**T.C.**
**FIRAT ÜNİVERSİTESİ**
**MÜHENDİSLİK FAKÜLTESİ**

# DERİN ÖĞRENME TABANLI BANKA LOGOSU TANIMA SİSTEMİ

Enes Safa ERBAŞ
Turgay HOPAL

Tez Danışmanı
Doç. Dr. Burhan ERGEN

BİTİRME TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

**ELAZIĞ**
**2019**

**T.C.**
**FIRAT ÜNİVERSİTESİ**
**MÜHENDİSLİK FAKÜLTESİ**
**BİLGİSAYAR MÜHENDİSLİĞİ**

# DERİN ÖĞRENME TABANLI BANKA LOGOSU TANIMA SİSTEMİ

Enes Safa ERBAŞ
Turgay HOPAL

BİTİRME TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Bu bitirme tezi 15/05/2019 tarihinde, aşağıda belirtilen jüri tarafından oybirliği/oyçokluğu ile başarılı/başarısız olarak değerlendirilmiştir.

| (İmza) | (İmza) | (İmza) |
|---|---|---|
| Prof. Dr. Mehmet KAYA | Doç. Dr. Burhan ERGEN | Dr. Öğr. Üyesi Erkan DUMAN |

## ÖZGÜNLÜK BİLDİRİMİ

Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini, alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

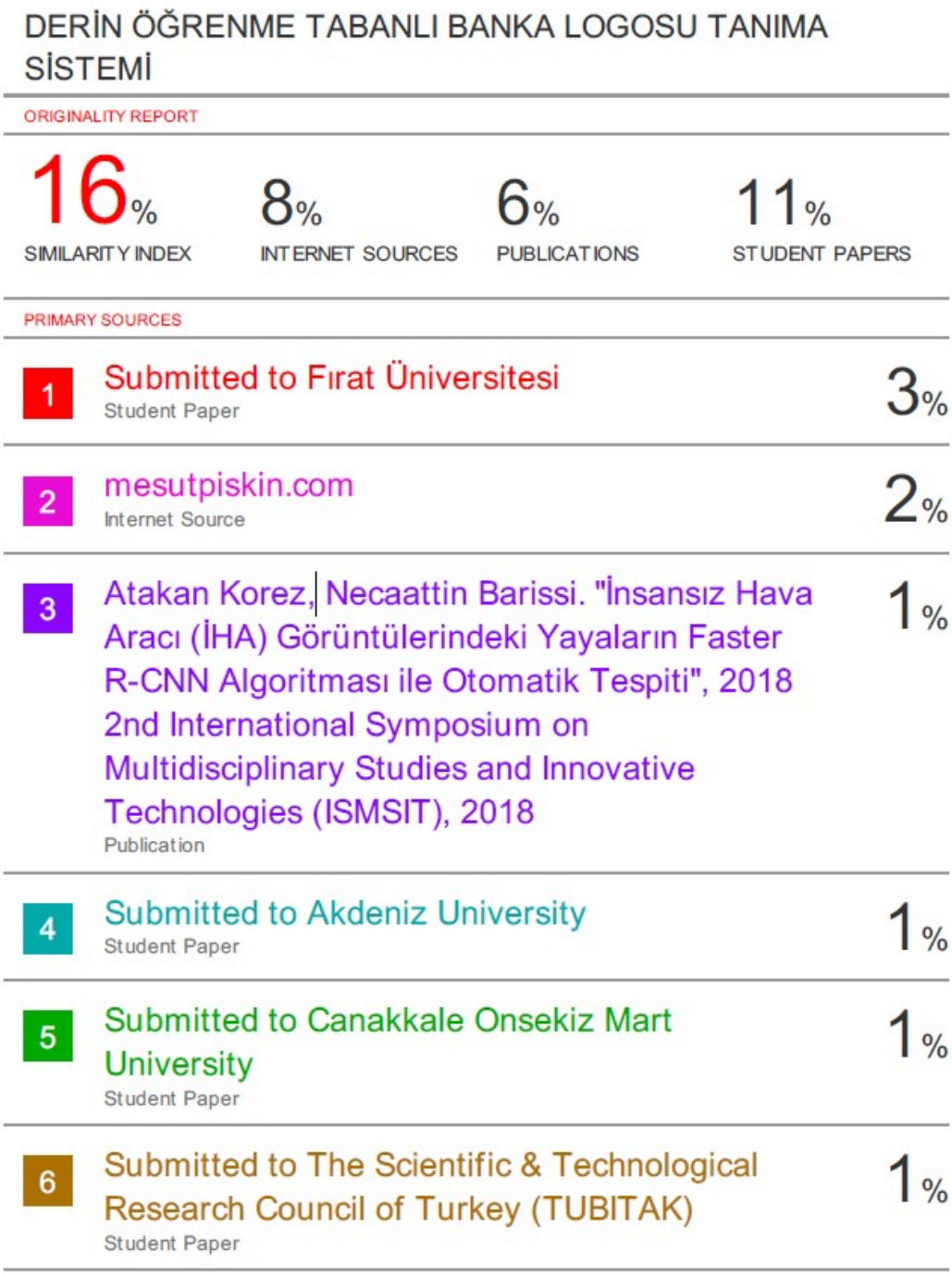
Fırat Üniversitesi
Bilgisayar Mühendisliği
23119 Elazığ

15/05/2019

Enes Safa ERBAŞ

Turgay HOPAL

**BENZERLİK BİLDİRİMİ**



DERİN ÖĞRENME TABANLI BANKA LOGOSU TANIMA SİSTEMİ

ORIGINALITY REPORT

| 16% | 8% | 6% | 11% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Submitted to Fırat Üniversitesi<br>Student Paper | 3% |
| 2 | mesutpiskin.com<br>Internet Source | 2% |
| 3 | Atakan Korez, Necaattin Barissi. "İnsansız Hava Aracı (İHA) Görüntülerindeki Yayaların Faster R-CNN Algoritması ile Otomatik Tespiti", 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), 2018<br>Publication | 1% |
| 4 | Submitted to Akdeniz University<br>Student Paper | 1% |
| 5 | Submitted to Canakkale Onsekiz Mart University<br>Student Paper | 1% |
| 6 | Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK)<br>Student Paper | 1% |

## TEŞEKKÜRLER

Bu çalışmamız, Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nde, Sayın Doç. Dr. Burhan ERGEN 'in yönlendirmesi ve gözetimi altında hazırlanmıştır. Projemizin hazırlaması sürecinde bilgi, görüş ve eleştirilerinden yararlandığımız hocamız Sayın Burhan ERGEN 'e en içten duygularımızla teşekkür ederiz.

## İÇİNDEKİLER

Sayfa No

## ŞEKİLLER LİSTESİ

## KISALTMALAR LİSTESİ

| | |
|---|---|
| **R-CNN** | **:** Region-Based Convolutional Neural Network |
| **SVM** | **:** Support Vector Machine |
| **FAST R-CNN** | **:** Fast Region-Based Convolutional Neural Network |
| **FASTER R-CNN** | **:** Faster Region-Based Convolutional Neural Network |
| **SSD** | **:** Single Shot Detector |

# ÖZET

Bu proje kapsamında günümüzde gelişmekte olan derin öğrenme algoritmalarını kullanarak tasarlanan görüntü işleme ile nesne tanıma yapılmaktadır. Bu proje Google tarafından geliştirilen derin öğrenme uygulamalarında kullanılan Tensorflow yardımıyla geliştirilmiştir.

Nesne tanıma projemizde hazır olarak eğitilmiş Tensorflow kütüphanesinde bulunan Faster R-CNN modelini, oluşturulan veri seti ile tekrar eğiterek istenilen nesneleri tanıyan bir model elde edilmiştir.

Tez kapsamında veri setinin oluşumundan modelin eğitimine kadar hangi aşamalardan geçtiği anlatılmaktadır. Herhangi bir nesne tanıma uygulaması için hangi modelin daha verimli çalışacağı konusunda araştırmalar yapılmıştır. Tensorflow'un hazır olarak eğitilmiş modelleri paylaşılmaktadır. Eğitilen modellerin daha iyi sonuçlar vermesi için, veri setinin nasıl oluşturulacağı hakkında bilgiler verilmektedir.

Günümüzün gelişen teknolojisinin sunmuş olduğu imkanlar neticesinde istenilen her nesnenin bilgisayarlı görü uygulamaları tarafından tanınması için gerekli tüm aşamalardan bahsedilmektedir.



**Anahtar kelimeler:** Tensorflow, nesne tanıma , veri seti , model

## ABSTRACT

It is an object recognition project with image processing designed using deep learning algorithms that are developing today. This project was developed with the help of Tensorflow used in deep learning applications developed by Google.

In our object recognition project, a model that recognizes the desired objects by retraining the Faster R-CNN model in the Tensorflow library, which is readily available in the Tensorflow library, has been obtained.

Within the scope of the thesis, the stages of the data set from the formation of the model to the education are explained. Research has been carried out on which model will work more efficiently for any object recognition application. Tensorflow's readily available models are shared. In order for the trained models to give better results, information is given about how to create the data set.As a result of the opportunities provided by today's developing technology, all the necessary stages for the recognition of the desired object from computerized applications are mentioned.



**Keywords:** Tensorflow, object recognition, data set , model

# 1.GİRİŞ

Gelişen teknolojinin insan hayatına olan adaptasyonunun çok önemli olduğu bilinmektedir. Özellikle bilgi ve bilgisayar teknolojilerinin insan hayatını kolaylaştırma, iyileştirme gibi bir misyonu olduğu ve gün geçtikçe kullanım alanlarının arttığı gözlenmektedir. Görüntü işleme ve nesne tanımadaki öneminin anlaşılmasının ardından insan hayatına şekil veren bir bilim alanı olarak ortaya çıkmıştır.

Bilgisayar teknolojisinin gelişmesi sayesinde sağlık, askeri, eğitim vb. birçok alanda en az iş gücü ve maliyetle en iyi performans sağlanması amaçlanmış ve başarılı olduğu gözlemlenmiştir. Bu sayede günümüzde cihazlar insan gibi düşünüp karar verme mekanizmasına yetecek kadar gelişme göstermiş yapay zeka alanı olarak çeşitli alanlarda kullanılmaktadır.

Nesne tanımada eğitilen algoritmaların farklı alanlarda kullanıldığı da bilinmektedir. Bu algoritmalar istenilen nesnenin veri setine göre eğitilmektedir. Bu yaklaşımla gerçekleştirilen projemizde kendi veri setimizi oluşturarak modelin eğitimini gerçekleştirdik. Bu doğrultuda istenilen herhangi bir alanda hazır veri setlerine ihtiyaç duyulmadan oluşturulan veri setini gerçek zamanlı, video veya resimler üzerinde nesne tanımanın gerçekleşebileceği anlatılmaktadır.

Bu tez kapsamında geliştirilen projenin ilk aşaması nesne tanımada kullanılan algoritmaları ve yöntemleri öğrenmektir. Bu aşamalar 2. bölümde anlatılmaktadır. Bu doğrultuda oluşturduğumuz veri seti ve eğitim kısmı 3. bölümde anlatılmaktadır. 3. bölümde ise sonuç ve gelecek çalışmalar üzerine anlatımlara yer verilmektedir.

## 1.1 Literatür Taraması

Tez kapsamında konu ile ilişkili olarak birçok araştırma yapılmıştır. Yapılan bu araştırmalarda proje ile alakalı benzerlikler ve yenilikler anlatılmaktadır.

Gangal ve Gürbüz tarafından yapılan çalışmada[2], önceden çekilmiş renkli fotoğraflarda bulunan farklı açılardaki insan yüzlerinin bulunması amaçlanmıştır. Önerilen yeni kayan pencere yaklaşımıyla önceden kullanılan metotlara gerek kalmadan, görüntüdeki mevcut yüz bölgelerinin tamamına yakın bir bölümü tespit edilebilmektedir. Uzun bir videoda işleme süresi sorununu çözmek için Viola ve Jones tarafından geliştirilen Haar Cascade benzeri özellik tabanlı kaskad (cascade) yüz sınıflandırıcısı ve RGB ile bir ten rengi sınıflandırıcısı birlikte kullanılmıştır. Bao Yüz Veri tabanı, UCD Renkli Yüz Veri tabanı ve internet üzerinden ücretsiz olarak elde edilen görüntüler üzerinde yapılan deneysel çalışma sonuçları oldukça verimli olduğu gözlemlenmiştir.

Gandhi tarafından yapılan çalışmada[4], nesne tespit algoritmaları ve sınıflandırma algoritmaları arasındaki fark, tespit algoritmalarında, resmin içine yerleştirmek için üzerinde durulan nesnenin etrafına sınırlayıcı bir kutu çizmeye çalışmak olduğundan bahsedilmiştir. Ayrıca, bir nesne algılama durumunda yalnızca bir sınırlayıcı kutu çizilmesi gerekmeyebilir, görüntüde farklı ilgi alanlarını temsil eden birçok sınırlayıcı kutu olabilir ve daha önce kaç tane olduğu bilinmeyebilir. Bunun sabit olmamasının nedeni, üzerinde durulan nesnelerin oluşum sayısının sabit olmamasıdır. Bu sorunu çözmek için saf bir yaklaşım, görüntüden farklı ilgi alanlarını almak ve o bölgedeki nesnenin varlığını sınıflandırmak için bir CNN kullanmak olacaktır. Bu yaklaşımla ilgili sorun, üzerinde durulan nesnelerin görüntü içinde farklı uzamsal konumlara ve farklı en/boy oranlarına sahip olabilmesidir. Bu nedenle, çok sayıda bölge seçmek zorunda kalınabilir ve bu hesaplamada istenmeyen sonuçlar getirebilir. Bu nedenle, R-CNN, YOLO gibi algoritmalar bu oluşumları bulmak ve hızlı bir şekilde bulmak için geliştirilmiştir.

Abbas ve DR.Singh tarafından yapılan çalışmada[21], araştırma çalışmaları R-CNN, Fast R-CNN, YOLO algoritmalarıdır. Bir görüntüdeki istenilen böyleyi çıkarmak için Bölge Teklif Ağı'nın (RPN) kullanılmasına odaklanmıştır. RPN, nesnellik puanına dayanan bir görüntü çıkarır. Çıktı nesneleri sınıflandırma için Roll Polling kullanılmıştır.Bu çalışma, görüntülere özel veri seti kullanarak daha hızlı R-CNN eğitimi üzerine odaklanır. Eğitilmiş ağlar sayesinde, birden fazla nesneden oluşan bir görüntüdeki nesneleri etkili bir şekilde algılamaktadır.

HSU ve arkadaşları tarafından yapılan çalışmada[24], araç tespiti için basitleştirilmiş hızlı bölge tabanlı evrişimli bir sinir ağı, yani Faster R-CNN önerilmektedir. Faster R-CNN, derin evrişim ağları kullanan nesne tanıma için iyi bilinen bir yöntemdir. Orijinal Faster R-CNN bölgesel teklif ve nesne tanıma olmak üzere iki ayrı bölümden oluşur. Fast R-CNN'deki nesne

tanıma bölümü, eğitim sürecini hızlandırmak için kaldırabilen bir sistem için gereksizdir. Deneylerde, tespit doğruluğunu göstermek için Virginia Tech Transportation Institute (VTTI) tarafından sunulan SHRP 2 NDS veritabanını kullanarak amaçlanan yöntemler test edilmektedir.

Pişkin tarafından yapılan çalışmada[1], nesne tespit ve tanıma süreçleri birkaç adımda gösterilmiştir. Bu aşamalar, veri girişi, veri ön işleme ve aşamaları, öznitelik çıkarımı ve son olarak da tanımlama olarak gösterilmektedir. Veri girişi aşamasında, hazırlanan veri gürültülerinden ayrıştırılmak istenilen formata getirilmek gibi amaçlara sisteme girdi olarak verilir. Ön işleme aşamasında, verilerin yapılacak olan işlemin amacına uygun hale gelmesi için hazırlamak veya engel teşkil etmesinin önüne geçmek için uygulanan bir takım sabit olmayan yöntemlerdir. Veri ön işlemenin nedenlerini sıralamak gerekirse; gürültülü parazitli veriler, tam olmayan eksik veriler ve tutarsız veriler olarak sayılabilmektedir. Veri ön işleme için birçok algoritma veya yöntem mevcuttur, bu yöntemler insan gözü ile denetleme olabileceği gibi karmaşık sinir ağları gibi yapılar da olmaktadır. Öz nitelik aşamasında, ön işleme tabi tutulmuş veri üzerinde daha önceden belirlenen nesnenin elde edilmesidir. Son olarak tanımlama aşamasında ise, tespit ettiğimiz görüntüden çıkarım yapabilmek, tanımak yani bu görüntünün ne olduğunu anlama aşaması olarak basitçe tanımlanabilir.

CHEN ve arkadaşları tarafından yapılan başka bir çalışmada[22], görüntülerin karmaşık arka planda birçok nesnesi bulunmaktadır. Bu nesnelerin ve buradaki ana nesnelerin nasıl tanımlanacağı ve ana nesneler ile diğer nesneler arasındaki ilişkinin nasıl anlaşılacağı bu çözümün odak noktası olduğu kanısına varılmıştır. Bu yazıda, görüntüdeki çok nesneyi algılamak ve tanımak için geliştirilmiş R-CNN ağı kullanılmıştır. Ardından, görüntü ana nesnelerini işaretlemek için ana nesneleri puanlama sistemi öne sürülmüştür. Deneysel sonuçlar, algoritmanın sadece R-CNN'nin üstünlüğünü korumakla kalmayıp aynı zamanda görüntünün ana nesnelerini de tespit ettiğini göstermektedir. Görüntüdeki ana nesnelerin aday bölge büyüklüğü ve nesnelerin görüntü üzerindeki gösterilme miktarları ile ilişkili olduğu tespit edilmiştir.

He ve Gkıoxari tarafından çalışmada[5], nesne örneği segmentasyonu için kavramsal olarak basit, esnek ve genel bir çerçeve sunulmuştur. Kullanılan yaklaşım görüntüdeki nesneleri verimli bir şekilde tespit ederken, aynı anda her örnek için yüksek kalitede bir segmentasyon maskesi oluşturmaktadır. Mask R-CNN olarak adlandırılan yöntem, sınırlayıcı kutu tanıma için mevcut alanla paralel olarak bir nesne maskesini öngörmek için bir alan ekleyerek Faster R-CNN'yi genişletmektedir. Mask R-CNN'nin eğitim aşamasının basit olduğundan da bahsedilmiştir. Ayrıca Maske R-CNN'nin, örneğin insan pozlarını aynı çerçevede tahmin edilmesine olanak sağlayan diğer algoritmalara göre kolay olduğu söylenilmiştir. Örnek segmentasyonu, sınırlayıcı kutu nesnesi tespiti ve kişi kilit noktası tespiti dâhil olmak üzere COCO grubunun üç parçasında da en

iyi sonuçlar gösterilmektedir. Basit ve etkili yaklaşımlarının sağlam bir temel olarak hizmet edeceğini ve örnek düzeyinde tanıma konusunda gelecekteki araştırmalara öncülük edeceği düşünülmektedir.

Jiang ve Learned-Miller tarafından yapılan çalışmada[6], Faster R-CNN kullanarak üç referans veri setinde en iyi sonuç veren yüz algılama performansı gösterilmiştir. Bu çalışmada yapılan deneysel sonuçlar, bunun etkinlik bölge teklif ağı (RPN) modül'ün den gelmektedir. Kıvrımlı katmanların RPN ve Faster R-CNN detektör modül'ü arasında paylaşılmasından dolayı, RPN de fazladan hesaplama yükü olmadan birden fazla kıvrımlı katman kullanılması mümkündür. Faster R-CNN genel nesne tespiti için tasarlanmış olmasına rağmen, uygun bir yüz tanıma eğitim setinde yeniden eğitildiğinde iyi performans veren yüz algılama performansı gösterdiğinden bahsedilmiştir.

Cao ve Chen tarafından yapılan çalışmada[7], CNN'nin donanım mimarilerine uyumunu sağlayan bir SNN'yi bir SNN'ye dönüştürmek için yeni bir yaklaşımı açıklamaktadır. Bu yaklaşımda önce CNN mimarisini SNN'nin gereksinimlerine uyacak şekilde düzenlenir, sonra özel CNN'yi CNN ile aynı şekilde eğitilir ve sonunda öğrenilen ağ ağırlıklarını özel CNN'den türetilen bir SNN mimarisine uygulanmaktadır. Sonuç olarak ortaya çıkan SNN'yi kamuya açık Defense Advanced Research Projects Agency (DARPA) Neovision2 Kulesi ve CIFAR-10 veri setleri üzerinde değerlendirilmekte ve orijinal CNN ile aynı nesne tanıma doğruluğunu gösterilmektedir.

Körez ve Barışçı tarafından yapılan çalışmada[8], nesne algılama uygulama literatüründe sıklıkla kullanılan Faster R-CNN algoritması, bu durumu otomatikleştirmek için değiştirilmiş ve insansız hava fotoğrafları ile çekilen görüntülerde sekmelerin otomatik olarak algılanması için kullanılmıştır.

Doyran ve Akın tarafından yapılan çalışmada[9], son yıllarda nesne tanıma alanında başarısını geliştirmiş bir ESA'yı (Faster R-CNN) zorlu bir veri kümesi olan SUN RGB-D veri kümesinde eğitilmiş ve geliştirilmiştir. Sonuç olarak derinlik alanı daha değişik yöntemler kullanarak eğitildiğinde küçük başarı artışları elde edilmiştir. Gelecek çalışmalarında daha başarılı sonuçlar verecek eğitim yöntemleri deneyerek derinlik bilgisini işleyen alanlardan daha fazla yararlanmayı amaçlamışlardır.

Orhan ve Baştanlar tarafından yapılan çalışmada[10] ise leoparların yerleri derin sinir ağları tarafından tanımlanmaya çalışılmıştır. Bu görevi gerçekleştirmek için iki farklı yöntem uygulanmıştır. Bunlardan ilki, tüm görüntüleri kullanarak sinir ağını eğitmek, ikincisi ise tam

boyutlu görüntülerden kesilmiş görüntü yamalarını kullanarak sinir ağlarını eğitmektir. Yama eğitimi modeli, görüntü eğitimli modelin sonuçlarından daha iyi performans verdiği göstermiştir.

Huang ve arkadaşları tarafından yapılan çalışmada[13], tek bir görüntüdeki birden fazla nesneyi yerelleştirme ve tanımlama yeteneğine sahip doğru makine öğrenme modelleri oluşturmak, bilgisayar vizyon'unda temel bir zorluk olmaya devam etmektedir. TensorFlow Nesne Algılama API'si, TensorFlow'un üzerine inşa edilmiş, nesne algılama modellerini oluşturmayı, eğitmeyi ve dağıtmayı kolaylaştıran açık kaynaklı bir çerçevedir. Google'da, bu kod tabanının kesinlikle bilgisayarlı görme gereksinimleri ve kullanacak kişiler için yararlı olacağını amaçlamışlardır.

Proje kapsamında araştırılan konular hakkında sunmuş olduğumuz literatür taramasında projede kullanılan yapılar anlatılmaktadır. Genel anlamda projenin, araştırılan konulardan farklı olan kısmı veri setidir. Literatür taraması kısmında anlatılmakta olan modellerde eğitimde hazır veri seti kullanılmıştır. Proje kapsamında veri seti hazır olarak kullanılmamıştır. Veri seti oluşturulmuştur.

## 1.2. Amaç ve Kapsam

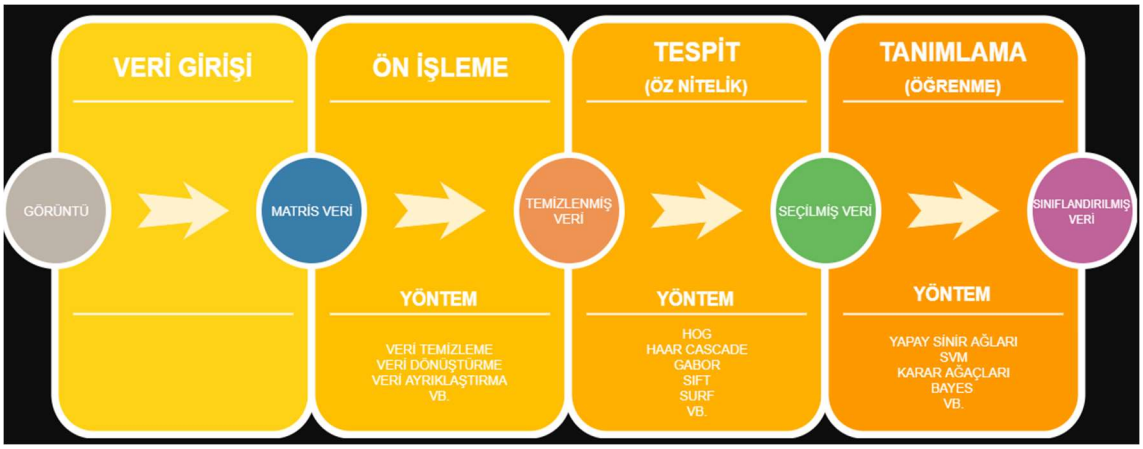
Tez kapsamında anlatılan derin öğrenme ile banka logosu tanıma projesinin amacı veri setinin oluşumu ile banka logo'larını ve banka isimlerini tanımaktır. Veri setinin oluşumu, hazır eğitilmiş modellerden farklı olup projeye yenilik katmaktadır.

İstenilen her alanda veri seti oluşturulup nesne tanıması yapılabileceği anlaşılmıştır. Hayatın her noktasında kullanım alanı mevcuttur. Aranılan bir ürünün tespitinde, suçlu bir insanın yakalanmasında, askeri alanda personele yardımcı olma konularında yardımcı olmaktadır.

# 2. GÖRÜNTÜ İŞLEME TEKNİKLERİ İLE NESNE TANIMA

## 2.1. Nesne Tanıma

Nesne tanıma uzun zamandır bilgisayarlı görü uygulamaları için vazgeçilmezdir. Yıllarca bu konu hakkında çalışılmış ve farklı algoritmalar geliştirilmiştir. Yakın zamanda ise kullanılmaya başlanan GPU teknolojisi ile derin öğrenmenin de katkısıyla daha doğru sonuçlar ve tespitler yapan yöntemler geliştirilmiştir. Nesne tanıma dört ana başlık altında tanımlana bilir.




**Şekil 2.1.** Nesne tespit ve tanımlama aşamaları [1]

Veri girişi hazırlanan verinin gürültülerden ayrıştırılarak istenilen formata dönüşümü için sisteme girdi olarak verilmektedir [1]. Şekil 2,1'de görüldüğü üzere nesne tanıma işlemi veri girişi, ön işleme, tespit, tanımlama aşamalarından oluşmaktadır.

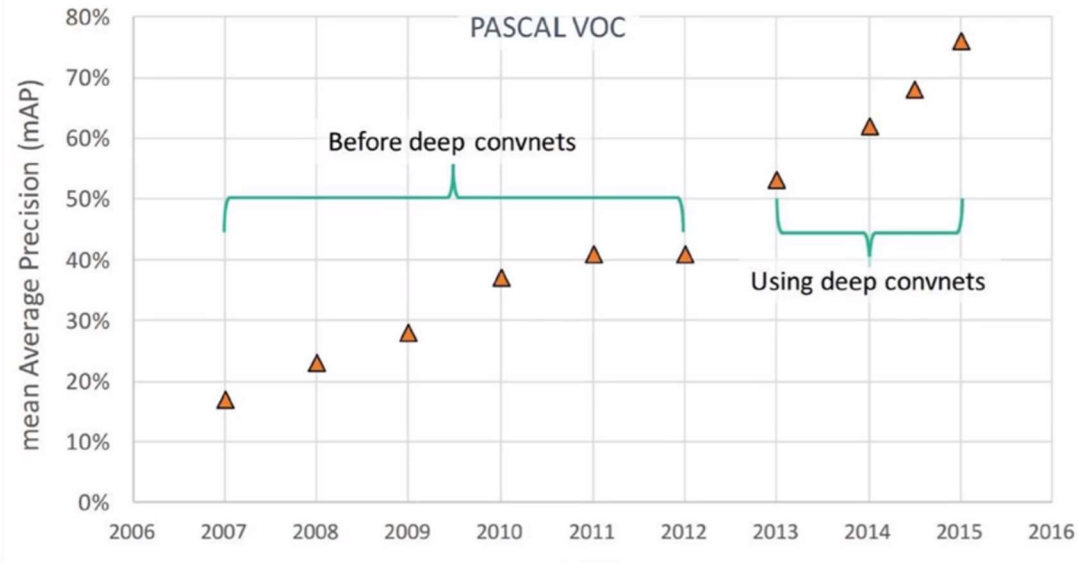
Sisteme girilen verilerin yapılacak işlemin amacına uygun hale gelmesi, hazırlamak ve engel teşkil etmesinin önüne geçmek için yapılan yöntemlere ön işleme denilmektedir. Tespit ise ön işleme tabi tutulmuş verinin tespiti için yapılır ve tanımlama aşamasında bu verinin tanınmasını sağlamaktadır [1].

### 2.1.1 Nesne Tanıma Tarihi

Nesne tanıma bilgisayar görüşü alanında uzun yıllar boyunca en önemli alanlardan birisi olmuştur. Nesne tanıma sistemleri, modern akıllı sistemlerin derinden bütünleşmiş ve her şeyi bilen bir bileşenidir. Nesne tanıma algoritmaları üzerine araştırmalar, optik karakter tanıma sistemleri, montaj hattı endüstriyel denetim sistemleri ve talaş kusur tanımlama sistemlerinin oluşturulması yoluyla fabrika ve ofis otomasyonunda gelişmelere yol açmıştır.

Şekil 2.2 de gösterildiği gibi 2012 yılına kadar pascal data setinde geliştirilen modeller yeteri kadar başarı göstermemektedir 2012 yılına bakıldığı takdirde bu alanda ilerleme kaydedilmemiştir. 2012 yılında alexnet ile birlikte konvolüsyonel sinir ağlarının

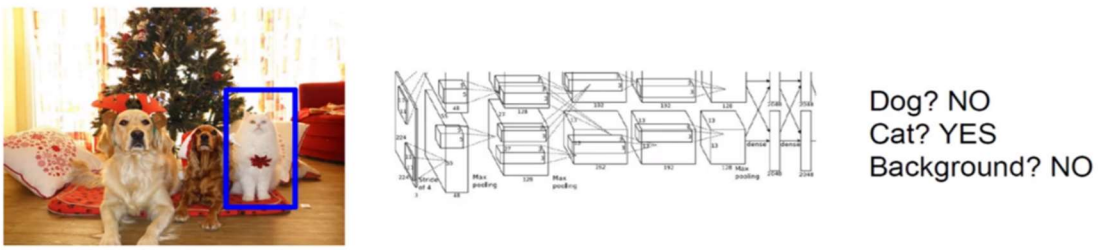
yaygınlaşmasıyla bu alan çok hızlı bir şekilde gelişmeye başlamıştır. Şekil 2.2'de görüldüğü üzere isabet oranının % 40 'tan % 80' e çıktığı görülebilmektedir. Günümde bu oran % 90'ların üzerinde bulunmaktadır.




**Şekil 2.2.** Nesne Tanıma Gelişim Süresi [17]

### 2.1.2 Pencere Kaydırma

Nesne tanımada akla gelen ilk yöntem resim üzerinde pencere oluşturmaktır. Daha sonra oluşan bu pencereyi konvolüsyonel sinir ağından geçirildikten sonra sinir ağı bir sınıflandırma yapıyor ve bu pencere de örnek olarak kedi veya köpek olmadığını belirliyor. Aradığımız sınıflar haricinde arka plan adında bir sınıf daha belirlenmektedir. Sinir ağı aradığı yerde kedi veya köpek bulamadığı takdirde bu alana arka plan diyebilecektir. Bu sayede her pencere için sinir ağını kedi veya köpek için seçim yapmaya zorlanmamış olacaktır [2]. Şekil 2,3'te görüldüğü gibi pencere kaydırma işlemi görüntünün her yerinde arama yaptıktan sonra istenilen nesneyi bulmuştur.



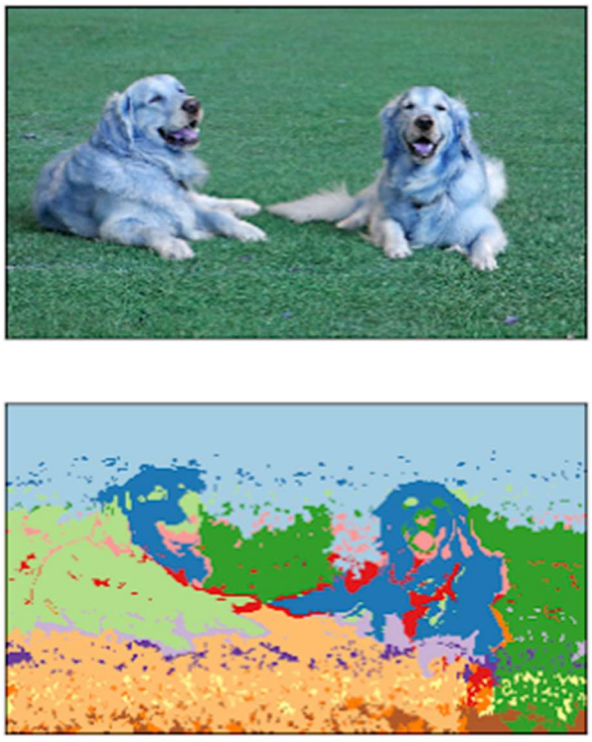

**Şekil 2.3.** Pencere kaydırma [17]

Görüntü üzerinde oluşan yen pencerede eğer sinir ağı iyi eğitilmiş ise bu pencerede kedi veya köpek olduğunu belirleyecektir. Ancak sinir ağı resimdeki kedi ve köpeklerin yerini belirleyemez, bu yüzden pencereyi resmin her yerinde gezdirmesi gerekmektedir.

Farklı boyutlarda pencereler oluşturup aranılan nesnenin bulunması için görüntü üzerinde pencereleri gezdirmesi gerekir ve oluşan her pencereyi tekrardan sinir ağına sokması ve tanımlama yapması gerekmektedir. Bu işlem çok fazla bilgisayar gücüne ihtiyaç duymaktadır [2].

### 2.1.3 Bölge Önerisi

Kayan pencere yöntemine alternatif olarak görüntüye bölge önerisi algoritmaları uyguladıktan sonra bulunan bölgelerde nesne tanıma işlemi gerçekleştirilmiştir. Resimlerde ortalama bin veya iki bin tane nesne bulunma ihtimali olan bölgeler oluşturulmuştur. Bu bölgeler resimdeki renklerden tonlarına göre işaretlenmiştir. Daha sonra işaretlenen alanlar konvolüsyonel sinir ağından geçirilerek aranan nesnenin olup olmadığı tespit edilebilmektedir. Bu yöntemde kullanılan Selective Search algoritması, yapılan işlem sayısını ciddi anlamda azalta bilmektedir [3].



**Şekil 2.4.** Bölge önerisi

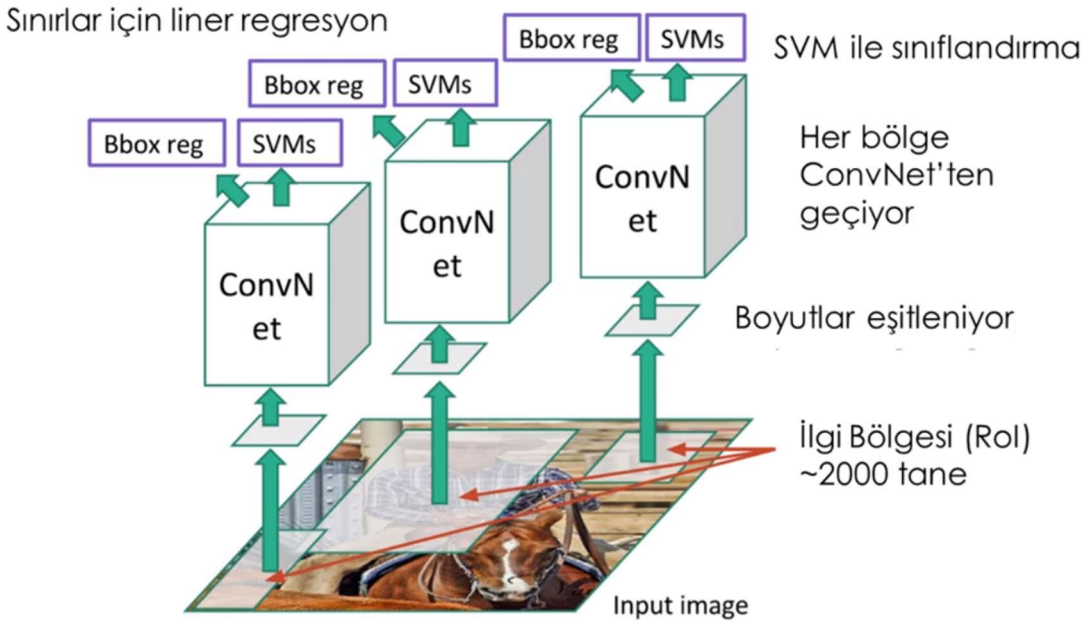
Selective Search, başlangıç bölgelerini bulduktan sonra her bir bölgeyi listeye eklemektedir. Daha sonra ise benzer olan bölgeleri gruplayarak bölge sayısını azaltmaktadır ve

bu işlemi iteratif olarak tekrarlamaktadır. Her bir iterasyonda daha büyük bir bölge adayı bulmakta ve aday listesine eklemektedir [3].

### 2.2 Bölge Önerisinde Kullanılan Algoritmalar

#### 2.2.1 R – CNN

Bölge önerisi mantığını kullanarak çalışmaktadır. Çok sayıda bölgede çalışmak yerine, RCNN algoritması görüntüde bir sürü kutu önermekte ve bu kutulardan herhangi birinin herhangi bir nesne içerip içermediğini kontrol etmektedir.

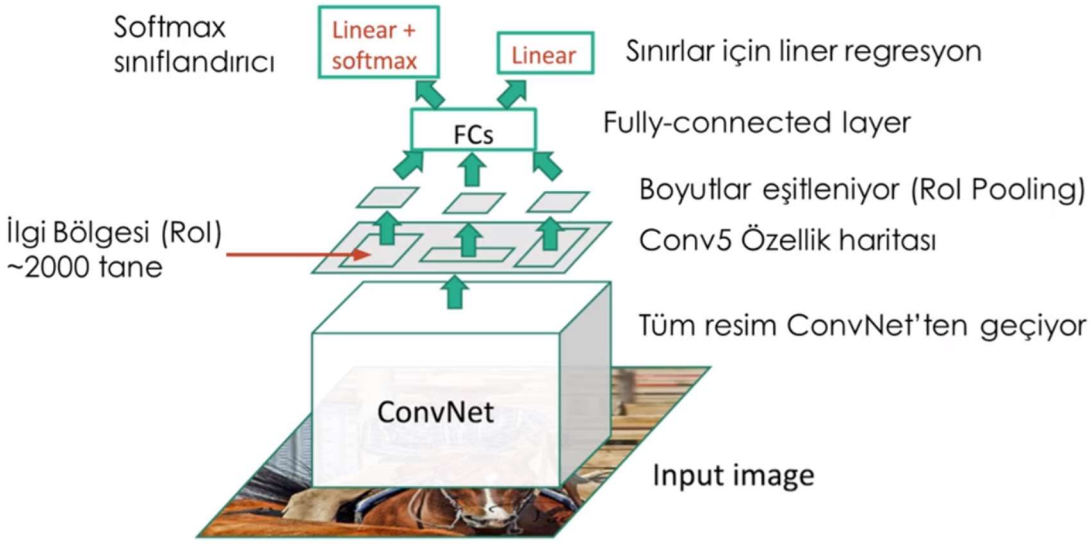



**Şekil 2.5.** R-CNN [18]

Şekil 2,5'te görüntüde önerilen bölgeleri konvolüsyonel sinir ağından geçirebilmek için boyutlar eşitlenmektedir. Daha sonra bu pencereler tek tek sinir ağından geçmektedir ve sonucunda o bölgede sınıflandırma yapmak için SVM kullanılmaktadır [4]. SVM gözetimli öğrenmede kullanılan bir makine öğrenimi yöntemi olarak bilinmektedir. Sınıflandırma için kullanılmaktadır. Pencerenin sınırları için ise Lineer Regresyon kullanılmaktadır. Bununla birlikte tespit edilen nesne etrafına bir dikdörtgen çizilmektedir [5]. Lineer Regresyon bir veya birden fazla bağımsız değişken ile başka bir bağımlı değişken arasındaki bağlantıyı modellemek için kullanılmaktadır. Nesnenin bulunduğu alanın tespiti için önemli bir yöntemdir [5] [6].

### 2.2.2 Fast R – CNN

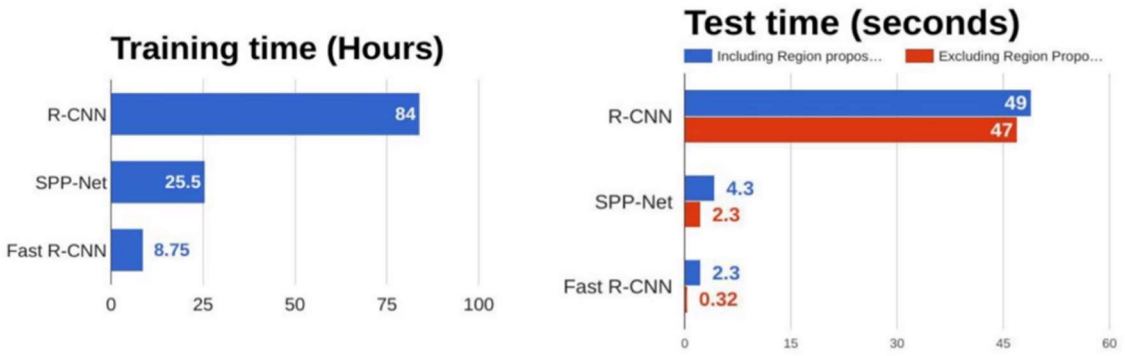
R –CNN'nin gelişmiş versiyonu olarak düşünüle bilir daha hızlı ve daha iyi sonuçlar almak için tasarlanmıştır. Görüntü başına 2.000 kez CNN çalıştırmak yerine, görüntü başına yalnızca bir kez çalıştırabilir ve tüm ilgilenilen bölgeleri tespit edebilir




**Şekil 2.6.** Fast R-CNN [18]

Şekil 2,6'da bölge önerisi yapmak yerine direkt olarak resmi konvolüsyonel sinir ağından geçirilmektedir [5]. Daha sonra orijinal resme uyan yüksek çözünürlüklü bir özellik haritası çıkartılmaktadır. Alınan özellik haritasında Selective Search ile bölge önerisi çıkartılmaktadır. Orijinal resimde iki bin tane bölge önerisi oluşturmak yerine özellik haritasında bölge önerisi yapılmaktadır [6]. Bu sayede her bölgeyi ayrı ayrı sinir ağından geçirmek yerine özellik haritasında bu işlemi yapmaktayız.

Bu işlemin ardından önerilen bölgelerin boyutları düzenlenmekte ve düzenlenen bölgeler Fully Connected Layer'a verilmektedir. Softmax sınıflandırıcısı ile sınıflandırma yapılmakta ve tespit edilen nesnenin sınırları çizilmektedir[6] [7].



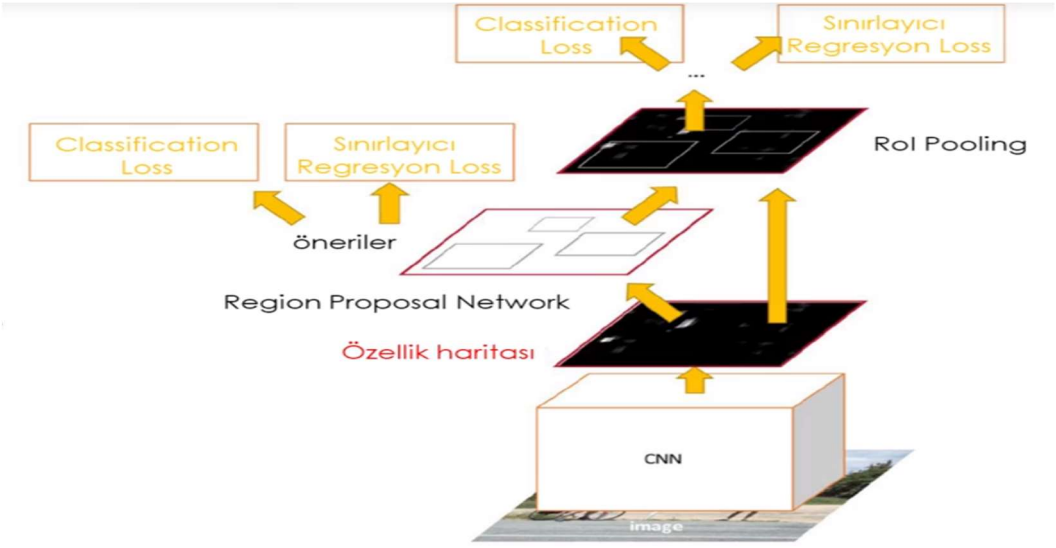

**Şekil 2.7.** Eğitim ve Test Verilerinin Karşılaştırılması [18]

Şekil 2,7'de eğitim süreleri görülmektedir R-CNN modeli 84 saat gibi bir sürede eğitilmektedir. Fast R-CNN modeli ise 8.75 saat sürede eğitilmektedir. Fast R – CNN modeli'nin ne kadar hızlı eğitildiği görülmektedir. Test süreleri ise R-CNN'nin 49 sn de gerçekleşmektedir. Fast R-CNN'nin ise 2.3 saniyede gerçekleşmektedir [7].

### 2.2.3 Faster R-CNN

Faster R-CNN iki ağdan oluşmaktadır.Bunlar bölge teklif ağı ve nesneleri tespit eden ağdır. Bölge teklif ağı, çapa olarak adlandırılan bölgesel alanları içerisinde bulunabilme oranına göre derecelendirir ve belirli bir oranı geçen bölgeleri nesne tespit ağına gönderir [8].

Şekil 2,8'de nesne tespit ağında iki işlem yapılmaktadır. İlk işlem bölge içerisindeki arka plan ve ön plan nesnelerin sınıflandırılmasıdır. Sınıflandırma sonucu ortaya çıkan ön plan nesnesi, farklı çapa boyutlarından dolayı birden fazla kutucuk ile temsil edilir [8].

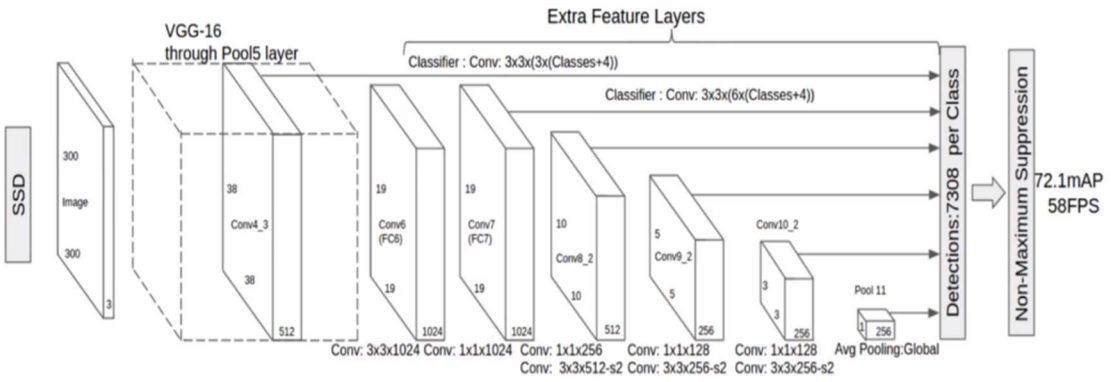



**Şekil 2.8.** Faster R-CNN [18]

Selective Search ile bölge önerisi almak yerine bu önerileri yukarıda anlatılan network içinde yaparak hız kazanılabilmektedir.Giriş olarak bir resim alınmakta ve bu resim konvolüsyonel sinir ağından geçirilerek özellik haritası çıkarılmaktadır [8]. Daha sonra bu aşamada Selective Search ile bölge önerisi almak yerine ayrı bir bölge önerisi ağı oluşturulmaktadır. Bölge önerileri artık bu ağ üzerinde yapılmaktadır. Belirlenen bölgeleri yeniden şekillendirdikten sonra fully connected layer'dan geçirilerek sınıflandırma işlemi yapılmaktadır. Böylece ortaya eğitilmesi gereken dört tane parametre çıkmış olur. Hem bölge önerisi veren ağı hem normal konvolüsyonel işlemleri yaptığımız ağ eğitilecektir [8] [9]. Bu eğitimlerin dengesini tutturmak zor olduğundan bölge önerisi alanının iki tane görevi vardır. Her öneri için orada nesne olup olmadığına karar vermesi gerekmektedir. Aynı zamanda önerilerin pencere büyüklüğünün belirlenmesi gerekmektedir. Daha sonra asıl ağ'a geçildiği zaman yine yapılacak iki farklı görev vardır. Asıl sinir ağı sınıflandırma işlemini gerçekleştirdikten sonra, taranan bölge içerisinde nesne olup olmadığının tespiti yapılacaktır. Daha sonra bulduğu nesnenin sınırlarını belirleyecektir [9] [10].

### 2.2.4 SSD

Tek seferde nesne tanıma yapıla bilen algoritmadır. Giriş olarak alınan resim konvolüsyonel sinir ağından geçirilmektedir. Bu şekilde karşımıza farklı boyutlarda özellik haritaları çıkmaktadır. Tüm özellik haritalarında 3x3 konvolüsyonel filtre yardımıyla az miktarda sınırlayıcı dikdörtgen elde edilmektedir. Her dikdörtgen için hem sınırlar hem de sınıflandırmalar belirlenmektedir. Bu dikdörtgenler her aktivasyon haritasında olduğu için hem küçük nesneleri hem de büyük nesneleri tanıyabilmektedir [10].

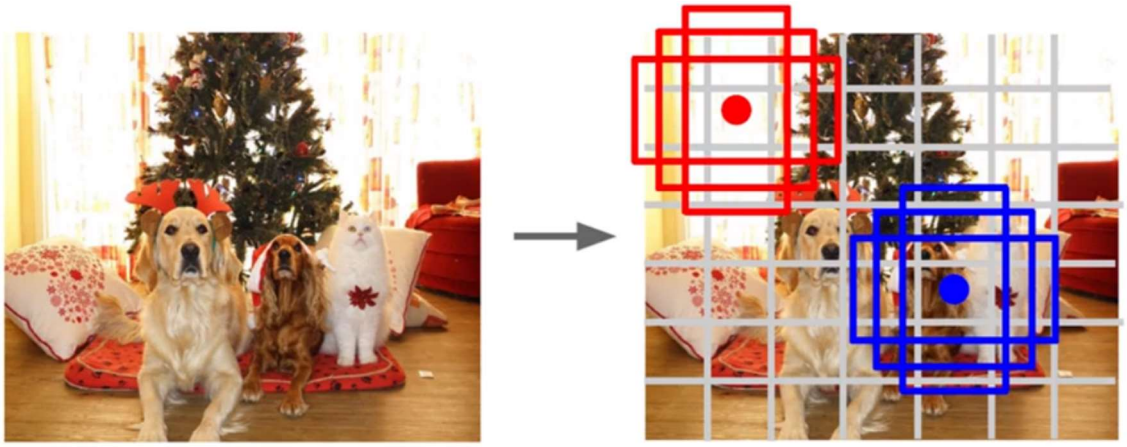



**Şekil 2.9.** SDD [19]

Şekil 2,9'da 300x300 boyutlarında bir resim alınmıştır. Ve boyutlar konvolüsyonel layer'larda zamanla küçük boyutlara düşürülmüştür. Bu sayede resimde küçük ve büyük nesnelerin tespiti yapılabilmektedir. Eğitim esnasında bu doğru olan sınırlar ile tahmin edilen

sınırlar karşılaştırılmaktadır. En iyi tahmini yapan ve 0.5 oranının üzerinde olan dikdörtgenler pozitif olarak etiketlenmektedir [11].

Her bölge için farklı işlemler yapmak yerine bütün tahminler tek seferde bir konvolüsyonel sinir alanı içerisinde yapılmaktadır. Eğitim ve tanıma aşaması çok hızlı olmakla birlikte nesne tanımada tam anlamıyla doğru çalışmamaktadır.



**Şekil 2.10.** SDD grid [19]

Şekil 2.10 da görülen resim grid şeklinde ayarlanmaktadır. Şekilde görüldüğü üzere buradaki örnekte resim 7x7 bölünmüş şekildedir. Bu gridlerin her birisinin içerisinde sınırlayıcı dikdörtgenler bulunmaktadır. Burada temsili olarak her grid için üç dikdörtgen tane görülebilmektedir. Her grid ve her dikdörtgen için nesnenin gerçek yerinin tahmini yapılabilmektedir [11].

### 2.3 Tensorflow ile Nesne Tanıma

TensorFlow, bir dizi görevde dataflow ve farklılaştırılabilir programlama için ücretsiz ve açık kaynaklı bir yazılım kütüphanesidir. Sembolik bir matematik kütüphanesi olan TensorFlow, ayrıca sinir ağları gibi makine öğrenme uygulamaları için de kullanılmaktadır.


**Şekil 2.11.** Tensorflow [20]

### 2.3.1 Tensorflow Object Detection Apı

Tek bir görüntüdeki birden fazla nesneyi yerelleştirme ve tanımlama yeteneğine sahip doğru makine öğrenme modelleri oluşturmak, bilgisayar vizyonunda temel bir zorluk olmaya devam etmektedir. TensorFlow Nesne Algılama API'si, TensorFlow'un üzerine inşa edilmiş, nesne algılama modellerini oluşturmayı, eğitmeyi ve dağıtmayı kolaylaştıran açık kaynaklı bir çerçevedir [12][13].


**Şekil 2.12.** Tensorflow Object Detection Apı [18]

### 2.3.2 Tensorfolw Kurulumu

Tensorflow kütüphanesini kullanabilmek için Anaconda ve Pycharm gibi programların kurulumlarına ihtiyaç duyulmaktadır. Programların kurulumlarının ardından Tensorflow kurulumu için iki farklı seçenek mevcuttur.Bu seçenekler CPU ve GPU dur. Bilgisayarınızda uygun ekran kartı mevcut ise GPU için kurulum yapmak gerekmektedir.Bunun sebebi model eğitimini CPU' ya göre daha hızlı gerçekleştirmesidir [14].

Nvidia' dan farklı bir ekran kartı mevcut ise CPU için kurulum gereklidir, GPU kurulamama sebebi tensorflow sadece nvidia marka ekran kartlarını desteklemektedir. Tensorflow GPU için kurulmak istendiğinde CUDA ve CuDNN kurulması gerekmektedir [14]. CUDA ve CuDNN kurulduktan sonra pip install yapmak gereklidir

```
C:\> pip install --upgrade tensorflow
```

**Şekil 2.13.** Pip install [13]

Tensorflow'un kurulumu şekil 2,13'teki pip install ile sonuçlandırmış olacaktır. İhtiyaç duyulan kütüphaneler olduğu zaman tekrar dan pip install yapılmalıdır. Bu projede ihtiyaç duyulan kütüphaneler için gerekli pip install şekil 2,14'te gösterilmiştir [14].

```
pip install opencv-python
pip install imageio
```

**Şekil 2.14.** Gerekli Kütüphaneler [13]

### 2.3.3 Tensorflow ile Eğitilmiş Modeller

Tensorflow bünyesinde COCO veri seti, Kiti veri seti, Open images veri seti, Ava v2.1 veri seti ve iNaturalist tür tespiti veri kümesinde önceden eğitilmiş bir tespit modelleri koleksiyonunu paylaşmaktadır [15].

| | | | |
|---|---|---|---|
| ssd_mobilenet_v1_coco | 30 | 21 | Boxes |
| ssd_mobilenet_v1_0.75_depth_coco ☆ | 26 | 18 | Boxes |
| ssd_mobilenet_v1_quantized_coco ☆ | 29 | 18 | Boxes |
| ssd_mobilenet_v1_0.75_depth_quantized_coco ☆ | 29 | 16 | Boxes |
| ssd_mobilenet_v1_ppn_coco ☆ | 26 | 20 | Boxes |
| ssd_mobilenet_v1_fpn_coco ☆ | 56 | 32 | Boxes |
| ssd_resnet_50_fpn_coco ☆ | 76 | 35 | Boxes |
| ssd_mobilenet_v2_coco | 31 | 22 | Boxes |
| ssd_mobilenet_v2_quantized_coco | 29 | 22 | Boxes |
| ssdlite_mobilenet_v2_coco | 27 | 22 | Boxes |
| ssd_inception_v2_coco | 42 | 24 | Boxes |
| faster_rcnn_inception_v2_coco | 58 | 28 | Boxes |
| faster_rcnn_resnet50_coco | 89 | 30 | Boxes |
| faster_rcnn_resnet50_lowproposals_coco | 64 | | Boxes |
| rfcn_resnet101_coco | 92 | 30 | Boxes |
| faster_rcnn_resnet101_coco | 106 | 32 | Boxes |
| faster_rcnn_resnet101_lowproposals_coco | 82 | | Boxes |
| faster_rcnn_inception_resnet_v2_atrous_coco | 620 | 37 | Boxes |
| faster_rcnn_inception_resnet_v2_atrous_lowproposals_coco | 241 | | Boxes |
| faster_rcnn_nas | 1833 | 43 | Boxes |
| faster_rcnn_nas_lowproposals_coco | 540 | | Boxes |
| mask_rcnn_inception_resnet_v2_atrous_coco | 771 | 36 | Masks |
| mask_rcnn_inception_v2_coco | 79 | 25 | Masks |
| mask_rcnn_resnet101_atrous_coco | 470 | 33 | Masks |
| mask_rcnn_resnet50_atrous_coco | 343 | 29 | Masks |

**Şekil 2.15.** Eğitilmiş modeller [13]

Şekil 2,15'te modeller arasından projeye göre en uygun model seçilip eğitimi gerçekleştirilebilir. Bu modelleri birbirinden ayıran özellikler nesne tanıma oranları ve eğitim süreleridir.

# 3. MODELLERİN EĞİTİMİ VE ÇALIŞMASI

Tensorflow bünyesinde hazır eğitimli modeller bulunmaktadır, Proje kapsamında hazır veri seti bulunmadığı için projede kullanılan veri setini oluşturmaktayız. Veri setini oluşturan toplam dört yüz adet resim mevcuttur. Geliştirilen model beş adet bankayı tanıması üzere eğitilmektedir.

## 3.1 Modellerin İçeriği ve Eğitim Aşaması

Projenin kapsamında üç farklı model eğitilmiştir. Bu üç farklı modelden ilk modelin içeriği sadece bankaların logolarını tanıyan modeldir. İkinci modelin içeriği ise sadece bankaların ismini, text kısmını tanıyan modeldir. Üçüncü model ise bankaların hem logo hem isim olarak tanıyan modeldir. Bu üç farklı modelin de eğitim kısımlarındaki aşamalar aynıdır farkları veri setlerinin farklı olmasıdır.

### 3.1.1 Model 1 İçeriği

Modellerimize tanıttığımız bankalar Ziraat Bankası, Garanti bankası, Kuveyt Türk bankası, Yapı kredi bankası ve Albaraka bankasıdır. İlk model sadece bu bankaların logolarını tanıyan modeldir.


**Şekil 3.1.** Albaraka bankasına ait logo

Şekil 3.1 de sadece logonun olduğu resimde %98 oranında doğru tanıma sağlanmaktadır.


**Şekil 3.2.** Kuveyt Bank

Şekil 3.2 de hem banka yazısı hem logo görünmekte, ancak eğitim sadece logo için olduğundan logoyu % 99 oranında doğru tanımaktadır.

### 3.1.2 Model 2 İçeriği

Bu modelin eğitimi sadece bankaların ismini tanımak için yapılmıştır. Şekil 3.3'teki görüntüde bankaların isimleri üzerinde tanıma yapılmaktadır.


**Şekil 3.3.** Model 2 de kullanılan görüntü

Şekilde görülen bankalar yüksek doğruluk oranı ile belirlenmektedir. Model 1 ve Model 2 tam anlamıyla istenilen sonuçları vermediği için üçüncü bir modele ihtiyaç duyulmuştur. Model 1 ve Model 2'nin bizlerin çekmiş olduğu videolardaki bazı hatalı tanımlamalarının sebebi veri setindeki resimlerin tam anlamıyla model eğitimine uygun olmamasından kaynaklanmaktadır. Ancak model 3'ün eğitilen modeller arasında en iyi çalışan model olduğu gözlemlenmiştir.

### 3.1.3 Model 3 İçeriği

Bu model tanımlanan bankaların hem logo hem isim, text kısımlarını birlikte tanıyan modeldir. Şekil 3.4 te görüldüğü gibi banka isim ve logosunu tanımaktadır. Aranılan nesneleri kare içine alarak doğruluk oranını göstermektedir.


**Şekil 3.4.** Garanti bankasına ait görüntü

Bu modelde hem logo hem banka isminin yüksek doğruluk oranlarında tanındığı görülmektedir. Modellerin eğitim aşaması ve veri setinin içeriği bölüm 3.2 de anlatılmaktadır.

## 3.2 Modellerin Eğitim Aşaması

Model eğitimine başlamadan önce gerekli ayarlamaların yapılması gerekmektedir. Bu ayarlamalara Tensorflow Object Detection kurulumu ve Anaconda Virtual Environment oluşturarak başlanması gerekmektedir. Bu ayarlamaları yaptıktan sonra model zoo dan bu proje için kullandığımız hazır eğitilmiş model olan Faster – RCNN-inception-V2-COCO'nun indirilmesi gerekmektedir [16].

Anaconda virtual environment için bazı pip installar gerekmektedir. Pip install'ları komut penceresinde yapmak gerekmektedir.

```
C:\> conda create -n tensorflow1 pip

C:\> activate tensorflow1

(tensorflow1) C:\> pip install --ignore-installed --upgrade tensorflow-gpu
(tensorflow1) C:\> conda install -c anaconda protobuf
(tensorflow1) C:\> pip install pillow
(tensorflow1) C:\> pip install lxml
(tensorflow1) C:\> pip install jupyter
(tensorflow1) C:\> pip install matplotlib
(tensorflow1) C:\> pip install pandas
(tensorflow1) C:\> pip install opencv-python
```

**Şekil 3.5.** Eğitim için gerekli kurulumlar[13]

Bu pip installları yaptıktan sonra PythonPath ayarlaması ve Protobuf derlemesi yapılmaktadır.

Modeller'in eğitimi için hazırlanmış olan veri setinin etiketlenme işlemi yapılmalıdır. Bu etiketleme işlemi veri setindeki her bir resim için gerçekleştirilmelidir. Bu işlem üç ayrı model için ayrı ayrı gerçekleşmiştir. Veri setlerinde yaklaşık dört yüz adet resim mevcuttur. Resimlerin boyutlarının küçük olması eğitim süresini etkilemektedir. Üç ayrı model için farklı resim boyutları kullanılmıştır.

Etiketleme işlemi Labellmg ile yapılmıştır. Her etiketlenen resim için xml dosyası oluşmaktadır. Bu xml dosyalarının %20 sinin object detection içindeki test klasörüne, %80'inin ise train klasörüne atılması gerekmektedir.

Eğitim verisi oluşturmak için etiketlenen resimlerin TFRecord'unu oluşturmak gerekmektedir. Etiket bilgilerini barındıran xml dosyalarını cvs dosyasına çevrilmesi gerekmektedir. Bunu gerçekleştirebilmek için komut penceresine "(tensorflow1) C:\tensorflow1\reseacrh\object_detection > python xml_to_cvs.py" komutunun yazılması gerekmektedir. Bu komut sonucunda iki adet cvs dosyası oluşmaktadır.

Modelin hangi id'de hangi sınıfta olduğunu belirtmesi için label map oluşturulmalıdır. Object detection training klasörüne yeni bir dosya oluşturulması gerekmektedir. Oluşturulan dosyanın ismine "labelmap.pbtxt" yazılmalıdır. Bu proje için label map şekil 3.6 da gösterilmektedir.

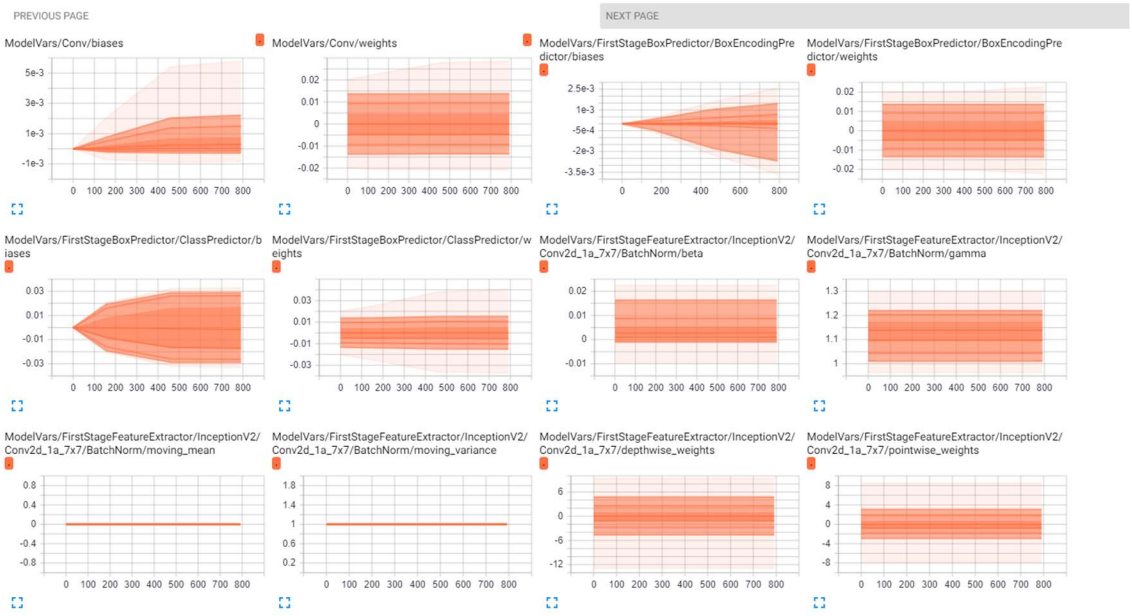
```
item {
  id: 1
  name: 'ziraatbank'
}

item {
  id: 2
  name: 'kuveytbank'
}

item {
  id: 3
  name: 'garantibank'
}

item {
  id: 4
  name: 'albarakabank'
}

item {
  id: 5
  name: 'yapikredibank'
}
```

**Şekil 3.6.** Label map [13]

Eğitimi gerçekleştirmek için "python train.py --logtostderr --train_dir=training/ --pipeline_config_path=training/faster _ rcnn _ inception _ v2 _ pets.config " komutunu komut penceresine yazdığımız takdirde eğitim başlamış olacaktır. Üç farklı model için de bu aşamalar yapılmaktadır. Her modelde farklı olan kısım veri setinin etiketlenme kısmıdır. Modellerin eğitim adımları ortalama altmış bin civarındadır. Her modelin eğitim süresi de farklı olamktdır, bunun sebebi resim boyutlarının her modelde farklı olmasıdır. Resim boyutları küçük olan model daha hızlı eğitilmiştir.

### 3.2.1 Eğitime Başlarken Tensorboard Çıktıları

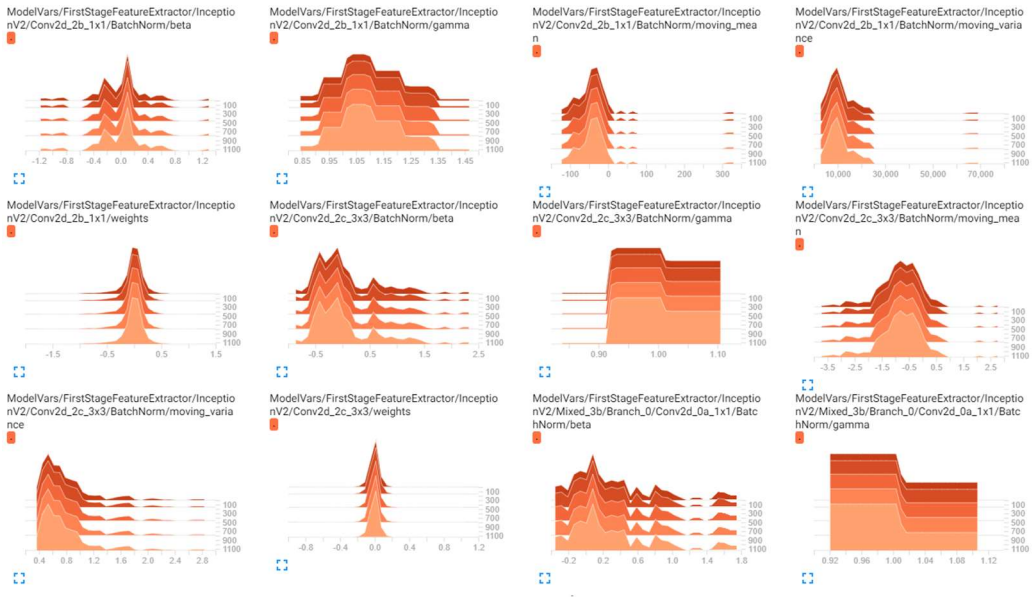
Tensorflow, modelleri eğitirken model'in veri setinden kendini nasıl eğittiğini grafikler şeklinde paylaşmıştır. Bu paylaşımları tensorboard yoluyla yapmaktadır. Bundan sonraki şekiller eğitimden önce model in tanıma oranlarını göstermektedir. Bu oranlar tüm modeller için ortalama aynıdır [16].




**Şekil 3.7.** Model 2 eğitim başlangıcı grafikleri 1


**Şekil 3.8.** Model 2 eğitim başlangıcı grafikleri 2

Şekil 3.7 ve şekil 3.8 de veri setinin eğitime başlarken, modelin kendini nasıl eğittiğini şekillerdeki grafiklerde gösterilmektedir. Bu grafikler eğitim adımları arttıkça değişmektedir. Her grafik farklı oranları göstermektedir.

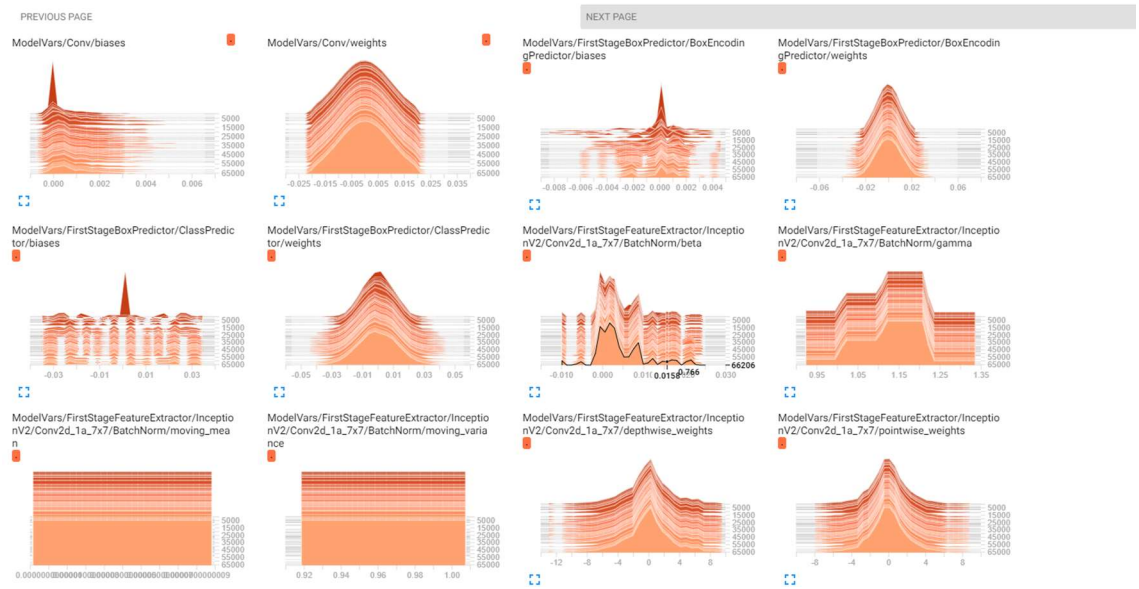



**Şekil 3.9.** Model 3 eğitim başlangıcı grafik

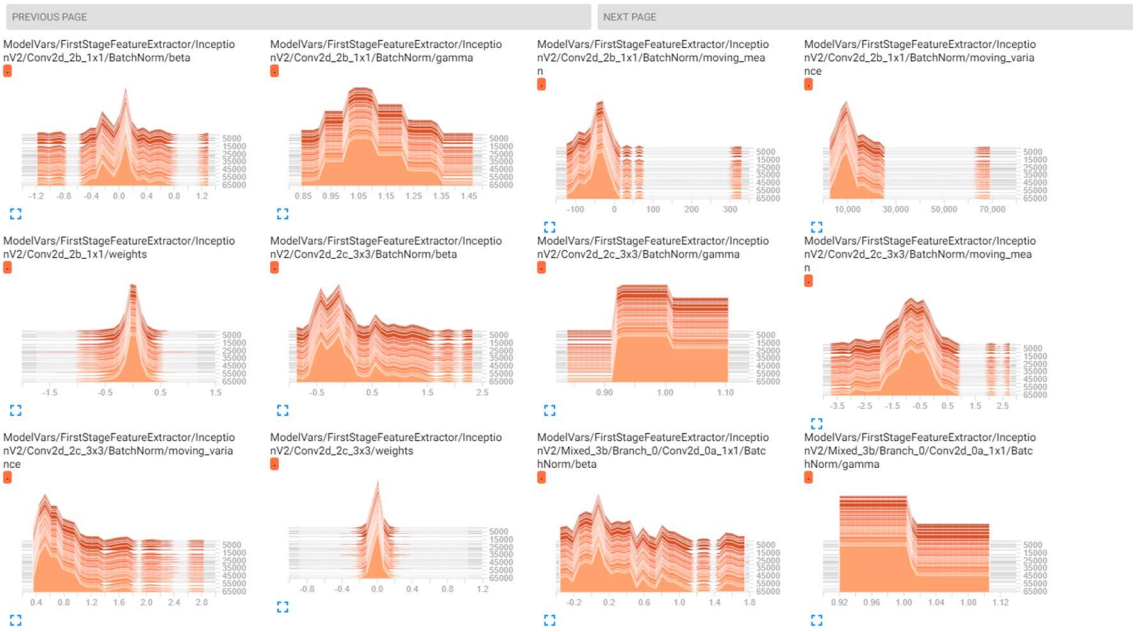
Şekillerde de görüldüğü gibi modellerin verileri hangi oranda tanımaya başladığı görülmektedir.

### 3.2.2 Eğitimden Sonra Tensorboard Çıktıları

Eğitimler ortalama altmış bin adım yürütülmüştür. Altmış bin adımda eğitimlerin durdurulmasının sebebi loss grafiklerinin durağan hale gelmesinden kaynaklanmaktadır.

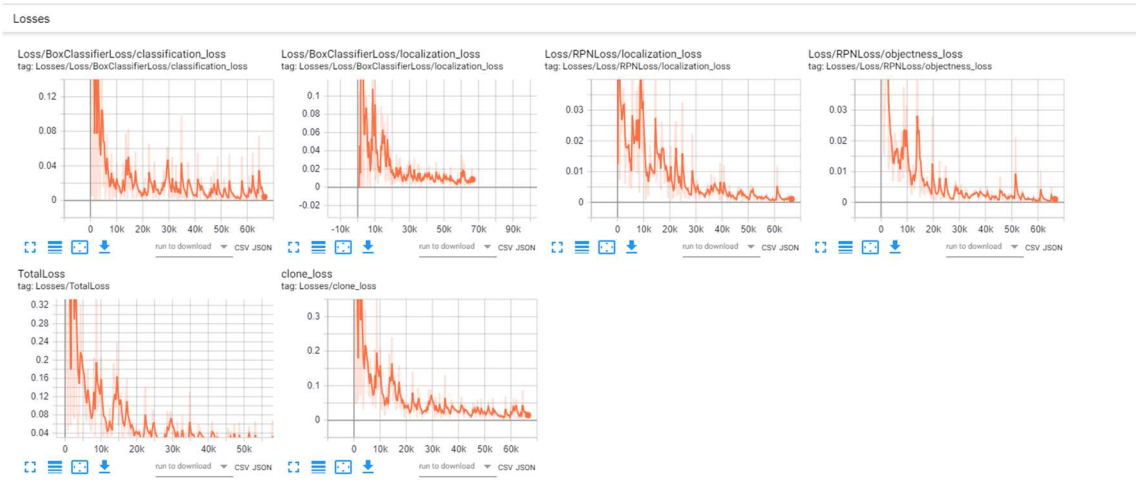



**Şekil 3.10.** Model 2 eğitim sonu grafikleri




**Şekil 3.11.** Model 3 eğitim sonu grafikleri

Şekil 3.10 ve şekil 3.11 de eğitim sonunda modellerin kendilerini kaç adımda eğittiklerini ve grafikleri kaç adımda doldurdukları görülmektedir. Grafikler deki sıklık arttıkça eğitimin sona yaklaştığı bilinmektedir. Bu grafiklerde eğitim altmış bin civarı devam ettirilmiştir.




**Şekil 3.12.** Model 2 loss grafikleri

Şekil 3.12 de model 2 için eğitimin kayıp değerleri grafiklerde verilmektedir. Bu değerlerden yola çıkarak eğitimi sonlandırma aşamasına geçilmiştir. Grafiklerde de görüldüğü üzere kayıp değerleri eğitim adımları ilerledikçe düşmektedir ve sabitlenmeye kayın bir değerde eğitime son verilmiştir.

### 3.2.3 Eğitim Sırasında Karşılaşılan Sorunlar Ve Çözümleri

Tez kapsamında üç farklı içeriğe sahip modeller eğitilmiştir. Her modelin eğitimi farklı sürelerde bitmiştir. Bunun sebebi modelin eğitimi için kullanılan veri setindeki resimlerin boyutlarının büyük olmasıdır. Bu üç farklı model için farklı boyutlarda veri setleri kullanılmıştır. İlk model için veri setindeki resimlerin boyutları 200 ile 300 kB arasında idi ve eğitim yaklaşık dokuz saat sürmüştür. Daha sonraki model eğitimlerinde veri setindeki resimlerin boyutlarının 50 ile 80 kB arasına düşürülmesi eğitim süresini beş ile altı saat arasına düşürmüştür.

Modellerin eğitimi sırasında gerekli yüklemelerde hatalarla karşılaşılmıştır. Bu hataların nedenleri genel anlamda gerekli yüklemeleri göz ardı etmek ve yanlış yapmak sebebiyle gerçekleşmiştir. Hataların çözümleri araştırılarak sorunlar giderilmiştir. Bu sorunların çözümleri projenin mantığının anlanmasında ve pratik kazanmada yardımcı olmuştur.

Eğitimler sonucunda modeller images, video, webcam için istenilen nesnenin tanınması konusunda çalışır duruma getirilmiştir. (görüntüden tanıma videodan tanıma gerçek zamanlı webcamden tanıma gibi bir şey yazın buraya) Eğitilen modeller içerisinde üçüncü model verimliliği en yüksek olan modeldir (şekil..'de de görüldüğü üzere en yük sek model olduğu belirlenmiştir tarzı bir cümle olsun bu). Bunun sebebi ise modelin hem banka logosuna göre hem de banka ismine göre eğitilmiş olmasıdır. Diğer modellerde bazı hatalı sonuçlar bulması ise veri setinde kullanılan resimlerin eğitime tam anlamıyla uygun (teorik bir açıklama olsun ne demek tam anlamıyla uygun olmaması ??) olmamasından kaynaklanmaktadır. Çevre faktörlerinin veri seti üzerindeki etkisi, yakınlık uzaklık derecesi, ışık açısı gibi resimleri etkileyen nedenler veri setlerinin model eğitimine uygun olmayacağı anlamına gelir. Model eğitimi için veri setlerinde kullanılan resimlerin düzgün ve farklı açılardan çekilmiş olması gerekmektedir. Proje kapsamında her model için yaklaşık dört yüz adet resim çekilmiştir.

# 4.SONUÇLAR

Bu tez çalışmasında hazır eğitilmiş bir modeli, oluşturulan veri seti ile istenilen nesnenin tanınması amacıyla tekrar eğitime sokulmuş ve nesne tanıması yapılmıştır. Proje kapsamında istenilen herhangi bir nesnenin bilgisayarlı görü ortamında tanınması sağlanabilmektedir.

Oluşturulan veri setindeki resimlerin model eğitimine uygun olması ve boyutlarının eğitim süresini etkilediği saptanmıştır. Proje kapsamında eğitilen üç farklı modelin % 80'lik doğruluk oranı ile çalıştığı gözlemlenmiştir. Bu oranı arttırmak için veri setinin eğitime daha uygun hale getirilmesi gerektiği (yapılan deneysel çalışmalar neticesinde …. Şeklinde bir cümle olsun).

Veri setinin eğitimini daha iyi bir hale getirebilmek için çekilen resimlerin hangi faktörlerden etkilendiğinin bilinmesi gerekmektedir. Işık açılarına, çevresel faktörlere, resim çekmek için kullanılan makinaya dikkat edildiği takdirde model eğitimi için daha iyi sonuç veren veri setleri elde edilebilir. Bu elde edilen veri setleri ile istenilen nesnenin tanınması konusunda daha iyi sonuçlar alınacağı belirlenmiştir.

Yapılan araştırmalarda nesne tanıma uygulamaları geliştirmek için eğitilmiş birçok model keşfedilmiştir. Yapılmak istenilen projenin yapısına göre ve veri setinin içeriğine göre istenilen model seçilebilmektedir. Proje kapsamında Faster R-CNN modeli eğitilmiştir. Ve nesne tanıma için gerekli ihtiyaçları karşıladığı görülmektedir. Veri setinin doğru planlandığı bir modelde istenilen nesnenin tanınması konusunda verimli şekilde çalıştığı gözlemlenmiştir.

## 5.KAYNAKLAR


[1]. http://mesutpiskin.com/blog/nesne-tespiti-ve-nesne-tanima.html

[2]. M. Ece GÜRBÜZ , Ali GANGAL "Döndürülmüş Kayan Pencereler Kullanarak İyleştirilmiş Hibrid Nesne Tespit Yöntemi" Eleco 2014 Elektrik – Elektronik – Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu, 27 – 29 Kasım 2014, Bursa.

[3]. http://ibrahimdelibasoglu.blogspot.com/2017/10/python-selective-search-segmentation.html

[4]. Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik "Rich feature hierarchies for accurate object detection and semantic segmentation " UC Berkeley

[5].Kaiming He , Georgia Gkioxari ,Piotr Doll´ar , Ross Girshick " Mask R-CNN"

[6]. Huaizu Jiang , Erik Learned-Miller "Face Detection with the Faster R-CNN" University of Massachusetts Amherst
Amherst MA 01003

[7]. Yongqiang Cao · Yang Chen · Deepak Khosla "Spiking Deep Convolutional Neural Networks for
Energy-Efficient Object Recognition"

[8]. Atakan KÖREZ, Necaattin BARIŞÇI "İnsansız Hava Aracı (İHA) Görüntülerindeki Yayaların Faster R-CNN Algoritması ile OtomatikTespiti" Bilgisayar Mühendisliği
Gazi Üniversitesi Ankara, Türkiye

[9]. Metehan Doyran , H. Levent Akın "Nesne Bulma ve Tanımada Çoklu Girisli Evri¸simsel Sinir Ağları" Bilgisayar Mühendisli˘gi Bölümü Bogaziçi Üniversitesi ˙Istanbul, Türkiye

[10]. Semih Orhan, Yalın Baştanlar "Parca Tabanlı Egitimin Evrisimli Yapay Sinir Agları ile Nesne Konumlandırma Uzerindeki Etkisi Effect of Patch Based Training on Object Localization with Convolutional Neural Networks"

[11]. https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11

[12]. https://emredurukn.github.io/2016/11/02/tensorflow-ile-derin-ogrenmeye-giris.html

[13]. https://github.com/tensorflow/models/tree/master/research/object_detection

[14]. https://www.udemy.com/bilgisayar-gorusu/learn/v4/t/lecture/9570298?start=1

[15].https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md

[16]. https://www.udemy.com/bilgisayar-gorusu/learn/v4/t/lecture/9544010?start=1

[17]. https://www.udemy.com/bilgisayar-gorusu/learn/v4/t/lecture/9608866?start=0

[18]. https://www.udemy.com/bilgisayar-gorusu/learn/v4/t/lecture/9608870?start=555

[19]. https://www.udemy.com/bilgisayar-gorusu/learn/v4/t/lecture/9608872?start=210

[20]. https://ctme.caltech.edu/project-management/deep-learning-caltech

[21]. Syed Mazhar Abbas, Dr. Shailendra Narayan Singh "Region-based Object Detection and Classification using Faster R-CNN" ASET, Amity University Uttar Pradesh, India

[22]. Liyan Yu, Xianqiao Chen, Sansan Zhou "Research of Image Main Objects Detection Algorithm Based on Deep Learning"School of Computer Science and Technology Wuhan University of Technology Wuhan, China

[23]. Koen E. A. van de Sande, Jasper R. R. Uijlings, Theo Gevers, Arnold W. M. Smeulders "Segmentation as Selective Search for Object Recognition" U niversity of Amsterdam Amsterdam, The Netherlands

[24]. Shih-Chung Hsu, Chung-Lin Huang, Cheng-Hung Chuang "Vehicle Detection using Simplified Fast R-CNN " *Dept. of Electrical Eng., National Tsing-Hua University, Hsin-Chu, Taiwan

## 6. ÖZGEÇMİŞ

**Enes Safa ERBAŞ**

**Kişisel Bilgiler**

**Adı Soyadı:** Enes Safa ERBAŞ

**Doğum yeri:** Kadıköy/İSTANBUL

**Doğum Tarihi:** 09/06/1996

**Askerlik Durumu:** Tescilli (31.12.2022)

**İletişim Bilgileri**

**Adres:** Aydıntepe mah. 100. Yıl cad. Arı sok. No:7 TUZLA/İSTANBUL

**Telefon:** 90 (539) 4811385

**E-Posta:** enessafaerbas@hotmail.com

**Eğitim Bilgileri**

**Lise:** Ömer Çam Anadolu İmam Hatip Lisesi (2010-2014)

**Lisans:** FIRAT ÜNİVERSİTESİ Mühendislik Fakültesi Bilgisayar Mühendisliği (2015-)

**Yabancı Dil**

**İngilizce:** Orta

**Arapça:** Temel

**Yetkinlikler**

**Bilgisayar:** MySQL, AutoCAD, C#, Java, Python , ISIS PROTEUS, Machine Learning, Data Mining, PHP, Eagle CAD, JavaScript, HTML/CSS , Arduino

**Ek Bilgiler**

**Ehliyet:** B Sınıfı

**Turgay HOPAL**

**Kişisel Bilgiler**

**Adı Soyadı:** Turgay HOPAL

**Doğum yeri:** Refahiye/ERZİNCAN

**Doğum Tarihi:** 01/08/1997

**Askerlik Durumu:** Tescilli (31.12.2022)

**İletişim Bilgileri**

**Adres:** Yeni doğan mah . Doğan Koç cad. REFAHİYE/ERZİNCAN

**Telefon:** 90 (553) 726 96 94

**E-Posta:** turgay24.th@gmail.com

**Eğitim Bilgileri**

**Lise:** Refahiye Anadolu Lisesi (2011-2015)

**Lisans:** FIRAT ÜNİVERSİTESİ Mühendislik Fakültesi Bilgisayar Mühendisliği (2015-)

**Yabancı Dil**

**İngilizce:** Orta

**Yetkinlikler**

**Bilgisayar:** AutoCAD, C/C++,Java, Android Programlama, Machine Learning, Data Mining, PHP, Eagle CAD, JavaScript, HTML/CSS, Python , Arduino , PIC Programlama

**Ek Bilgiler**

**Ehliyet:** B Sınıfı