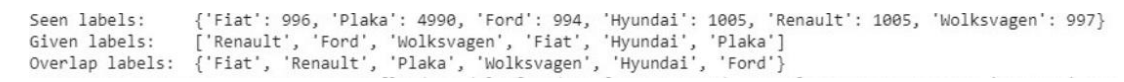
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 416, 416, 3)	0	
model_1 (Model)	(None, 13, 13, 1024)	50547936	input_1[0][0]
DetectionLayer (Conv2D)	(None, 13, 13, 55)	56375	model_1[1][0]
reshape_1 (Reshape)	(None, 13, 13, 5, 11	0	DetectionLayer[0][0]
input_2 (InputLayer)	(None, 1, 1, 1, 10,	0	
lambda_2 (Lambda)	(None, 13, 13, 5, 11	0	reshape_1[0][0] input_2[0][0]

Total params: 50,604,311
Trainable params: 50,583,639
Non-trainable params: 20,672

Şekil 5.6. Model Parametreleri

Şekil 5.6 dikkatli incelendiğinde eğitilebilir model parametreleri yaklaşık 50 Milyon parametre bulunmaktadır. Bu parametreler Evrişim katmanında bulunan filtreler ve sınıflandırma katmanına bulunan ağırlıklardan oluşmaktadır.

Modelde kullanılan etiketler (label) şekil 5.7 'deki gibi yazdırılmıştır.



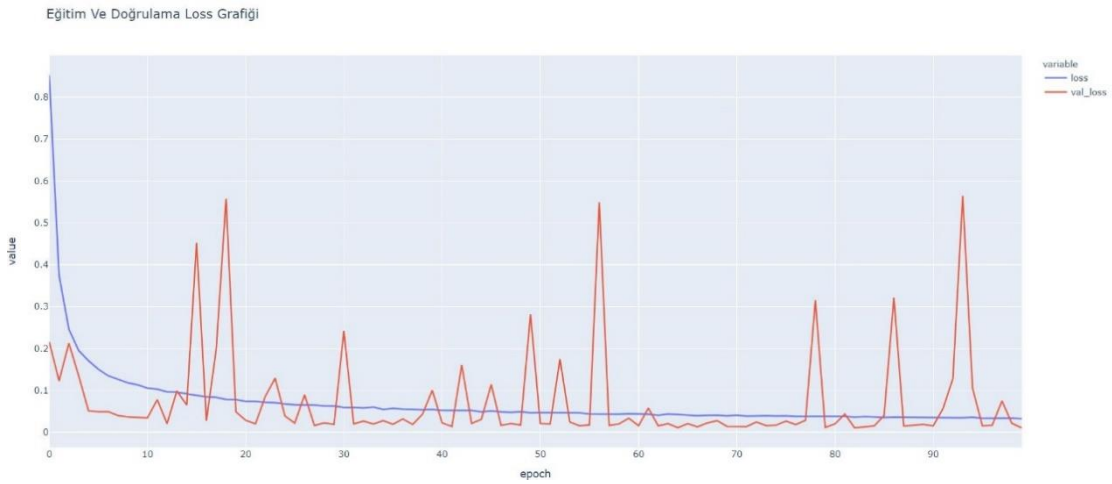
```
Seen labels:    {'Fiat': 996, 'Plaka': 4990, 'Ford': 994, 'Hyundai': 1005, 'Renault': 1005, 'Wolksvagen': 997}
Given labels:   ['Renault', 'Ford', 'Wolksvagen', 'Fiat', 'Hyundai', 'Plaka']
Overlap labels: {'Fiat', 'Renault', 'Plaka', 'Wolksvagen', 'Hyundai', 'Ford'}
```

Şekil 5.7. Modelde Kullanılan Sınıflar ve Adetleri

Şekil 5.7 incelendiğinde ise, toplam 6 farklı sınıf etiketi bulunmaktadır. Sınıfların adetleri ise her bir sınıf için (plaka hariç) yaklaşık 1000 adettir.

5.2. Eğitim Sonucunda Elde Edilen Modelin Değerlendirilmesi

Eğitim işlemi yapıldıktan sonra, elde edilen model hem validasyon hem de test verisi üzerinden model başarı değerlendirmesi yapılmaktadır. Dikkat edilecek husus, Eğitim işlemi sonucunda elde edilen modelin verileri genelleyebilmesidir. Eğitim skoru ile test skoru arasında çok büyük bir fark olması beklenmemekte, aksi halde model veriyi genelleyememiş, ezberlemiş (overfit) olmaktadır.



Şekil 5.8. Eğitim ve Doğrulama veri Setleri ile Oluşturulmuş Doğrulama Grafiği

Şekil 5.8 'deki grafikte her bir epoch için eğitim ve doğrulama grafiği görünmektedir. Bazı epoch noktalarında kayıp fonksiyonunun pik değerler ürettiği gözlemlenmektedir. Bunun nedeni resim üzerinde etiketleme yaparken etiketleme yapılan alanın dışarısındaki bir piksel de plaka var diye etiketlenmiş olmasıdır. Şekil tümüyle incelendiğinde ise eğitim ve doğrulama loss fonksiyonlarının düştüğü gözlemlenmektedir.

Tablo 5.1. Eğitim Sonucu

	Ford	Hyundai	Renault	Fiat	Wolksvagen	Plaka
Accuracy Değerleri	0.93	0.96	0.96	0.92	0.96	0.99

Tablo 5.1 incelendiğinde, model doğrulama seti üzerinden gayet iyi bir sonuç vermiştir. Araç sınıflarında sonuçlar birbirine yakınken; plaka sınıfı bunlardan yüksektir. Bunun nedeni veri setinde, plaka sınıfı sayısının araç sınıfı sayısından fazla olmasından kaynaklanmaktadır.

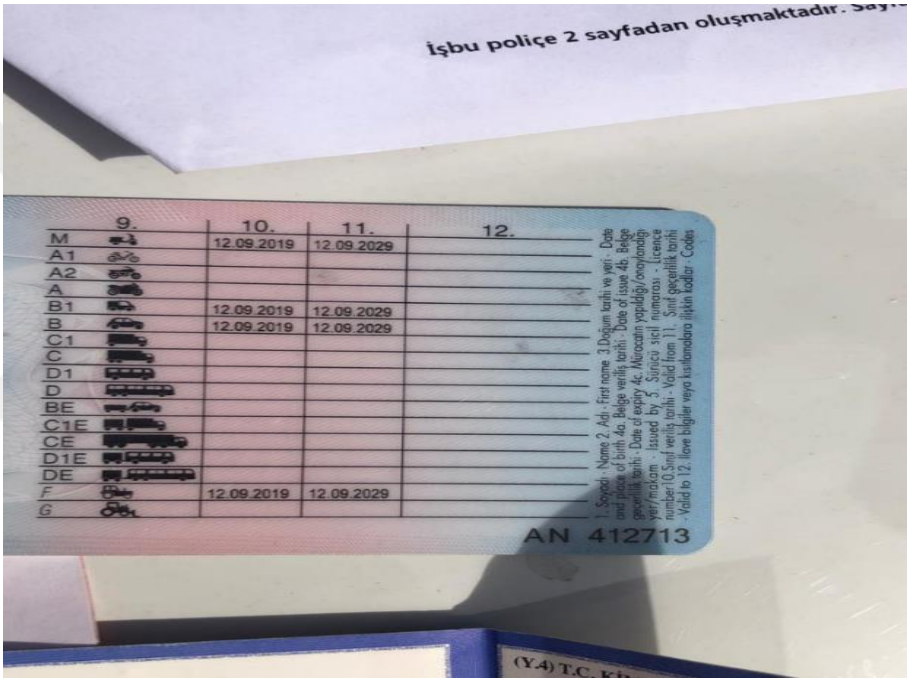
Burada metric olarak accuracy değeri üzerinden sonuçlar paylaşılmaktadır. Literatür incelendiğinde özellikle sınıflandırma problemlerinde farklı parametreler üzerinden model başarım metriği belirlenmektedir.

Eğer eldeki veri seti unbalanced veri seti olmuş olsaydı farklı sınıflandırma metriği üzerinden (racall, confusion matrix, f1 score v.b.) model başarısı değerlendirilebilirdi. Ama veri seti incelendiğinde accuracy değerine bakılması ve bu accuracy değeri

özelinde model başarısını değerlendirme kısmında veri seti balance bir veri seti olduğundan herhangi bir sakınca görülmemektedir.

5.2.1. Oluşturulan modelin tahmin ile gerçek değerinin karşılaştırılması

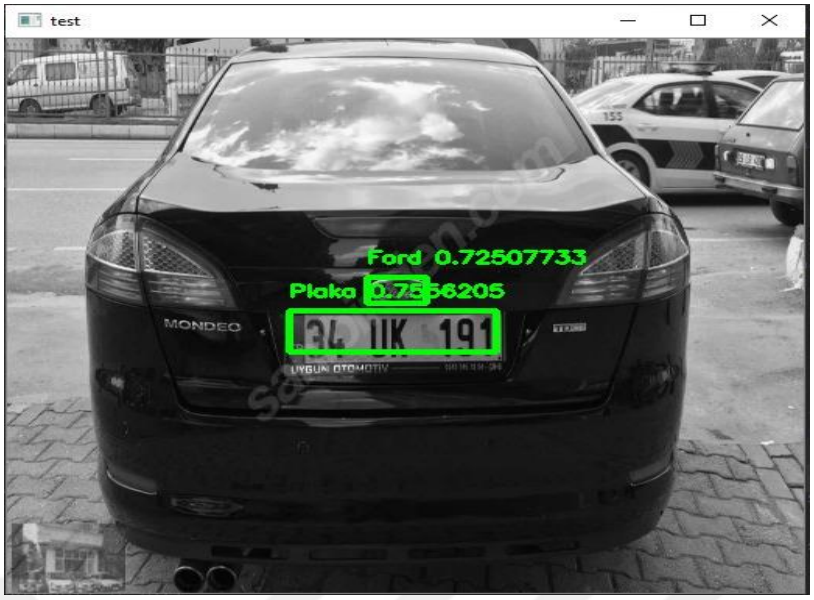
Model eğitme işlemi tamamlandıktan sonra, elde edilen model üzerinden eğitim ve doğrulama esnasında kullanılmayan veriler üzerinde test çalışması yapılmıştır. Yapılan test çalışmasında hiçbir şekilde araç içermeyen bir fotoğraf modele verilmiş ve bu fotoğraf üzerinde eğitim işleminde gördüğü sınıflardan hangisine ait olduğunu tespit edilmesi beklenmiştir. Sonuç çıktısı şekil 5.9 ‘da gösterildiği gibidir.



Şekil 5.9. Hiç Araç ve Plaka Olmayan Resim Üzerinde Sınıflandırma

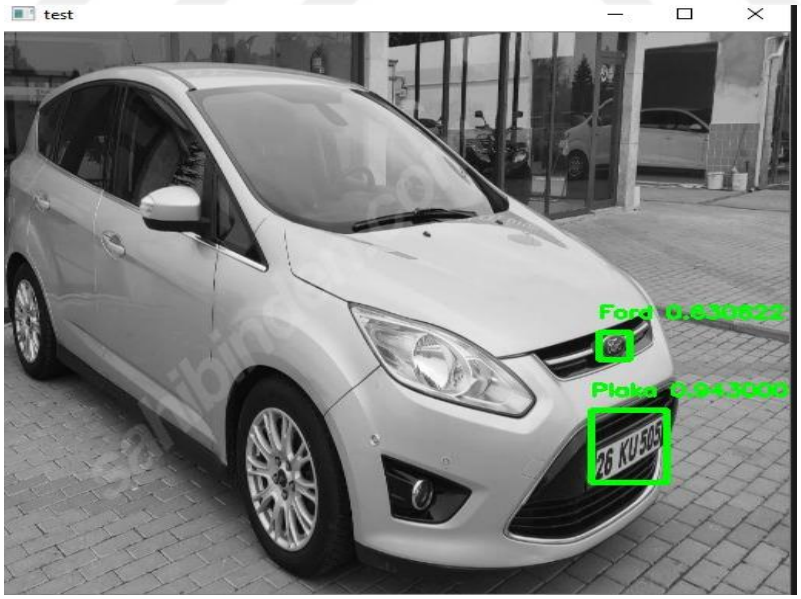
Şekil 5.9 ‘da görüldüğü üzere algoritma resim üzerinde plakaya araç bulamamıştır ki bu resim incelendiğinde olması gereken beklenen bir durumdur. Şekil 5.10 ‘da araç ve plaka içeren yeni bir resim algoritmaya verildiğinde ise resim üzerinde gayet başarılı bir şekilde araç sınıfını ve plakayı bulmaktadır.

Elde edilen model, görüntü üzerindeki nesneyi tespit edebilmesinin yanında, görüntü üzerinde aranan nesne yok ise ilgili nesneyi görüntü üzerinde bulamayacaktır. Buradan elde edilen sonuçla, model ilgili nesnelerin sınıflandırılmasını iyi öğrenmiş, ilgili nesne görüntü üzerinde olmadığında ise ilgili nesneyi görüntü üzerinde bulamaması istenilen bir durumdur.



Şekil 5.10. Resim Üzerinde Araç ve Plakanın Bulunması (a)

Şekil 5.10 'da görüldüğü gibi resim üzerinde araç ve plaka tespiti yapılmış olup her iki sınıfta yaklaşık %75 doğrulukta tespit edilmiştir (Şekil 5.11). Daha sonra elde edilen çıktılardan plakanın tam konumunu bulabilmek adına, görüntü işleme yöntemleri kullanılarak plakanın tam okunurluğu sağlanabilmektedir.



Şekil 5.11. Resim Üzerinde Araç ve Plakanın Bulunması (b)

6. SONUÇLAR VE ÖNERİLER

Yapılan bu çalışmada çoklu nesne algılama (sınıflandırma + lokalizasyon) problemlerinden biri olan görüntü üzerinde araç ve plaka tanıma çalışması farklı bir yaklaşımla ele alınmıştır. Burada Türkiye ‘de trafikte en fazla bulunan ilk 5 araç alınmış olup, bu araçlar üzerinde nesne algılama (araç marka ve plaka) işlemi gerçekleştirilmiştir.

Bu çalışma diğer araç ve plaka tanıma çalışmalarından farklı olarak şekil 5.8 ve şekil 5.9 ‘da görüldüğü üzere sadece araç üzerinde bulunan marka amblemi üzerinden araç tanıma yapılmaktadır. Bu mantığın sağladığı avantaj, amblem üzerinden araç tanıma yapıldığı için ilgili markaya ait tüm modellerin sınıflandırılmadı yapılabilmektedir. Geçmişten günümüze ve hatta geleceğe ait araçların marka amblemleri çok sık değişmediğinden, çok az veri setiyle ilgili araç markasına ait modeller, derin öğrenme yöntemi ile oluşturulan model üzerinden ve ilgili derin öğrenme modelini güncellemeye gerek kalmadan, tüm araç modelleri için araç markası tespit edilebilmektedir.

Çalışmada farklı boyutlarda resimler kullanılmış olup, uygulanan mimariye göre ön işlem çalışması ile tüm görüntüler 512x515 boyutlarına indirgenmiştir. Sonrasında da bu çalışma Yolo –v2 mimarisine göre 416x416 olarak tekrar resize edilerek görüntü boyutu indirgenmiştir.

Tablo 6.1. YOLO Mimari Karşılaştırması [73]

	Katman Sayısı	**Başarım Oranı (%)**	**Süre (ms)**
Yolo-v1	17	93.53	21
Yolo-v2	19	95	23
Yolo-v3	53	99.5	29

Yolo -v2 mimarisi, Yolo -v1 mimarisine göre daha hızlı, daha gelişmiş ve daha iyi bir algoritmadır. Yolo -v3 algoritması ise Yolo -v2 algoritmasına göre doğruluğu yüksek ancak yavaş çalışmaktadır. Bunun da nedeni içlerinde bulunan ağ mimarilerindeki katman sayılarından kaynaklanmaktadır. Katman sayısı arttıkça, model başarım oranı

ve süre artmaktadır (Tablo 6.1). Kısaca katman sayısı ile model başarım oranı ve süre arasında doğru orantı bulunmaktadır. Yapılan çalışmada eldeki veri seti tablo 6.2 ‘de gösterilmektedir.

Tablo 6.2. Her bir Sınıf İçin Elde Edilen Veri Seti Dağılımı

	Eğitim	**Doğrulama**	**Toplam**
Plaka	4000	1000	5000
Renault	800	200	1000
Hyundai	800	200	1000
Ford	800	200	1000
Wolksvagen	800	200	1000
Fiat	800	200	1000

Veri setindeki dağılım incelendiğinde her bir araç üzerinde plaka olduğu düşünüldüğünde toplam plaka sınıfı sayısı denklem (6.1) ‘deki gibi ifade edilmektedir.

$$\text{Plaka Sayısı} = \text{Araç Sınıfı Sayısı} * 1000 \quad (6.1)$$

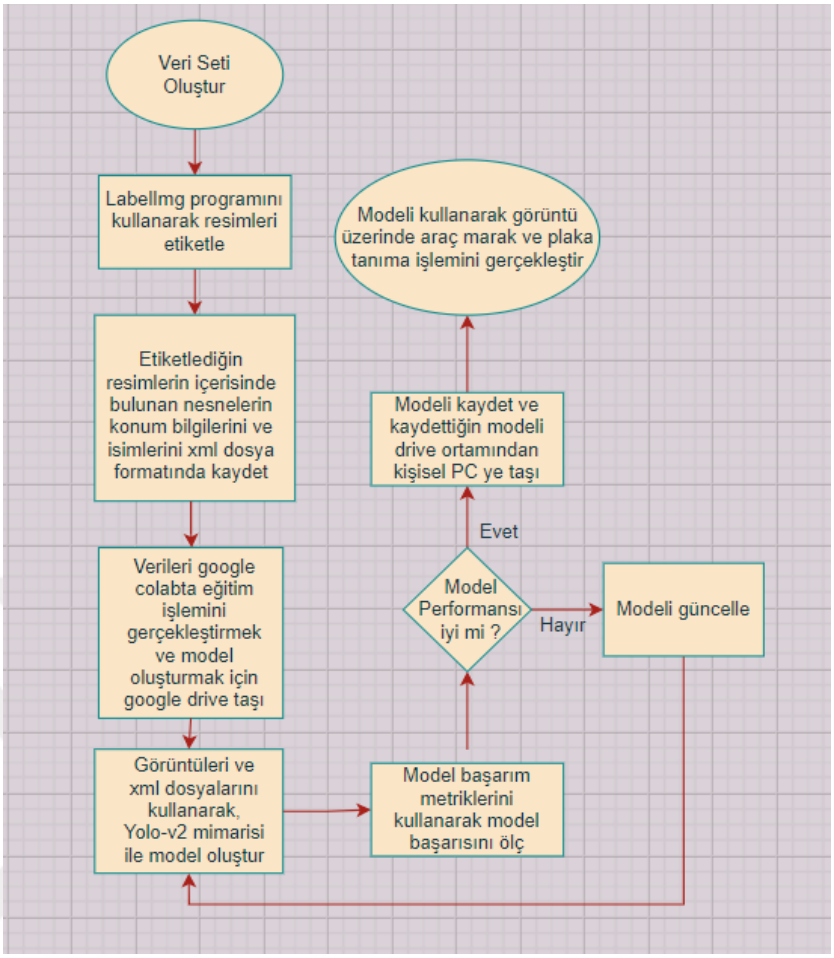
Her bir araç sınıfı için 800 adet eğitim, 200 adet doğrulama olmak üzere toplam 1000 adet görüntü, her görüntüde plaka bulunduğu için toplam 5000 adet plaka görüntü verisi kullanılmıştır.

Mevcuttaki veri seti incelendiğinde veriler arasında homojen bir dağılım olduğu gözlemlenmiş olup, araç sınıflarında elde edilen görüntü resim adetleri aynı bulunmaktadır.

Araç sınıflarının adet dağılımlarının aynı olması model başarı performans metriklerinden accuracy metriği kullanarak model başarı performansını ölçmek için yeterlidir. Bunun yanında model başarı performansları için ROC Eğrisi yada karmaşıklık matrisi gibi model başarım ölçüsü parametreleri de kullanılabilir.

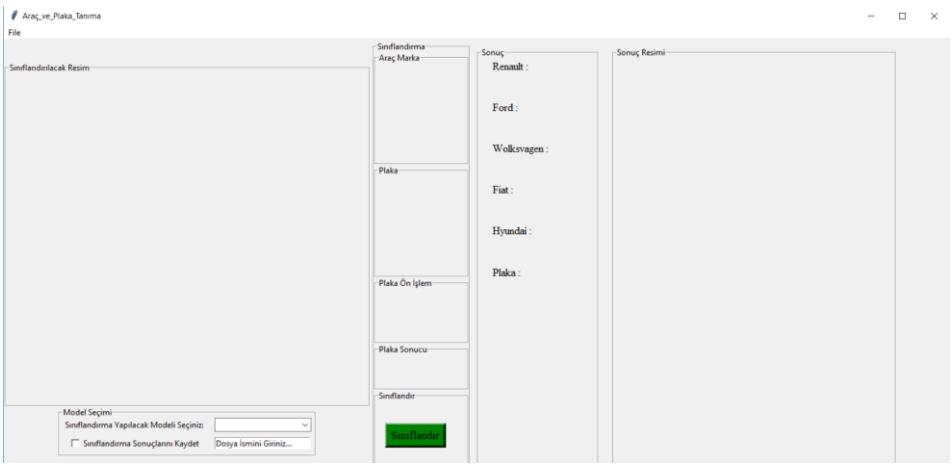
Derin öğrenme yöntemi ile oluşturulan model için artık test verileri ile model doğrulama işlemine geçilebilmektedir. Yani, eğitim verileri ile model oluşturulurken aynı zamanda doğrulama verileri ile de model doğrulandı.

Sonuç olarak modelin hiç görmediği görüntü verileri ile model test etme aşamasına gelindi. Modelden beklenen eğitim setinde görmediği nesneleri tespit edememesi, elde edilen sonuca paralellik gösterdiği için model başarı performansı burada da başarılı olarak değerlendirildi. Çalışmanın akış diyagramı şekil 6.1 ‘de gösterilmektedir.



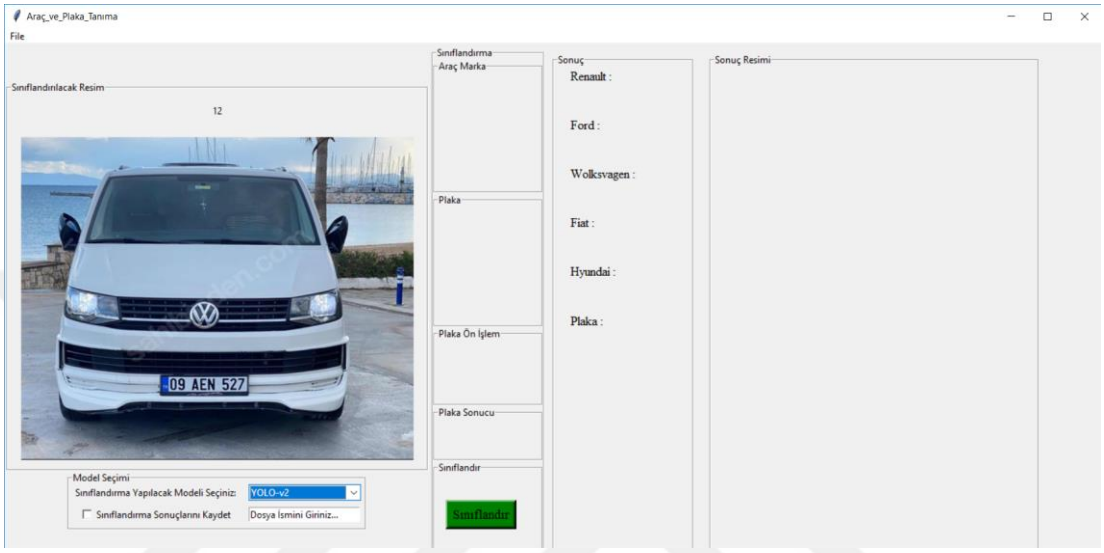
Şekil 6.1. Yapılan Çalışmanın Akış Diyagramı

Model test performansını değerlendirmek ve görüntü üzerinde bulduğu araç marka ve plaka bilgisini görüntü üzerinde gösterip, model test başarımını değerlendirmek için şekil 6.2 ‘deki gibi python programlama dili kullanılarak bir ara yüz oluşturulmuştur.



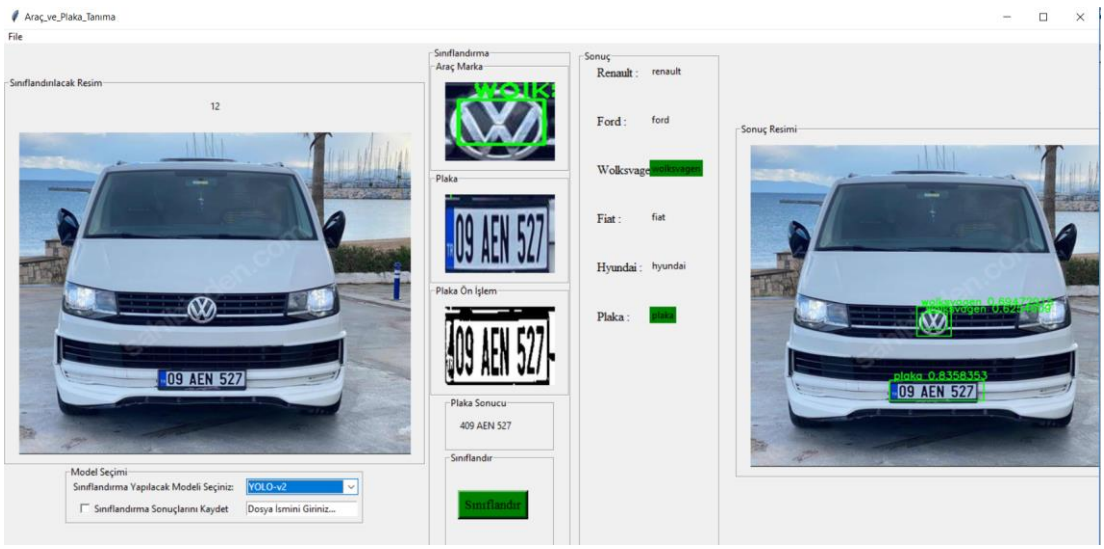
Şekil 6.2. Araç Görüntü Sınıflandırma Ara Yüz

Şekil 6.2 incelendiğinde görüntü üzerinde araç sınıflandırma yapan bir ara yüz programı gösterilmektedir. Bu ara yüzde file kısmından araç marka ve plaka bilgileri bulunacak olan görüntü yüklenir ve sınıflandırma yapılacak olan derin öğrenme modeli seçilerek yeşil renkte gösterilen sınıflandır butonuna basılarak (Şekil 6.3) görüntü üzerinde varsa plaka ve araç marka bilgileri bulunur.



Şekil 6.3. Araç Görüntüsünün Yüklenmesi

Elde edilen model ile şekil 6.4 'teki gibi görüntü üzerinde araç ve marka sınıflandırma gerçekleştirilmiş olunmaktadır.



Şekil 6.4. Görüntü Üzerinde Araç ve Plaka Tanıma

Görüntü üzerinde araç ve plakanın bulunduğu bölgeler tespit edildikten sonra plaka kısmının çözümlenebilmesi için OCR Kütüphanesi kullanılmıştır. Bu sisteme (OCR)

plaka diye bulunan alan gönderilerek ilgili alan üzerinde karakter tanıma yapılmaktadır. Düzgün karakter tanıma yapılabilmek için görüntü üzerinde birtakım ön işlem (preprocessing) yapılmaktadır. Ön işlem yapılan görüntü üzerinde pytesseract kütüphanesi (OCR) yardımıyla karakter çözümleme işlemi yapılmıştır.

Aşağıdaki Tablo 6.3 'de farklı çalışmalarda yapılan derin öğrenme tabanlı plaka tanıma/tespiti çalışmalarının yöntem, başarım oranı ve hesaplama süreleri bakımından karşılaştırılması verilmektedir.

Tablo 6.3. Derin Öğrenme Tabanlı Plaka Tanıma/Tespiti Çalışmalarının Karşılaştırılması

Metot	**Uygulama**	**Kullanılan Yöntemler**	**Başarım Oranı (%)**	**Süre (ms)**
Laroca ve ark.[77]	Plaka Tanıma	YOLO	93.53	21
Rafique ve ark. [78]	Plaka Tespiti	RCNN, SVM	94.53	350
Li ve ark. [79]	Plaka Tanıma	CNN, RCNN	96.57	400
Bulan ve ark. [80]	Plaka Tanıma	Gizli Markov Modelleri, CNN	99.00	2000
Önerilen Model	Plaka Tanıma	**YOLO-v2**	**99.00**	**23**
Bayram F. [12]	Plaka Tanıma	Mask-RCNN	98.46	400
Zhuang ve ark. [81]	Plaka Tanıma	DeeplabV2 ResNet-101	99.25	-

Bu tez çalışmasında önerilen model gerek başarım oranı ve gerekse hesaplama süreleri göz önüne alındığında diğer önerilen yöntemler arasında tercih edilebilir. Bu tez kapsamında yapılan çalışmanın daha hızlı ve daha yüksek bir başarıma sahip olabilmesi için aşağıdaki iyileştirmeler uygulanabilir.

- ✓ Elde edilen görüntülerin daha doğru etiketlenmesi model başarımını arttıracaktır
- ✓ Nesne tespitinin daha hızlı yapılabilmesi için Yolo-v2 mimarisi dışında diğer nesne tespit algoritma mimarileri de denenebilir
- ✓ Veri setindeki görüntüler daha yüksek çözünürlükte olabilir ve bu yüksek çözünürlük model başarı performansını arttıracaktır
- ✓ Bu çalışmada kullanılan Yolo-v2 mimarisinde sinir ağında kullanılan görüntü 416x416 boyutlarına indirgenmiştir.

Özetle, bu tez çalışması ile araç markasının amblem üzerinden tespiti ve plaka tanıma işlemleri derin öğrenme tabanlı Yolo-v2 mimarisi ile %99 başarım oranı ile gerçekleştirilmiştir. Yapılan diğer çalışmalardan da görüleceği üzere konu hala üzerinde çalışmaların yürütüldüğü aktif bir çalışma alanı olup, yapılan bu tez çalışması ile özellikle bu alanda çalışan/çalışacak olan kişi/kişilere yön göstermesi açısından bir rehber olduğu düşünülmekte ve üzerinde yapılacak iyileştirmeler ile genişletilerek yeni çalışmalar yapılabilme imkânı sunmaktadır.

KAYNAKLAR

[1] Doğan F., Türkoğlu İ., Derin Öğrenme Modelleri ve Uygulama Alanlarına İlişkin Bir Derleme, *DÜMF Mühendislik Dergisi*, 2019, **10**(2), 409-445.

[2] Topal Ç., Alan Turing'in Toplum Bilimsel Düşünü: Toplumsal Bir Düş Olarak Yapay Zeka, *DTCF Dergisi*, 2017, **57**(2), 1340-1364.

[3] Aalami N., Derin Öğrenme Yöntemi Kullanarak Görüntülerin Analizi, *ESTUDAM Bilişim Dergisi*, 2020, **1**(1), 17-20.

[4] Sonka M, Hlavac V., Boyle R., *Image Processing, Analysis and Machine Vision*, 2rd ed., Cengaga Learning, Stamford, 2014.

[5] Korkmaz Y., Boyacı A., Adli Bilişim Açısından Ses İncelemeleri, *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 2018, **30**(1), 329-343.

[6] Kılınç D., Borandağ E., Yücalar F., Tunalı V., Şimşek M., Özçift A., KNN Algoritması ve R Dili ile Metin Madenciliği Kullanılarak Bilimsel Makale Tasnifi, *Marmara Fen Bilimleri Dergisi*, 2016, **28**(3), 89-94.

[7] Burgaz M., Derin Öğrenme Algoritmaları Kullanarak İnsansız Hava Araçlarında Silah Tespiti, Yüksek Lisans Tezi, Batman Üniversitesi, Fen Bilimleri Enstitüsü, Batman, 2020, 636954.

[8] Searle J. R., *Speech Acts*, 23rd ed., Cambridge University Press, Cambridge, 1999.

[9] İlgen B., Adalı E., Tantuğ A. C., Exploring Feature Sets For Turkish Word Sence Disambugation, *Turkish Jurnal Of Electrical Engineering & Computer Science*, 2016, **24**(1), 4391-4405.

[10] Şeker A., Diri B., Balık H. H., Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme, *Gazi Mühendislik Bilimleri Dergisi*, 2017, **3**(3), 47-64.

[11] Goodfellow I., Bengio Y., courville A., *Derin Öğrenme*, 1.rd ed., Buzdağı, Ankara, 2018.

[12] Bayram F., Automatic License Plate Recognition Based On Deep Learning, *Journal of Polytechnic*, 2020, **23**(4), 955-960.

[13] Çetin N. M., GPU Hızlandırmalı Veri Denetleme Algoritmasının İncelenmesi, *Online Academic Jurnal Of Information Technoloy*, 2013, **4**(12), 20-59.

[14] Tan Z., Derin Öğrenme Yardımıyla Araç Sınıflandırma, Yüksek Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ, 2019, 572632.

[15] Uçar A., Bingöl M. S., Derin Öğrenmenin Caffe Kullanılarak Grafik İşleme Kartlarında Değerlendirilmesi, *DÜMF Mühendislik Dergisi*, 2018, **9**(1), 39-49.

[16] Yeşilkaynak V. B., Yapay Zeka, Makine Öğrenmesi ve Derin Öğrenme Kavramları Arasındaki Fark Nedir ?, Evrim Ağacı, https://evrimagaci.org /yapay-zeka-makine-ogrenmesi-ve-derin-ogrenme-kavramlari-arasindaki-fark -nedir-8889/ (Ziyaret tarihi : 10 Şubat 2021).

[17] Tienin B. W., Özyıldırım B. M., Cloud Coverage Prediction With Deep Learning Methods, *Çankırı Üniversitesi Fen ve Mühendislik Bilimleri Dergisi*, 2018, **39**(11), 50-57.

[18] Gonzales R. C., Woods E. R., Digital Image Processing, 4rd ed., Pearson International Education, Tennessee, 2014.

[19] Eriş M., Derin Öğrenme Yöntemleri Kullanarak Adli Bilişim İncelemelerinde Delil Çıkarımının Gerçekleştirilmesi, Yüksek Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ, 2018, 524217.

[20] Yaraş N., Vehicle Type Classification With Deep Learning, Yüksek Lisans Tezi, İzmir Yüksek Teknoloji Enstitüsü, Lisansüstü Eğitim Enstitüsü, İzmir, 2020, 631254.

[21] Ateş H., Pothole Detection In Asphalt Images Using Convolutional Neural Networks, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Ankara, 2019, 546712.

[22] Zontul M., Yangın A., Yapay Sinir Ağı Teknikleri Kullanarak Eğitim Yayıncılığı Sektöründe Metin Madenciliği, *Aurum Muhendislik Sistemleri ve Mimarlık Dergisi*, 2017, **1**(2), 1-15.

[23] Kumar V., Boluanger D., Explainable Automated Essay Scoring: Deep Learning Really Has Pedagogical Value, *School of Computing and Information Systems*, 2020, **30**(3), 294-299.

[24] Taghipour K., Ng H. T., A Neural Approach to Automated Essay Scoring, Association For Computational Linguistics, 2016, **16**(11),1882-1891.

[25] Ser G., Bati C. T., Derin Sinir Ağları İle En İyi Modelin Belirlenmesi: Mantar Verileri Üzerine Keras Uygulaması, *Yüzüncü Yıl Üniversitesi Tarım Bilimleri Dergisi*, 2019, **29**(3), 406-417.

[26] Aktan E., Büyük Veri: Uygulama Alanları, Analitiği ve Güvenlik Boyutu, *Ankara Üniversitesi Bilgi Yönetim Dergisi*, 2018, **1**(1), 1-22.

[27] Ergin T., Evrişimsel Sinir Ağları (Convolutional Neural Network), DevHunter, https://devhunteryz.wordpress.com/2018/04/08/evrisimsel-sinir-aglariconvolutional-neural-network/ (Ziyaret tarihi : 15 Şubat 2021).

[28] Çarkacı N., Derin Öğrenme Uygulamalarında En Sık Kullanılan Hiper-Prametreler, Medium, https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4/ (Ziyaret tarihi : 18 Aralık 2020).

[29] Şengül V., Derin Öğrenme Çalışmalarında Optimizayon Algoritması Seçimi, Medium, https://medium.com/path-internet/derin-%C3%B6%C4%9Frenme-%C3%A7al%C4%B1%C5%9Fmalar%C4%B1nda-optimizasyon-algoritmas%C4%B1-se%C3%A7imi-74bf9d430135, (Ziyaret tarihi : 18 Aralık 2020).

[30] Kulikovskikh I., Proghorov S., Lapic T., Legoviç T., Simuç T., BioGD: Bio-inspired Robust Gradient Descent, *Plos One Jurnals*, 2019, **14**(7), 1-19.

[31] İnik Ö., Ülker E., Derin Öğrenme ve Görüntü Analizinde Kullanılan Derin Öğrenme Yöntemleri, *Gaziosman Paşa Bilimsel Araştırma Dergisi*, 2017, **6**(3), 85-104.

[32] Aksakallı K. I., Bayındır L., Derin Öğrenme Kullanarak Oda Seviyesinde Wi-Fi Parmak İzi Tabanlı İç Ortam Konumlandırma, *Erzincan Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 2020, **13**(2), 483-503.

[33] Domingos P., Bayesian Averaging of Classifiers and the Overfitting Problem, *Department of Computer Science and Engineering*, 2010, **7**(6), 223-230.

[34] Budharija A., Dropout Deep Machine Learning, Medium, https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5/ (Ziyaret tarihi : 19 Aralık 2020).

[35] Karasulu B., Çoklu Ortam Sistemleri İçin Siber Güvenlik Kapsamında Derin Öğrenme Kullanarak Ses Sahne ve Olayların Tespiti, *İstanbul Üniversitesi Acta Infologica Dergisi*, 2019, **3**(2), 60- 82.

[36] Pençe İ., Cetişli B., El Yazı Karakterlerinin Kapalı Cebirsel Eğrilerle Modellenmesi ve Sınıflandırılması, *Mühendislik ve Fen Bilimleri Dergisi*, 2013, **2**(8), 1-7.

[37] Gündüz G., Cedimoğlu İ. H., Derin Öğrenme Algoritmalarını Kullanarak Görüntüden Cinsiyet Tahmini, *Bilgisayar ve Bilişim Sistemleri Dergisi*, 2019, **8**(1), 2208-2228.

[38] Krizhevsky A., Sutskever I., Hinton G. E., ImageNet Classification with Deep Convolutional Neural Networks, *Jurnals University of Toronto*, 2013, **6**(2), 84-90.

[39] Toğaçar M., Burhan Ergen B., Fatih Özyurt F., Evrişimsel Sinir Ağı Modellerinde Özellik Seçim Yöntemlerini Kullanarak Çiçek Görüntülerinin Sınıflandırılması, *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 2020, **32**(1), 47-56.

[40] Ferguson M., Ak R., Yung Tsun Lee Y. T., Lax K. H., Automatic Localization Of Casting Defetcs With Convolutional Neural Network, *Research Gate*, 2017, **12**(1), 1726-1735.

[41] Kızrak A., Derin Bir Karşılaştırma : Inception & Res-Net Veriyonları, Medium, https://ayyucekizrak.medium.com/deri%CC%87n-bi%CC%87r-kar%C5%9Fila%C5%9Ftirma-inception-res-net-versiyonlar%C4%B1-f5cfb83df131/ (Ziyaret tarihi : 21 Aralık 2020).

[42] Zhao B., Feng J., Wu X., Yan S., A Survey On Deep Learning-Based Fine-Grainet Object Classification and Semantic Segmentation, *International Jurnal Of Otomation and Computing*, 2017, **14**(2), 119-135.

[43] He K., Zhang X., Ren S., Sun J., Deep Residual Learning For Image Recognition, *Institute of Engineers and Everyone Else*, 2015, **11**(4), 770-778.

[44] Amidi S., Convolutional Neural Network Hand Books, Standford, https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-convolutional-neural-networks/ (Ziyaret tarihi : 22 Aralık 2020).

[45] Gandhi R., R-CNN, Fast RCNN, Faster R-CNN, YOLO-Object Detection Algorithams, Towards Data Science, https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e/ (Ziyaret tarihi : 22 Aralık 2020).

[46] Kumar A., R-CNN or Region-Based Convolutional Neural Network, Stackoverflow, https://stackoverflow.com/questions/44255411/which-is-best-for-object-localization-among-r-cnn-fast-r-cnn-faster-r-cnn-and/ (Ziyaret tarihi : 22 Aralık 2020).

[47] Bektaş M., Tensorflow Ortamında Nesne Tanıma Uygulaması, Medium, https://medium.com/@mbektas/tensorflow-ortam%C4%B1nda-nesne-tan%C4%B1ma-uygulamas%C4%B1-defect-type-detection-using-faster-r-cnn-1be1e1f59c8c/ (Ziyaret tarihi : 22 Aralık 2020).

[48] Zeren M. T., Aytulun S. K., Kırelli Y., Comparison of SSD and Faster R-CNN Algorithms to Detect the Airports with Data Set Which Obtained From Unmanned Aerial Vehicles and Satellite Images, *European Journal of Science and Technology*, 2020, **19**(1), 643-658.

[49] Aktaş A., Derin Öğrenme Yöntemleri ile Görüntü İşleme Uygulamaları, Yüksek Lisans Tezi, Marmara Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2018, 411130.

[50] Redmon J. C., YOLO: Real-time Object Detection, Pjreddie, https://pjreddie.com/darknet/yolo/ (Ziyaret tarihi : 23 Aralık 2020).

[51] Dikbayır H. S., Bülbül H. İ., Derin Öğrenme Yöntemleri Kullanarak Gerçek Zamanlı Araç Tespiti, *Türk Bilim Araştırma Vakfı Dergisi*, 2020, **13**(3), 1-14.

[52] Hanbay K., Üzen H., Nesne Tespit ve Takip Metotları: Kapsamlı Bir Derleme, *Türk Doğa ve Fen Dergisi*, 2017, **6**(2), 40-49.

[53] Pişkin M., Nesne Tespiti ve Nesne Tanıma Süreçleri, Blog, https://mesutpiskin.com/blog/nesne-tespiti-ve-nesne-tanima.html/ (Ziyaret tarihi : 25 Aralık 2020).

[54] Balcı M., Altun A. A., Taşdemir Ş., Görüntü İşleme Teknikleri Kullanılarak, Napolyon Tipi, Kirazların Sınıflandırılması, *Selçuk Teknik Dergisi*, 2016, **15**(3), 221-237.

[55] Elen A., Görüntü İkileştirme için Global Eşikleme Yöntemleri Üzerine Bir İnceleme, *Mühendislik Bilimleri ve Araştırma Dergisi*, 2020, **2**(2), 38-49.

[56] Yalçın M., Normal Dağılım ve Veri Bilimi'ndeki Yeri, Medium, https://medium.com/datarunner/normaldagilim-589846bb850a/ (Ziyaret tarihi : 23 Aralık 2020).

[57] Kahraman A. S., Farshi T. R., Demirci R., Renkli Görüntülerin Çok Seviyeli Eşiklenmesi ve Sınıflandırılması, *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, 2018, **6**(4), 846-859.

[58] Perihanoğlu G. M., Dijital Görüntü İşleme Teknikleri Kullanılarak Görüntülerden Detay Çıkarım, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2015, 389200.

[59] Atalı G., Özkan S. S., Karayel D., Image Damage Analysis With Morphological Image Processing Technique Using Artificial Neural Network, *Akademik Platform Mühendislik ve Fen Bilimleri Dergisi*, 2019, **4**(1), 1-7.

[60] Daş R., Polat B., Gürkan Tuna G., Derin Öğrenme ile Resim ve Videolarda Nesnelerin Tanınması ve Takibi, *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 2019, **31**(2), 571-581.

[61] Uğur N., Kocaeli Üniversitesi Amblemi, Basın ve Halkla İlişkiler Müdürlüğü, http://www.kocaeli.edu.tr/index.php/ (Ziyaret tarihi : 28 Aralık 2020).

[62] Aydın Y. E., OpenCV 'de Pramit Oluşturma ve Dörtgen Ekleme, Mobilhanem, https://www.mobilhanem.com/opencvde-piramit-olusturma-dortgen-ekleme/ (Ziyaret tarihi : 28 Aralık 2020).

[63] Erkan U., Görkem L., Tuz-Biber Gürültüsünde Tekrarsız Median Filtre, *Gaziosmanpaşa Bilimsel Araştırma Dergisi*, 2017, **6**(2), 11-19.

[64] Özer F., Özkaya U., Süperpiksel Algoritmalarının Gürültülü İmageler İçin Bölütleme Performansının İncelenmesi, *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, 2018, **7**(4), 225- 232.

[65] Aydın A., Kocaeli Üniversitesi'nin Açılış Tarihi Belli Oldu, Tek Kocaeli, https://www.tekkocaeli.com/kocaeli-universitesinin-acilis-tarihi-belli-oldu/ (Ziyaret tarihi : 10 Ocak 2021).

[66] Kızrak A., Türkçe Derin Öğrenme Kaynakları, Github, https://github.com/ayyucekizrak/turkce-derin-ogrenme-kaynaklari/ (Ziyaret tarihi : 10 Ocak 2021).

[67] Yaraş N., Vehicle Type Classification With Deep Learning, Yüksek Lisans Tezi, İzmir Teknoloji Enstitüsü, Lisansüstü Eğitim Enstitüsü, İzmir, 2020, 631254.

[68] Kılıç S., Klinik Karar Vermede ROC Analizi, *Jurnal Of Mood Disorders*, 2013, **3**(3), 135-140.

[69] Kasturi S. N., Underfitting and Overfitting Machine Learning and How to Deal With, Towards Data Science, https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf/ (Ziyaret tarihi : 10 Mart 2021).

[70] Kamber E., Bolatan G. İ., Endüstri 4.0 Türkiye Farkındalığı, *Mehmet Akif Ersoy Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, 2019, **11**(30), 836-847.

[71] Somuncu Ş., Türkiye 'de Son 15 Yılda En Çok Hangi Marka Otomobil Satıldı, Euronews, https://tr.euronews.com/2019/12/26/turkiye-de-son-15-yilda-en-cok-hangi-marka-otomobil-satildi/ (Ziyaret tarihi : 13 Şubat 2021).

[72] Çağıl G., Yıldırım B., Bir Montaj Parçasının Derin Öğrenme ve Görüntü İşleme ile Tespiti, *Zeki Sistemler Teori ve Uygulamaları Dergisi*, 2020, **3**(2), 31-37.

[73] Joseph C. R., YOLO: Real-time Object Detection, Pjreddie, https://pjreddie.com/darknet/yolo/ (Ziyaret tarihi : 23 Şubat 2021).

[74] Murat S., İnsansız Hava Aracı Görüntülerinden Derin Öğrenme Yöntemleri ile Nesne Tanıma, Yüksek Lisans Tezi, Maltepe Üniversitesi, Lisansüstü eğitim Enstitüsü, İstanbul, 2021, 631128.

[75] Mesci Y., YOLO Algoritmasını Anlamak, Medium, https://medium.com/deep-learning-turkiye/yolo-algoritmas%C4%B1n%C4%B1-anlamak-290f2152808f (Ziyaret tarihi : 12 Mart 2021).

[76] Kavitha N., Chandrappa D. N., Optimized YOLOv2 Based Vehicle Classification and Tracking Intelligent Transportation System, *Elsevier Enhanced Reader*, 2021, **2**(21), 1-8.

[77] Laroca R., Severo E., Zanlorensi L. A., Oliveira L. S., Gonçalves R., Schwartz W. R., Menotti D., A Robust Real-Rime Automatic License Plate Recognition Based YOLO Detector, *IEEE International Joint Conference on Neural Networks*, Rio de Jenerio, Brazil, 8-13 Temmuz 2018.

[78] Rafique M. A., Pedrycz W., Jeon M., Vehicle License Plate Detection Using Region-Based Convolutional Neural Networks, *Soft Computing*, 2018, **22**(19), 6429-6440.

[79] Li H., Wang P., Shen C., Toward End-to-End Car License Plate Detection and Recognition With Deep Neural Networks, *IEEE Transactions on Intelligent Transportation Systems*, 2018, **99**(2), 1-11.

[80] Bulan O., Kozitsky V., Ramesh P., Shreve M., Segmentation and Annotation Free License Plate Recognition With Deep Localization and Failure Identification, *IEEE Transactions on Intelligent Transportation Systems*, 2018, **18**(9), 2351-2363.

[81] Zhuang J., Hou S., Wang Z., Zha J., Towards Human-Level License Plate Recognition, *In Proceedings of the European Conference on Computer Vision*, Munich, Germany, 8-14 Eylül 2018.

KİŞİSEL YAYIN VE ESERLER

Kırca S., Kelekçi E., Ayaz M., Bir Depolama Tesisi İçin Otomasyon Sistemi Tasarımı, *Pamukkale Üniversitesi Mühendislik ve Doğa Bilimleri Dergisi*, 2019, **25**(2), 157-164.

Kırca S., Gürel H. H., Araç ve Plaka Tanıma, *IMASCON Uluslararası Marmara Fen Bilimleri Kongresi*, Kocaeli, Türkiye, 04-05 Aralık 2020.

ÖZGEÇMİŞ

Serkan KIRCA, 2010 yılında girdiği Kocaeli Üniversitesi Mekatronik Mühendisliği Bölümü'nden 2013 yılında derece ile mezun oldu. 2018 yılında Aynı üniversitede Bilişim Sistemleri Mühendisliği bölümünde Yüksek Lisansa başlamıştır. Değişik sektörlerde görüntü işleme ve derin öğrenme ile alakalı TÜBİTAK, TEY-DEB destekli projelerde görev almış ve başarılı bir şekilde tamamlamıştır. Şuan Anadolu Türk Sigorta Anonim Şirketi'nde Yapay Zekâ ve Veri Bilimi Mühendisi olarak çalışmaktadır.