İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

YOUTUBE VİDEOLARI ÜZERİNDE GÖRÜNTÜ SINIFLAMASI YAPILMASI

Hazırlayan : MURAT KAZANÇ 1306180073

Danışman : DR. ÖĞR. Ü. MUSTAFA DAĞTEKİN

HAZİRAN - 2020

ÖNSÖZ

Günümüzde yapay zeka; üretimde, üretilen ürünlerde ve insan hayatının neredeyse her alanında aktif olarak kullanılmaktadır. Her gün yapay zekanın kullanıldığı alanlar gelişmekte ve insanlara yeni imkanlar sağlamaktadır.

Youtube dünyanın en popüler video izleme platformudur. Politikaları gereği yüklenen videolara bir filtre uygulamaktadır. Ancak bu filtre videoları sadece genel hatları ile incelemektedir. Yapılan araştırmalar sonucunda Youtube'da sunulan videoların içerikleri hakkında sadece kategori gibi genel bir sınıflama olduğu görülmüştür. Bizim hedefimiz izlenen videoların gerçek zamanlı olarak incelenerek, takip edilmek istenen nesne veya durumların tespitinin yapılmasıdır.

Resim sınıflama ile resimler içerisinde evrişimli sinir ağlarından faydalanılarak bir model geliştirilmesi sonucu istenilen nesnelerin veya durumların tespit edilmesi amaçlanmaktadır. Geliştirilen model kullanılarak, tarayıcı üzerinde bir eklenti ve mobil uygulama yazılarak ikinci kişilerin izlenilen video içeriğinden haberdar olması, kayıt altına alınması veya denetim sağlanması hedeflenmektedir.

Bu çalışma boyunca gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Sayın Dr. Öğr. Ü. Mustafa DAĞTEKİN'e en içten dileklerimle teşekkür ediyorum.

Ayrıca tüm tez yazımı sürecinde bana destek olan aileme, motivasyonumun her zaman yüksek olmasını sağlayan arkadaşlarıma, mühendislik tamamlama sürecine birlikte başladığımız desteklerini hiç bir zaman esirgemeyen meslektaşlarıma en içten dileklerimle teşekkürlerimi sunuyorum.

Murat KAZANÇ 1309180073

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

SEMBOL LİSTESİ

h	: hipotez
W	: ağırlık
b	: bias
α	: öğrenme katsayısı
γ	: ölçekleme parametresi
β	: kaydırma parametresi
n	: eğitim örneği sayısı
i	: veri setindeki eğitim örneği
Θ_0	: başlangıç değer parametresi
Θ_1	: eğim parametresi
y_i	: eğitim örneği için doğruluk
$\hat{y}_i$	: eğitim örneği için tahmin

KISALTMA LİSTESİ

CPU	**:**	Merkezi İşlem Birimi
GPU	**:**	Grafik İşlem Birimi
RAM	**:**	Random Access Memory
PCA	**:**	Principal Component Analysis
LDA	**:**	Linear Discriminant Analysis
ANN	**:**	Artifical Neural Network
CNN	**:**	Convolutional Neural Network
RGB	**:**	Reg Green Blue
IDE	**:**	Integrated Development Environment
JDK	**:**	Java Development Kit
SDK	**:**	Software Development Kit
IoT	**:**	Internet of Things
JS	**:**	Java Script
HTML	**:**	Hyper Text Markup Language
CSS	**:**	Cascading Style Sheets
SQL	**:**	Structured Query Language
CORS	**:**	Cross-Origin Resource Sharing
URL	**:**	Uniform Resource Loader

ÖZET

YOUTUBE VİDEOLARI ÜZERİNDEN RESİM SINIFLAMASI YAPILMASI

Youtube tüm dünyada her yaştan insanın bilgisayar ve mobil cihazlardan sürekli erişim sağladığı video içerik sunan bir sağlayıcıdır. Genel olarak bakıldığında videoların içerikleri ile ilgili olaral çok kısıtlı birkaç konu dışında her hangi yasaklamaları bulunmamaktadır.

Youtube'un bu serbest ortamında özellikle çocukları ve gençleri zararlı içeriklerden korumak veya sorumlu kişilerin haberdar edilmesi için derin öğrenmeden faydalanılması düşünülmektedir.

Günümüzde web sitelerinde html kodları içerisinde video etiketleri ile videolar görüntülenmektedir. Youtube sitesi içerisinde de bu etiketten ve derin öğrenmeden faydalanılarak iş yeri veya kütüphane gibi ortamlar içinde bireylerin neler izledikleri incelenebilir.

Projede ilk olarak bir derin öğrenme metodu kullanılarak tütün mamülleri, zararlı içecek ve silah gibi nesnelerin sınıflamasını yapabilecek bir model Keras ile geliştirilmiştir.

Sonrasında bu model Tensorflow JS'e dönüştürülerek chrome tarayıcısı için bir eklenti geliştirilmiştir. Bu eklenti ile izlenen videolardan anlık resimler alınarak eğitilen model üzerinde sınıflamalar yapılmıştır.

Videolar üzerinde yapılan sınıflamalarda gerekli görülen sonuçlar Google'ın bulut hizmeti Firebase RealTimeDatabase'e kaydedilmiştir. Bu veri tabanı kullanılarak daha önceden kötü içerik tespiti yapılmış videoların en baştan engellenmesi sağlanmasıdır.

Son olarak model Tensorflow Lite'a dönüştürülerek Android mobil cihazlar için güvenli Youtube video izleme uygulaması geliştirilmiştir. Bu uygulamada yine izlenen videolardan anlık resimler alınarak model ile sınıflama yapılmıştır. Gerekli hallerde video veri tabanına eklenmiştir. Veri tabanına kayıtlı videolarin görüntülenmesi engellenmiştir.

Yapılan uygulamalar ile gerçekleştirilen deneyler sonucunda model eğitimi için kullanılan veri setinin yetersiz olduğu görülmüştür. Daha iyi bir veri seti ile hedefin gerçekleştirilebilir olduğu düşünülmektedir.

SUMMARY

IMAGE CLASSIFICATION ON YOUTUBE VIDEOS

Youtube is a provider of video content that allows people of all ages, all over the world, to access from computers and mobile devices. In general, there is no prohibition except for a very clear some issues in terms of video content.

In this free environment of Youtube, it is considered to benefit from deep learning especially in order to protect children and young people from harmful contents or to inform responsible persons.

Nowadays, videos are displayed in html codes and video tags on web sites. Not only on youtube, but also on other websites, this label and deep learning can be utilized to see what persons are watching in work environments or libraries.

In the project, a model that can classify objects such as tobacco products, harmful drinks and weapons using a deep learning method was first developed with Keras.

Later, this model was converted to Tensorflow JS and an extension was developed for the chrome browser. By taking snapshots from the videos watched with this extension, classifications were made on the model trained.

The results deemed necessary in the classifications made on videos were recorded in Google's cloud service Firebase RealTimeDatabase. It is to prevent the videos that have been detected bad content by using this database from the beginning.

Finally, by converting the model to Tensorflow Lite, a secure Youtube video viewing application has been developed for Android mobile devices. In this application, snapshots were taken from the videos watched and classification was made with the model. If necessary, the video has been added to the database. The viewing of videos recorded in the database is blocked.

As a result of the experiments carried out with the applications, the data set used for the model education was found to be insufficient. The target is considered to be achievable with a better data set.

1. GİRİŞ

Yapay zeka günümüzde hem akademik hem de sıradan insanlar için çok popüler bir konudur. Yapay zeka kavramı insanlar için, insan zekasını aşan uygulamalar olabileceği gibi bazıları için ise tüm veri işleme teknolojileri yapay zekadır. Peki yapay zeka tam olarak nedir? Yapay zekanın farklı yönlerini gösteren bazı örnekler şunlardır;

Yapay zekâ, bir bilgisayarın veya bilgisayar kontrolündeki bir robotun çeşitli faaliyetleri zeki canlılara benzer şekilde yerine getirme kabiliyetidir [1].

Otonom araçlar yapay zeka ile ilgili en popüler konulardan birisidir. Bir çok yapay zeka tekniğinin bir arada kullanılmasını gerektirmektedir: başlangıç ve bitiş noktasındaki en uygun rotayı bulmak, engelleri tanımlamak ve karmaşık bir çevrede kazaları önlemek gibi çok kompleks işlemleri kusursuz yapması beklenmektedir. Gelecekte bu teknoloji insansız hava araçları ve gemilere de uygulanması düşünülmektedir.

Gün boyu karşı karşıya geldiğimiz bir çok bilgi bize özel kişiselleştirilmiştir. Google'da yaptığımız aramalara göre karşımıza reklamlar gelmektedir. Sosyal medya hesaplarındaki görüntülediğimiz içeriklere göre karşımıza öneriler veya yine reklamlar gelmektedir. Online müzik veya film uygulamaları önceki içerik tercihlerimize bakarak bize daha önce tecrübe etmediğimiz içerikleri önermektedir [2].

Resim ve video üzerine işlemler hayatımızda sürekli kullanmaya başladığımız teknolojilerdir. Çoğu akıllı telefon artık yüz tanıma ile kilidini açmaktadır. E–Sınav uygulamalarında sınav sorumlulukları ve adayların kimlik tespitini yine yüz tanıma ile yapabilmektedir. Eğlence amaçlı olarak mobil cihazlar içerisindeki uygulamalar ile yaşlandırma, gençleştirme veya daha sıra dışı düzenlemeler yapılabilmektedir. Araç otoparklarında ve ücretli otoyol geçişlerinde araçların plakası tespit edilebilmektedir. Canlı MOBESE kameraları üzerinden kişi tespiti yapılabilmekte. Bir grup insanın sayısı, cinsiyetleri ve yaşları gibi parametreler ile analiz yapılabilmektedir.

Yapay zekayı tanımlamak istersek iki önemli nokta bulunmaktadır. Birincisi otonomi bir kişi tarafından sürekli rehberliğe ihtiyaç duymadan karmaşık ortamlarda görevleri yerine getirebilmesidir. İkinci olarak deneyimlerden öğrenerek performansını arttırabilmesidir. Sonuç olarak yapay zeka tek boyutlu olarak düşünülemez ve bilimin bir çok dalını içinde barındıran bir çatı olarak düşünülebilir.
Yapay zekanın alt kategorilerine geçmeden önce deneyimi temsil eden veriden bahsetmek gerekmektedir. Yapay zekanın eğtilmesi için elimizde veriler olmalıdır. Bu verinin boyutu çok büyük miktarlarda olmak zorundadır. Genellikle veriler ham şekilde olduğundan önce verilerin işlenmesi gerekmektedir.

Verilerin işlenmesini verinin düzenli bir hale getirilmesi, içerisinde hatalar varsa düzeltilmesi, gereksiz kısımların çıkarılması ve gereken dönüşümlerin yapılması ile kullanımımıza uygun hale getirilmesi olarak tanımlayabiliriz. Bu işlemleri gerçekleştirmek için Python programlama dilinin hazır kütüphaneleri bulunmaktadır. Kütüphanelerden bazıları şunlardır: Numpy, Pandas, Matplotlib ve Seaborn' dur. Bu kütüphanelerden Numpy ve Pandas veri analizinde kullanılmaktadır. Matplotlib ve Seaborn ise veri görselleştirme yapmaktadır. Bu kütüphaneler dışında veri madenciliği için kullanılan farklı kütüphaneler de mevcuttur ancak burada onlara değinilmeyecektir.

Yapay zekanın alt kategorilerinden biri makine öğrenmesidir. Kendi içinde bulunan önemli üç ana dalı şunlardır.

1. Danışmanlı Öğrenme (Supervised Learning): danışmanlı öğrenmede hedef eldeki eski verilere bakarak yeni veriler hakkında doğru bir tahminde bulunmaktır. Örneğin bir evin fiyatını tahmin etmek. Bir diğer hedef ise verilen bir verinin hangi sınıfa ait olduğunu bulmaktır. Örneğin bir hastanın kanser olması ya da olmaması durumu.

2. Danışmansız Öğrenme (Unsupervised Learning): Buradaki hedef verilen çok büyük miktardaki verilere bakarak verilerden bir anlam çıkartmaktır. Bu verilerin yoğunlaştığı bölgelere bakılarak yapılabilir. Yüz tanıma ve obje tespitinde kullanılır.

3. Destekli Öğrenme (Reinforcement Learning): Girişe verilen verilere karşılık çıkış bilgisi verilmemektedir. Ancak çıkan sonucun doğru veya yanlış olduğu sisteme gösterilmektedir. Örnek olarak bir köpeğe atılan topu geri getirmesi öğretilmek istenmektedir. Köpek doğru hareketi yaptıkça ödül verilmektedir. Ama köpek ne yapacağını kendisi seçer / keşfeder [3].

Yukarı da bahsedilen Makine öğrenmesi metodları problem çözümünde çok başarılı sonuçlar versede bazı problemlerin çözümünde yetersiz kalmaktadır. Bundan dolayı araştırmalarda insan beynindeki nöronlardan esinlenerek yapay sinir ağları geliştirilmiştir.

Yapay sinir ağları ile insan beynindeki nöronların matematiksel olarak modellenerek eğitilebilir, adaptif ve kendi kendine organize olabilir yapısı taklit edilmeye çalışılmıştır. Aynı insan beynindeki gibi gelen giriş verileri yapay sinir ağlarına iletilir ve sonraki katmandaki sinir hücrelerine iletilirek bir ağ yapısı oluşturularak çıkış elde edilir.

Resim sınıflama yapılmak isteniyorsa veri seti içerisindeki resimlerin içindekilerin belirgin özelliklerinin ortaya çıkarılması gerekir. Bunun için evrişimli sinir ağları geliştirilmiştir. Evrişimli sinir ağlarının görevi, özellik çıkarmaktır. Örnek olarak eğriler, kenarlar gibi özelliklerin belirginleşmesi bu ağlar ile sağlanabilir [4].

Yapay sinir ağlarının çok katmanlı ve karmaşık bir yapıda kullanılması ile oluşturulan modeller derin öğrenme olarak isimlendirilmektedir. Derin öğrenme kullanılarak bu projede bir resim sınıflama modeli eğitilecektir.

Eğitilen modelin kullanılacağı iki ürün ortaya konması hedeflenmektedir. Birincisi Chrome tarayıcısı eklentisidir. İkincisi Android mobil uygulamasıdır.

Tarayıcı eklentileri görüntülenen web sitenin içerisine normalde sitenin içeriğinde olmayan ekstra özellikler eklemek için kullanılabileceği gibi tarayıcıda bulunmaya özellikler eklemek için de kullanılmaktadır. Örneğin tarayıcılar için web sitelerindeki reklamları engeleyen eklentiler yoğun bir şekilde kullanılmaktadır. Bu projede normalde

görüntülenen sitenin içeriğinde olmayan ekstra javascript kodları eklenerek işlemler gerçekleştirilecektir. Tarayıcı eklentisinin çalışma mantığı html sayfa içerisinde ki video etiketini bularak oynatılan video içerisindeki görüntülerde sınıfların bulunmasıdır. Yapılan tespitler bir bulut veri tabanında depolanarak bir katalog oluşturulacaktır.

Mobil uygulamada eğitilen modelin tarayıcı eklentisine benzer şekilde çalıştırılması hedeflenmektedir.

Tartışma ve sonuç bölümünde farklı ağ yapıları ile eğitilen modellerin karşılaştırılması ve ürünlerin performans değerlendirilmeleri yer alacaktır.

2. GENEL KISIMLAR

2.1. VERİ SETİ OLUŞTURULMASI

Bir modelin beklenilen işi gerçekleştirebilmesi için çok fazla veriye ihtiyaç vardır. Çok miktardaki verinin işlenip sonuçlar çıkarmaya hazır hale getirilmesi veri madenciliği adında ayrı bir bilgisayar bilimidir.

Veri setinin elde edilmesi aşamasında ilk olarak Kaggle web sitesindeki hazır datasetler kullanılmıştır. Gerektiği durumda bu datasetler içerisinde düzenlemeler yapılacaktır. Kaggle dışında farklı kaynaklarda değerlendirilmiştir.

Bu projede modelin eğitilmesi için Keras Framework kullanılacaktır. Keras verinin hazırlanması için bir çok hazır metod sunmaktadır. Ancak yine de makine öğrenmesinde çokça kullanılan bazı hazır veri işleme kütüphanelerinin bilinmesi gerekmektedir.

2.1.1 Numpy Kütüphanesi

Numpy bilimsel hesaplama işlemlerini kolaylaştırmak için kullanılan bir Python matematik kütüphanesidir. Bu kütüphane ile çok az kod yazarak özellikle diziler üzerinde bir çok işlem gerçekleştirilebilmektedir. Bazı önemli metodlar şu şekildedir.

- np.array([1,2,3,4]) →dizi oluşturma. Bu metodun haricinde rastgele sayılar ile dizi oluşturan veya belli bir aralıkta belli adımla dizi oluşturan alternatifleri de mevcuttur.
- np.zeros((3,4)) →0' lardan oluşan 3x4' lük bir dizi oluşturur.
- np.ones((5,2)) →1' lerden oluşan 5x2' lük bir dizi oluşturur.
- np.eye((4)) →4x4' lük birim matris oluşturur. Makine öğrenmesinde çıkışlar verisi için zeros, ones ve eye metodları kullanılmaktadir.
- np.reshape(6,5) →30 elemanı olan tek boyutlu diziyi 6x5'lik diziye dönüştürür. Eğer boyut belirtirken değer -1 girilirse numpy kendisi uygun boyutu ayarlar.
- np.ravel() →çok boyutlu diziyi tekrar tek boyutlu yapar [5].

Numpy kütüphanesi içersinde diziden istatistiksel bilgiler almamızı sağlayan hazır metodlar vardır. Ayrıca dizi birleştirme ve matris çarpımı gibi matematiksel işlemler yapmamızı sağlayan hazır metodlar da mevcuttur.

2.1.2 Pandas Kütüphanesi

Pandas python programlama dili için yazılmış veri analizi sağlayan bir kütüphanedir. Bu kütüphane ile "cvs", "xls", "txt" ve hatta "html" dosyalarındaki veriler üzerinde çok az kod ile hızlı bir şekilde işlemler gerçekleştirilebilir.

Pandas da ilk yapı serilerdir. Seriler indeks değerleri olan tek boyutlu diziler olarak düşünülebilir. Ancak indeks sayısal değerler olmak zorunda değildir. Seri oluşturmak için python listeleri veya sözlükler ya da numpy dizileri kullanılabilir. Seri oluşturma söz dizimi şu şekildedir: `pandas.Series(data, index, dtype, copy)` burada data verilerin dizisini, index indeks değerlerin dizisini, dtype serinin tipini ve copy gerekirse verinin kopyalanması işlevini sağlar.

Pandas serileri üzerinde istatistiksel veri analizlerini hızla yapmamızı sağlayan hazır metodlar mevcuttur. `head()` metodu verinin ilk 5 satırını döndürürek veriye bir göz atmamızı sağlar. Benzer şekilde `tail()` metodu verinin son 5 satırını döndürür.

Dataframe yapısı satır ve sütunlardan oluşan çok boyutlu diziler olarak düşünülebilir. Bu tarz veriler genellikle harici bir dosyadan okunur. Bunu sağlayan metod

`pandas.read_csv('dosya yolu')`'dur. Dataframeler için bazı önemli fonksiyonlar şöyledir:

- shape → dataframe satır sütun sayısını verir.
- info → index sayısı, sütunlar ve satırlarda boş değerler var mı bilgilerini gösterir.
- value_counts() → hangi değerden kaç tane olduğunu verir.
- dropna() →NaN yani boş değer olan satırı siler. Ya da sütun adı belirtilerek sadece bir sütuna bakması sağlanabilir.
- fillna() → tüm NaN değerlere 1 değerini girer.
- isin() → sütunun belirtilen değeri içermesi durumuna göre bir boolen değer dizisi döndürür. Benzer şekilde çalışan isnull() ve notnull() komutları kullanılarak veri üzerinde filtreleme işlemleri gerçekleştirilir.
- unique() →sütunda tekrarsız bir şekilde kayıtların listesini verir. nunique() ise kaç tane benzersiz değer olduğunu verir.
- loc["Ali"]→ index değeri yazılarak satıra ulaşılır. Unutulmaması gereken metinsel ifadelerde index olabilmektedir.
- iloc[0,3] →index sütunu ne olursa olsun 0. indeksli ve 3. sütundaki veriyi döndürür.
- nsmallest(3, columns="yas") →yas sütunundaki en küçük değerli 3 satırı verir. nlargest() ise belirtilen parametrelere göre en büyük değerli satırları döndürür [6].

Dataframeler için sütun ekleme, çıkarma ve sütunları isimlendirme gibi işlemler yapılabilir. Bunlar dışında gruplama, sorgular oluşturma, yazdığımız bir metodun uygulanması ve dataframelerin birleştirimesi gibi bir çok işlem gerçekleştirilebilir.

2.1.3 Mathplotlib Kütüphanesi

Karmaşık ve dağınık haldeki verilerin daha anlamlı hale gelmesi için veri görselleştirme yapılır. Matplotlib kütüphanesi python için yazılmış en çok kullanılan veri görselleştirme kütüphanesidir. Temel olarak grafikler çizmemizi sağlar. Kütüphanenin import edildiği varsayılarak kullanımında bilinmesi gerekli temel komutlar şunlardır:

- plt.figure(figsize=(10,6)) →grafiğin büyüklüğünü belirler.
- plt.plot(dataX,dataY) →dizi olarak verilen x ve y eksen değerlerini belirler.

- plt.show() →grafiği ekrana çizer [7].

Yukarıdaki komutlar ile varsayılan olarak bir çizgi grafiği çizilir. Grafiğe başlık eklemek, x eksenine başlık eklemek, y eksenine başlık eklemek ve renk paletini değiştirmek yine yapılabilecek işlemlerdendir.

İstenilen durumlarda bir grafik çerçevesinde birden fazla çizgi çizdirilebileceği gibi birden fazla grafik çerçevesi bir seferde ekrana çizdirilebilir. Çizgi kalınlıkları, stili veya keşisim noktalarını belirten marker denilen işaretçiler için de bir çok özelleştirme mevcuttur.

Mathplotlib ile scatter (dağılım), histogram (çubuk), step (adım), piechart (pasta) ve daha bir çok farklı grafik çeşitini çizmek mümüküdür. İki farklı grafiğin combinasyonlarıda mevcuttur. Bu konu ile ilgili detaylı bilgi kütüphanenin kendi web sitesinde bulunmaktadır.

2.1.4 Seaborn Kütüphanesi

Mathplotlib altyapısını kullanan bir python kütüphanesidir. Temel olarak Seaborn kütüphanesi daha az kod ile daha güzel görünümlü grafikler oluşturmak için kullanılır. Bunu bir örnek ile açıklayalım: Pandas ile okuduğumuz 5 sütundan oluşan bir dataframemiz olsun. Biz dataframe içerisindeki bütün sütunların bir biri ile kıyaslanmasını sağlayan grafikler çizdirmek istiyor olalım. Çok sayıda kod yazmak yerine sadece seaborn'un `pairplot()` metodunu kullanmamız yeterli olacaktır [8].

Seaborn'un öne çıkan bazı grafik türleri şunlardır: heatmap, clustermap, boxplot ve violinplot. Bunun dışında seaborn geniş görselleştirme özellikleride sunmaktadır.

2.2. MACHINE LEARNING (MAKİNE ÖĞRENMESİ)

Makine öğrenmesi yapay zekanın bir alt dalı olan bir bilim dalıdır. Amaç matematiksel ve istatistiksel işlemler ile veriler üzerinde tahminler ve çıkarımlar yapmaktır. Çıktıların durumuna göre sayısal bir çıktı varsa regresyon eğer kategorik bir çıktı varsa sınıflama olarak adlandırılır. Benzer özelliklere sahip çıktıları ayırarak onları gruplamak

kümeleme olarak adlandırılır. Makine öğrenmesi öğrenme çeşitine göre 3 ana gruba ayrılarak incelenebilir.

2.2.1 Supervised Learning (Danışmanlı Öğrenme)

Bu yöntemte giriş verileri ile çıkış verileri arasında bir fonksiyon üretilir. Danışmanlı öğrenmeyi kullanan bir çok algoritma vardır.

2.2.1.1 Linear Regression (Doğrusal Regresyon)

Doğrusal regresyonu anlamanın en iyi yolu giriş verilerine karşılık gelen çıkış verilerini bir grafik ile göstermektir. Bu grafiğe bakılarak grafik üzerinde çizilecek bir doğru ile veri seti dışından gelen bir girişe karşılık çıkış verisini çizilen doğru üzerinden bulunarak çıkışın tahmin edilmesidir. Bu doğrunun bulunması makine öğrenmesinin konusudur. Öncelikle doğru için bir hipotez belirlenir. Örneğin;

$$h_\theta(x) = \theta_0 + \theta_1 x$$

hipotezi için parametreleri bulma işlemine maliyet fonksiyonu denir. Girişlerimize karşılık bulduğumuz çıkışların gerçek çıkış değerleri ile farkı bizim maliyet fonksiyonumuzdur. Hedef bu farkı minimize etmektir.

Maliyet (Cost ya da Loss) fonksiyonu $J(\Theta_1)$ olarak ifade edilir ve hesaplanmasında bir çok farklı metrik bulunmaktadır. Bunlardan bazıları:

$$\text{MSE(Mean Squared Error} - \text{ortalama hatanın karesi)} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

$$\text{RMSE(Root Mean Squared Error} - \text{ortalama hatanın karesinin kökü)} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

$$\text{MSE(Mean Absolute Error} - \text{ortalama mutlak değeri)} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n}$$

Amaç bu maliyet fonksiyonunu minimize etmektir. Böylece en doğru tahmin elde edilebilir.

Gradient Descent (Dereceli Azaltma) basitçe bir fonksiyonun minimumunu bulmamazı sağlar. Gradient Descent'in kendi içinde çeşitleri vardır. Batch Gradient Descent'te her bir adımda bütün eğitim örnekleri kullanılır. Ancak 5 milyon giriş verimiz varsa her adım için bütün hatanın hesaplanması bir sorundur. Stochastic Gradient Descent'te

(olasılıksal dereceli azaltma) her adımda rastgele bir giriş örneği kullanılır. Gürültü oluşsa da minimum değere çok daha hızlı erişim sağlar. Mini-Batch Gradient Descent'te ise toplam örnek içerisinden n adet örnek seçilerek uygulanır.

Gradient Descent hesaplanırken bir α öğrenme oranı kullanılır. Bu değer çok küçük seçilirse fonksiyonun minimumuna çok yavaş erişilir. Çok büyük seçilmesi durumunda hem osilasyon oluşacaktır hem de minimum değer ıskalanabilir.

$$\theta_1 := \theta_1 - \alpha \frac{\partial(J(\theta_1))}{\partial \theta_1}$$

Gradient Descent optimizasyon algoritmaları da vardır. Amaç önceden tanımlanmış bir işlem sayısında (epoch-devir) mümkün olan en iyi parametre değerlerini bulmaktır. Bunun için de bir çok optimizasyon algoritması bulunmaktadır. Bunlardan bazıları: Adagrad, Rmsprop, Adam, AdaMax ve AMSGrad' tır. Her optmizasyon algoritmasının kendine has avantajları ve dezavantajları bulunmaktadır. Veriler üzerinde deneyler yapılarak en uygun algoritma bulunabilir.

Gradient Descent ile ilgili bir bağımsız değişken olması durumunda yukarıda yazılan hipotez kullanılmaktadır. Ancak birden fazla bağımsız değişken olması durumunda çok değişkenli doğrusal regresyon olarak adlandırılır. Bu durumda sadece hipotezdeki parametre sayısı artmakta işlemler matris çarpımları şeklinde gerçekleştirilmektedir. Bazı durumlarda doğru çizmek yerine bir eğri çizilmesi çözüm için daha uygun olmaktadır. Bu durumda polinom regresyon olarak adlandırılır ve sadece hipotez denklemi değiştirilerek geri kalan işlemler aynen yapılır.

2.2.1.2 Logistic Regression (Lojistik Regresyon)

Hedef bağımlı değişken ile bir dizi bağımsız değişken arasında ki ilişkiyi açıklamak için en uygun modeli bulmaktır. Tahmin etmek istediğimiz değişkenimizin tipi kategoriktir. Örneğin bir kişinin kanser olup olmadığı ya da bir kişiye kredi verilip verilmeyeceği. Mantıksal olarak çıkış değerimiz 1'den büyük olamaz ve 0'dan küçük olamaz. Çıkış değeri için seçilen fonksiyonunda bu iki değer arasında sonuç üretmesi gerekmektedir. Sigmoid fonksiyon çıkış değeri ile ya tetikleme değeri olan 1'i ya da 0 verdiği için çıkış değerini belirlemede kullanılmaktadır.

2.2.1.3 KNN (K En Yakın Komşu)

Sınflandırma ve regresyon için kullanılan bir algoritmadır. En büyük farkı eğitim aşaması yoktur. Sınıfları belli olan bir örnek veri setine katılacak olan yeni verinin mevcut verilere göre uzaklığı hesaplanarak k sayıdaki yakın komşularına mesafesine bakılır. Uzaklığı en düşük olan komşularının olduğu sınıfa dahil edilir. Uzaklığı hesaplamak için bazı fonksiyonlar şunlardır: Euclidean, Minkowski ve Manhattan'dır [9].

2.2.1.4 Decision Tree (Karar Ağaçları)

Karar ağaçları hem sınıflamada hem de regresyonda kullanılan bir yöntemdir. Çok sayıda ve çok boyutlu bir veri kümesini karar kuralları uygulayarak daha küçük kümelere bölmek için kullanılan yapıdır.

Karar kuralları için farklı algoritmalar vardır. Sınıflandırma için en çok kullanılan algoritmalar Entropi ve Gini Sınıflandırma Hatasıdır. Regresyon için En Küçük Karaler yöntemi uygulanır.

Her karar kuralı ile bölünen veri kümesi homojenleşir. Entropi verilerimiz ile ilgili belirsizliğin ölçüsüdür. Belirsiziliğin artması demek entropinin yüksek olması anlamına gelir. Karar düğümleri kullanılarak entropi düşürülmektedir [10].

En çok kullanılan karar ağacı modelleme algoritmaları şunlardır: ID3,C4.5, C5.0 ve CART.

2.2.1.5 Support Vector Machines (Destek Vektör Makineleri)

Linear Support Vector Machines özellikle sınıflamada kullanılan bir metottur. Veri kümesinde iki veri grubu arasına bir sınır çizgisi çizilerek ayrılabilir. Bu sınırın çizileceği yer iki gruba da en uzak yer olmalıdır. Support Vector Machines ile bu işlem şu şekilde yapılır. Her gruba en yakın ve bir birine paralel iki sınır çizgisi çizilir bunlar destek doğrularıdır. Sonra bu çizgiler bir birine yaklaştırılarak üst üste bindikleri yer karar doğrusunu oluşturur. Support Vector Machines de sınıflandırmada (-1,+1) sınıf etiketleri ile kullanılır [11].

Nonlinear Support Vector Machines veri kümesinin doğrusal olarak ayrılamadığı durumlarda kernel trick denilen çözümler uygulanır. Bunlardan bazıları Polynomial Kernel ve Gaussian Radial Basis Function Kernel'dir.

2.2.1.6 Naïve Bayes

Bir sınıflandırma algoritmasıdır. Olasılık ilkelerine göre çalışmaktadır. Veri kümesindeki her özniteliğin birbirinden bağımsız olduğu ve öğrenilmek istenilen kavramın tüm bu özniteliklere bağımlı olduğu bir ağ oluşturularak sınıflama gerçekleştirilir. Mantık olarak olasılık değeri 0'dan küçük ve 1'den büyük olamayacağı için veri kümesi içerisinde 0 değerleri olmaması gerekmektedir. Bu duruma Zero Frequency denir ve veri kümesinin düzeltilmesi gerekir.

2.2.2 Unsupervised Learning (Danışmansız Öğrenme)

Danışmansız öğrenme algoritmalarında giriş verilerine bakılarak model eğitilir. Çıkış verileri eğitimde yoktur. Bu modellerde ne aradığımızı kesin bilmeme durumu vardır.

2.2.2.1 K-Means

Giriş verisi k sayıda kümeyi gruplamak için geliştirilmiştir. Algoritmanın çalışması şu şekildedir. Küme sayısı belirlenir ve veriler içinden noktalar küme merkezi olarak belirlenir. Her nokta değerlendirilir ve uygun kümeye dahil edilir sonrasında küme merkezi noktaları tekrar güncellenir. Yapılan güncellemeden sonra küme merkezlerinde bir önceki değerler ile bir fark yoksa işlem sonlandırılır.

Noktalar değerlendirilirken küme merkezine uzaklıkları hesaplamak için kullanılan yöntemler şunlardır: Euclid, Manhattan ve Minkowski uzaklıklarıdır [12].

2.2.2.2 Hierarchical Clustering

İki ana çeşiti bulunmaktadır.

Agglomerative (parçadan bütüne) Hierarchical Clustering'te önce tüm veriler bir kümedir. N eleman varsa N küme vardır. Sonra birbirine yakın olan kümeler birleştirilerek yeni bir küme oluşturulur. Sistem kararlı hale gelene kadar devam edilir.
Divisive Hierarchical Clustering tüm veri tek bir kümedir. Her adımda k adet küme kalana kadar işlemler tekrar edilir [13].

Mesafe hesaplamak için yollar şu şekildedir: Single Linkage (en kısa mesafe), Complate Linkage (en uzak mesafe) ve Average Linkage (ortalama mesafe hesaplama)'dır. Bu algoritmada sıklıkla kullanılan Dendrogram (öbek ağacı) ile benzer veri kümeleri arasındaki ilişkileri veya hiyerarşik kümelemeyi gösteren bir ağaç diyagramıdır.

2.2.2.3 PCA(Principal Component Analysis) ve LDA(Linear Discriminant Analysis)

PCA bir boyut indirgeme algoritmasıdır. Burada amaç birden fazla sütundan oluşan verinin içerisindeki gereksiz olan sütunların eğitimden çıkarılmasıdır. Makine öğrenmesinde yaygın olarak kullanılır ve hazır kütüphanelerde bu işlemi yapan metotlar mevcuttur. Gereksiz olan sütunların belirlenmesi verilerin kendisi incelenerek gerçekleştirilir. Boyut indirgeme işlemi hatanın biraz artmasına sebep olabilir. Ama çok büyük veriler ile çalışırken bize maliyet yönünden büyük avantaj sağlar [14].

LDA'da yine bir boyut indirgeme algoritmasıdır. PCA'dan farklı olarak çıkıştaki sınıflarında kullanılmasıdır. PCA ile verinin tamamını en iyi ifade eden algoritma bulunurken LDA ile sınıfları en iyi ifade eden algoritma bulunur. Başarımı PCA'ya göre daha yüksektir.

2.2.3 Reinforcement Learning (Takviyeli Öğrenme)

Hem danışmanlı hem de danışmansız öğrenme yöntemlerine benzer yanları vardır. Sisteme danışmansız öğrenmede olduğu gibi sadece giriş verileri verilirken sistem çıkışını doğru veya yanlış olduğu sisteme bildirilir. Yani sistem giriş verilerine göre ne çıkış vermesi gerektiğini bilmemektedir. Çıktı hesaplandıktan sonra danışman doğru çıktıların pekiştirilmesini sağlar.

Kullanım alanları arasında robotik vardır. Örneğin robot bir engele çarptığında negatif geri bildirim alarak sonraki çarpışmalardan kaçınmayı öğrenebilir. Başka bir örnek olarak dijital oyunlarda deneme yanılma yöntemi ile oyunda belli bir hareket ile ödül kazanılabileceği öğrenilebilir.

2.3. DEEP LEARNING (DERİN ÖĞRENME)

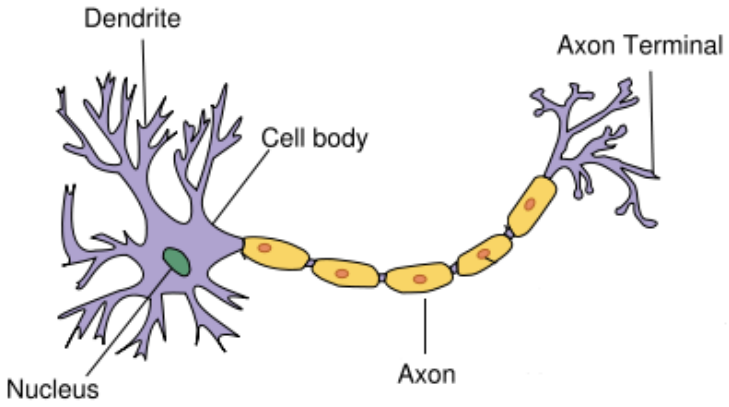
Deep learning ham veriden kademeli olarak daha yüksek seviyeli özellikler çıkarmak için çoklu katmanlar kullanan bir makine öğrenmesi alt sınıfıdır. Örneğin görüntü sınıflamada alt katmanlar kenarları tanımlarken yüksek katmanlar rakam, harfler veya yüzler gibi kavramları tanımlayabilir [15].

2.3.1 Artificial Neural Network (Yapay Sinir Ağları)

Yapay zeka yöntemlerinden biri olarak insan beynini modelleyerek taklit edilmeye çalışılması ile yapay sinir ağları ortaya çıkmıştır. Modellemede insan beyninde bulunan biyolojik sinir hücreleri (nöron) örnek alınmıştır. Sisteme sunulan örnekleri kullanarak problemi öğrenebilir ve sonrasında daha önce görmediği yeni bir durum karşısında çözüm üretebilirler yani adaptasyon özellikleri vardır. Genel olarak sınıflandırma, tahmin, yorumlama ve değerlendirme işlemlerini gerçekleştirebilirler.

2.3.1.1 Biyolojik Sinir Hücresi

Nöronlar bilgileri duyu organları vasıtası ile alırlar. Bilgi sonrasında alıcı nöronlar tarafından işlenip bir sonraki nörona aktarılmaktadır. Bu şekilde işlenmiş bilgi merkezi sinir sistemine oradan da üretilen sinyal bir tepkiye dönüştürülmektedir. Biyolojik nöron 4 kısımdan oluşmaktadır. Bunlar aşağıdaki şekilde görülmektedir.



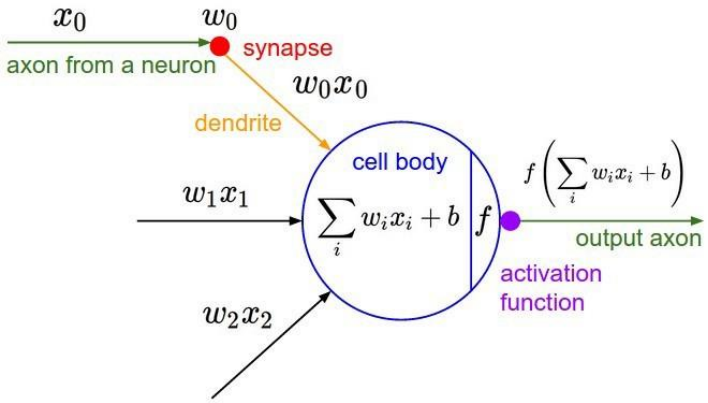
Şekil 1 Biyolojik Nöron [16]

- Dendrite: diğer nörondan iletilen sinyali hücre çekirdeğine iletmek ile görevlidir.
- Nucleus: Dendrite'den gelen sinyallerin toplandığı ve axona iletildiği bölümdür.
- Axon: nucleustan gelen toplanmış sinyalin ön işlemden geçirilerek axon terminale (synapse) iletilmesini sağlar.

Axon Terminal (Synapse): diğer nöron hücresi ile bağlantıyı sağlar.

2.3.1.1 Perceptron

Biyolojik sinir hücresini matematiksel modeli aşağıda ki gibidir.



Şekil 2 Biyolojik Sinir Hücresi Matematiksel Modeli [17]

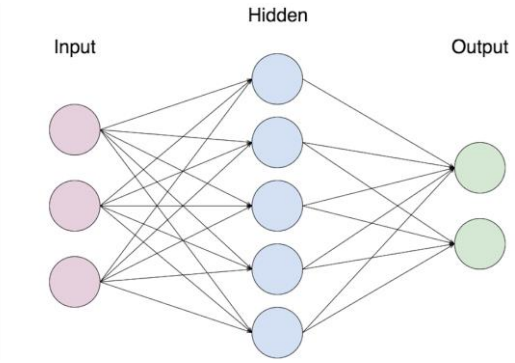
Perceptron kendisine gelen giriş sinyallerini şu şekilde işlemektedir. Şekil 2'de görüldüğü üzere her bir giriş için

$$y = Wx + b$$

şeklinde bir formül vardır. Burada:

- y: x bağımsız değişkenine bağımlı bir değer verir.
- x: bağımsız değişken girişidir.
- W: ağırlık parametresidir. O girişin modelde ki etkisini belirler.
- b: bias değeri gelen sinyalin eşik değerinin belirlenmesi için kullanılır.[17]

Perceptron yapay sinir ağının en küçük parçasıdır. Perceptronlardan oluşan katmanların aralarında birbirleri ile oluşturdukları bağlantılar ile ANN oluşmaktadır. ANN'de yapılan temel işlem modelin en iyi sonucu vereceği W ve b parametrelerini hesaplamaktır. Aşağıda çok katmanlı bir ANN örneği görülmektedir.



Şekil 3 Tek Gizli Katmanlı ANN Yapısı [17]

ANN modeli oluşturulurken birden fazla gizli katman olması mümkündür. Katman içindeki nöronların birbirleriyle ilişkileri yoktur. Ardışık katmanlardaki nöronların birbirlerine etkisi vardır. Katman sayısı ve katmandaki nöron sayısının artması ile her zaman iyi sonuç vermesi beklenmemelidir.

2.3.1.2 Toplama Fonksiyonu

Perceptrona her girişten gelen sinyale yukarıdaki formül uygulandıktan sonra bir "Toplama Fonksiyonu" uygulanmaktadır. Toplama fonksiyonunda kendi içinde çeşitleri bulunmaktadır.

- Toplam: bulunan y değerleri toplanarak elde edilir.
- Çarpım: bulunan y değerleri çarpılarak elde edilir.
- Maksimum: bulunan n adet y değeri içerisinden en büyüğü seçilerek elde edilir.
- Minimum: bulunan n adet y değeri içerisinden en küçüğü seçilerek elde edilir.
- Çoğunluk: bulunan n adet y değeri işaretlerine göre pozitif ve negatif olanların sayısı bulunur. Büyük olan sayı fonksiyon sonucu olur.
- Kümülatif toplam: daha önce bulunan toplama fonksiyonu ile yeni hesaplanan toplama fonksiyonu değeri toplanarak elde edilir [18].

2.3.1.3 Activation Function (Aktivasyon Fonksiyonu)

Perceptronun kendisine gelen sinyali işlemesindeki son aşama olarak toplama fonksiyonundan gelen değer Aktivasyon Fonksiyonu'na sokulurak nöron çıktısı elde edilir. Aktivasyon fonksiyonu kullanılmasının sebebi

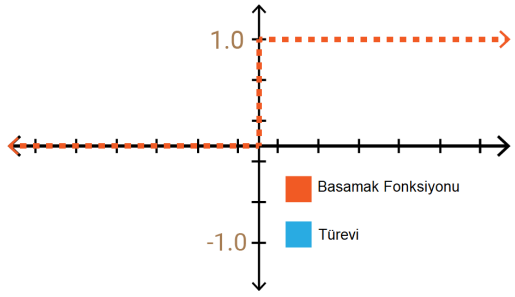
$$y = Wx + b$$

formülü gereği doğrusal bir fonksiyondur ve her nöron geçisinde bu değer katlanarak büyüyüp gidecektir. Oysa biz bu değerin 0 ile 1 yada -1 ile 1 arasında kalmasını tercih

ederiz. Bu durum çıktıların çok büyümesini engelleyecek ve işlemlerin kolaylaşmasını sağlayacaktır. Bir diğer nokta ise aktivasyon fonksiyonu olmaması halinde ANN sadece Linear Regression gibi davranacaktır. Oysa ANN' den beklenen evrensel fonksiyonların tamamına çözüm üretebilmesidir [19].

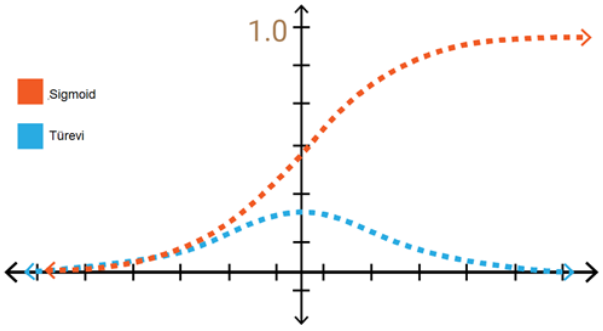
Çok çeşitli aktivasyon fonksiyonları bulunmaktadır. Her aktivasyon fonksiyonunun farklı bir performansı vardır. Ancak bir çözüm için çok verimli olan aktivasyon fonksiyonu başka bir çözüm için istenilen verimi veremeyebilmektedir. İstenildiği ve yeterli bilgiye sahip olunulduğu takdirde özel aktivasyon fonksiyonları yazılabilmektedir. Çok kullanılan bazı aktivasyon fonksiyonları şu şekildedir.

Step (Basamak) fonksiyon ikili değer alan bir fonksiyondur. İkili sınıflayıcı olarak kullanılabilir. Bu sebeple çıkış katmanlarında tercih edilir. ANN' de gizli katmanlarda kullanılamamasının sebebi türev değer olmamasıdır. Türev değerinin bizim için önemi ANN' de Back Propagation algoritmasında kullanılmasıdır.



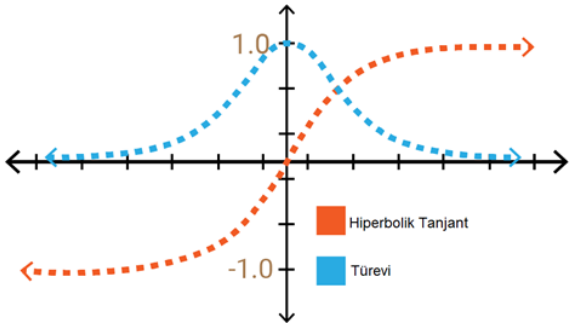
Şekil 4 Step Fonksiyonu [19]

Sigmoid fonksiyon çok kullanılan bir aktivasyon fonksiyonudur çünkü doğada çoğu problem doğrusal değildir. Fonksiyon 0 ile 1 arasında değer alabilir. Türevi de öğrenme değeri sağlayacak şekildedir. Ancak fonksiyonun uçlarına doğru x ekseninde ki değişikliklere çok az tekpi vermektedir.



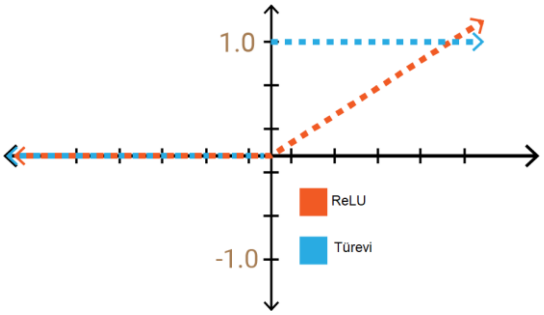
Şekil 5 Sigmoid Fonksiyonu [19]

Tanh yada Tangent Hyperbolic (tanjant) fonksiyonu sigmoid fonksiyona oldukça benzerdir. Farklı olarak -1 ile +1 arasında değerler alabilir. Ayrıca türev değeri aralığı daha yüksektir. Yine sigmoid fonksiyon ile aynı soruna sahiptir.



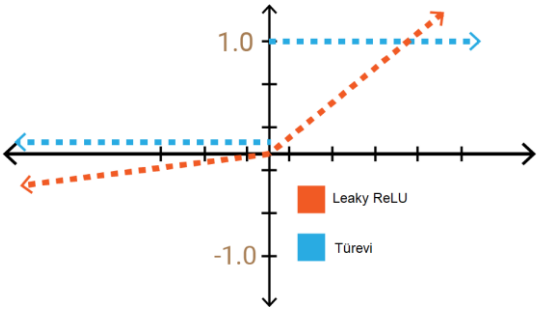
Şekil 6 Tanh Fonksiyonu [19]

ReLU (Rectified Linear Unit) Fonksiyonu x in negatif değerlerinde 0'dır ve x in pozitif değerlerinde +∞' a doğru gider. Bu bize x negatif olduğu tarafta 0 değerleri alarak ağın daha hızlı çalışmasını sağlayacaktır. Ancak negatif tarafta türev değeri 0 olduğu için öğrenmeyi etkileyecektir. İşlem yükünü azaltması ANN'de çokça tercih edilmesine sebep olmaktadır.



Şekil 7 ReLU Fonksiyonu [19]

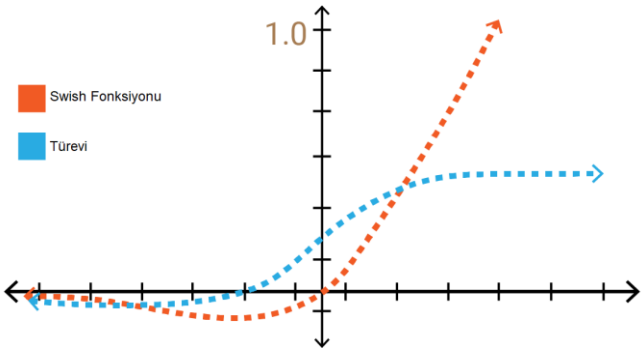
Leaky (Sızıntı) ReLU Fonksiyonu x in negatif olduğu bölgeleri için küçük bir türev değeri üreterek o bölgede ki öğrenmenin yok olmasını englleyen bir yapısı vardır.



Şekil 8 Leaky ReLU Fonksiyonu [19]

Softmax Fonksiyonu sigmoid fonksiyona benzer yapıdadır. Sınıflayıcı olarak çok iyi performas sergilemektedir. İkiden fazla sınıf olan durumlarda derin öğrenme modellerinin çıkış katmanında kullanılır. Bunun sebebi Sigmoid fonksiyon sadece 0 ile 1 olarak iki sınıfı temsil edebilirken Softmax fonksiyonu tüm sınıfların değerleri (olasılıkları) toplamı 1 olacak şekilde değer üretmesidir. Her sınıf için ayrı olasılıksal bir sonuç üretilmektedir.

Swish Fonksiyonu ReLu'dan farklı olarak x'in negatif bölgesinde de doğrusal olmayan bir türev değeri üretebilmesidir.



Şekil 9 Swish Fonksiyonu [19]

2.3.1.4 Back Propagation (Geri Yayılım)

Bir ANN modeli oluşturuduğunda başlangıç W (Ağılık) değerleri çoğunlukla rastgele seçilir. Giriş katmanı ve gizli katmanlardan geçerek gelen sinyal çıkış katmanından önce gizli katmanlara geri beslenerek W değeri güncellmesi yapılmasına Back Propagation denir. Bu ağlara örnek olarak Hopfield, SOM, Elman ve Jordan verilebilir.

2.3.1.5 Loss (Cost-Maliyet) Function

Makine Öğrenmesi Linear Regression konusu içerisinde Loss (Maliyet) fonksiyonu anlatılmıştır. ANN'ler içinde bir Loss fonksiyonu kullanılır. Yine Loss fonksiyonu optimize eden fonksiyonlar ANN' ler içinde geçerlidir.

2.3.2 Convolutional Neural Network (Evrişimli Sinir Ağları)

ANN bir önce ki bölümde detaylı olarak incelendi. ANN'lerin görevi veriden bir anlam çıkarmaktır. Bu bölümde incelenecek olan CNN ile verinin özelliklerinin ortaya

çıkarılması hedeflenmektedir. Örnek olarak eğriler, kenarlar gibi özelliklerin belirginleşmesi bu ağlar ile sağlanabilir. CNN genellikle resim sınıflama yaygın olarak kullanılır.

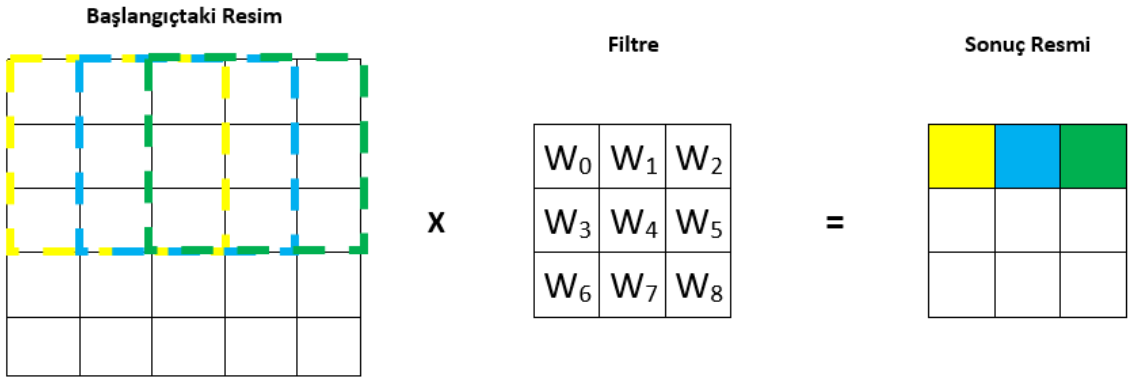
2.3.2.1 Bilgisayar Resmi Nasıl Görür

Dijital ortamdaki resimler pixel denilen noktaların birleşiminden oluşurlar. Yatay ve dikeyde ki pixellerin çarpım şeklinde gösterilmesine çözünürlük denilir. Her pixelin bir renk kodu vardır. Renk kodu olarak kullanılan birden fazla sistem mevcut olsa da burada RGB (Red Green Blue) kullanılacaktır. Üç ana renk kırmızı, yeşil ve mavi nin 0 ile 255 arasında ki değerlerinin karışımı ile gördüğümüz renk temsil edilir.

Bir resmin, modelin eğitiminde kullanılabilmesi için yapılması gereken işlem resimdeki her bir pixelin renk özelliklerini ve konumunu bir diziye aktarmaktır. Sonuç olarak eğer resmimizin çözünürüğü 180x180 ise elimizde boyutları (180,180,3) olan bir dizi olması gerekmektedir. Sonda ki 3 ile renk kodları (RGB) tutulacaktır.

2.3.2.2 Convolutional Filters/Kernels (Evrişim Filtreleri)

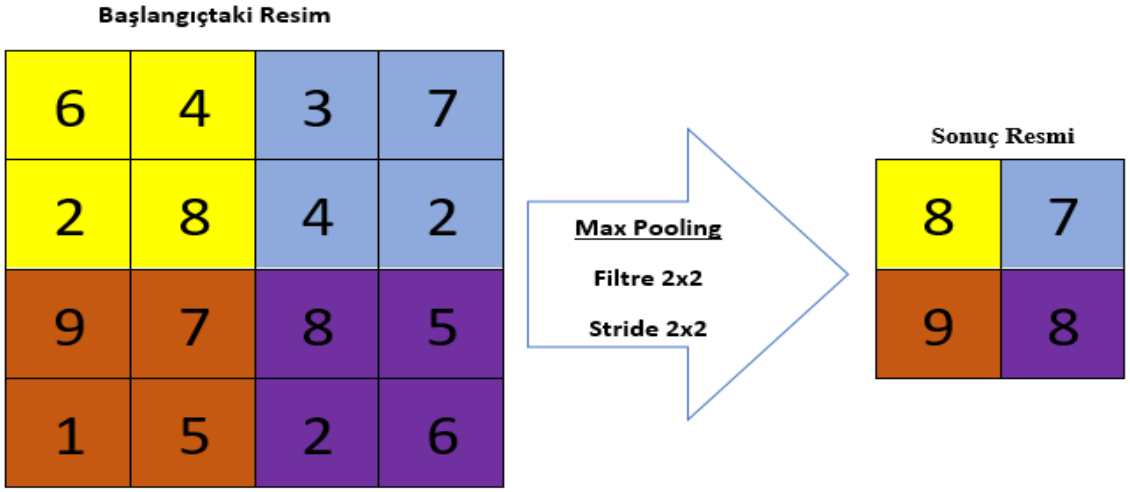
Filtreler ile resimde belli özelliklerin ortaya çıkarılması sağlanır. Örneğin 3x3' lük bir filtrenin özelliğine göre değerleri vardır. Bu filtre resim üzerinde her renk katmanı için ayrı ayrı tüm satırlarda kaydırılarak yapılan matris çarpımları sonucu ile görselin değiştirilmiş bir versiyonu elde edilir. Kaydırma işleminde 1 adım hareket edilebileceği gibi daha büyük adımlarda atılabilir bu parametreye "stride" denir. Bu versiyon ilk görselden daha küçük bir boyutta olacaktır. Böylece hem resimdeki bir özellik öne çıkarılmış olacak hem de boyut düştüğünden dolayı sonraki katmanlarda işlem yükü azalacaktır.



Şekil 10 Convolutional Filtre Uygulaması

2.3.2.3 Pooling Layer

Büyük çözünürlükteki resimlerde özellik kaybetmeden boyut düşürmek için kullanılır buna “dimensionality reduction” denir. Farklı pooling teknikleri mevcut olsa da en çok kullanılan ve en işlevsel olan “Maxpooling” tir. Filtrelere benzer şekilde bir boyut belirlenir örneğin 2x2’lik bir kutu. Yine filtrelere benzer şekilde bu kutular aynı yeri taramayacak şekilde resim üzerinde kaydırılır. Yani 2x2’lik kutu için her kaydırmada 2 adım ilerlenir. Her adımda kutu içerisinde ki en büyük değer alınır ve yeni bir matrise yerleştirilir. Bu şekilde o kutu içinde kalan en baskın özellik ya da en önemli detay saklanarak kayıp olmadan boyut yarı yarıya düşürülmüş olur.



Şekil 11 Maxpooling Uygulaması

Bir diğer metot ise Avaragepooling’tir. Filtre olarak seçilen kutu içerisindeki değerlerin ortalaması alınarak yeni değer elde edilir. Bu metodun pek tercih edilmemesinin sebebi örneğin filtre 2x2 ise 4 değerin ortalaması alındığı için çıkan sonuç aslında resimde olmayan bir değer olabilmesidir.

2.3.2.4 Flattening Layer

Bu Katmanın görevi yukarıda anlatılan filtreleme ve pooling işlemleri sonucunda oluşan boyutu düşürülmüş (2x2’ ye kadar düşürülebilir) çok boyutlu matrisimizi tek boyutlu matris haline getirmektir.

2.3.2.5 Fully Connected Layer

Flattening katmanından sonraki işlemlerde CNN yapısında katmanlar ile sınıflama (anlama) işlemi gerçekleştirilir. Bu katman ile çıkarılan özelliklere göre sınıflama işlemi

burada gerçekleşir. Kesin bir katman biçimi olmamak ile birlikte modelin performansını çok etkilemektedir. Bu katmanın sonunda bir softmax katmanı bulunur. Bu katmanda sistemin tespit edeceği sınıf sayısı kadar nöron bulunmaktadır.

2.3.1.6 Batch Normalization

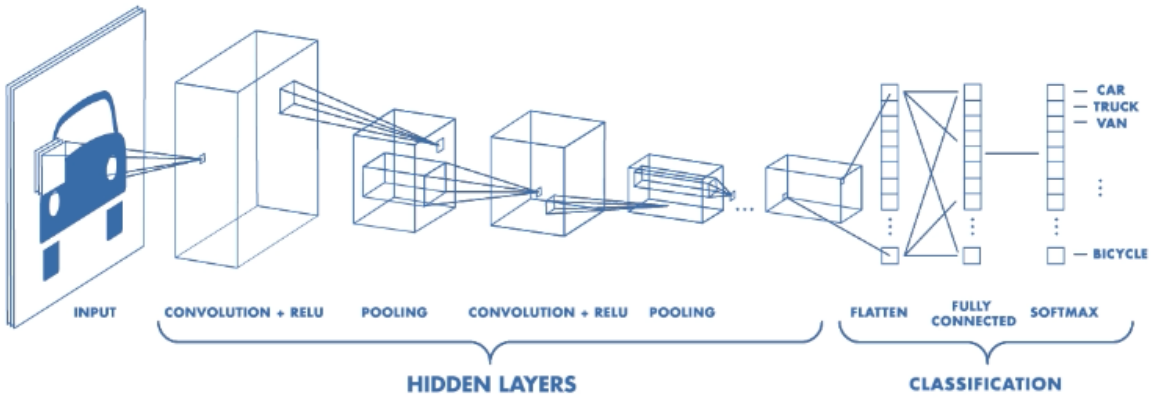
ANN' de hız, performans ve kararlılığı geliştiren bir tekniktir. Batch normalization, kendinden önceki çıkış katmanı vektörünü yeniden ortalayarak ve yeniden ölçeklendirerek giriş katmanını normalleştirmek için kullanılır. Son olarak normalize edilmiş veri γ (Ölçekleme) parametresi ile çarpılarak ve β (Kaydırma) parametresi ile toplanarak işlem tamamlanır. β ve γ parametreleri model eğitimi sırasında bize veriler üzerinde normalleştirme işleminin uygulamaya devam edilmesini sağlar [20].

2.3.1.7 Dropout (Overfitting)

Loss fonksiyon değeri çok küçülürse yani hata sıfıra yaklaşırsa doğruluk %100'e yaklaşır. Bu durum baştan çok iyi gibi görünsede modelin veri setini ezberlemiş olma ihtimali vardır. Modelin eğitimi sonucunda doğruluğun %85 civarında olması ezberleme olmadığını göstermektedir [21].

Dropout işlemi uygulması şu şekilde olur. Her epoch'ta (devirde) gizli katmandaki bazı nodelar (nöronlar) rastgele iptal edilerek modelin ezberlenmesi engellenebilir. İptal edilecek nodelar Bernolulli dağılımı ile 0 ve 1 vektörü oluşturulup nörona çarpan olarak eklenirek uygulanır. Çıkış katmanına dropout yapılamaz. Varsayılan olarak 0 ile çarpılacak nöronlar katmanda ki nöronların yarısı olarak seçilebilir.

CNN de bu aşamaya kadar anlatılan konuları göstermek için aşağıdaki şekil iyi bir örnektir.



Şekil 12 CNN Genel Uygulama [22]

2.3.1.8 Epoch ve Batch-Size

Epoch eğitim sırasında tüm eğitim verilerinin ağa kaç kere gösterileceğini belirleyen sayıdır. Modelin başarımını etkileyen faktörlerden birisidir. Batch-Size ağa verilen örnek sayısıdır. Bu parametre büyük tutulması durumunda büyük veriler ile çalışırken RAM yetmemesi gibi durumlar ortaya çıkabilmektedir.

2.3.2.9 Data Augmentation (Veri Artırma)

Veri artırma işlemidir. Modeli eğitmek için elimizdeki resim sayısı az ise olan resimleri çoğalmamıza imkan verir. Resimler manupüle edilerek çoğaltılır. Örneğin belli bir kısma zoom yapmak, resmi döndürmek veya resmin belli bir kısmını almak gibi.

2.3.2.10 State of Art Models

Resim sınıflamadaki en iyi modellere verilen isimdir. Yarışmalarda ödül alan modellerdir. Keras kütüphanesinde hazır olarak kullanılabilen modeller ve başarımları EK-A' daki tabloda görülmektedir.

2.4. DEEP LEARNING FRAMEWORK' LERİ

Derin öğrenme için kullanılan algoritmalar genel olarak çok kompleks ve kodlanması oldukça meşakkatli yapılardır. Geliştiriciler bu algoritmaları tek tek kodlamak yerine geliştirme için derin öğrenme frameworklerini kullanmaktadır. Bunlardan bazıları Keras, Tensorflow, Caffe, PyTorch' tur.

2.4.1 Keras Framework

Bu projede model geliştirilmesinde kullanılacak framework Keras olacaktır. Keras ile ANN ve CNN modelleri çok hızlı bir şekilde geliştirilebilmektedir. Modelin eğitimi sürecinde CPU ve GPU kullanılabilmektedir. Keras arka planda Tensorflow kullanmaktadır. Yakın zamanda yapılan güncelleme ile Tensorflow 2.0 versiyonu ile daha iyi çalışacağını web sitesinden duyurmaktadır. Aslında güncel hali ve sonraki alacağı güncellemeler de Tensorflow framework içerisine dahil edilmiş olduğu bilgisi de verilmektedir. Bundan dolayı geliştirme aşamasında tf.keras şeklinde kodlanan yeni versiyon kullanılacaktır.

Bir modelin geliştirilmesinin yaşam döndüsü 4 aşamadan oluşmaktadır. Bu aşamalar sırası aşağıda verilmiştir.

2.4.1.1 Modelin Tanımlanması

Modelin tipi tanımlanmaktadır. İki çeşit tanımlama yapılabilir. Bunlar:

- `Sequential()` : Katman katman modeli sıralı olarak oluşturmamızı sağlar. Genellikle sıfırdan bir model geliştiriyorsak bunu tercih ederiz.
- functional API : Katmanların yalnızca önceki ve sonraki katmanlardan daha fazlasına bağlandığı modeller tanımlayabileceğiniz daha fazla esnekliğe sahip modeller oluşturmanıza olanak tanır. Örneğin transfer öğrenme kullanılacaksa model bu şekilde oluşturulur.

2.4.1.2 Compile Model

Model eğitilmeden önce bu metod ile ağın eğitiminde gerekli parametreleri vermemiz gerekir. Bu parametreler şunlardır.

- Loss : model içerisinde ki nöronların hangi loss (Maliyet) fonksiyonunu kullacağı belirtilir. Linear Regression konusunda bu fonksiyonların görevi tanıtılmıştır. Keras'ta kullanılabilen loss fonksiyonlara Tensorflow resmi web sitesinden ulaşılabilir. Ancak özellikle CNN' de binary_crossentropy (sadece iki sınıf varsa), categorical_crossentropy (ikiden fazla sınıf varsa) ve sparse_categorical_crossentropy (yine ikiden fazla sınıf olduğu durumda ama çıkış biçimi farklıdır) fonksiyonları kullanılmaktadır.

- Optimizer : Loss fonksiyonun optimizasyonu için seçilen fonksiyon burada belirtilir. Her optimizerin her model ve veri seti için farklı başarım göstermektedir. Bundan dolayı eğitim sonuçlarına göre farklı optimizerler seçilerek denemeler yapılması daha iyi sonuç alınmasını sağlayacaktır. tf.keras'ta kullanılabilecek optimizerler Tensorflow resmi web sitesinde dökümanlar kısmında mevcuttur.
- Metric : Eğitim ve test sırasında modelin performansını değerlendirmek için kullanılan parametredir. Bunlardan bazıları acc (0 ile 1 arasında doğruluğu verir), BinaryAccuracy (iki sınıf için hangi sınıf kaç eşleşme olduğunu verir), BinaryCrossentropy (iki sınıf için etiketler ve tahminler arasındaki çapraz-ölçüm metriğini hesaplar), CategoricalAccuracy (çoklu sınıf için hangi sınıf kaç eşleşme olduğunu verir), CategoricalCrossentropy (çoklu sınıf için etiketler ve tahminler arasındaki çapraz-ölçüm metriğini hesaplar) daha fazlası Tensorflow resmi web sitesinde dökümanlar kısmında mevcuttur.

2.4.1.3 Fit veya Fit_generator Model

Modelin eğitiminin gerçekleştirildiği fonksiyondur. Genellikle bu fonksiyonu çağırmadan önce `summary()` fonksiyonu ile eğitim öncesi model katman yapısı, eğitilecek parametre sayısı ve eğtilmeyecek parametre sayısı görüntülünebilir. Bu iki fonksiyonda verilmesi gereken parametreler ise şunlardır:

- Giriş Verisi : her iki fonksiyon için de veri bir ön işlemden geçirilmesi gerekir. `fit()` fonksiyonu için bir numpy dizi gerekirken `fit_generator()` için bir veri jenaratörü kullanılır.
- Çıkış Sınıfları: Sadece `fit()` fonksiyonunda çıkış sınıfları verisini belirtmek için kullanılır.
- Validation_data: Eğitim esnasında modelin doğruluğunu kontrol etmek için kullanılan test datalarını işaret eder.
- Epoch: Eğitim esnasında verilerin model üzerinde kaç kere eğitime sokulacağı sayısını belirler.
- Step_per_epoch: her epoch işleminin kaç adımdan oluşacağını belirler. Bu sayısnın düşük olması veri setinin büyüklüğüne göre RAM yetersizliğine sebep olabilmektedir.

- validation_step: modelin doğruluğu kontrol edilirken verilerin kaç adımda verileceğini belirler.

Her iki fonksiyon da çalışmasını tamamladıktan sonra bir History nesnesi döndürür. Bu nesne içerisinde eğitim esnasında hesaplanan loss ve metric değerlerini içinde tutmaktadır. Bu veriler ile mathplotlib gibi kütüphaneler kullanılarak grafikler çizilerek sonuçların görselleştirilmesi gerçekleştirilir.

Eğitime başlamadan önce kullanabileceğimiz bir diğer fonksiyon evaluate() test modunda model için loss değerini ve metric değerlerini döndürmektedir.

2.4.1.4 Predict (Tahmin)

Eğitilen modelin daha önce hiç görmediği bir giriş verisi verilerek sonuç vermesini sağlayan fonksiyondur. Giriş verisi eğitim için modele sokulan veriler gibi önce işlenmesi gerekmektedir. Fonksiyon sonucunda bir numpy dizisi döndürmektedir.

2.4.2 Tensorflow Framework

Tensorflow'da Keras'tan bağımsız olarak modeller oluşturup eğitebileceğimiz bir frameworktür. Yine cpu ve gpu desteği mevcuttur. Tensorflow'un önemli özellikleri arasında web tarayıcı, mobil cihazlar ve iot cihazlarda çalışabilen türevleri olmasıdır. Tensorflow'un kendisi doğrudan kullanılmayacağı için detaylarına girilmeyecektir. Bu projede tensorflow bünyesinde yer alan tensorflow JS ve tensorflow Lite kullanılacaktır.

2.4.2.1 Tensorflow JS

Java script kullanılarak tamamen tarayıcıda modeller eğitmek ve çalıştırmak için kullanılacak tek kütüphane Tensorflow JS' tir. Otomatik olarak WebGL desteği mevcuttur. Tarayıcı üzerinde geliştirilen uygulamalar ile kamera kullanılarak kimlik kontrolu, harekete duyarlı oyunlar, resimlerde sınıflama, tahmin ve yorum işlemleri gerçekleştirilebilir. Tensorflow JS bize eğitilmiş modelleri tarayıcılarda veya node.js ile kullanmamıza olanak sağlamaktadır. Bu çalışmada model tf.keras ile eğitilerek sonrasında eğitilmiş modelin dosyası Tensorflow JS model dosyalarına çevrilerek kullanılacaktır.

2.4.2.1 Tensorflow Lite

Tensorflow Lite mobil ve IoT cihazlar da kullanılmak için geliştirilmiştir. Tensorflow Lite kullanılarak model eğitilebilir ve çalıştırılabilir. Özellikle IoT cihazlarda sınırlı işlemçi gücü bulunmaktadır. Bundan dolayı model Tensorflow Lite'a çevirilirken Quantization işlemi uygulanarak modelin içindeki değerlerin ve işlemlerin hassasiyetini azaltarak hem modelin boyutunu hem de çıkarım için gereken süreyi azaltılabilir. Ancak bu işlem doğruluk kaybına sebep olmaktadır. Yine tf.keras ile eğitilen modelin Tensorlow Lite'a çevrilerek geliştirilecek mobil uygulama da kullanılacaktır.

2.5. TARAYICI EKLENTİSİ

Tarayıcı eklentisi en basit hali ile bir tarayıcı uygulamasıdır. Tarayıcının görevi web sayfalarını görüntülemek olduğu için uygulamalarda istemci tarafında çalışan HTML, CSS ve JavaScript kodlarından oluşan programlardır.

Bir tarayıcı eklentisinin kalbi "manifest" dosyasıdır. Bu dosya içerisinde JSON formatında eklentimiz ile ilgili tüm tanımlamalar yapılmaktadır. Bu tanımlamalardan bazıları şunlardır.

- Eklentinin temel isim , versiyon ve kısa tanım bilgileridir.
- Eklenti için kullanılacak simge dosyası
- "permission" ile geliştireceğimiz eklentinin tablar, yerimleri gibi sayfa dışında ki öğelere erişim izni verilmesi sağlanabilir. Yada eklentinin hangi web sayfalarında çalışacağını belirleyebiliriz.
- "page_action" eklentimizin hangi sayfada aktif hangi sayfada pasif olduğunu görmemizi sağlayan bir buton olmasını sağlayabiliriz. Bu buton bir pencere açılmasını da sağlayacaktır.
- "browser_action" ile eklenti eğer bir pencere açacak ise burada açılacak html sayfası adını, pencereyi açaçak butonun simgesini ve üzerine gelindiğinde yazılacak mesaj burada belirtilir.
- "background" ta belirtilen bir JavaScript dosyası ile kullanıcının göremeyeceği arka planda çalışan kodlar bulunmaktadır.

- "content_scripts" ta belirtilen bir JavaScript dosyası ile o an aktif olan HTML sayfasında ki DOM yapısına müdahale ederek yapısal manupülasyonlar yapılabilir [23].

Chrome eklentisi içerisinde kullanılan page_action veya browser_action' da tanımlanan html sayfalarda rahatça dışarıdan kaynak çağırabilmek mümkündür. Ancak background ve content_scripts için yazılan html veya js dosyalarına dışarıdan direkt olarak kaynak dahil edemeyiz. Bunun için "manifest" dosyamızda özel tanımlamalar yapmamız gerekir.

Ayrıca eklentiler yapısı gereği js dosyaları arasında direkt olarak veri alışverişi yapamazlar. Bundan dolayı iki dosya arasında "chrome.runtime" özel fonksiyonları kullanılarak haberleşme sağlanır.

Bir eklenti içerisinde farklı kaynaklara erişim sağlanması gerekmektedir. Erişim izinleri "manifest" dosyasında belirtilir. Bunlardan bazıları şunlardır.

- tabs: Tarayıcıda ki tablara erişim izni sağlar.
- alarms: Bir kez veya periyodik olarak çalışacak fonksiyon izni sağlar.
- storage: Yerel depolamaya erişim izni sağlar.
- notification: Bildirim oluşturma izni sağlar.
- bookmarks: Yer imlerine erişim izni sağlar.
- contextMenus: Fare sağ tıklaması menüsüne ekleme yapma izni sağlar.
- Cookies: Çerezlere erişim izni sağlar.
- desktopCapture: Pencere veya tab' ın görüntüsünü kaydetme izni sağlar.
- downloads: İndirilenlere erişim izni sağlar.
- geolocation: Lokasyon bilgisine erişim izni sağlar.
- history: Geçmişe erişim izni sağlar.

Yukarıda sayılan izinlerin dışında daha bir çok faydalı izin mevcuttur. Proje hedefi doğrultusunda geliştirilecek eklenti ile tensorflow JS' e çevrilen modelin video etiketlerini takip ederek gerekli işlemleri yapması sağlanacaktır.

2.6. FIREBASE

Firebase Google firmasının bulut tabanlı bir çok hizmet sağladığı bir çatı platformdur. En önemli özelliği cross platform (IOS, Android, Web ve C++ gibi) ve gerçek zamanlı olarak çalışmasıdır. Cloud Firestore hizmeti ile NoSql veri tabanı hizmeti sunmaktadır. Authentication hizmeti ile yine cross platform güvenli oturum hizmeti sunmaktadır. Cloud Storage hizmeti ile resim ve medya depolama hizmeti sunmaktadır. Web siteleri için Hosting hizmeti de yine hizmetler içerisinde bulunmaktadır [24].

Bu sayılan hizmetler içerisinden projede kullanılması düşünülen hizmetler RealTime Database ve cloud storage hizmetleridir. Tarayıcı eklentisi ile yapılan işlemlerin sonuçları NoSql veritabanına kaydedilecektir. Bu şekilde bir katalog oluşturulacaktır. Sonraki video görüntülemelerinde video URL'ine bakılarak izlenilen videonun kayıtlı olup olmadığına bakılarak çalışan sistemde ki iş yükü azaltılmaya çalışılacaktır. Ayrıca kullanılacak model firebase cloud stroge içerisinde depolanacaktır.

2.7. MOBİL UYGULAMA GELİŞTİRİLMESİ

Adroid cihazlar için uygulama geliştirmek için birkaç alternatif bulunmakla birlikte en geçerli yöntem "Android Studio" kullanmaktır. Jet Brains tarafından geliştirilen Android Studio IDE (integrated development environment) olarak tüm ihtiyaçları karşılamaktadır.

Android stüdyo kurulumundan önce JDK (Java Development Kit) kurulumu yapılması gereklidir. Android studio kurulumu yapıldıktan sonra SDK kurulumu yapılmasıda gerekir.

Bu proje ihtiyaçları doğrultusunda. Geliştirme için "webview" komponenti kullanılacaktır. Bu komponent temel olarak uygulama ekranı içerisinde bir web sayfasını görüntüleme işlemi yapmaktadır. Webview içerisine JS enjekte ederek izlenilen videodan kareler kaydetmek. Resim bilgisinin java kodları ile ön hazırlık yapılması ve Firebase ML Kit hizmetinden faydalanılarak local olacak şekilde resim sınıflaması yapılmasıdır. Sonuçlar yine bulut veritabanına kaydedilecek ve görüntülenen video bulut veritabanında kayıtlı mı kontrol edilecektir.

3. KULLANILAN ARAÇ VE YÖNTEM

3.1. ÇALIŞMA ORTAMLARININ HAZIRLANMASI

Modelin geliştirilmesi için Anaconda ve PyCharm kurulumları yapıldı. Anaconda hem gerekli kütüphaneleri hazır sunduğu hemde Jupyter Notebook ve Spyder IDE kurulumlarını yaptığı için tercih edildi. PyCharm python için virtual environment sağlaması ve kod yazmayı kolaylaştıran yapısı için tercih edildi.

Jupyter Notebook kodlar ve veriler için web tabanlı bir etkileşimli geliştirme ortamıdır. Jupyter Notebook esnektir. Kullanıcı ara yüzü veri bilimi, bilimsel hesaplama ve makine öğreniminde çok çeşitli iş akışlarını desteklemektedir. Başlangıçta sadece python dili için tasarlanmış olsa da sonradan başka dillere de destek vermeye başlamıştır [25].

Model için bir diğer geliştirme ortamı olarak Colab kullanıldı. Colab bulut bir hizmet olarak Python kodlarımızı GPU (çalımaların yapıldığı sırada kullanılan GPU NVIDIA Tesla P4) üzerinde çalıştırmamıza imkan sağlaması ile local bilgisayarımız ile çok uzun sürecek modelin eğitilmesi işlemini en az 5 kat daha hızlı yaparak eğitimin hızlanmasını sağlamış oldu. Bir diğer tercih sebebi de Google Drive ile bağlantılı çalışarak veri setinin bulut ortamda bulunması ile neredeyse bilgisayar bağımsız bir çalışma ortamı sağlanmasıdır.

Eklenti geliştirilmesi için VsCode kurulumu yapıldı. VsCode geniş eklenti seçenekleri ile Java Script yazmayı ve html kodlarını otomatik tamamlama sağlaması sebebi ile tercih edildi.

Local sunucu olarak Web Server for Chrome kullanılacaktır. Geliştirme sırasında bulut hizmet kullanmak yerine local sunucu kullanmak bir avantaj sağlayacaktır.

Mobil uygulama geliştirilmesi için JDK ve Android Studio kurulumları yapıldı. Android Studio sorunsuz hazır bir geliştirme ortamı sunduğu için tercih edildi.

3.1. KULLANILAN PROGRAMLAM DILLERI VE PLATFORMLAR

Modelin eğitilmesi için Python (3.6) dili kullanıldı. Python programlama dili her geçen gün artan popülerliği ve özellikle bilimsel çalışmalar ve yapay zeka için sunduğu geniş kütüphane seçeneklerinden dolayı tercih edildi.

Eklentinin geliştirilmesi VSCode (1.45) programı kullanıldı. Kodlama için Java Script, HTML ve CSS kullanıldı. Eklenti için başka bir alternatif geliştirme seçeneği bulunmamaktadır.

Bulut depolama, veri tabanı ve makine öğrenmesi özelliklerinden dolayı Firebase seçildi. Ayrıca Firebase ücretsiz olması, gerçek zamanlı olması ve stabil çalışması sebebi ile tercih edildi.

Mobil uygulamanın geliştirilmesi için Android Studio (4.0 RC1) kullanıldı. Programlama dili olarak Java kullanıldı. Kotlin ve Dart (Flutter ile kullanılır) gibi farklı diller bulunmasına rağmen tam uyumluluktan dolayı Java tercih edilmiştir. Emülatör için GenyMotion (3.1) programı kullanıldı. GenyMotion çalışması için Oracle VirtualBox (6.1) programları kullanıldı.

4. SİSTEMİN GERÇEKLENMESİ

4.1. VERI SETI OLUŞTURULMASI

CNN için kullanılacak veri seti resimlerden oluşmaktadır. Sistemde ki her bir sınıf için çok fazla resime ihtiyaç vardır (mümkünse 10.000 ve üstü sayılarda). Yine her bir sınıf için yaklaşık olarak aynı sayıda resim olması en ideal durumdur.

Şekil 13 Veri Setinden Örnekler

Bu çalışmada üç sınıf olacak şekilde tasarım yapılacaktır. Bu sınıflar tütün mamulleri, silah ve zararlı içeceklerdir. Yapılan araştırmalar sonucunda öncelikle Kaggle web sitesinde tütün mamülleri [26], silah [27] ve zararlı içecekler [28] için hazır veri setleri temin edildi. Özellikle silah sınıfı için veri setinde resim sayısının yetersiz olmasından dolayı veri setini geliştirmek için çalışmalar yapıldı. Bu çalışmalardan sonuç alınan "Bulk Bing Image Downloader" [29] oldu. Bir python script'i üzerinden verilen anahtar kelime ile ilgili arama sonuçlarında gelen resimlerin kaynak dosyalarının bilgisayara indirilmesi sağlandı.

Görsel olarak veri seti üzerinde yapılan incelemelerden sonra sınıflardaki resim sayıları şu şekilde oluştu: tütün mamülleri 2147 resim, silah 728 resim ve zararlı içecekler 2928 resim. Veri setine bakıldığında bu sayıların yetersiz olduğu görüldü ancak özellikle sınıfların çok iyi konular olmadığı da göz önüne alınarak eldeki veri seti ile model eğitimlerine başlandı.

4.2. MODELIN EĞITILMESI

4.2.1 Google Colaboratory Kullanımı

Google Colaboratory bulut hizmeti ile Jupyter Notebook kullanarak model eğitim işlemleri gerçekleştirilecektir. Bu hizmet bir uzak sanal makinede kodlarımızı çalıştırmamızı sağlamaktadır. Sanal makinenin fiziksel özellikleri şöyledir: Intel(R) Xeon(R) CPU @ 2.20GHz, Tesla P4 (GPU), 12.72GB RAM, 68.40GB Disk alanına

sahiptir. İşletim sistemi olarak Ubuntu kuruludur ve makine öğrenmesinde yaygın kullanılan kütüphaneler içerisinde yüklü bulunmaktadır. Bazı kütüphanelerin çalışma yapıldığı sırada ki versiyonları şunlardır:

- tensorflow 2.2.0-rc4
- keras 2.3.1
- google-colab 1.0.0
- google 2.0.3
- oauth2client 4.1.3
- ipython 5.5.0
- numpy 1.18.4
- matplotlib 3.2.1

Sistemde her oturum açıldığında sanal makine sıfırlanmaktadır. Bu hizmetin tercih edilmesinde iki temel sebep vardır. Birincisi GPU üzerinde kodların çalıştırılması CPU üzerinde çalıştırılmaya göre çok daha hızlı olmaktadır. İkinci olarak veri seti drive' a yüklenerek eğitim sırasında tamamen bilgisayar bağımsız bir çalışma ortamı kazanmış olmamızdır.

Bu çok faydalı hizmet ücretsiz olmakla birlikte bazı kullanım sınırları mevcuttur. Bunlar bir bilgisayardan aynı anda sadece 2 tane Jupyter Notebook çalıştırabilmemiz. GPU kullanım verisi hesaplanmakta ve günlük kullanımın bir üst sınırı bulunmaktadır.
Google Colaboratory kullanmak için bir google hesabı ile oturum açtıktan sonra google drive hizmetinde "Yeni", "Diğer" altında ki "Daha Fazla Uygulama Bağla" seçildikten sonra "Google Colaboratory" seçilerek uygulama drive'ımıza bağlanmış olur. Yeniden "Google Colaboratory" seçilerek yeni bir Jupyter Notebook oluşturulduktan sonra istenilen isim verilir ve drive içerisinde "Colab Notebooks" içerisine kaydedilmiş olur.
Yazılan python kodlarının çalıştırılması için istenilirse code bloğunun solunda ki "run" butonu ile çalıştırılır yada CTRL+Enter tuş kombinasyonu da kullanılabilir. Çalışmadaki tüm code blocklarının çalıştırılması için CTRL+F9 tuş kombinasyonu kullanılabilir. Bir diğer önemli özellik için "Runtime" menüsü altında "Change Runtime Type" seçilir. Gelen pencereden "Hardware accelerator" dan GPU seçilir. Son olarak "SAVE" seçilerek artık kodlarımızın GPU' da yürütüleceği ayar yapılmış olur.

Drive'ımızda yüklü olan dosya ve klasörlere erişmek için bir "Google Drive File Stream" oluşturmamız gerekir. Aşağıdaki kodları çalıştırdığımızda bizden tarayıcıda açmamız istenilen bir link verir.

```
from google.colab import drive
drive.mount('/content/drive')
```

Bu linke tıklandığında bir google hesabı ile oturum açmamız ve drive ile ilgili gerekli erişimleri vermemiz istenecektir. Son olarak karşımıza bir kod gelecektir. Bu kodu kopyalayıp çalıştırılan kod bloğumuzda gerekli yere yapıştırıp enter'a basarak işlemi tamamlamış oluruz. Sol taraftaki klasör simgesine tıklayarak drive içerisindeki dosyalarımızı görüntüleyebiliriz.

4.2.2 Ağı Bizim Oluşturduğumuz (Custon) Model Eğitilmesi

Modelin tasarlanmasında bir çok deneme yapılmış ve en verimli olan modelin kodları EK-B' da verilmiştir. Şimdi bu kodları açıklayalım.

Kodlarda önce "Google Drive File Stream" oluşturularak Drive ile bağlantı sağlanmıştır.

Gerekli kütüphanelerin eklenmesinin ardından eğitimde kullanılacak "train_dir" ve doğrulamada kullanılacak "validation_dir" yollarını tutacak değişkenlere atamalar yapılmıştır.

Sıfırdan bir model eğittiğimiz için Sequential (Sıralı) olarak kodlanmaya başlanmıştır. Modelimizin giriş katmanını oluşturacak ilk Conv2D (evrişim) katmanımızda 256 adet nöron ve 2x2'lik bir filtre tanımladıktan sonra "input_shape" parametresi ile 224x224x3 olan giriş biçimimizi belirttik. Bu parametre ile sistemin eğitiminde ve eğitim sonrasında tahmin için modele verilecek resimlerin ebatlarının bu olması gerekmektedir. Bu katman için aktivasyon fonkiyonu "relu" seçilmiştir. Genel olarak giriş ve ara katmanlarda bu fonksiyon tercih edilmektedir. Sonrasında tanımlanan "MaxPooling2D" ile 2x2 pooling işlemi yapılmakta ve girişe uygulanan resim verisi yarılanarak 112x112x3 boyutuna indirilmektedir. Son olarak "BatchNormalization" işlemi ile bu ara katman tamamlanmış olur.

Sonraki ara katman Conv2D' da 512 adet nöron ve 2x2'lik bir filtre tanımladıktan sonra aktivasyon fonksiyonu yine "relu" seçilmiştir. Sonrasında tanımlanan "MaxPooling2D" ile 2x2 pooling işlemi yapılmakta ve girişe uygulanan resim verisi yarılanarak 56x56x3 boyutuna indirilmektedir. Son olarak "BatchNormalization" işlemi ile bu ara katman tamamlanmış olur.

Bundan sonraki 3 ara katmanda sade nöron sayısı değiştirilerek özellik çıkarma işlemine devam edilmiştir. En sonunda elimizde 7x7x3' lük bir tensor kalmıştır.
Flatten katmanı ile tensor bir vektöre dönüştürülürek özellik çıkarma işlemi tamamlanmış olur. Bu aşama sonrasında modeltasarımında artık sınıfları öğrenme işlemi başlar.

Dense katmanında (Fully Connected Layer) 256 nöron tanımlanmıştır. Aktivasyon fonksiyonu evrişim katmanlarında olduğu gibi "relu" seçilmiştir. Modelin veriyi ezberlememesi için 0.2'lik bir "Dropout" katmanı eklenmiştir. Son olarak "BatchNormalization" işlemi ile bu katman tamamlanmış olur.

Yukarıda tanımlanan katmanın aynısından iki kere daha tanımlama yapılmıştır.

Modelimizin ağ yapısının sonuna gelirken. Son bir Dense katmanı eklenir. Bu katman 3 nörondan oluşur. Çünkü modelimiz 3 sınıf için tahmin yapacaktır. Yeni modelimizde sınıf sayısı kadar olmak zorundadır. Yoksa hata verecektir. Bu katmanın aktivasyon fonksiyonu "softmax" seçilerek modelin ağ yapısı tamamlanmış olur.

"ImageDataGenerator" fonksiyonu kullanılarak eğitim için kullanılacak resimlerde hangi veri artırımları yapılacak (Dat augmentation) belirlenir. Bunlar:

- rescale : normalize işlemi yapar.
- rotation_range : resmin ne kadar döndürülebileceğini belirler.
- width_shift_range : resmi yatayda çerçevesi içerisinde kaydırmayı sağlar
- height_shift_range : resmi yatayda çerçevesi içerisinde kaydırmayı sağlar
- shear_range : resmi eğmeyi sağlar.
- zoom_range : resme yakınlaştırma yapılmasını sağlar.

- horizontal_flip : resmi yatayda aynalama yapılabilmesini sağlar.
- fill_mode : yukarıda sayılan özellikler uygulandığında resim çerçevesinde yapılan işlemden dolayı boş alanlar kalırsa nasıl doldurulacağını belirtir. Biz "nearest" kullandık yani en yakın piksele göre dolduracaktır.

"flow_from_directory" fonksiyonu ile eğitim için gerekli şu parametreler belirlenir:

- validation_dir : eğitim için kullanılacak resimlerin yolu (path)
- batch_size : eğitimde her adımda kaç resim işleneceği genelde 32 seçilir.
- shuffle : resimlerin rastgele bir sırayla modele verilmesini belirler.
- class_mode : sınfı modudur. Bizim ikiden fazla sınıfımız olduğu için 'categorical' olmak zorundadır.
- target_size : modele verilecek resmin boyutunu yeniden ayarlamayı sağlar. Bizim modelimiz için 224x224 olmak zorundadır.

Bir "ImageDataGenerator" de doğrulama resimleri için yapılır ancak sadece rescale işlemi yapılır. Sonrasında ki doğrulama için "flow_from_directory" fonksiyonu aynıdır. Optimizer seçiminde yapılan bir çok denemeden sonra en iyi sonuç "RMSprop" ile alındığı için tercih edilmiştir. "RMSprop" un bir özelliğide eğitim sonunda kaydedilen modelin boyutunun bir miktar daha küçük olmasıdır.

"compile" fonksiyonu ile loss fonksiyonumuzu belirleriz. Burada ikiden fazla sınıfımız olduğu için "categorical_crossentropy" seçilmiştir. Optimizer daha önce tanımlanmıştı. "metric" parametresinde eğitim sırasında her epochun ara adımlarında bize yol gösteren parametredir. Burada sadece "accuracy" seçilerek doğruluğu görmek bizim için yeterlidir.

"summary" fonksiyonu ile model eğitimine başlamadan önce ağımızı gözden geçirmemiz, eğitilebilir ve eğitilemeyen parametre sayımızı görebiliriz. Burada parametre sayısını görmek bizim için çok önemlidir. Çünkü ağımız büyüdükçe parametre sayısı artacak bu hem eğitim süresini uzatacak hem de eğitim sonunda ortaya çıkacak model dosyasının boyutunu büyütecektir. Sonraki aşamalarda geliştirilmek istenilen uygulamalar düşünülerek model başarımının azalması durumuna rağmen çok yüksek parametre değerli ağlar oluşturulmamıştır.

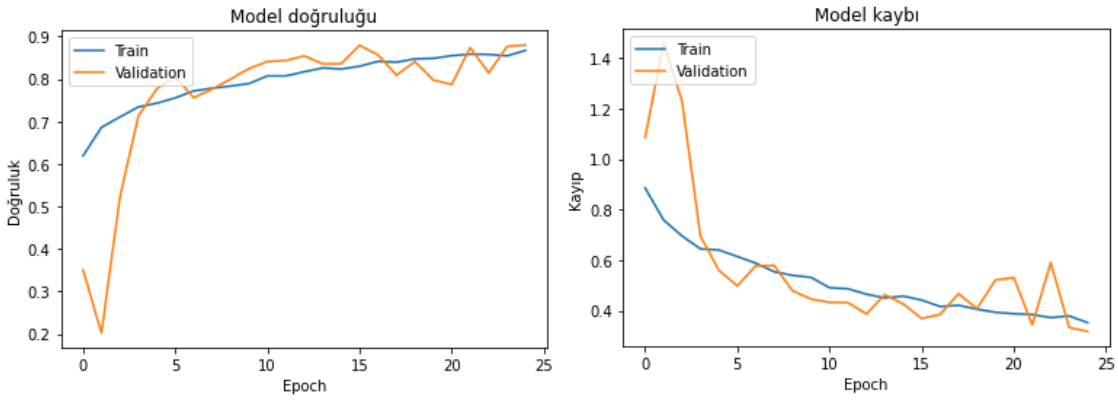
“fit_generator” modelimizin eğitimini gerçekleştirecek fonksiyonumuz budur. Şu parametreleri tanımlamamız gerekmektedir.

- generator :eğitim için oluşturulan generator belirtilir.
- validation_data : doğrulama için oluşturulan generator belirtilir.
- steps_per_epoch : her epoch’ ta kaç adım olacak belirtilir. Bunu hesaplamak için eğitim için resim sayısı / batch_size ile bulunu. Bizim modelimiz için 4640/32 = 145 olur.
- validation_steps : doğrulama için adım sayısı aynı formülle hesaplanır.
- epochs : eğitim verileri öğrenme için modele kaç kere verileceğini belirler.

Epoch sayısı önemli bir parametredir. Az olması modelin tam öğrenememesine yol açabilir. Ancak çok fazla epoch sayısı modelin ezberlemesine de sebep olabilir. Ya da bir noktadan sonra epoch sayısı ne kadar artırılırsa artırılsın modelin başarımı yatay bir eğriye dönüşür ve bir gelişim gösteremez. “fit_generator” çalıştırıldığında ekranda her epochta ki adımlar için hesaplanan doğruluk ve loss değerleri görüntülenmektedir. Tabi bu çok görsel bir sunum olmamaktadır.

“save” fonksiyonu ile eğitimi tamamlanan model drive’ a kaydedilir.

Son kısımdaki kodlar ile eğitim sırasında oluşan doğruluk ve doğrulamadaki doğruluk bir grafik olarak gösterilir. Loss ve doğrulamadaki Loss’ta ayrı bir grafik ile görselleştirilerek programımız tamamlanır.



Şekil 14 Custom Model Eğitim Grafiği

Yukarıdaki grafiklerde de görüldüğü gibi eğitilen model %90'a yakın bir başarım göstermektedir. Bu çok iyi bir sonuç olarak görülebilir. Ancak gerçek uygulamalar üzerinde yapılan denemeler de bu kadar yüksek başarım maalesef elde edilememiştir. Kayıt edilen h5 dosyasının boyutu 59MB olmuştur. Bu kabul edilebilir bir boyuttur.

4.2.3 Transfer Öğrenme ile Model Eğitilmesi

4.2.3.1 Neden Transfer Öğreme?

Oluşturulan sınırlı veri seti ile sıfırdan geliştirilen modele alternatif olarak transfer öğrenme kullanılarak deneylere devam edilmiştir.

Transfer öğrenme veri setinin yetersiz olduğu durumlarda daha önceden çok büyük veri setleri ile eğitilmiş state of art modellerin ağırlıklarını yani özellik çıkartma becerilerini kullanmamıza imkan vermektedir.

Transfer öğrenmeden faydalanırken kullanılacak önceden eğitilmiş modelin (pre-trained) çıkış katmanları model tasarımından çıkarılır ve katmanlar dondurularak (eğitilmeyip eski ağırlıklarını tutması sağlanarak) üzerine bizim çıkış katmanlarımız eklenir. Eğitim az miktarda eğitilebilir parametre ile gerçekleştirilir. Bu durum eğtimin çok daha kısa sürmesinin yanı sıra çok daha başarılı modeller elde etmemizi sağlamaktadır.

Ön eğitimli modellerden MobileNetV2, DenseNet121, NasnetMobile, Resnet50V2 kullanılarak deneyler gerçekleştirilmiştir.

4.2.3.2 Çıkış Katmanının Başarımına Etkisi

İlk olarak çıkış katmanı için ağ yapısı ile ilgili denemeler yapıldı. Çıkış katmanında kullanılan katman sayısı ve nöron sayısı artırıldıkça başarım artırılmaktadır. Ancak eğitilmiş modelin boyutunun çok büyümesi tercih edilmemektedir. Ayrıca bir noktadan sonra çıkış katmanlarının büyütülmesi çok fazla iyi yönde değişime sebep olmamaktadır. Aşağıda ki tabloda bu durum görünmektedir.

Tablo 1 Çıkış Katmanı Ağ Büyüklüğünün Etkisi

Model Adı	Eğitilebilir Parametre Sayısı	Eğtim Başarım	Doğrulama Başarım	Model Boyutu
MobileNetV2	395011	0.6506	0.58	12MB
	593155	0.6534	0.5035	14MB
	2762243	0.6511	0.5503	30MB
DenseNet121	329475	0.7362	0.7474	30MB
	527619	0.736	0.7622	32MB
	2631171	0.7442	0.7318	48MB

4.2.3.3 Optimizerin Eğitim Sürecine Etkisi

Model eğitimi sırasında seçilen optimizer kaç epoch sayısında maksimum öğrenme düzeyine ulaşılacağını belirlemede önemli rol oynamaktadır. Optimizer algoritmalarının kendilerine has özellikleri vardır. Kimisi hızla loss değerini düşürürken çok gürültü ile ilerler kimisi daha az gürültü ile daha uzun sürede maksimum başarıma ulaşabilir. Ancak bilinmelidir ki epoch sayısının çok fazla olması başarımı sürekli yukarı doğru yükselten bir durum oluşturmamaktadır. Bir seviyeden sonra başarım düzeyi yatay eğri haline gelecektir.

EK-C'deki deney sonuçlarının grafikleri incelendiğinde kullanılan veri seti için eğitim sırasında "Nadam" algoritmasının en az gürültü ile en kısa sürede başarımın yükseldiği görülmüştür. Ancak doğrulama sırasında tüm optimizerlerin çok görültü oluşturduğu görülmekle birlikte en iyi sonucu "RMSprop" vermiştir.

Loss değerleri incelendiğinde, eğitim sırasında yine "Nadam" algoritmasının en hızlı sonucu verdiği görülmüştür. Ancak yine doğrulama verileri incelendiğinde "RMSprop" un daha iyi sonuç verdiği görülmektedir.

Tüm sonuçlar göz önünde bulundurulduğunda “Nadam” başarılı gibi görünse de doğrulamada çok fazla gürültü oluşturmaktadır. “RMSprop” ve ardından “Adam” algoritmları başarılı optimizerler olarak görünmektedir.

4.2.3.4 Ön Eğitimli Modellerin Karşılaştırılması

Ön eğitimli modeller arasında ağ yapısının oluşturulması ve parametre sayılarında farklar vardır. Doğal olarak aralarında görece küçük başarım farkları olmaktadır. Genellikle daha fazla parametreye sahip modeller daha başarılı olacağı düşünülebilir ama böyle bir durum yoktur. Tablo 1 incelenirse bu durum görülebilmektedir. Uygulama içerisinde kullanılırken modeldeki hesaplanacak parametre sayısının çok olması daha fazla işlem gücü gerektirecektir. Uygulama geliştirilecek platformların işlem gücü göz önünde bulundurularak seçim yapılmalıdır. Yine dosyasının büyüklüğü bizim için önem arz etmektedir. Çünkü modelin çalışabilmesi için modelin buluttan veya localden RAM’e yüklenmesi gerekecektir. Çok büyük boyuttaki model dosyalarının yüklenmesi sistemin başlatılmasında ciddi gecikmelere sebep olacaktır. Bundan dolayı küçük dosya boyutuna sahip modeller tercih edilecektir. Bizim veri setimiz ile yapılan deneylerde aşağıdaki sonuçlar elde edilmiştir.

Tablo 2 Ön Eğitimli Modellerin Karşılaştırılması

Model	**Eğitim**		**Doğrulama**		**Boyut**
	Doğruluk	**Hata**	**Doğruluk**	**Hata**	
DenseNet121	0.7442	0.6205	0.6432	0.7318	48MB
NasnetMobile	0.7373	0.6370	0.6953	0.7318	39MB
MobileNetV2	0.7957	0.6511	0.5503	1.1368	30MB
ResNet50V2	0.6836	0.7424	0.7318	0.6674	115MB
EfficientNetB0	0.9088	0. 2578	0.1286	0.9566	32MB

Yukarıdaki deneylerde sadece “EfficientNetB0” farklı bir ağ yapısına sahiptir. Diğer modeller aynı ağ yapısı ve parametreler kullanılarak yapılmıştır. Sonuçlar göstermektedir ki dropout uygulanmasına rağmen “EfficientNetB0” ezberleme (Overfitting) yaptığı düşünülmektedir. Maalesef bu eğitilen model tensorlfow JS ve tensorlow Lite modellerine çevrilemedeği için çalışan sistem üzerinde test edilememiştir.

4.2.3.5 Eğitim İçin Gerekli Kodların Yazılması

EK-D' de transfer öğrenme için kullanılan temel kodumuz verilmiştir. Kodlarda önce "Google Drive File Stream" oluşturularak Drive ile bağlantı sağlanmıştır.
Gerekli kütüphaneler eklenmiştir. Çalışma esnasında hangi ön eğitimli model ile eğitim yapılacaksa onla ilgili kütüphaneler eklenmiştir ."train_dir" ve doğrulamada kullanılacak "validation_dir" yollarını tutacak değişkenlere atamalar yapılmıştır.

"ImageDataGenerator" ve "flow_from_directory" fonksiyonları custom model eğitimi kısmında anlatıldığı için burada tekrar anlatılmayacaktır. Bir tek "ImageDataGenerator" fonksiyonunda "preprocessing_function=preprocess_input" parametresi eklenerek seçilen ön eğitimli modelin kendi veri ön hazırlık özelliği de kullanılmıştır.

Ön eğitimli modelin adı olan fonksiyon ile eğitilecek modelin temeli hazırlanır. Bu fonksiyonda 3 parametre kullanılır.

- include_top : modelin çıkış katmanlarının kullanılmayacağını ifade eder.
- weights : modelin özellik çıkartmada temel alacağı ağırlıkların ne olacağı belirlenir. "imagenet" tüm dünyada bir çok kişinin ortak çalışması ile oluşturulmuş çok büyük bir veri seti üzerinden modelin eğitilmiş ağırlıklarını kullanarak en iyi özellik çıkartma ağırlıklarına sahip olmamızı sağlayacaktır.
- input_shape : parametresi ile 224x224x3 olan giriş biçimimizi belirttik. Ön eğitimli modellerin farklı farklı giriş biçimleri olduğu için bu parametre varsayılan değer ile bırakılmamalıdır.

Bir döngü ile modelin katmanlarının eğitilebilme özellikleri devre dışı bırakılır.
Kendi çıkış katmanlarımızı bu aşamada oluşturmaya başlarız. Dense katmanlar ile fully connected katmanlar arasında ki bağlantıyı sağlamak için "GlobalAveragePooling2D" katmanı kullanılır. Bu katmanın temelde yaptığı iş örneğin 8x8x3 olan bir tensorun değerlerin ortalamalarını alarak 1x1x3'lük bir tensor haline getirmektir.

İlk Dense katmanı 512 nöron ile tanımlanmıştır. Aktivasyon fonksiyonu "ReLU" seçilmiştir. Ardından "BatchNormalization" işlemi ile bu katman tamamlanmış olur.

Sonrasında 1024,1024 ve 512 nöron ile aynı katman tanımlaması tekrar yapıldı. En son katmanda “BatchNormalization” işlemi yapılmadı.

Ayrı bir değişkende bir Dense katmanı tanımlanır. Bu katman 3 nörondan oluşur. Çünkü modelimiz 3 sınıf için tahmin yapacaktır. Modelin sınıf sayısı kadar olmak zorundadır. Yoksa hata verecektir. Bu katmanın aktivasyon fonksiyonu “softmax” seçilir.

Model ağımızı tamamlamak için bir değişkende “Model()” fonksiyonu ile giriş ve çıkış katmanlarını tutan değişkenler vasıtasıyla tanımlama yapılır.

“compile” fonksiyonu için seçilen parametreler şunlardır; optimizer “RMSprop”, loss fonksiyonu için “categorical_crossentropy” ve metric için “accuracy” seçildi.

“summary” fonksiyonu ile model ağı ve parametre sayılıra gözlemlenmiştir.

“fit_generator” fonksiyonu parametreleri custom model eğitiminde ki aynı olduğu için burada tekrar anlatılmayacaktır.

Son olarak model dosyasının kaydedilmesi ve eğitim değerlerinin grafiğe çizdirilmesi gerçekleştirilmiştir.

4.2.4 Tarayıcı Eklentisinin Geliştirilmesi

Tarayıcı eklentisinin kodları EK-E’ de paylaşılmıştır.

4.2.4.1 Modelin tensorflow JS’ e Çevirilmesi

Eğtilen modelin tarayıcıda kullanılabilmesi için tensorflow JS modeline dönüştürülmesi gerekmektedir. Öncelikle bu kısımda modelin eğitimi sırasında kullanılan “Google Drive File Stream” kullanılmamıştır. Çünkü modelin çevirilmesi sırasında yazılan kodun dosya yolunda bulunan boşluk karakterinden dolayı çalışmamasıdır. Bundan dolayı sanki local sabit diskimizde çalışıyormuşuz gibi çalışan yine drive’ımıza erişen bir kod yazılmıştır.

“os” kütüphanesi ile konum tam çevirilecek model dosyasının olduğu klasör konumana ayarlanmıştır.

“tensorflowjs” kütüphanesi varsayılan olarak yüklü olmadı için onun “pip” paket yöneticisi ile yüklemesi gerçekleştirilmiştir.

“tensorflowjs_converter” komutu ile modelimizin çevirilmesi işlemi gerçekleştirilmiştir. Ön eğitimli modeller üzerinde yapılan çevirme işlemi esnasında herhangi bir hata alınmamasına rağmen tarayıcı eklentisinde test edilmek istendiğinde şu sonuçlar elde edilmiştir.

- NasnetMobile ve EfficientNetB0 modellerinde hata verdi ve çalışmadı
- MobileNetV2 bir bug sonucu tüm katman isimlerine bir son ek eklemesi sonucu tek tek el ile düzeltme işlemi gerektirdi.
- Resnet50V2 çalıştı video görüntüsünde sekmelere sebep oldu.
- Custom Model ve DenseNet121 sorunsuz çalıştı.

4.2.4.2 Firebase ile İlgili İşlemler

Uygulamalarımızda veri tabanı için firebase kullanılacağı daha önce ifade edilmişti. Ilk olarak “https://console.firebase.google.com” web sitesinde oturum açılır. “Proje Oluştur” tıklanır. Proje ismi verilir, firebase kullanım şartları onaylanarak “Devam” tıklanır. “Google Analytics” kullanım onayı seçilir, son olarak konum “Türkiye” seçilerek “Proje Oluştur” a tıklanarak işlem tamamlanır.

Firebase birçok servisten oluşan komple bir “back end” çözümüdür. Bu proje içerisinde sadece “RealTime Database” ve “Storage” özelliklerini kullanacağız.

“Database” sayfasında “RealTime Database” bulunarak “Veritabanı oluşturun” a tıklanır. “Kilitli modda başla” seçilir ve “Etkinleştir” tıklanır.

Veritabanı oluşturulduktan sonra ilk iş olarak “Kurallar” sekmesi altında veritabanımıza erişim ile ilgili ayarları yapmamız gerekmektedir. False olan “read” ve “write” izinleri true yapılmalıdır. Bu yapılan ayarlar bir güvenlik açığına sebep

olmaktadır. Bundan dolayı ticari bir ürün için daha fazla güvenlik kuralına ihtiyaç vardır.

“Storage” hizmetini başlatmak için sol panelden seçilir. “Başla” seçilerek devam edilir. Bizden sunucu konumu seçimi yapmamız istenir. Bunun değiştirilemeyeceği ile ilgili bir uyarıda bulunmaktadır. Artık istenirse “Dosya Yükle” tıklanarak model dosyaları buluta yüklenebilir. Bu hizmet içinde “Rules” adı altında güvenlik kuralları mevcuttur. Ancak varsayılan olarak okuma izni açık olduğu için değişiklik yapmamıza gerek yoktur.

Bir diğer konu “Storage” hizmetine eklentiden eriştiğimizde “CORS” ile ilgili hata almamamız için ayar yapılması gerekmektedir. “CORS” bir güvenlik önlemidir. Çalışma mantığı bir web sitesi görüntülenirken o web sitesinin başka bir siteye istek yapmasını engellemektir. Bu şekilde kullanıcı bilgilerinin çalınması engellenmektedir. Geliştirilecek olan eklentide Youtube sayfasında çalışacağı için bizim Storage hizmetine erişmemiz engellenecektir. Bu engeli kaldırmak için “https://console.cloud.google.com” sitesine giriş yapılır. İlk defa giriş yapıyorsak hizmet şartlarını kabul etmemiz istenecektir. “Proje seçin” butonuna tıklanarak proje seçimi yapılır. Ardından sağ üst köşede ki terminal (>_) butonuna tıklanır. Pencere altında “Cloud Shell” açılacaktır “Devam” tıklanır. Bir süre bekledikten sonra terminal başlayacaktır. “nano cors.json” komutu ile nano editöründe gerekli dosya oluşturulacaktır. Ardından içerisine şu kod bloğu eklenir:

```
[
  {
    "origin": ["*"],
    "method": ["GET"],
    "maxAgeSeconds": 3600
  }
]
```

“CTRL + X” ardından “y” tuşuna basılarak dosyaya kodlar kaydedilir. “gsutil cors set cors.json gs://mktez-c305f.appspot.com” kodu çalıştırılarak işlem tamamlanır. Son kod içerisindeki gs ile başlayan kısım Firebase’deki “Storage” hizmeti bağlantımızdır [30].

Son olarak eklentiden “RealTime Database” erişebilmek için yapılması gereken bir işlem vardır. Firebase projesinin ana sayfasında “Uygulama ekle” tıklanarak “</>” web seçilir. Uygulama için bir isim seçilir ve “Uygulama kaydet” seçilir. Bu aşamada bize web sayfamızda kullanmak üzere gerekli sdk kodlarını verecektir. Bu kodları daha sonra kullanılmak üzere saklanır. Gerekirse uygulama ayarlarından da sonra erişim sağlanabilmektedir.

4.2.4.3 manifest.json Dosyası

Chrome eklentisinin kalbi bu dosyadır. Uygulamanın ismi, açıklaması, versiyonu ve iconların tanımlanması yapılan tanımlamaların sadece bir kısmıdır.

“page_action” kısmında sadece belirli bir sayfada ya da sayfalarda eklentinin aktif olmasını sağlayacaktır. Eklentinin iconu bu sayfada aktif olacak ve tıklanması durumunda bir pencere görüntülenecektir.

“background” kısmında arka planda çalışacak kodlarımızın yazıldığı dosya belirtilir.
“content_scripts” kısmında eklentinin “https://www.youtube.com/*” ve alt sayfalarında aktif olacağı belirtildi. Bu web sayfası içerisine “content.js” ve “tfjslatest.js” dosyaları enjekte edilecektir.

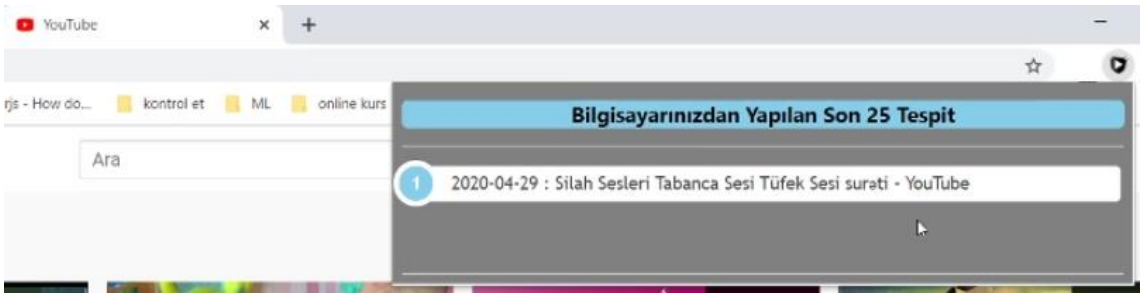
“permisson” kısmında kullanacağı özellikler belirtilmiştir.

“content_security_policy” kısmında Firebase ile çalışabilmemiz için gerekli kütüphanelerin yüklenmesini sağlayacak güvenlik izinleri verildi.

4.2.4.4 popup.html ve popup.js Dosyaları

“popup.html” dosyasında gerekli Firebase kütüphaneleri ve “popup.js” yüklendikten sonra tarayıcıda eklenti aktif olduğunda tıklandığı zaman açılacak pencere ile ilgili CSS kodları yazılmıştır. “body” etiketi içerisinde bir başlık etiketi eklendikten sonra o bilgisayardan yapılan tespitlerin listeleneceği bir sıralı liste etiketi yazılarak kod tamamlanmıştır.

“popup.js” dosyası içerisinde önce Firebase veritabanına bağlanacak tanımlamalar yapılmıştır. Veritabanı nesnesi oluşturulmuştur. Tarayıcının yerel depolamasından “kimlik” bilgisi okunarak o bilgisayardan yapılan son 25 tespitin tarih ve video başlığı olacak şekilde listelenmesi sağlanmıştır.



Şekil 15 popup.html Görüntüsü

4.2.4.5 content.js Dosyası

“manifest.json” dosyasında “content_script” bölümünde tanıtılan iki dosyadan kodlarını bizim yazdığımız dosya “content.js” dir. Diğer dosya “tfjslatest.js” dosyası Tensorflow JS’ in çalışması için gereken kaynak kütüphanedir.

“content.js” dosyasının ilk satırında “eventpage.js” gönderilen mesaj ile uygulama ikonunun aktif olması yani “popup.html” sayfasının görüntülenebilir olmasını söyler.
model, canvas ve context global değişkenleri tanımlandı. Adından da anlaşılacağı gibi model değişkeni içerisinde modelimiz tutulacaktır. Diğer iki değişken canvas oluşturulması ve video içeriğinin canvas’a yüklenmesini sağlamak için kullanılacaktır.

“eventPage.js” dosyasından gelecek mesajları dinlemek için bir listener oluşturuldu. Bu listener içerisinde ilk olarak dinlenen mesajdan bağımsız olarak modelin yüklenmesini sağlayan asenkron “loadModel()” fonksiyonu yazılmıştır.

Sonrasında if bloğu ile gelen mesaj eğer “modeliBaslat” ise bir sefer çalışacak şekilde modelin yüklenmesi ve canvas oluşturulması sağlanmıştır. Deneyler aşamasında canvas özellikle web sayfasının sol üst kısmında görünecek şekilde konumlandırılmıştır.

Eğer gelen mesaj “tanimlamaYap” ise öncelikle sayfanın URL’ içerisinde bir video id’si var mı kontrol edilir. Ardından sayfada video elementi seçilerek videonun oynatılması

durumu kontrol edilir. Eğer aktif ve oynatılan bir video varsa görüntüsü alınarak canvas' a yüklenir. "tahmin()" fonksiyonu çalıştırılır.

"tahmin()" asenkron bir fonksiyondur. Canvas içerisinde ki resim Tensorflow JS modele uygulanabilecek hale getirilir. Tahmin için kullanılacak sınıflar tanımlanır. Global "model" değişkeni kontrol edilerek modelin yüklenmesinin tamamlanması beklenir. Asenkron "predict()" fonksinyonu ile tahmin işlemi gerçekleştirilir. Sonuçlar tanımlanan sınıf isimleri ile map edilerek en büyük olasılık ve sınıf adı birer değişkene alınır. İzlenen videonun başlığı bir değişkene aktarılır. Eğer %90 üzerinde bir sınıf için olasılık varsa "eventPage.js"e bir mesaj gönderilir. Bu mesaj içerisinde işlemin ne olduğu (addToList), videonun id'si, videonun başlığı ve tespit edilen sınıf gönderilir. Sonrasında sayfa URL'i Youtube ana sayfasına yönlendirilir. En son kullanılan değişkenlerin içerikleri temizlenerek "content.js" dosyasında ki kodlar tamamlanır.

4.2.4.6 eventPage.js ve eventPage.html Dosyaları

"eventPage.html" dosyası içerisinde Firebase için gerekli kütüphaneler dahil edildikten sonra "eventPage.js" dosyası dahil edildi.

"eventPage.js" dosyası içerisinde önce Firebase veritabanına bağlanacak tanımlamalar yapılmıştır. Veritabanı nesnesi oluşturulmuştur.

Bir Youtube videosu yüklendiğinde URL' de videonun id' si yer almaktadır. Bu id daha önce "RealTimeDatabase" e kaydedildi mi kontrol edildi. Eğer yer alıyorsa bir bildirim (notification) gösterilelerek sayfanın URL' i youtube ana sayfaya yönlendirildi.

"content.js" ten gelecek mesajları dinlemek için bir listener oluşturuldu. Eğer gelen mesaj "showPageAction" ise eklentinin ikonu aktif olması sağlandı. "content.js" e "modelBaslat" mesajı gönderildi. 3 saniyede bir izlenen videodan görüntü alınması için "alarm" tanımlaması yapıldı.

Eğer "content.js" ten gelen mesaj "addToList" ise bu bir tespit yapıldığı anlamına gelir. Bir bildirim oluşturulur. Önce gelen paket veri içerisindeki video id'si veritabanında

kayıtlı mı kontrol edildikten sonra eğer id kayıtlı değilse o bilgisayara özel kimlik bilgisi ile kayıt edilmesi sağlandı.

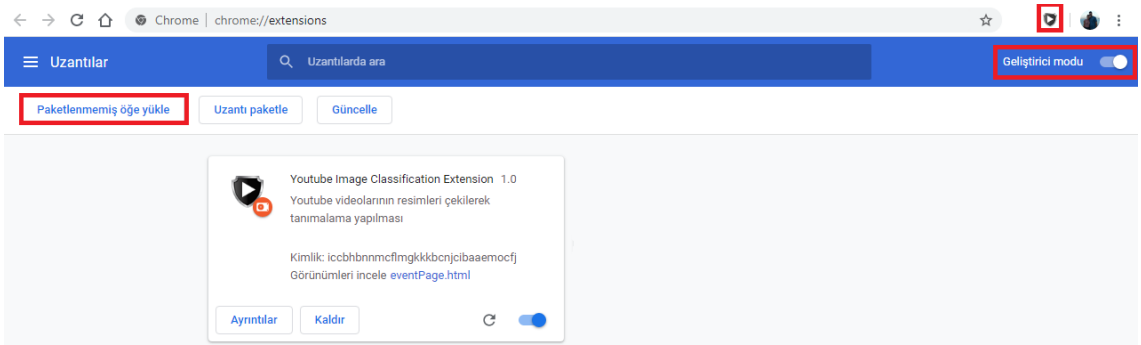
“RealTimeDatabase” yapısı gereği gerçek zamanlı çalışmaktadır. NoSQL ya da doküman tipi bir veritabanıdır. Kayıt edilen bilginin yapısı şu alanlardan oluşmaktadır: baslik, icerik, tarih, videoID ve yapan(kimlik bilgisini tutar).

3 saniyede bir çalışacak “alarm” ın kodları burada yazıldı. Eğer aktif tab "https://www.youtube.com/" adresi ile başlıyor ve adres sadece "https://www.youtube.com/" değilse “content.js” e “tanimlamaYap” mesajı gönderilcektir.

Eğer eklenti kurulduysa, güncellendiyse veya tarayıcı başlatıldıysa “kimlik()” adında benzersiz bir kimlik oluşturan bir fonksiyon çağırıldı. “kimlik()” fonksiyonu eğer local storage içerisinde kayıtlı bir kimlik bilgisi yoksa oluşturup local storage içerisine kaydedilmesi sağlandı.

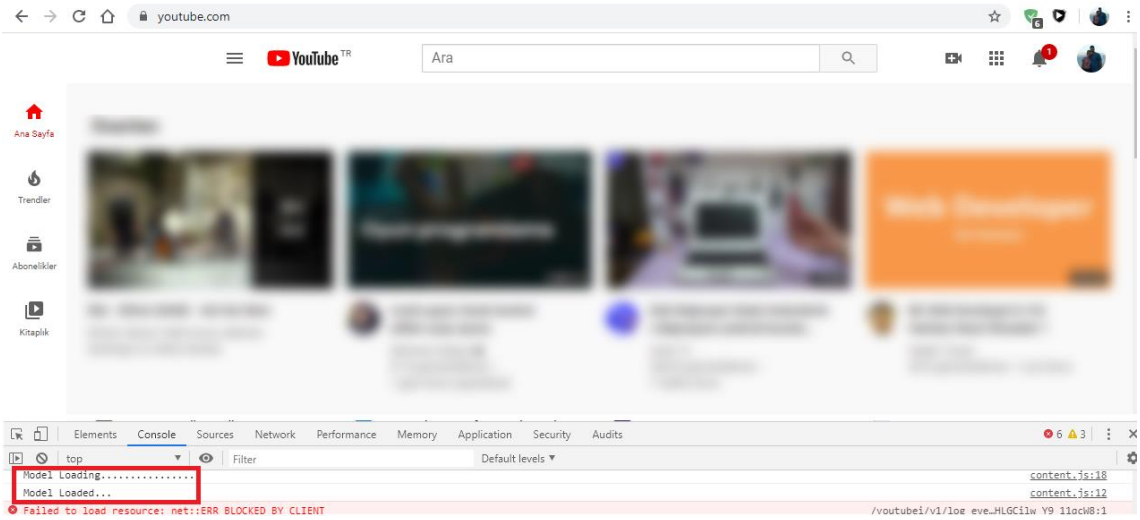
4.2.4.6 Eklentinin Çalıştırılması

Chrome da uzantılar penceresi açılır ve “Geliştiri modu” aktif edilir. “Paketlenmemiş öğe yükle seçilerek” eklentiyi içeren klasör seçilerek eklentinin yüklenmesi tamamlanmış olur. Eklentinin ikonu görünür hale gelecektir.



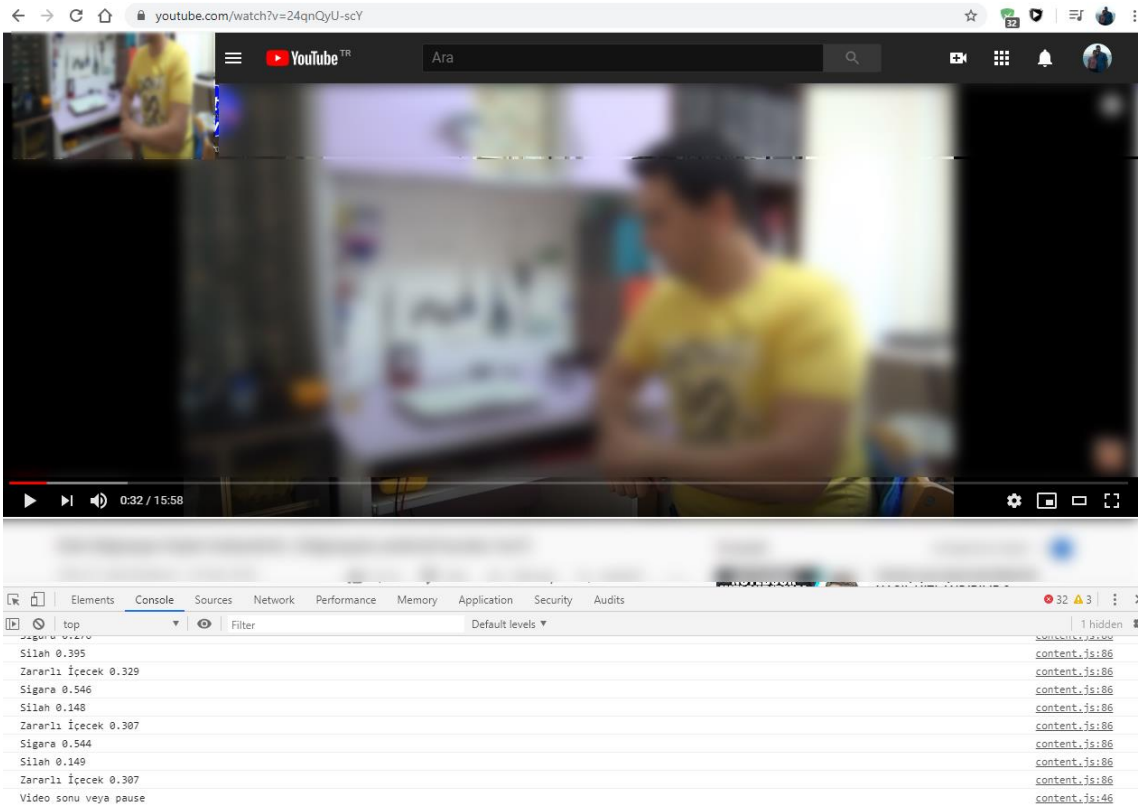
Şekil 16 Eklentinin Yüklenmesi

Eğer kullanıcı Youtube web sayfasına giriş yaparsa model arka planda yüklenecektir. Ama aktif video olmadığı için model tahmin yapmayacaktır. Bir video görüntülendiğinde model tahmin yapmaya başlayacaktır.



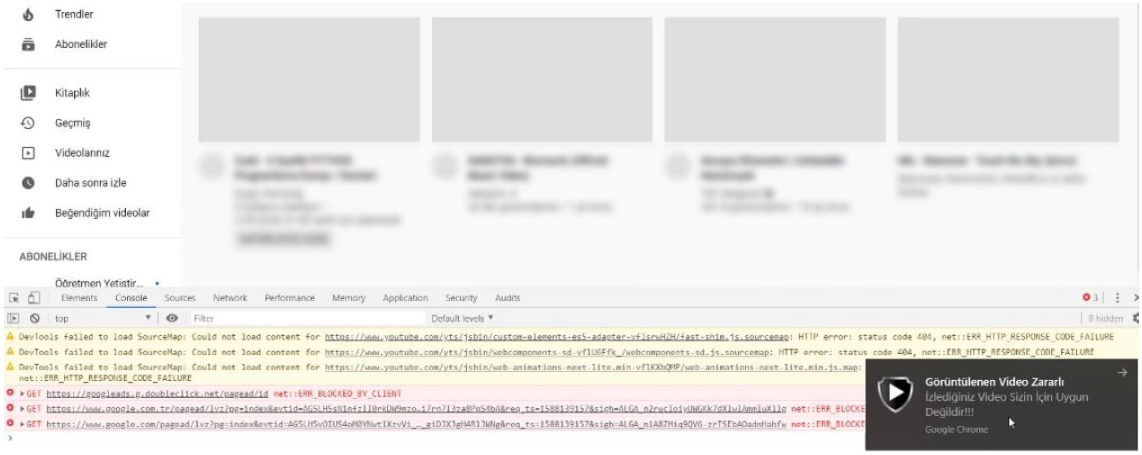
Şekil 17 Modelin Arka Planda Yüklenmesi

Bir video izlenmesi esnasında ekranın sol üst köşesinde çekilen görüntüler görülecek ve modelin tahminleri konsoldan takip edilecektir.



Şekil 18 Modelin Çalışması ve Tahminler

Model bir tespit yaptığında ya da daha önce veritabanına id'si kaydedilmiş bir videoyu görüntülediğinde youtube ana sayfa görüntülenecek ve kullanıcı bir bildirim ile uyarılacaktır.



Şekil 19 Tespit Yapılması

Eklentinin geliştirilmesi aşamasında modelin buluttan yüklenmesi yerine local sunucu kullanılarak geliştirme yapılmıştır. Yapılan birçok denemeler sonucunda yüksek parametre sayısına sahip modellerin örneğin "Resnet50V2" daha başarılı olduğu ancak local sunucuda dahi modelin yüklenme süresinin uzun olduğu ve gecikmelere sebep olduğu görülmüştür. Bir diğer dikkate alınması gereken konu da büyük boyutlu modellerin çalışma sırasında sayfada duraksamalara sebep olduğu da görülmüştür. Eğitilen custom modelin gerçek uygulamada başarımının düşük olduğu eğitim sırasındaki yüksek başarımın yanıltıcı olduğu tespit edilmiştir.

4.2.5 Mobil Uygulamanın Geliştirilmesi

Mobil uygulamanın kodları EK-F' de paylaşılmıştır. Mobil geliştirmeye bir Android Studio projesi açarak başlandı. "Phone and Tablet" sekmesinden "Empty Activity" seçildi. Proje adı, lokasyonu belirtildi. "Language" olarak Java seçildi. "Minimum SDK" olarak "API 22: Android 5.1(Lollipop)" seçildi. Bu sayede android ekosisteminde ki tüm cihazların %92,3' ünde uygulamanın çalışması sağlandı.

4.2.5.1 Modelin Tensorflow Lite' a Çevirilmesi

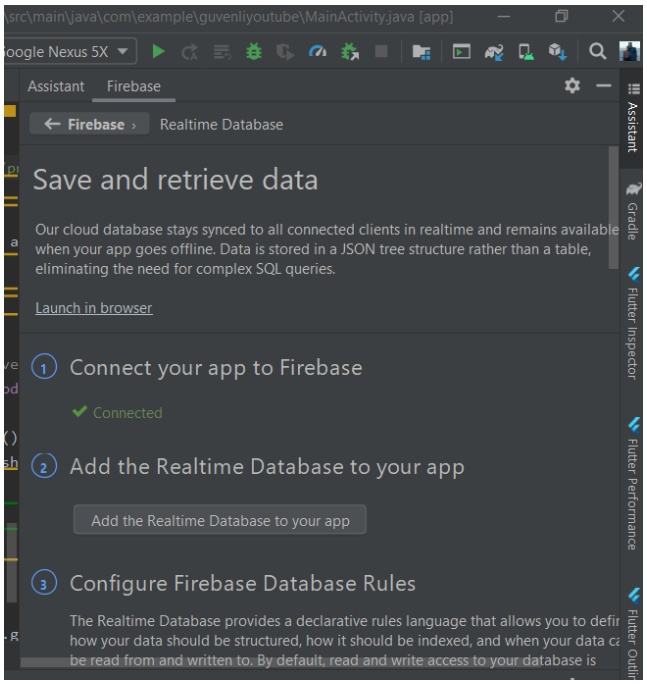
Eğitilen modelin mobil cihazlarda kullanılabilmesi için tensorflow Lite modeline dönüştürülmesi gerekmektedir. Öncelikle bu kısımda modelin eğitimi sırasında kullanılan "Google Drive File Stream" kullanılmamıştır. Çünkü modelin çevirilmesi sırasında yazılan kodun bulut sunucu içerisinde bulunan dosya yolunda boşluk karakteri içermesinden dolayı çalışmamasıdır. Bundan dolayı sanki local sabit diskimizde çalışıyormuşuz gibi çalışan yine drive'ımıza erişen bir kod yazılmıştır.

“os” kütüphanesi ile konum tam çevirilecek model dosyasının olduğu klasör konumana ayarlanmıştır.

“tflite_convert” komutu ile belitrilen keras modeli (h5 dosyası) çevirilmesi işlemi gerçekleştirilmiştir. Ön eğitimli modeller üzerinde yapılan çevirme işlemi esnasında EfficientNet çevirilememiş ve “Swish” aktivasyon fonksiyonu sebebiyle hata vermiştir. MobilenetV2, NasnetMobile, DenseNet121 ve Resnet50V2 modellerinde hatasız çevirme işlemi gerçekleşmiştir. Ayrıca Custom Model de sorunsuz olarak çevirilebilmiştir.

4.2.5.2 Firebase ile İlgili İşlemler

“Android Studio” da oturum açmak için Tools → Firebase →Açılan Pencereden “RealTime Databse” seçilir → “Connect your app to Firebase” seçilerek oturum açılır. Eklenti geliştirilmesi sırasında veritabanı oluşturulmuştu.

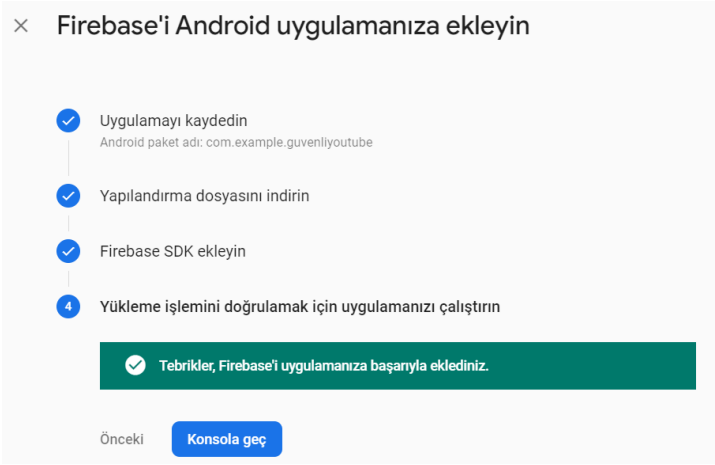


Şekil 20 Firebase’ e bağlanma

Sonrasında “Firebase console” giriş yaptığımızda yeni bir uygulama bağlandığı ile ilgili uyarı görünmektedir. “SDK kurulumuna devam et” denilerek “goole-services.json” dosyasını indirildi. Bu dosya projenin “app” klasörü altına yapıştırıldı.

“build.gradle” proje gradle dosyası içerisinde “repositories” altına “google()” eklendi. “dependencies” altına “classpath "com.android.tools.build:gradle:4.0.0-beta04"” satırı eklendi. “Sync Now” ile kütüphaneler projeye dahil edilir.

“build.gradle” modül gradle dosyası içerisinde dosyanın başına “apply plugin: 'com.google.gms.google-services'” satırı eklendi. Son olarak “implementation 'com.google.firebase:firebase-analytics:17.2.2'” satırı da eklendikten sonra yine “Sync Now” denilerek kütüphaneler eklendi. Uygulama çalıştırıldı ve “Firebase console” tekrar dönüldüğünde uygulamanın eklendiği ile ilgili mesaj görüldü.



Şekil 21 Firebase SDK’ nın eklenmesi

“Realtime Database” kullanmak için modül “build.gradle” içerisinde “dependencies” e “implementation 'com.google.firebase:firebase-database:19.3.0'” satırını eklendi “Sync now” ile kütüphane eklendi.

“ML Kit” hizmetini kullanabilmek için modül “build.gradle” içerisinde “dependencies” e “implementation 'com.google.firebase:firebase-ml-model-interpreter:22.0.3'” satırını eklendi. Model dosyasının sıkıştırılmasını engellemek için “android” süslü parantezleri arasına “aaptOptions {noCompress "tflite"}” kod satırı eklendi. “Sync now” ile kütüphane eklendi.

Model dosyasının uygulamaya dahil edilmesi için “main” klasörü altına “assets” klasörü eklendi. Model dosyası bu klasör altına kopyalanarak işlemler tamamlandı.

4.2.5.3 AndroidManifest.xml Dosyası

Bu dosyada uygulamada kullanılacak “INTERNET” erişimi ve “ACCESS_NETWORK_STATE” network durumu izinleri eklendi.

Uygulamanın görünecek ismi, logosu burada belirlendi. Logo için “res\mipmap\ic_launcher” klasörü altına gerekli resim dosyaları kopyalandı.

Uygulamanın teması burada “NoActionBar” seçilerek uygulama başlığı görüntülenmemesi sağlandı.

Uygulamanın açılış aktivitesi “SplashActivity” olarak belirlendi.

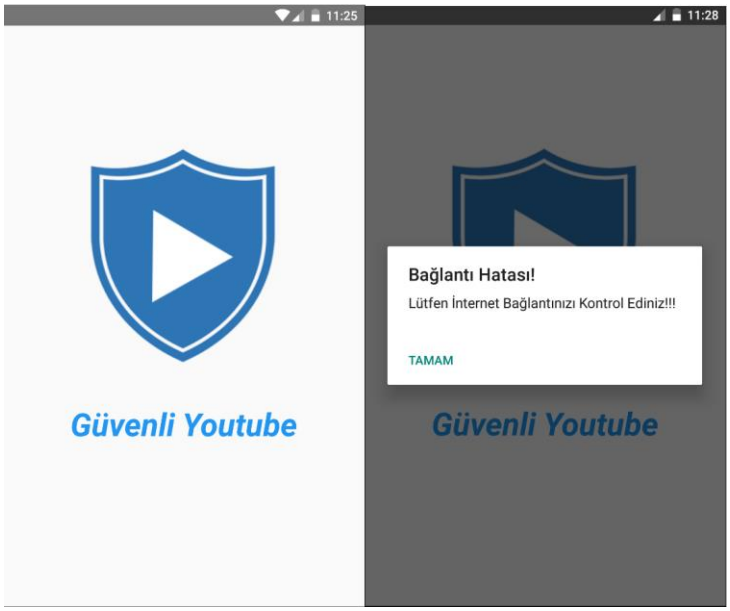
“MainActivity” ye “orientation|screenSize” özelliği verilerek cihaz yan çevrildiğinde uygulamada kullanılan “WebView” in yeniden yüklenmesi engellendi.

4.2.5.4 build.gradle (project) ve build.gradle (modul) Dosyaları

Bu dosyalarda yapılacak eklemeler Firebase ile ilgili işlemlerde anlatıldı. Genel olarak bu dosyalar ile projemizin kullandığı “SDK” (geliştirme yapılan android versiyonunu belirler), “gradle” (projenin derlenmesinden sorumludur) ve bağımlılıkların bilgisini içermektedir.

4.2.5.5 SplashActivity.java (Açılış Aktivitesi) Dosyası

Bu aktivitenin tasarımında sadece bir resim ve yazı bulunmaktadır. Asıl görev olarak cihaz aktif internet bağlantısına sahip mi kontrol edilir. Eğer internet yoksa bir uyarı mesajı verilerek uygulama durduruldu. Eğer cihaz internete bağlı ise 4 saniye sonra “MainActivity” e geçiş yapıldı.



Şekil 22 Splash Ekran Görüntüsü

4.2.5.6 Kayit.java Dosyası

Yapılan tespitlerin "RealTime Database" e kaydedilmesi için oluşturulan sınıftır. Alanları şunlardır: baslik, içerik, tarih, videoID ve yapan (tespiti yapan cihazın ID' si).

Şekil 23 RealTime Database Yapısı

4.2.5.7 RealTimeVeritabani.java Dosyası

Önce tarih bilgisi oluşturuldu. Sonrasında yeni bir "Kayit" nesnesi oluşturularak yapılan tespitin "RealTime Database" e kayıt edilmesi işlemi gerçekleştirildi.

4.2.5.7 CustomModel.java Dosyası

Model dosyamızın çalıştırıldığı ve tahminlerin yapıldığı kodlar burada işletilmektedir.

İlk olarak model dosyası “assets” klasöründen okundu ve bir “interpreter” oluşturuldu. Ardından “MainActivity.java” den gelen “bitmap” bilgisi bir metot ile “array”e çevrildi. Modelin tahmin sonuçları ayrı bir metot ile değerlendirilerek log’a yazdırıldı. Eğer yüksek değerli bir tahmin varsa izlenilen videonun veri tabanına kaydedilmesi sağlandı.

4.2.5.8 MainActivity.java Dosyası

Uygulamanın kullanıcı tarafından kullanılacak aktivitesi budur. Tasarımda “ActionBar” yoktur. Sadece tüm ekranı kaplayan bir “WebView” vardır.

Kodlarda öncelikle “CustomModel”, “RealTimeVeritabani”, “WebView” ve “SharedPreferences” nesneleri oluşturuldu. “SharedPreferences” uygulamada basit bir değer ya da birkaç değerin saklanması için kullanılan yöntemdir. Bu projede “WebView” içerisindeki URL değişiminin takibi için kullanıldı. Ayrıca bir video id’sinin veritabanında var olması ile ilgili bir kontrol değişkeni ve cihaz id’sini tutacak bir değişkende oluşturuldu.

“onCreate” metodu içerisinde “WebView” tanımlandı. Java Script etkin hale getirildi. “WebView” içerisine enjekte edilecek Java Script kodları için bir “WebClient” nesnesi oluşturuldu. “WebView” içerisinde çalışacak Java Script kodlarından veri okumak için “WebAppInterface” nesnesi oluşturuldu ve haberleşme için Java Script tarafında tanımlı fonksiyon adı belirlendi. Son olarak “WebView” in görüntüleyeceği URL atandı.

“SharedPreferences” kullanılarak bir “urlDegisim” değeri oluşturuldu içerisine varsayılan bir değer tanımlandı. Bu şekilde izlenen videoların değişimi takip edildi.

Yapılan tespitlerin kimin tarafından yapıldığı bilinebilmesi için son olarak mobil cihazın id’si tanımlanan değişkene aktarıldı.

“WebClient” sınıfı “WebViewClient” sınıfından implement edilmiştir. “shouldOverrideUrlLoading” metodu override edilerek “WebView” in ilk tanımlanan URL dışında başka URL’leri de görüntülenmesi sağlandı.

“onPageFinished” metodu override edilerek web sayfasının yüklenmesi tamamlandıktan sonra içerisine yazılan Java Script kodlarının enjekte edilmesi sağlandı.

Bu metod içerisindeki Java Script kodları ile önce bir web sayfasının sonuna yorumlardan hemen öncesine konumlanacak şekilde bir “canvas” elementi oluşturuldu. Bu elementin genişliği ve yüksekliği 224px olarak ayarlandı. Sonra 3 saniyede bir “snap()” fonksiyonun çalışması sağlandı. Bu metod ile URL içerisindeki video id’ si varsa video elementi aranır. Video elementi eğer çalışır durumda ise görüntü alınarak “canvas”a yerleştirildi. Ardından resim dataya dönüştürülerek “Android()” fonksiyonu ile video id’si, video başlığı ve resim datası gönderildi.

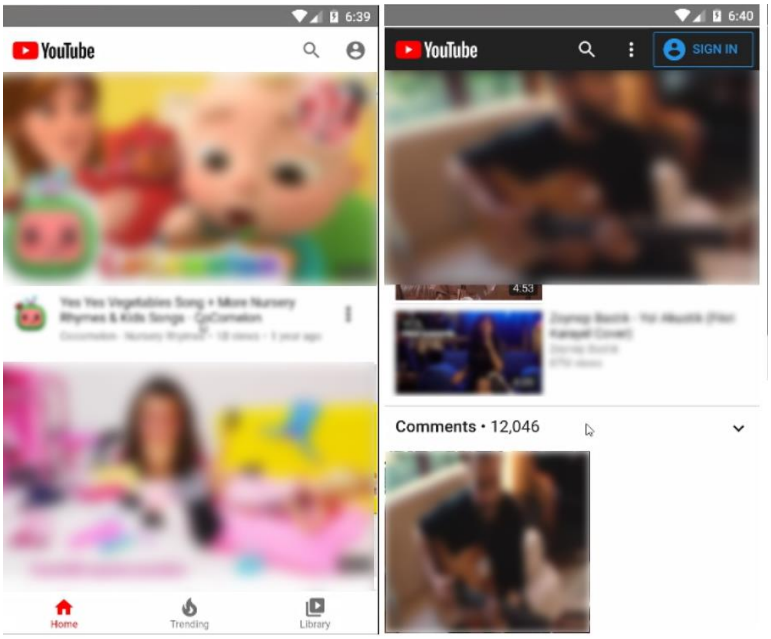
“WebAppInterface” sınıfında gelen bilgilerin değerlendirilmesi yapıldı. İlk olarak modelin yüksek tahmin yapması durumunu saklayacak değişken oluşturuldu. Ardından “SharedPreferences” kullanılarak URL değişimi kontrol edildi. Eğer değişim varsa video id’ si kaydedildi. Yeni video id’si “RealTime Database” de kayıtlı olup olmadı kontrol edildi. Kayıtlı ise bir “Toast” mesajı gösterilerek Youtube ana sayfa görüntülendi. Firebase yapısı gereği asenkron çalışmasından dolayı yapılan tespitin bir den fazla kez veri tabanına kaydedilmemesi için “MainActivity”nin başında oluşturulan bir kontrol değişkeni kullanıldı.

Java Script’ten gelen resim bilgisinin başlığı kaldırılmasının ardından “bitmap”e dönüştürüldü. Verinin modelin değerlendirmesi sonucunda bir tespit yapıldıysa ve veritabanın da kayıtlı olmadığı kontrolü tamamlandıysa veri tabanına kayıt işlemi gerçekleştirilir. Bir “Toast” mesajı gösterilir ve Youtube ana sayfası görüntülenmesi sağlanır.

4.2.5.9 Mobil Uygulamanın Çalıştırılması

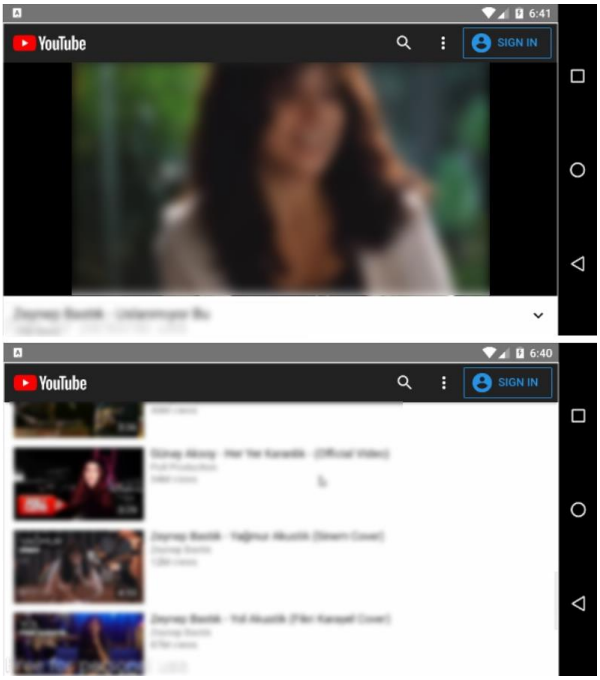
Uygulamamızın test edilmesi için “GenyMotion” programında oluşturulan sanal cihazlar kullanıldı. Ayrıca fiziksel cihaz üzerinde de testler yapıldı. Farklı sanal cihazların denenmesi sonucunda Android versiyonu 8 ve üzeri cihazlarda sorunsuz çalıştığı gözlendi. Uygulama Android 5.1 ve üzeri versiyonlarda çalışacak şekilde programlanmasına rağmen emulatörde videoların yürütülmemesi sorunu oluştu. Ancak fiziksel cihazda sorun olmadı görüldü.

Uygulamanın çalıştırılması ile “Splash” ekran görüntülendi. Sonrasında “MainActivity” e geçildi.



Şekil 24 Uygulama Çalışma Görüntüsü Dikey

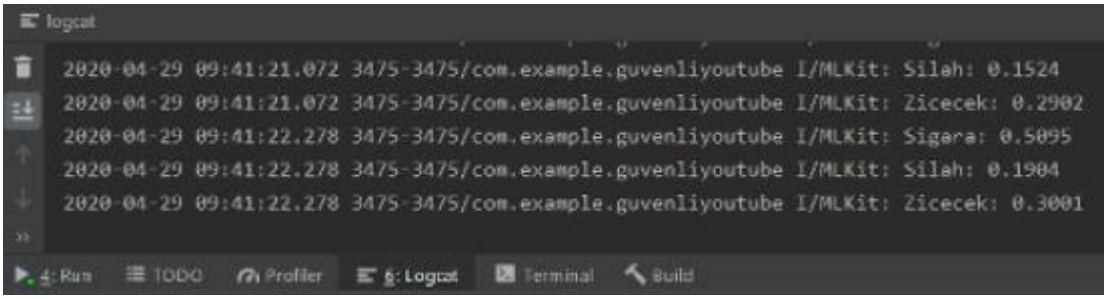
Şekil 24’ de sol tarafta uygulamanın ilk çalışma ekranı görülmektedir. Sağ tarafta ise bir videoya tıklanması durumunda videonun oynatılması ve sayfanın en alt kısmında “canvas” görülmektedir.



Şekil 25 Uygulama Çalışma Görüntüsü Yatay

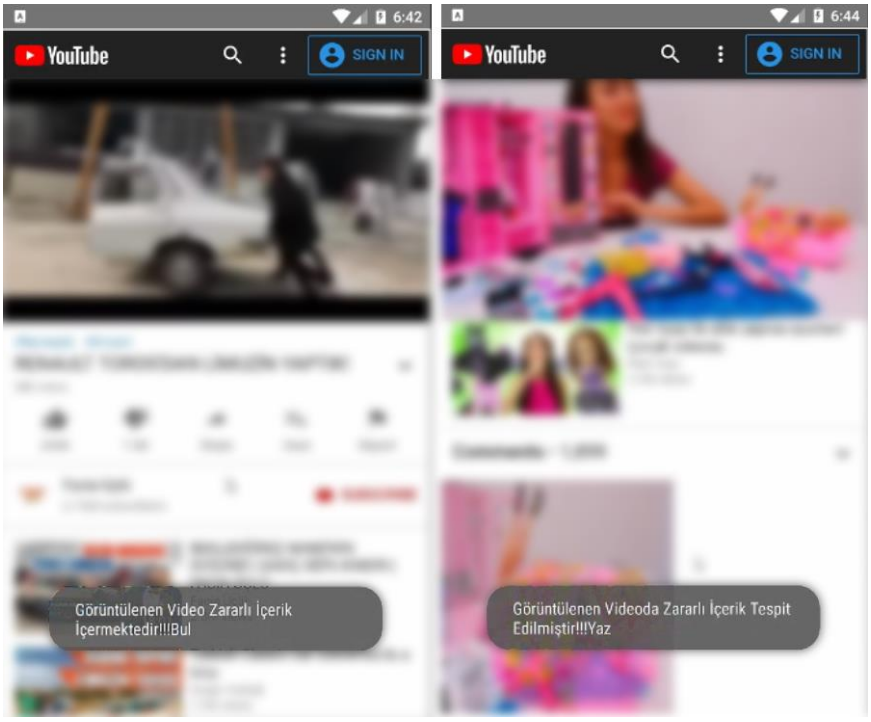
Şekil 25’ te uygulamanın yatay durumda çalışması görünmektedir. Üstte video oynatması ve altta da arama sonuçları veya ilk açılış ekranı görünmektedir.

Uygulamada bir video izlenirken arka planda model resimler üzerinde tahminler yapmaktadır. Bu tahminler “logcat” üzerinden görüntülenmektedir.



Şekil 26 Modelin Tahminleri

“logcat” üzerinde ayrıca veritabanı ile ilgili işlemler içinde mesajlar yazdırılmıştır. Model yüksek değerli bir tahmin yaptığında kullanıcıya bir mesaj gösterip “WebView” Youtube ana sayfaya yönlendirilecektir.



Şekil 27 Tespit Yapılması

Şekil 27’ te sol tarafta daha önceden veri tabanında kayıtlı bir video açıldığında tespit edilmesi görülmektedir. Sağ tarafta veri tabanına kayıtlı olmayan bir video da modelin görüntüde istenmeyen içerik tespit yapması görünmektedir. Uyarı mesajının

görünmesinden yaklaşık bir saniye sonra Youtube ana sayfaya yönlendirme işlemi gerçekleşmektedir.

5. TARTIŞMA VE SONUÇ

Bu çalışmada CNN ile bir model eğitilerek bu modelin iki farklı uygulamada çalıştırılması hedeflenmiştir.

Hazırlık aşamasında model için gerekli dataset oluşturulurken hazır kaynaklardan faydalanmanın yanı sıra mümkün olduğunca arama motorlarından da faydalanılmıştır. Bu çalışma sırasında bir kez daha ispatlanmıştır ki bilgi günümüzde en önemli kaynaktır. Ne kadar uğraşılırsa uğraşılsın bir noktadan sonra veri setinin daha fazla geliştrilmesi mümkün olmamıştır. Bu durum ile model eğitimi sırasında veri seti üzerinde data augmentation (veri artırma) yapılarak çözülmeye çalışılsada çok büyük bir iyileştirme sağlanamamıştır. Sonuç olarak dataset içerisinde ki resim sayıları yeterli düzeye ulaştırılamaması sonra ki aşamalarda modelin yeterince başarılı sonuçlar verememesine sebep olmuştur.

Modelin geliştirilmesi aşamasında iki metot benimsenmiştir. Bunlardan ilki sıfırdan geliştirilen model de farklı parametreler ile yapılan denemelerde ağ büyüklüğünün özellik çıkarma konusunda avantaj sağladığı görülmüştür. Yine çıkış katmanı için ağ büyüklüğünün artırılması olumlu sonuçlar vermiştir. Ancak uygulamalarda kullanılacak modelin çok fazla parametreye sahip olması hem işlem yükünü artıracağı hem de modelin dosya boyutunu büyüteceği için parametre sayısı başarım arasında bir denge kurulması gerekmiştir.

Modelin geliştirilmesi için benimsenen bir diğer metot transfer öğrenme olmuştur. Burada Keras' ın ön eğitimli modelleri kullanılarak yapılan deneylerin sonuçları daha önce ki kısımlarda paylaşılmıştır. Transfer öğrenme kesinlikle özellik çıkartma tarafında çok daha başarılıdır. Hazırlanan farklı çıkış katmanları ile yapılan deneylerde parametre

sayısını çok fazla artırmadan yapılan deneyler ile sıfırdan eğitilen modelden daha başarılı olunduğu görülmüştür.

Her iki metot ile yapılan farklı optimizer deneylerinin sonuçları önceki kısımlarda paylaşılmıştır. Optimizer'in model eğitiminde maksimum başarıma ulaşılması ve kaç epoch ile maksimuma ulaşılacağı konusunda önemli bir rol oynadığı görülmüştür.
Epoch sayısı modelin eğitiminde bir diğer önemli parametre olmaktadır. Bu sayı içinde bir denge yakalanması gerekmektedir. Çok az epoch ile model yeterince eğitilemezken, epoch sayısının çok fazla olması belirsiz bir epoch sayısından sonra modelin dataseti ezberlemesine sebep olabilmektedir. Ayrıca bu durum başarımın yatay bir eğriye dönüşmesi sonucu gereksiz işlem yapılmasına ve eğitim süresinin gereksiz yere uzamasına sebep olabilmektedir.

Modelin eğitilme ortamı olarak "Google Colab" kullanılmıştır. Bunun en önemli sebebi ücretsiz olarak GPU kullanımı sağlamasıdır. Bir diğer önemli sebep her hangi bir bilgisayara bağımlı kalmaksızın eğitimin gerçekleştirilmesidir. Dataset yine bir diğer google hizmeti olan "Google Drive" üzerinde barındırılarak çalışmaya çok büyük esneklik katılmıştır. Bulut olarak bu hizmeti veren farklı çözümler de incelenmiştir. Daha fazla özelliklere sahip olmalarına rağmen ücretli talep edilmesi veya ücretsiz verilen hizmetlerin çok sınırlı olması sebebiyle en iyi ücretsiz hizmeti vernin "Google Colab" olduğu görülmüştür.

Geliştirilen uygulamalar için kullanılan veri tabanı çözümü Firebase' in "RealTime Database" i olmuştur. Bu hizmetin tercih edilmesinin iki sebebi vardır. Birincisi gerçek zamanlı olması ve çok basit bir şekilde back end işlerini halletmesidir. İkincisi ise yine geliştirme aşamasında tamamen ücretsiz olarak kullanabilmemizdir.

Geliştirilen iki uygulamadan ilki tarayıcı eklentisidir. Tarayıcı üzerinde CNN modelini kullanabilmemizi sağlayan teknoloji tensorflow JS'dir. Uygulamamız temelde izlenen videodan resimler almakta ve değerlendirme yapmaktadır. Değerlendirme sonucunda istenmeyen içerik tespiti durumunda videonun bilgileri bulut veritabanı kaydedilmekte ve kullanıcı uyarılarak başka bir sayfaya yönlendirilmektedir. Eğer veri tabanında

kayıtlı bir video görüntülenmek istenirse yine kullanıcı uyarılmakta ve başka bir sayfaya yönlendirilmektedir.

Eklentinin kodları sorunsuz bir şekilde çalışmaktadır. Eklenti de kullanılan modelin başlangıçta yüklenmesi bir süre almaktadır. Ayrıca çok fazla parametreye sahip modeller sistem tamamen asenkron çalışmasına rağmen yine de video görüntüsünde sekmelere sebep olduğu gözlemlenmiştir. Bundan dolayı Custom Modelimiz, MobileNetV2 ve DenseNet121 ön eğitimli modelleri uygulanabilir kabul edilmektedir. Yapılan deneylerde başarımın iyi olmadığı gözlenmiştir.

Yapılan araştırmalar sonucunda daha önce tarayıcı eklentisinde makine öğrenmesi kullanan az sayıda eklenti olduğu görülmüştür. İncelemeler derinleştirildiğinde eklentilerin genelinin node js ile kullanıldığı tespit edilmiş olup çalıştırılmalarının zahmetli ve sorunlu oldukları tespit edilmiştir. Bu çalışmada geliştirilen eklentide modelin depolanmasının Firebase “Storage” hizmetinde yapılması kullanım ve güncelleme kolaylığı sağlamaktadır. Ayrıca video akışından resim alarak değerlendirilme yapılmasının benzersiz bir özellik olduğu da görülmüştür.

Geliştirilen iki uygulamadan ikincisi mobil uygulamadır. Makine öğrenmesinin mobil cihazlarda kullanılabilmesini sağlayan teknoloji tensorflow Lite’tır. Uygulama tarayıcı eklentisi ile aynı mantıkta çalışmaktadır. Uygulamada hala beta sürecinde olan Firebase’in “ML Kit” hizmeti kullanılmıştır. Modelin bulut olarakta depolanmasını sağlayan bu hizmette sadece local olarak çalışacak şekilde kodlama yapılmıştır. Mobil veriden tasarruf edilmesi düşünülmüştür.

Uygulamanın kodları sorunsuz bir şekilde çalışmaktadır. Tensorflow Lite Beta sürecinden çıkalı çok fazla zaman geçmemiştir. Eğitilen modeller ile yapılan deneylerde başarımın iyi olmadığı gözlenmiştir.

Yapılan araştırmalar sonucunda mobil dünyada makine öğrenmesinin çok yeni olduğu görülmüştür. Bu çalışmada geliştirilen uygulama ile benzer mantıkta çalışan herhangi bir uygulama görülmemiştir.

Bu çalışma süresince makine öğrenmesi dünyasının ne kadar dinamik ve hızla ilerleyen yapıda olduğu da deneyimlenmiştir. Keras framework ile başanılan model eğitimi "Google Colab" in güncelleme alması ile tf.keras (tensorflow) ile devam edilmiştir. Yakın zamanda stabil olmayan "nightly" sürümü ile ön eğitimli modellere eklemeler yapılmıştır. Ayrıca model için dataset'i hazırlayan yeni metot eklenmiş ve bu çalışmada modelin eğitiminde kullanılan metotların yakında yeni versiyon da kullanılmayacağı duyurulmuştur. Tüm bunlar 6 ay gibi kısa bir sürede gerçekleşmiştir. Uzun vadede makine öğrenmesinde daha başarılı sonuçlar alınacağı ve hayatımızı daha fazla kolaylaştıracağı düşünülmektedir.

Sonuç olarak geliştirilen uygulamaların testleri göstermiştir ki yeterli bir dataset ile eğitilecek model tarayıcıda ve mobil cihazlarda sorunsuz olarak çalışacak ve videolar üzerinde sınıflar tespit edebilecektir. Bu proje ile CNN'i daha önce kimsenin kullanmadığı bir şekilde kullanarak yeni bir vizyon oluşturulmuştur. Sonraki yapılacak çalışmalara ilham kaynağı olunacağı düşünülmektedir.

KAYNAKLAR

1. Artificial intelligence [online]
   https://www.britannica.com/technology/artificial-intelligence#ref219078

2. What is AI? [online]
   https://course.elementsofai.com/1/1

3. Reinsforment Lerning, 2019 [online]
   https://www.tutorialspoint.com/machine_learning/machine_learning_tutorial.pdf

4. Fatih Erikli, 2019, *Yapay Sinir Ağları*[online]
   https://medium.com/@fthrkl/yapay-sinir-aglari-27e50fd83267

5. Numpy Tutorials [online]
   https://numpy.org/devdocs/user/quickstart.html

6. Pandas Tutorials [online]
   https://pandas.pydata.org/pandas-docs/stable/

7. Mathplotlib Tutorials [online]
   https://matplotlib.org/3.1.1/contents.html

8. Seaborn Tutorias [online]
   https://seaborn.pydata.org/tutorial.html

9. E. Kaan Ulgen, 2017, *Makine Öğrenimi En Yakın Komşuluk* [online]
   https://medium.com/@k.ulgen90/makine-öğrenimi-bölüm-2-6d6d120a18e1

10. E. Kaan Ulgen, 2017, *Makine Öğrenimi Karar Ağaçları* [online]
    https://medium.com/@k.ulgen90/ makine- öğrenimi-bölüm -5-c90bd7593010

11. Şadi Evren ŞEKER, 2008, *Support Vector Machine* [online]
    http://bilgisayarkavramlari.sadievrenseker.com/2008/12/01/svm-support-vector-machine-destekci-vektor-makinesi/

12. Erdinç UZUN, *K-Means Algoritması* [online]
    https://erdincuzun.com/makine_ogrenmesi/kumeleme-algoritmalari-k-means-algoritmasi/

13. T.Tugay BİLGİN, *Kümeleme Analizi* [online]
    http://www.ttbilgin.com/2018-2019-guz/veri-bilimine-giris/hafta7/kumeleme-analizi.pdf
14. Aaron Hertzmann, David J. Fleet and Marcus Brubaker, 2015, *PCA Principal Component Analysis* [online]
    http://www.cs.toronto.edu/~mbrubake/teaching/C11/Handouts/PCA.pdf

15. Deep Learning [online]
https://en.wikipedia.org › wiki › Deep_learning

16. Sinir Hücresi [online]
https://tr.wikipedia.org/wiki/Sinir_hücresi

17. Ayyüce KIZRAK, 2019, *Yapay Sinir Ağları* [online]
https://medium.com/@ayyucekizrak/su-kara-kutuyu-açalim-yapay-sinir-ağları-7b65c6a5264a

18. Yrd.Doç.Dr. Altınç Yılmaz, 2018, *Yapay Zeka*, Syf 73

19. Ayyüce KIZRAK, 2019, *Aktivasyon Fonksiyonlarının Karşılaştırması* [online]
https://medium.com/@ayyucekizrak/derin-öğrenme-için-aktivasyon-fonksiyonlarının-karşılaştırılması-cee17fd1d9cd

20. Sergey Ioffe, Christian Szegedy, 2015, *Batch Normalization: Accelerating Deep Network Training b y Reducing Internal Covariate Shift* [online]
https://arxiv.org/pdf/1502.03167.pdf

21. N.Srivastava, G.Hinton, A.Krizhevsky, I.Sutskever, R.Salakhutdinov, *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*, [online]
http://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf

22. Chandra Churh Chatterjee, 2019, *How a classic CNN (Convolutional Neural Network) work?* [online]
https://towardsdatascience.com/basics-of-the-classic-cnn-a3dce1225add

23. Chrome Extension Tutorial [online]
https://developer.chrome.com/

24. Firebase Documentation [online]
https://firebase.google.com/docs/database/web/

25. Jupyter Notebook
https://jupyter.org/

26. Han LEE, *Cigarette Smoker Detection* [online]
https://www.kaggle.com/vitaminc/cigarette-smoker-detection

27. Sai SASANK, *Guns Object Detection* [online]
https://www.kaggle.com/issaisasank/guns-object-detection

28. Khalil NOUISSER, *Alcoholic Drinks* [online]
https://www.kaggle.com/khalilnouisser/alcoolicdrinks5

29. Gabriel OSTROLUCKY, 2020, *Bulk Bing Image Downloader* [online]
https://github.com/ostrolucky/Bulk-Bing-Image-downloader

30. Bahasa Indonesia, 2019, *How to Fix CORS Issue in Firebase or Google Cloud Storage* [online]
https://jefrydco.id/en/blog/fix-cors-issue-firebase-google-cloud/

31. Keras Applications [online]
https://keras.io/api/applications/

EKLER

EK-A

KERAS MEVCUT MODELLER

Tablo 3 Keras' ta mevcut State of Art Modeller [31]

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-
EfficientNetB0	29 MB	-	-	5,330,571	-
EfficientNetB1	31 MB	-	-	7,856,239	-
EfficientNetB2	36 MB	-	-	9,177,569	-
EfficientNetB3	48 MB	-	-	12,320,535	-
EfficientNetB4	75 MB	-	-	19,466,823	-
EfficientNetB5	118 MB	-	-	30,562,527	-
EfficientNetB6	166 MB	-	-	43,265,143	-
EfficientNetB7	256 MB	-	-	66,658,687	-

EK-B

CUSTOM MODEL EĞİTİM KODLARI

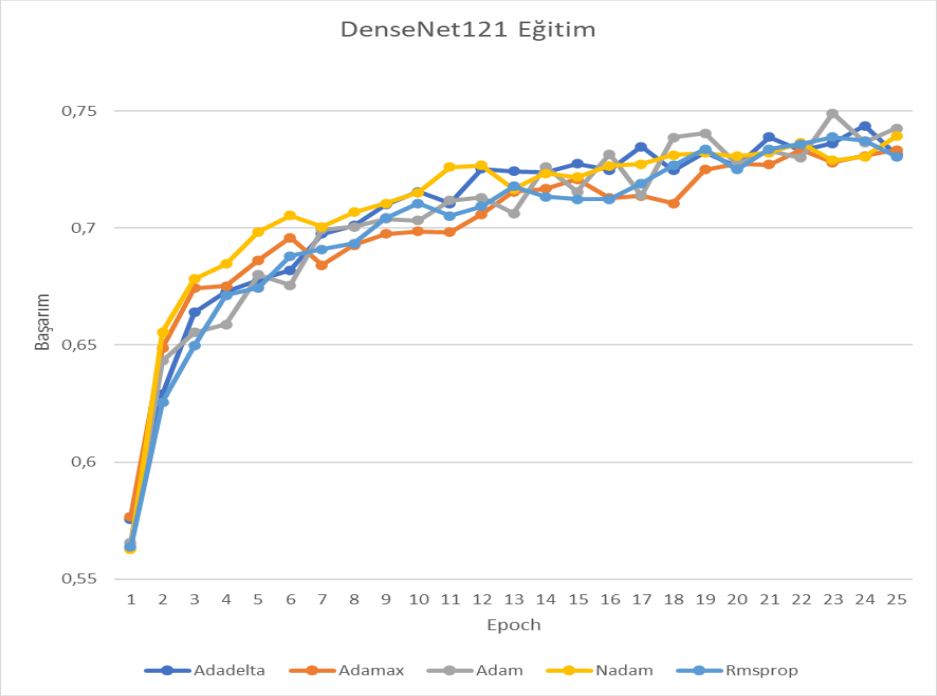
```
from google.colab import drive
drive.mount('/content/drive')

import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense, Batc
hNormalization

train_dir = '/content/drive/My Drive/datasets/UcClass/train'
validation_dir = '/content/drive/My Drive/datasets/UcClass/val'

model = Sequential()
model.add(Conv2D(256, (2, 2), input_shape = (224, 224, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size =(2, 2)))
model.add(BatchNormalization())

model.add(Conv2D(512, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size =(2, 2)))
model.add(BatchNormalization())

model.add(Conv2D(1024, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size =(2, 2)))
model.add(BatchNormalization())

model.add(Conv2D(512, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size =(2, 2)))
model.add(BatchNormalization())
```

```
model.add(Conv2D(256, (2, 2)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size =(2, 2)))
model.add(BatchNormalization())

model.add(Flatten())
model.add(Dense(256))
model.add(Activation('relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(Dense(256))
model.add(Activation('relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(Dense(256))
model.add(Activation('relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(Dense(3))
model.add(Activation('softmax'))

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest')

train_generator = train_datagen.flow_from_directory(
    train_dir,
    batch_size=32,
    class_mode='categorical',
    target_size=(224,224))

validation_datagen = ImageDataGenerator(rescale=1./255)
```

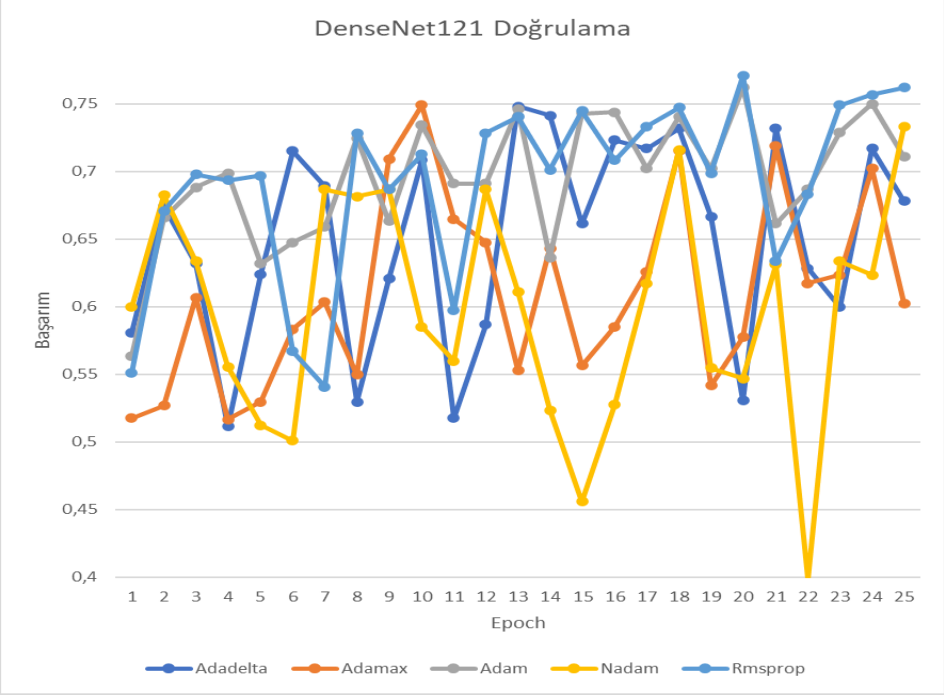
```
validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    batch_size=32,
    shuffle=False,
    class_mode='categorical',
    target_size=(224,224))

optimizer = tf.keras.optimizers.RMSprop(lr=1e-4)
model.compile(loss='categorical_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])

model.summary()

history = model.fit_generator(
    generator=train_generator,
    validation_data=validation_generator,
    steps_per_epoch=train_generator.n//train_generator.batch_size,
    validation_steps=validation_generator.n//validation_generator.batch_size,
    epochs=25)

model.save('/content/drive/My Drive/datasets/UcClass/customModel6RmspropE25.h
5')

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model doğruluğu')
plt.ylabel('Doğruluk')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model kaybı')
plt.ylabel('Kayıp')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

EK-C

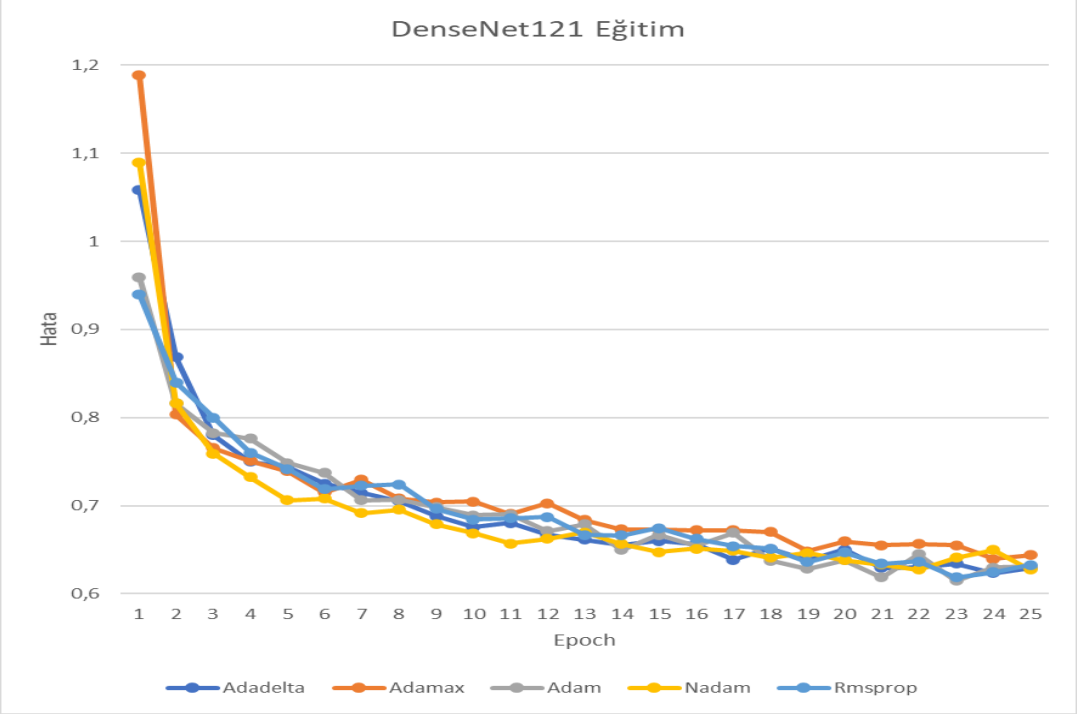
MODEL EĞİTİMİNDE OPTİMİZERLERİN EĞİTİM SÜRECİNE ETKİSİ



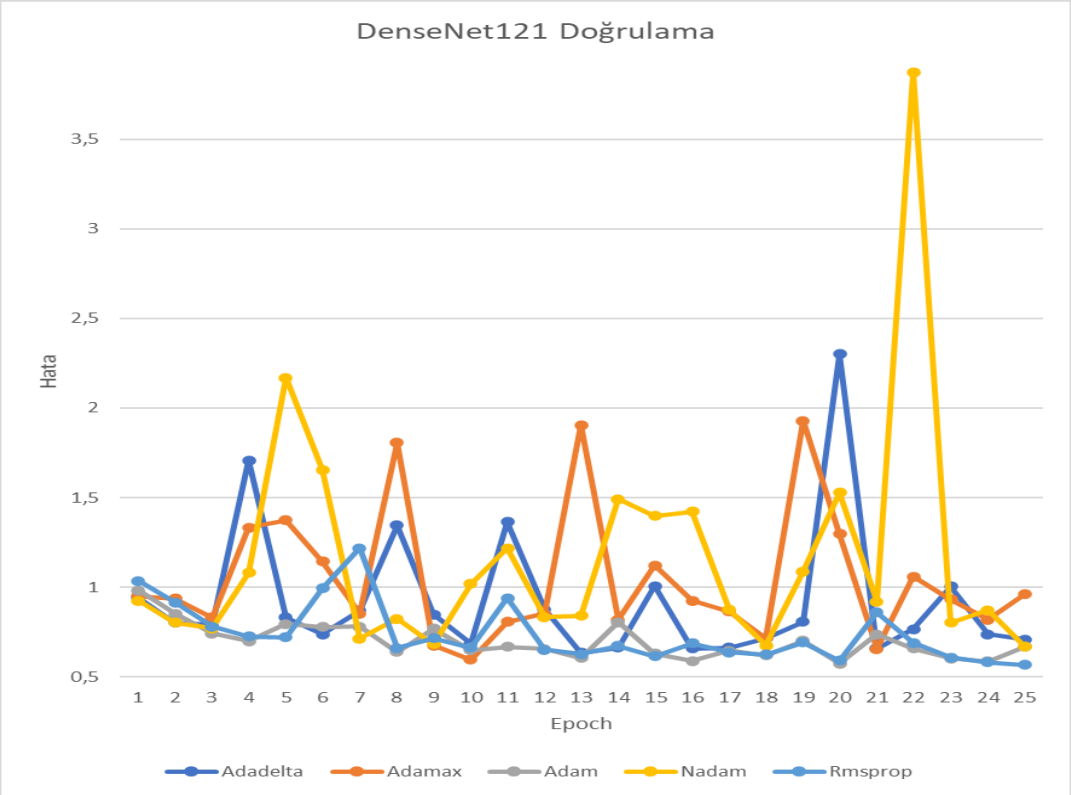
Şekil 28 DenseNet121 Eğitim Optimizer Başarım Grafiği



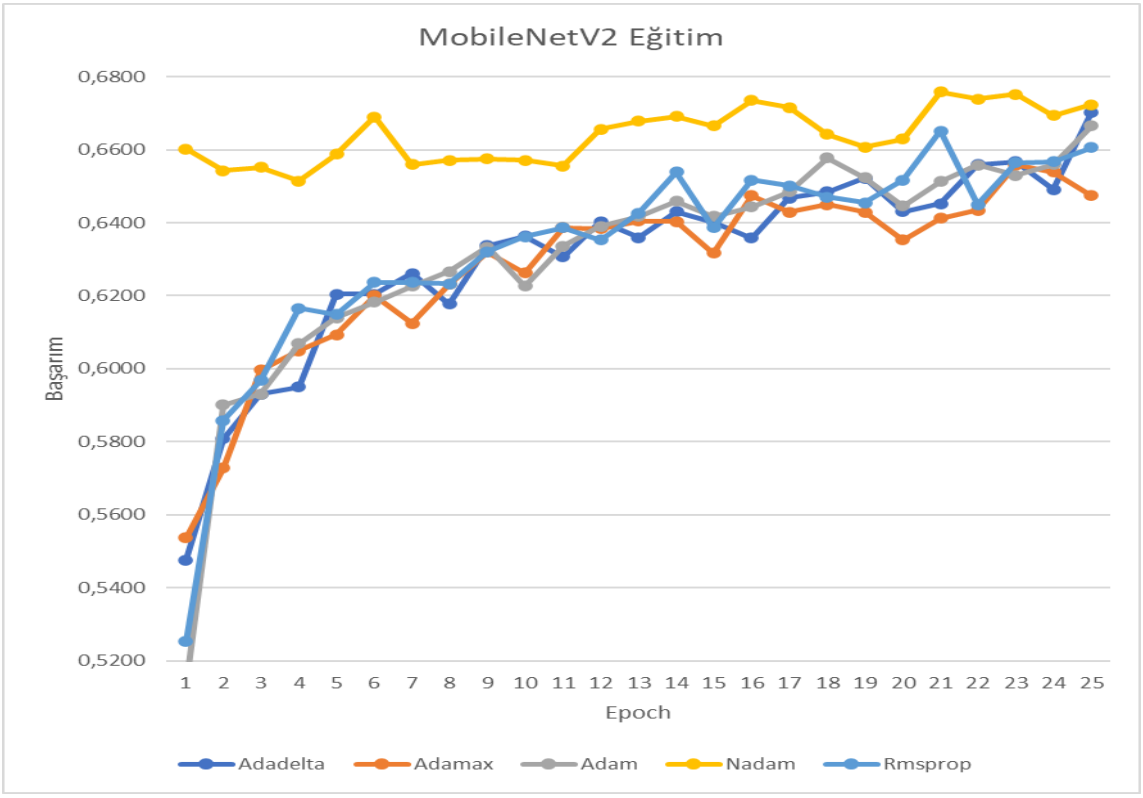
Şekil 29 DenseNet121 Doğrulama Optimizer Başarım Grafiği



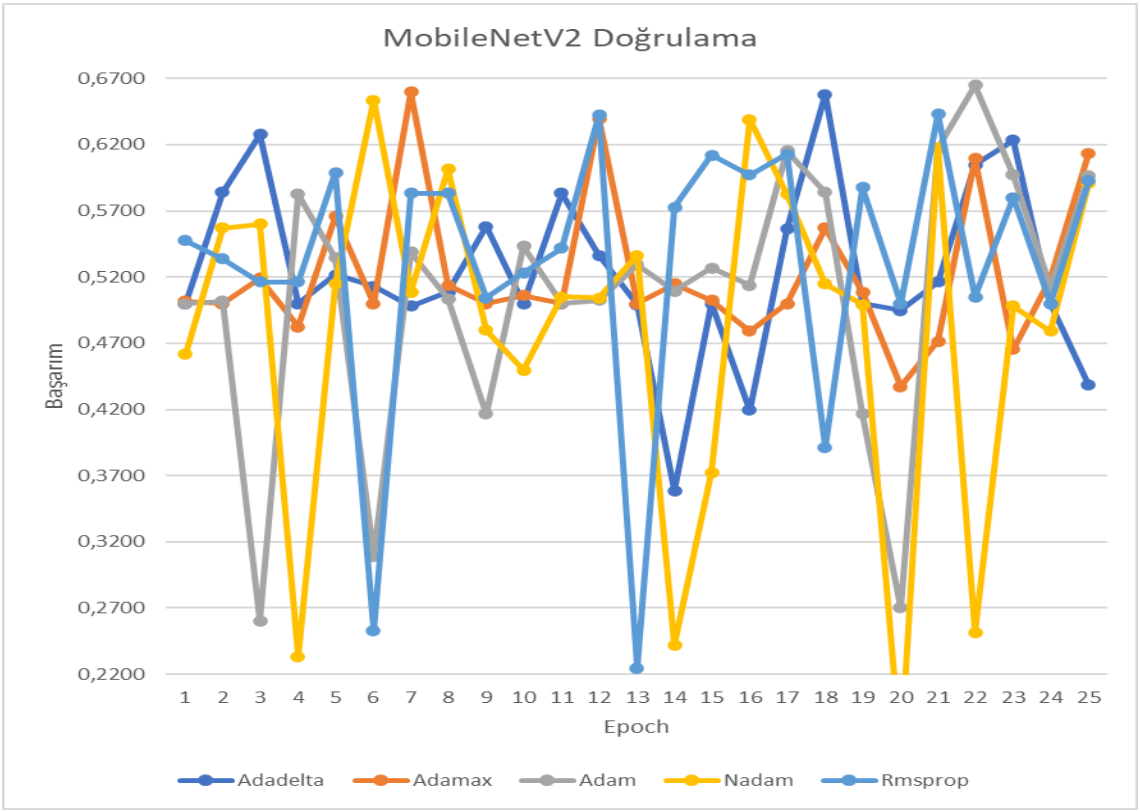
Şekil 30 DenseNet121 Eğitim Optimizer Loss Grafiği



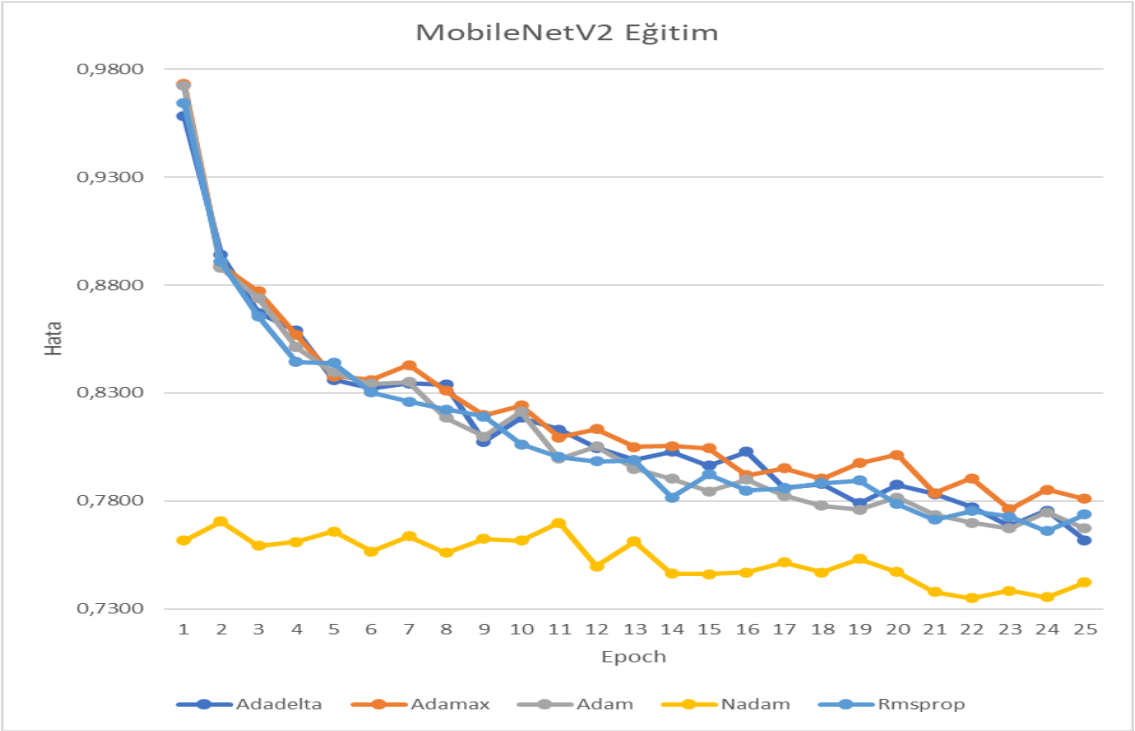
Şekil 31 DenseNet121 Doğrulama Optimizer Loss Grafiği



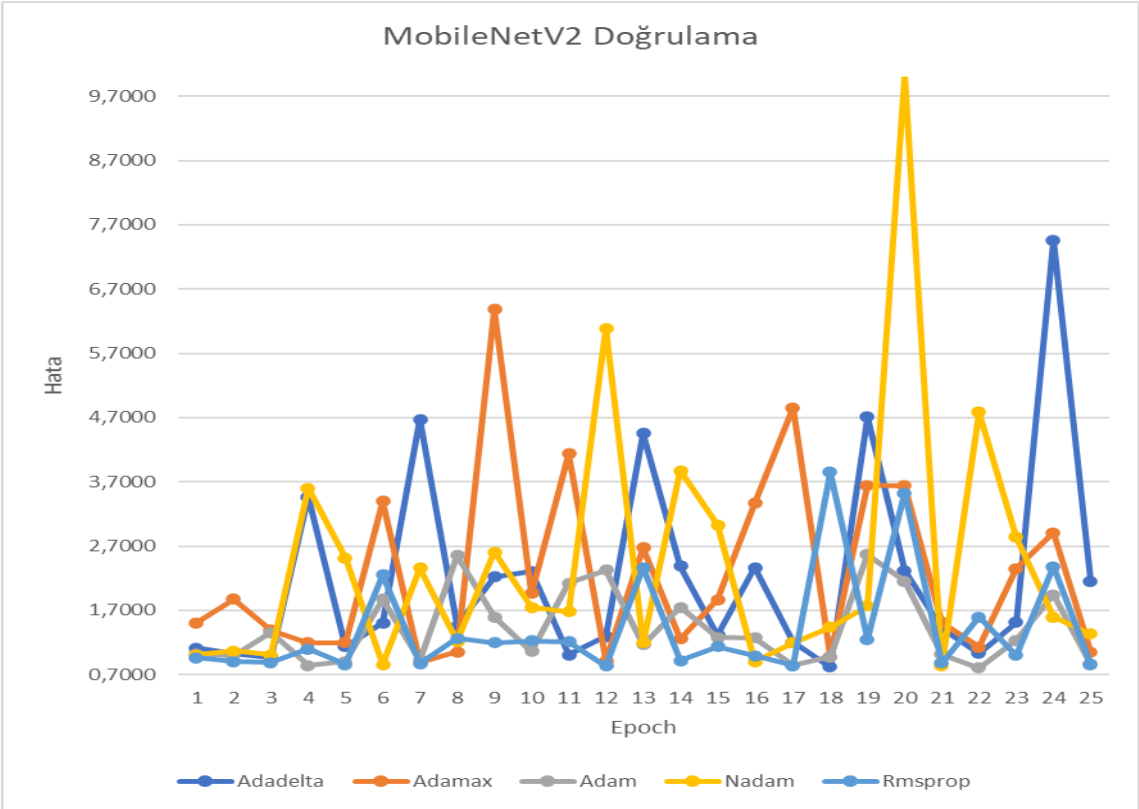
Şekil 32 MobileNetV2 Eğitim Optimizer Başarım Grafiği



Şekil 33 MobileNetV2 Doğrulama Optimizer Başarım Grafiği



Şekil 34 MobileNetV2 Eğtim Optimizer Loss Grafiği



Şekil 35 MobileNetV2 Doğrulama Optimizer Loss Grafiği

EK-D

TRANSFER ÖĞRENME MODEL EĞİTİM KODLARI

```
from google.colab import drive
drive.mount('/content/drive')

import numpy as np
import matplotlib.pyplot as plt

import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras import Model, layers

train_dir = '/content/drive/My Drive/datasets/UcClass/train'
validation_dir = '/content/drive/My Drive/datasets/UcClass/val'

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest',
    preprocessing_function=preprocess_input)

train_generator = train_datagen.flow_from_directory(
    train_dir,
    batch_size=32,
    class_mode='categorical',
    target_size=(224,224))

validation_datagen = ImageDataGenerator(
    rescale=1./255,
    preprocessing_function=preprocess_input)
```

```
validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    batch_size=32,
    shuffle=False,
    class_mode='categorical',
    target_size=(224,224))

conv_base = MobileNetV2(include_top=False,
                    weights='imagenet',
                    input_shape=(224, 224, 3))
for layer in conv_base.layers:
    layer.trainable = False

x = conv_base.output
x = layers.GlobalAveragePooling2D()(x)
x = layers.Dense(512, activation='relu')(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(1024, activation='relu')(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(1024, activation='relu')(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(512, activation='relu')(x)
predictions = layers.Dense(3, activation='softmax')(x) #3 class sayımız
model = Model(conv_base.input, predictions)

optimizer = tf.keras.optimizers.RMSprop(lr=1e-4)
model.compile(loss='categorical_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])

model.summary()

history = model.fit_generator(
    generator=train_generator,
    validation_data=validation_generator,
    steps_per_epoch=train_generator.n//train_generator.batch_size,
    validation_steps=validation_generator.n//validation_generator.batch_size,
    epochs=25)
```

```
model.save('/content/drive/My
Drive/datasets/UcClass/A71modelMobileNetV2RmspropE25.h5')

plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model doğruluğu')
plt.ylabel('Doğruluk')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model kaybı')
plt.ylabel('Kayıp')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```

EK-E

MODELİN TENSORFLOWJS' MODELİNE ÇEVİRİLMESİ

```
!apt-get install -y -qq software-properties-common python-software-properties
module-init-tools
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null
!apt-get update -qq 2>&1 > /dev/null
!apt-get -y install -qq google-drive-ocamlfuse fuse
from google.colab import auth
auth.authenticate_user()
from oauth2client.client import GoogleCredentials
creds = GoogleCredentials.get_application_default()
import getpass
!google-drive-ocamlfuse -headless -id={creds.client_id} -
secret={creds.client_secret} < /dev/null 2>&1 | grep URL
vcode = getpass.getpass()
!echo {vcode} | google-drive-ocamlfuse -headless -id={creds.client_id} -
secret={creds.client_secret}

!mkdir -p drive
!google-drive-ocamlfuse drive

import os
os.chdir("drive/datasets/UcClass/yeniModel")

!pip install tensorflowjs

!tensorflowjs_converter --input_format keras customModel3RmspropE50.h5
JScustomModel3RmspropE50
```

MANIFEST.JSON DOSYASI

```
{
    "manifest_version": 2,
    "name": "Youtube Image Classification Extension",
    "version": "1.0",
    "description": "Youtube videolarının resimleri çekilerek tanımalama
yapılması",
    "icons": {
```

```
        "128": "icon128.png",
        "48": "icon48.png",
        "16": "icon16.png"
    },
    "page_action": {
        "default_icon": "icon16.png",
        "default_popup": "popup.html",
        "default_title": "YoutubeImageClassification"
    },
    "background": {
        "page": "eventPage.html",
        "persistent": false
    },
    "content_scripts": [
        {
            "matches": ["https://www.youtube.com/*"],
            "js": ["content.js","tfjslatest.js"]
        }
    ],
    "permissions": [
        "tabs",
        "alarms",
        "storage",
        "notifications",
        "https://www.youtube.com/*"
    ],
    "content_security_policy": "script-src 'self' https://www.gstatic.com/
https://*.firebaseio.com https://*.firebase.com https://www.googleapis.com;
object-src 'self'"
}
```

POPUP.HTML DOSYASI

```
<!DOCTYPE html>
<html lang="tr">
    <head>
        <title>PageFontStyle</title>
        <meta charset="UTF-8">
        <script src="https://www.gstatic.com/firebasejs/7.6.2/firebase-
app.js"></script>
```

```
        <script src="https://www.gstatic.com/firebasejs/7.6.2/firebase-
database.js"></script>
        <script src="popup.js"></script>
        <style>
            body{
                background-color: grey;
            }
            h2{
                text-align: center;
                width:615px;
                background-color: #87ceeb;
                border-radius: .3em;
            }
            .numberlist{
                width:600px;
                margin-left: 1em;
            }
            .numberlist ol{
                counter-reset: li;
                list-style: none;
                font: 15px 'trebuchet MS', 'lucida sans';
                padding: 0;
                margin-bottom: 4em;
            }
            .numberlist li{
                position: relative;
                display: block;
                padding: .4em .4em .4em 2em;
                margin: .5em 0;
                background: #FFF;
                color: #444;
                border-radius: .3em;
            }
            .numberlist li:hover{
                background: #cbe7f8;
            }
            .numberlist li:before{
                content: counter(li);
                counter-increment: li;
```

```
                position: absolute;
                left: -1.3em;
                top: 50%;
                margin-top: -1.3em;
                background: #87ceeb;
                height: 2em;
                width: 2em;
                line-height: 2em;
                border: .3em solid #fff;
                text-align: center;
                font-weight: bold;
                border-radius: 2em;
                color:#FFF;
            }
        </style>
    </head>
    <body>
        <h2><strong>Bilgisayarınızdan Yapılan Son 25 Tespit</strong></h2>
        <hr>
        <div class="numberlist"><ol id="liste"></ol></div>
        <hr>
    </body>
</html>
```

POPUP.JS DOSYASI

```
var firebaseConfig = {
    apiKey: "AIzaSyDaV6JiLDHKA3cnyHogPdimpBN3pXXXXXX",
    authDomain: "mktez-c305f.firebaseapp.com",
    databaseURL: "https://mktez-cXXXX.firebaseio.com",
    projectId: "mktez-cXXXX",
    storageBucket: "mktez-cXXXX.appspot.com",
    messagingSenderId: "1023827XXXXXX",
    appId: "1:1023827940164:web:e1ec7f65227b2139XXXXXX"
};

firebase.initializeApp(firebaseConfig);
const database=firebase.database();
const rootref=database.ref("tespit");
```

```
chrome.storage.local.get(['kimlik'], function(result) {

rootref.orderByChild("yapan").equalTo(result.kimlik).limitToLast(25).on("valu
e", snapshot =>{
        document.getElementById("liste").innerHTML = '';
        snapshot.forEach(item => {
            var node = document.createElement("li");
            var textnode = document.createTextNode(item.val().tarih + " : " +
item.val().baslik);
            node.appendChild(textnode);
            document.getElementById("liste").appendChild(node);
        });
    });
});
```

CONTENT.JS DOSYASI

```
chrome.runtime.sendMessage({todo: "showPageAction"});
var model;
var canvas;
var context;
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse){
    async function loadModel(){
        model = undefined;
        const MODEL_URL = 'http://127.0.0.1:8887/model.json';
        //const MODEL_URL =
'https://firebasestorage.googleapis.com/v0/b/mktez-
c305f.appspot.com/o/model.json?alt=media&token=b8bcb795-3ff4-4a05-9696-
cf646c83952d';
        model = await tf.loadLayersModel(MODEL_URL);
        console.log("Model Loaded...");
    }


    if (request.todo == "modeliBaslat") {
        loadModel();
        console.log("Model Loading.............");
        var cnvs = document.createElement("canvas");
        var yeri = document.getElementById("container");
        yeri.insertBefore(cnvs, yeri.childNodes[0]);
```

```
        canvas = document.querySelector('canvas');
        canvas.width = 224;
        canvas.height = 224;
        context = canvas.getContext('2d');
    }else if (request.todo == "tanimlamaYap") {
        var video_id = window.location.search.split('v=')[1];
        var ampersandPosition = video_id.indexOf('&');
        if(ampersandPosition != -1) {
            video_id = video_id.substring(0, ampersandPosition);
        }

        var video = document.querySelector('video');
        if (!video.ended && !video.paused) {
            context.drawImage(video, 0, 0, 224, 224);
            tahmin();
        } else {
            console.log("Video sonu veya pause");
        }
        async function tahmin(){
            var image = tf.browser.fromPixels(canvas);
            var offset = tf.scalar(255.0);
            var normalized = tf.scalar(1.0).sub(image.div(offset));
            var batchedImage = normalized.expandDims(0);
            const MODEL_CLASSES = {
                0: 'Sigara',
                1: 'Silah',
                2: 'Zararlı İçecek'
            }
            if (model == undefined) {
                console.log("Model Henüz Yüklenmedi...");
            }else{
                var predictions  = await model.predict(batchedImage).data();
                var results = Array.from(predictions)
                .map(function (p, i) {
                    return {
                        probability: p,
                        className: MODEL_CLASSES[i]
                    };
                });
```

```
                var sinifAdi = "";
                var olasilik = 0;
                results.forEach(function (p) {
                    console.log(p.className + " " +
p.probability.toFixed(3));
                    if (parseFloat(p.probability)>olasilik) {
                        olasilik=p.probability;
                        sinifAdi=p.className;
                    }
                });
                var videoBaslik=
document.querySelector("title").childNodes[0].textContent;
                if (olasilik>0.95) {
                    chrome.runtime.sendMessage({
                        todo: "addToList",
                        video_id: video_id,
                        videoBaslik: videoBaslik,
                        sinifAdi: sinifAdi
                    });
                    location.replace("https://www.youtube.com/");
                }

                image.dispose();
                offset.dispose();
                normalized.dispose();
                batchedImage.dispose();
                results= null;
                predictions= null;

                video_id= null;
                ampersandPosition= null;
                olasilik= null;
                videoBaslik= null;
                sinifAdi= null;

            }
        }
    }
```

```
});
```

EVENTPAGE.HTML DOSYASI

```
<!DOCTYPE html>
<html lang="tr">
    <head>
        <title></title>
        <meta charset="UTF-8">
        <script src="https://www.gstatic.com/firebasejs/7.6.2/firebase-
app.js"></script>
        <script src="https://www.gstatic.com/firebasejs/7.6.2/firebase-
database.js"></script>
        <script src="eventPage.js"></script>
    </head>
    <body></body>
</html>
```

EVENTPAGE.JS DOSYASI

```
var firebaseConfig = {
    apiKey: "AIzaSyDaV6JiLDHKA3cnyHogPdimpBN3pz0XXXX",
    authDomain: "mktez-cXXXX.firebaseapp.com",
    databaseURL: "https://mktez-c cXXXX.firebaseio.com",
    projectId: "mktez-c cXXXX ",
    storageBucket: "mktez-c cXXXX.appspot.com",
    messagingSenderId: "102382794XXXX",
    appId: "1:1023827940164:web:e1ec7f65227b2139f6 cXXXX "
};
firebase.initializeApp(firebaseConfig);
const database=firebase.database();
const rootref=database.ref("tespit");

chrome.tabs.onUpdated.addListener( function (tabId, changeInfo, tab) {
    if (changeInfo.status == 'complete' && tab.active &&
tab.url.startsWith("https://www.youtube.com/") &&
tab.url!="https://www.youtube.com/") {
        console.log("Yüklenen Sayfa :" + tab.url);
        var video_id = tab.url.split('v=')[1];
```

```
        rootref.orderByChild("videoID").equalTo(video_id).on("value",
snapshot =>{
            if (snapshot.val()!=null) {
                var notifOptions = {
                    type: 'basic',
                    iconUrl: 'icon48.png',
                    title: 'Görüntülenen Video Zararlı',
                    message: 'İzlediğiniz Video Sizin İçin Uygun Değildir!!!'
                };
                chrome.notifications.create('tespitNotif', notifOptions);
                chrome.tabs.query({currentWindow: true, active: true},
function (tab) {
                    chrome.tabs.update(tab.id, {url:
"https://www.youtube.com/"});
              });
            }
        });
    }
});

chrome.runtime.onMessage.addListener(function(request, sender, sendResponse){
    if (request.todo == "showPageAction") {
        chrome.tabs.query({active: true, currentWindow: true},
function(tabs){
            chrome.pageAction.show(tabs[0].id);
            chrome.tabs.sendMessage(tabs[0].id, {todo: "modeliBaslat"});
            chrome.alarms.create('refresh', { periodInMinutes:0.05 });
        });
    }else if(request.todo == "addToList"){
        var notifOptions = {
            type: 'basic',
            iconUrl: 'icon48.png',
            title: 'Zararlı İçerik Tespit Edildi!',
            message: 'İzlediğiniz Video Sizin İçin Uygun Değildir!!!'
        };
        chrome.notifications.create('tespitNotif', notifOptions);

        console.log("Tespit: " + request.sinifAdi + " videoID:" +
request.video_id + " baslik: " + request.videoBaslik);
```

```
        rootref.orderByChild("videoID").equalTo(request.video_id).on("value",
snapshot =>{
            if (snapshot.val()==null) {
                const autoid=rootref.push().key;

                chrome.storage.local.get(['kimlik'], function(result) {
                    var today = new Date();
                    var dd = String(today.getDate()).padStart(2, '0');
                    var mm = String(today.getMonth() + 1).padStart(2, '0');
//January is 0!
                    var yyyy = today.getFullYear();
                    today = yyyy + '-' + mm + '-' + dd;
                    document.write(today);

                    rootref.child(autoid).set({
                        videoID: request.video_id,
                        baslik: request.videoBaslik,
                        icerik: request.sinifAdi,
                        tarih: today,
                        yapan: result.kimlik
                    });
                });
            } else {
                console.log(snapshot.val());
            }
        });
    }
});

chrome.alarms.onAlarm.addListener((alarm) => {
    console.log(alarm.name); // refresh
    chrome.tabs.query({active: true, currentWindow: true}, function(tabs){
        if (tabs[0].url.startsWith("https://www.youtube.com/") &&
tabs[0].url!="https://www.youtube.com/") {
            chrome.tabs.sendMessage(tabs[0].id, {todo: "tanimlamaYap"});
        }
    });
});
```

```
chrome.runtime.onInstalled.addListener(() => {
    console.log('onInstalled....');
    kimlik();
});

chrome.runtime.onStartup.addListener(() => {
    console.log('onStartup....');
    kimlik();
});

function kimlik() {
    chrome.storage.local.get(['kimlik'], function(result) {
        if (result.kimlik===undefined) {
            var deger=uuidv4();
            function uuidv4() {
                return ([1e7] + -1e3 + -4e3 + -8e3 + -1e11).replace(/[018]/g,
c =>
                (c ^ crypto.getRandomValues(new Uint8Array(1))[0] & 15 >> c /
4).toString(16)
                )
            }
            chrome.storage.local.set({kimlik: deger}, function() {
                console.log('Kimlik Olusturuldu: ' + deger);
            });
        } else {
            console.log('Gecerli Kimlik: ' + result.kimlik);
        }
    });
}
```

EK-F

MODELİN TENSORFLOW LITE MODELİNE ÇEVİRİLMESİ

```
!apt-get install -y -qq software-properties-common python-software-properties
module-init-tools
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null
!apt-get update -qq 2>&1 > /dev/null
!apt-get -y install -qq google-drive-ocamlfuse fuse
from google.colab import auth
auth.authenticate_user()
from oauth2client.client import GoogleCredentials
creds = GoogleCredentials.get_application_default()
import getpass
!google-drive-ocamlfuse -headless -id={creds.client_id} -
secret={creds.client_secret} < /dev/null 2>&1 | grep URL
vcode = getpass.getpass()
!echo {vcode} | google-drive-ocamlfuse -headless -id={creds.client_id} -
secret={creds.client_secret}

!mkdir -p drive
!google-drive-ocamlfuse drive

import os
os.chdir("drive/datasets/UcClass/yeniModel/G3")

!tflite_convert --keras_model_file=customModel3RmspropE50.h5 --
output_file=customModel3RmspropE50.tflite
```

ANDROIDMANIFEST.XML DOSYASI

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.guvenliyoutube">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"
/>

    <application
```

```
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppCompat.Light.NoActionBar">
        <activity android:name=".SplashActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity"
            android:configChanges="orientation|screenSize"/>
    </application>

</manifest>
```

BUILD.GRADLE (MODULE) DOSYASI

```
apply plugin: 'com.android.application'
apply plugin: 'com.google.gms.google-services'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId "com.example.guvenliyoutube"
        minSdkVersion 22
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
```

```
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }

    aaptOptions {
        noCompress "tflite"
    }
}

dependencies {
    implementation fileTree(dir: "libs", include: ["*.jar"])
    implementation 'androidx.appcompat:appcompat:1.1.0'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    implementation 'com.google.firebase:firebase-analytics:17.2.2'
    implementation 'com.google.firebase:firebase-database:19.3.0'
    implementation 'com.google.firebase:firebase-ml-model-interpreter:22.0.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.1'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'
}
```

BUILD.GRADLE (PROJECT) DOSYASI

```
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath "com.android.tools.build:gradle:4.0.0-beta04"
        classpath 'com.google.gms:google-services:4.3.3'
    }
}

allprojects {
    repositories {
        google()
        jcenter()
```

```
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

SPLASHACTIVITY.JAVA DOSYASI

```
package com.example.guvenliyoutube;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.os.Bundle;

public class SplashActivity extends AppCompatActivity {

    public boolean internetVarMi (final Context context) {
        final ConnectivityManager connectivityManager =
((ConnectivityManager)
context.getSystemService(Context.CONNECTIVITY_SERVICE));
        return  connectivityManager.getActiveNetworkInfo() != null &&
connectivityManager.getActiveNetworkInfo().isConnected();
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash);

        if (internetVarMi(this)){
            Thread splashThread = new Thread() {
                @Override public void  run(){
                    try {
                        synchronized (this) {
```

```
                        wait(4000);
                    }
                } catch (InterruptedException ex){

                } finally{
                        startActivity(new Intent(getApplicationContext(),
MainActivity.class));
                        finish();
                }
            }
        };
        splashThread.start();
    } else {
        final AlertDialog alert = new AlertDialog.Builder(this).create();
        alert.setTitle("Bağlantı Hatası!");
        alert.setMessage("Lütfen İnternet Bağlantınızı Kontrol
Ediniz!!!");
        alert.setButton(DialogInterface.BUTTON_NEUTRAL, "Tamam", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                int pid = android.os.Process.myPid();
                android.os.Process.killProcess(pid);
                dialog.dismiss();
            }
        });
        alert.show();
    }
  }
}
```

KAYIT.JAVA DOSYASI

```
package com.example.guvenliyoutube;

public class Kayit {

    public String baslik;
    public String icerik;
    public String tarih;
```

```
    public String videoID;
    public String yapan;

    public Kayit(String baslikG, String icerikG, String tarihG, String
videoIDG, String yapanG) {
        baslik = baslikG;
        icerik = icerikG;
        tarih = tarihG;
        videoID = videoIDG;
        yapan = yapanG;
    }
}
```

REATIMEVERITABANI.JAVA DOSYASI

```
package com.example.guvenliyoutube;

import android.util.Log;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import java.text.SimpleDateFormat;
import java.util.Date;

public class RealTimeVeritabani {

    public void realTimeWrite(final String gVideoBaslik, final String
gVideoID, final String donenVeri, final String androidIDG) {
        Date cDate = new Date();
        final String fDate = new SimpleDateFormat("yyyy-MM-
dd").format(cDate);
        Log.d("Veritabani", "Kayıt Bilgileri: " + gVideoBaslik + " " +
donenVeri + " " + fDate + " " + gVideoID + " " + androidIDG);

        FirebaseDatabase mDatabase = FirebaseDatabase.getInstance();
        DatabaseReference mDatabaseReference =
mDatabase.getReference("tespit");
        String kayitID = mDatabaseReference.push().getKey();//unique ID
oluştur
```

```
        Kayit kayit = new Kayit(gVideoBaslik, donenVeri, fDate, gVideoID,
androidIDG);
        mDatabaseReference.child(kayitID).setValue(kayit);//kayıt işlemini
gerçekleştir.
    }
}
```

CUSTOMMODEL.JAVA DOSYASI

```
package com.example.guvenliyoutube;

import android.graphics.Bitmap;
import android.graphics.Color;
import android.util.Log;

import androidx.annotation.NonNull;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.ml.common.FirebaseMLException;
import com.google.firebase.ml.custom.FirebaseCustomLocalModel;
import com.google.firebase.ml.custom.FirebaseModelDataType;
import com.google.firebase.ml.custom.FirebaseModelInputOutputOptions;
import com.google.firebase.ml.custom.FirebaseModelInputs;
import com.google.firebase.ml.custom.FirebaseModelInterpreter;
import com.google.firebase.ml.custom.FirebaseModelInterpreterOptions;
import com.google.firebase.ml.custom.FirebaseModelOutputs;

import java.io.IOException;

public class CustomModel {
    public String yapilanTespit ="YOK";

    private FirebaseModelInterpreter
createInterpreter(FirebaseCustomLocalModel localModel) throws
FirebaseMLException {
        FirebaseModelInterpreter interpreter = null;
        try {
            FirebaseModelInterpreterOptions options =
```

```
                    new
FirebaseModelInterpreterOptions.Builder(localModel).build();
            interpreter = FirebaseModelInterpreter.getInstance(options);
        } catch (FirebaseMLException e) {
            Log.i("MLKit_Interpreter", e.getMessage());
        }
        return interpreter;
    }

    private FirebaseModelInputOutputOptions createInputOutputOptions() throws
FirebaseMLException {
        FirebaseModelInputOutputOptions inputOutputOptions =
                new FirebaseModelInputOutputOptions.Builder()
                        .setInputFormat(0, FirebaseModelDataType.FLOAT32, new
int[]{1, 224, 224, 3})
                        .setOutputFormat(0, FirebaseModelDataType.FLOAT32,
new int[]{1, 3})
                        .build();
        return inputOutputOptions;
    }

    private float[][][][] bitmapToInputArray(Bitmap bitmap) {//oluşturulan
ham resim verisi burada işleniyor
        bitmap = Bitmap.createScaledBitmap(bitmap, 224, 224, true);

        int batchNum = 0;
        float[][][][] input = new float[1][224][224][3];
        for (int x = 0; x < 224; x++) {
            for (int y = 0; y < 224; y++) {
                int pixel = bitmap.getPixel(x, y);
                input[batchNum][x][y][0] = Color.red(pixel)  / 255.0f;
                input[batchNum][x][y][1] = Color.green(pixel) / 255.0f;
                input[batchNum][x][y][2] = Color.blue(pixel) / 255.0f;
            }
        }
        return input;
    }
```

```
    public String runInference(Bitmap bitmapGelen) throws FirebaseMLException
{//modelin çalıştırıldı kısım burasıdır
        FirebaseCustomLocalModel localModel = new
FirebaseCustomLocalModel.Builder()
                .setAssetFilePath("modelK.tflite")
                .build();//model Assets klasöründen okundu
        FirebaseModelInterpreter firebaseInterpreter =
createInterpreter(localModel);
        float[][][][] input = bitmapToInputArray(bitmapGelen);
        FirebaseModelInputOutputOptions inputOutputOptions =
createInputOutputOptions();

        FirebaseModelInputs inputs = new FirebaseModelInputs.Builder()
                .add(input)
                .build();

        firebaseInterpreter.run(inputs, inputOutputOptions)
                .addOnSuccessListener(
                        new OnSuccessListener<FirebaseModelOutputs>() {
                            @Override
                            public void onSuccess(FirebaseModelOutputs
result) {
                                float[][] output = result.getOutput(0);
                                float[] probabilities = output[0];//sonuçlar
                                try {
                                    yapilanTespit =
useInferenceResult(probabilities);
                                    if(!yapilanTespit.equals("YOK")){//burada
sonuç YOK ise bir tespit yapılmamış demektir.
                                        Log.i("MLKit", "TESPİT EDİLEN :" +
yapilanTespit);
                                    }
                                } catch (IOException e) {
                                    e.printStackTrace();
                                }
                            }
                        })
                .addOnFailureListener(
                        new OnFailureListener() {
```

```
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                Log.i("MLKit", "runInference ERROR : " + e);
                            }
                        });
        return yapilanTespit;
    }

    private String useInferenceResult(float[] probabilities) throws
IOException {
        String tespitDurum ="YOK";
        String label = "";
        String sinifAdi = "";
        float olasilik = 0;

        for (int i = 0; i < probabilities.length; i++) {
            if(probabilities[i]>olasilik){
                olasilik = probabilities[i];
                if (i==0) {sinifAdi = "Sigara";} else if (i==1){sinifAdi =
"Silah";} else {sinifAdi = "Zicecek";}
            }

            if (i==0) {label = "Sigara";} else if (i==1){label = "Silah";}
else {label = "Zicecek";}
            Log.i("MLKit", String.format("%s: %1.4f", label,
probabilities[i]));

            if(olasilik>0.95){
                tespitDurum = sinifAdi;
            }
        }
        return tespitDurum;
    }
}
```

MAINACTIVITY.JAVA DOSYASI

```
package com.example.guvenliyoutube;

import android.content.Context;
```

```
import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.provider.Settings;
import android.util.Log;
import android.webkit.JavascriptInterface;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.Query;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.ml.common.FirebaseMLException;

public class MainActivity extends AppCompatActivity {

    public CustomModel myCustomModel =new CustomModel();

    public RealTimeVeritabani veriler = new RealTimeVeritabani();

    WebView webView;
    private String url = "https://m.youtube.com/";

    private byte[] mDecodedImage;

    SharedPreferences sharedPref;
    SharedPreferences.Editor editor;

    public boolean listedeVarMi=false;

    public String androidID;
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        webView = findViewById(R.id.myWebView);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.setWebViewClient(new WebClient());
        webView.addJavascriptInterface(new WebAppInterface(this), "Android");
        webView.loadUrl(url);

        sharedPref = this.getPreferences(Context.MODE_PRIVATE);
        editor = sharedPref.edit();
        editor.putString("urlDegisim","myoutube");
        editor.commit();

        androidID =
Settings.Secure.getString(getContentResolver(),Settings.Secure.ANDROID_ID);
    }

    public class WebClient extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            view.loadUrl(url);
            return true;
        }

        @Override
        public void onPageFinished(WebView view, String url)
        {
            view.loadUrl(//sayfa yüklenirken inject edilecek js kodlarım
                    "javascript:(function() { " +
                            "var cnvs = document.createElement('canvas');" +
                            "var yeri = document.getElementById('initial-
data');" +
                            "yeri.insertBefore(cnvs, yeri.childNodes[0]);" +
                            "var canvas = document.querySelector('canvas');"
+
                            "var context = canvas.getContext('2d');" +
                            "canvas.width = 224;" +
```

```
                            "canvas.height = 224;" +
                            "window.setInterval( snap, 3000);" +
                            "function snap() {" +
                                "var video_id =
window.location.search.split('v=')[1];" +
                                "if(video_id!=undefined) {" +
                                    "var video =
document.querySelector('video');" +
                                    "if(!video.ended && !video.paused &&
typeof Android !== 'undefined' && Android !== null) {" +
                                        "var videoBaslik=
document.querySelector('title').childNodes[0].textContent;" +
                                        "context.drawImage(video, 0, 0, 224,
224); " +
                                        "var dataURL = canvas.toDataURL();" +
                                        "Android.resimGonder(video_id,
videoBaslik, dataURL.toString());" +
                                    "}" +
                                "}"+
                            "}"+
                        "})()");
        }
    }

    public class WebAppInterface {
        Context mContext;
        String donenVeri = "YOK";
        String gVideoID;
        String gVideoBaslik;

        WebAppInterface(Context c) {
            mContext = c;
        }

        @JavascriptInterface
        public void resimGonder(String videoId, String videoBaslik, String
base64Image) {
            gVideoID = videoId;
            gVideoBaslik = videoBaslik;
```

```
            String savedString = sharedPref.getString("urlDegisim","Kayıt
Yok");

            if(!savedString.equals(videoId)) {
                editor.putString("urlDegisim",videoId);
                editor.commit();
                Log.d("Veritabani", "URL Değişim Kontrolü Tetiklendi" +
videoId);

                Query query =
FirebaseDatabase.getInstance().getReference("tespit").orderByChild("videoID")
.equalTo(videoId);
                query.addListenerForSingleValueEvent(new ValueEventListener()
{
                    @Override
                    public void onDataChange(@NonNull DataSnapshot
dataSnapshot) {
                        if(dataSnapshot.exists()){
                            listedeVarMi=false;
                            Log.d("Veritabani", "Tespit Listesinde Var!!! "+
gVideoID);
                            Toast.makeText(mContext, "Görüntülenen Video
Zararlı İçerik İçermektedir!!!Bul", Toast.LENGTH_SHORT).show();
                            runOnUiThread(new Runnable() {//ui ile ilgili bir
hata veriyor onu önleyecekmiş
                                @Override
                                public void run() {
                                    webView.loadUrl(url);

editor.putString("urlDegisim","myoutube");
                                    editor.commit();
                                }
                            });
                        }else {listedeVarMi=true;}
                    }

                    @Override
                    public void onCancelled(@NonNull DatabaseError
databaseError) {
```

```
                        Log.d("Veritabani", "Error is: " +
databaseError.toException());
                    }
                });
            }

            String cleanBase64Image =
base64Image.replace("", "");
            try
            {
                mDecodedImage = android.util.Base64.decode(cleanBase64Image,
android.util.Base64.DEFAULT);
                if (mDecodedImage.length == 0) {
                    Log.d("resim", "Uzunluk Sıfır!!!!");
                    return;
                }
                if (mDecodedImage != null) {
                    final Bitmap bitmap =
BitmapFactory.decodeByteArray(mDecodedImage, 0, mDecodedImage.length);
                    if (bitmap != null) {

                        runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                try {
                                    donenVeri =
myCustomModel.runInference(bitmap);
                                    if(!donenVeri.equals("YOK")){
                                        if (listedeVarMi==true){

veriler.realTimeWrite(gVideoBaslik,gVideoID,donenVeri,androidID);
                                        }
                                        Toast.makeText(mContext,
"Görüntülenen Videoda Zararlı İçerik Tespit Edilmiştir!!!Yaz",
Toast.LENGTH_SHORT).show();
                                        runOnUiThread(new Runnable() {
                                            @Override
                                            public void run() {
                                                webView.loadUrl(url);
```

```
editor.putString("urlDegisim","myoutube");
                                                editor.commit();
                                            }
                                        });
                                    } //if(!donenVeri.equals("YOK"))
                                } catch (FirebaseMLException e) {
                                    e.printStackTrace();
                                }
                            } //public void run()
                        });

                    } else {
                        Log.d("resim", "Hata Oluştu!!!!");
                    }
                }
            } catch (Exception e)
            {
                Toast.makeText(mContext, "Byte Conversion Error !
\n"+e.getMessage(), Toast.LENGTH_SHORT).show();
            }
        }
    }
}
```

ÖZGEÇMİŞ

Murat KAZANÇ

1983 İstanbul doğumlu. İlkokulu Kemalpaşa İlkokulu'nda tamamladı. Ortaokulu Gültepe Orta Okulu'nda, liseyi de Zeytinburnu Ensüstri Meslek Lisesi'nde tamamladı. 2006 yılından Marmara Üniversitesi Bilgisayar Kontrol Öğretmenliğini (İngilizce) tamamladı. 2008 yılından beri Milli Eğitim Bakanlığında Bilgisayar Öğretmeni olarak görev yapmakta. 2018 yılından beri İstanbul Cerrahpaşa Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümüne Mühendislik Tamamlama öğrencisi olarak devam etmektedir.