2021-2022 Güz Dönemi

Görüntü İşleme

Proje 2 Rapor

Derin Öğrenme ile Duygu Tespit Uygulaması

Emel KAYACI – 05180000087

Mehmet Anıl TAYSİ – 05170000022

21.1.2022

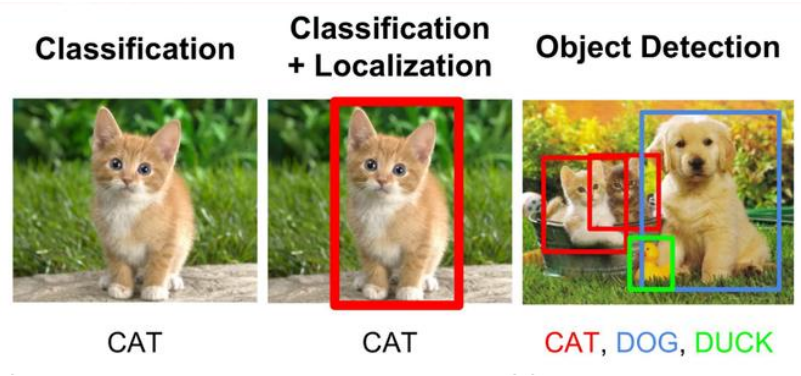
İçindekiler

1. Image Classification, Object Detection ve Image Segmentation kavramları arasındaki farklar nelerdir?

Bu kavramların arasındaki farklar basitten karmaşığa doğru yaptıkları işlerden anlaşılmaktadır çünkü birinin yaptığı işin üstüne katılarak sonraki kavrama ulaşılmaktadır.

Image classification yani resmin sınıflandırılmasında modele girdi olarak bir resim verilir. Çıktı olarak bu resmin ait olduğu sınıfın yanında bir metrik (olasılık, loss, accuracy vb.) elde edilir.

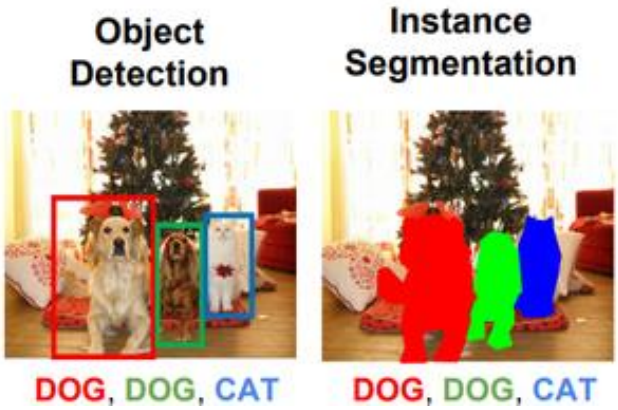
Object detection kavramına geçmeden önce image localization kavramının tanıtılması gerekmektedir. Image localization, verilen görüntüdeki tek bir nesnenin yerini belirlememize yardımcı olur. Birden fazla nesnemiz olması durumunda object detection kavramına başvurmamız gerekir. [1]



Resim 1. Classification, localization ve detection kavramları

Object detection'ın bazı dezavantajları bulunmaktadır. Örneğin bunlardan biri bounding box yani nesneyi kapsayan çerçevenin şeklinin her zaman dikdörtgensel olmasıdır. Bu nedenle, eğer nesne eğrilik kısmını içeriyorsa, nesnelerin şeklinin belirlenmesine yardımcı olmaz. Ayrıca bazı durumlarda oldukça kritik olabilecek nesnenin alanı, çevresi vb. bilgileri doğru tahmin edemez.

Image segmentation bu dezavantajları dikkate alarak bounding box yerine her nesne için oluşturulan piksel bazında maskeler aracılığıyla bir nesnenin varlığını belirtir. [2]



Resim 2. Detection, segmentation kavramları

2. (Ön Çalışma) Derin Öğrenme ile görüntü tanıma örnekleri

Ön çalışma kapsamında aşağıdaki konu başlıklarında çalışmalar yapılmıştır.

2.1 Görüntü tanıma için derin öğrenme neden gereklidir?

Görüntü tanıma iki şekilde gerçekleştirilebilir: geleneksel görüntü işleme teknikleri ya da modern derin öğrenme ağları ile.

Geleneksel teknikler kullanılırken eğitim için önceden hazırlanmış ve sınıflandırılmış resimleri içeren bir veri seti gerekmemektedir fakat kompleks bir senaryo içeren resimde (benzer renkler, birbirleriyle çakışan objeler, gölgeler, yanlış ışıklandırma) performansı da oldukça düşük olmaktadır. [3]

2.2 Derin öğrenme ile görüntü tanıma için ünlü modeller nelerdir?

Bu tanımı yapmadan önce tek ve iki aşamalı modellerin açıklanması gerekmektedir. Görüntü tanımada iki görev bulunur. Bunlardan ilki nesnelerin bulunması ikincisi ise her bir nesnenin sınıflandırılıp bounding box ile belirtilmesidir.

İki aşamalı modeller önce nesnelerin bulunma ihtimallerinin olduğu bölgeleri önerip daha sonra sınıflandırmayı gerçekleştirirken tek aşamalı modellerde bu işlem aynı anda yapılmaktadır. 2014'ten sonra RCNN ve SPPNet'in tanıtılmasıyla derin öğrenme ile görüntü tanıma çağına girilmiştir.

En önemli **iki aşamalı** nesne algılama algoritmaları

1. RCNN and SPPNet (2014)
2. Fast RCNN and Faster RCNN (2015)
3. Mask R-CNN (2017)
4. Pyramid Networks/FPN (2017)
5. G-RCNN (2021)

En önemli **tek aşamalı** nesne algılama algoritmaları

1. YOLO (2016)
2. SSD (2016)
3. RetinaNet (2017)
4. YOLOv3 (2018)
5. YOLOv4 (2020)
6. YOLOR (2021) [3]

2.3 Derin öğrenme ile görüntü tanıma gerçek hayatta kullanımları

Bu bölümde görüntü tanımanın yaygın olarak kullanıldığı alanlara göre örnekler tanıtılacaktır. [4]

2.3.1 Üretim (Manufacturing)

- Şantiye veya fabrikalarda güvenlik protokollerine uyumun izlenmesine yardımcı olur. Örneğin kask veya maske takmayan işçiler tespit edilebilir.
- Üretilen mallardaki kusurlar tespit edilebilir. Derin öğrenme metotları geleneksel metotlara göre gerçek zamanlı uygulamalarda çok daha başarılıdır. Kusur tespitinin de gerçek zamanlı yapılması gerektiğinden bu alanda derin öğrenme çok kullanılır. Örneğin üretilen vidalardaki şekil bozuklukları, toplanan domateslerin renklerine göre ayrılması.

2.3.2 Sağlık

- Kanser, korona virüsü ve hasarlı hücrelerin tespit edilmesinde kullanılır. Hastlaığın ilerleyiş evresine kadar çıkarımlar yapılabilir.
- Pose estimation yani görüntüden hareketlerin tespit edilmesiyle çeşitli nörolojik ve kas-iskelet sistemi hastalıkları tespit edilebilir ve tedavileri de bu yöntemle gerçekleştirilebilir. Örneğin fizyoterapi seanslarında hareketlerin hastalar tarafından doğru yapılıp yapılmadığı yanlarında herhangi biri olmadan bu yöntemle tespit edilebilir, öneriler sunulabilir.

2.3.3 Tarım

- Akıllı tarım son dönemlerde oldukça gelişmektedir. Tarım otomasyonu sayesinde hayvanlarım sayımları drone yardımlarıyla yapılabilmektedir. Böylece otomatik besleme mekanizmaları oluşturulabilmektedir.
- Bitkilerin büyümesi ve besin gereksinimlerini insanların aksine sürekli izlenmesi sağlanır. Çeşitli bitki hastalıkları tespit edilebilir. Örneğin yapraklarda hastalığın nerelerde olduğu ve neyden kaynaklandığı tespit edilebilmektedir.
- Böceklerin hızlı ve doğru tanınması ile haşere kontrolü verimli bir şekilde yapılabilir.

2.3.4 Ulaşım

- Trafik yoğunluğunun tespit edilmesinde kullanılır.
- Otoparklarda doluluk tespitlerinde kullanılır. Daha doğru tahmin yapılabilmesi için görüntü işlemeye ek termal kameralar da kullanılabilir.
- Plaka okumada kullanılır. Böylece hem trafik yoğunluğu hem de otomasyon sayesinde bu işe ayrılmış bütçe azalır.
- Yaya, trafik işaretleri, yol durumu (çukur vs.) tespit edilmesinde kullanılır.
- Sürücünün yorgunluk durumu veya emniyet kemeri takıp takmadığı tespit edilebilir. Böylece daha güvenli bir sürüş deneyimi elde edilebilir. Yorgunluk durumu çeşitli

faktörler elde edilerek ölçülebilir, bu faktörlerden biri belli zaman aralığında gözlerini ne kadar kapalı tuttuğu olabilir.

2.3.5 Pazarlama (Retail)

- Cadde üzerinde yoğunluk ölçülerek müşterilerin yoğunlaştığı mağzalar tespit edilebilir.
- Hırsız veya şüpheli hareketler tespit edilebilir.
- Sosyal mesafe tespit edilebilir.

Ayrıca bu araştırmalara ek Coursera platformundaki Convolutional Neural Networks kursundan da yaralanılmıştır. Nesnelerin tespiti için kullanılan tekniklerin detayları hakkında bilgi sahibi olunmuştur.

3. Derin Öğrenme ile Duygu Tespit Uygulaması

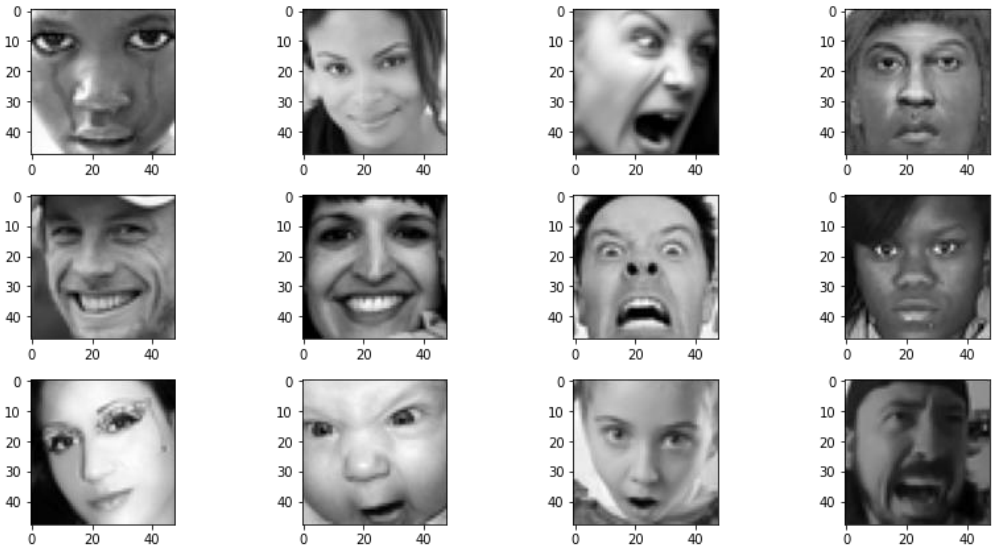
3.1 Konu Belirlenmesi ve Problemin Tanımı

Konu olarak yüz ifadelerinden duygu tespit uygulaması yapılmaya karar verildi. Duyguların tespit edilmesi sosyal yaşamda oldukça büyük öneme sahiptir. Aşağıda uygulamanın gerçek hayatta kullanılabileceği örnekler yer almaktadır.

1. Duygu tespitine ek yaş, cinsiyet, kimlik tespiti de eklendiğinde kalabalıktaki insanlar arasında potansiyet suçluları tespit edebilir. [5]
2. Mülakat veya sınavlarda kişinin stres durumunu ölçerek kişiyi rahatlatabilir veya aksine yüksek stresli bir durumda kişinin vereceği cevapları not edebilir.
3. Ders esnasında öğrencilerin duygularını ölçerek konu değişikliğine gidilebilir böylece dersim verimi arttırılabilir.
4. Sosyal konularda çeşitli video röportajlardan yararlanarak kişilerin konu hakkındaki duyguları toplanabilir ve bunların ışığında çeşitli analizler gerçekleştirilebilir.

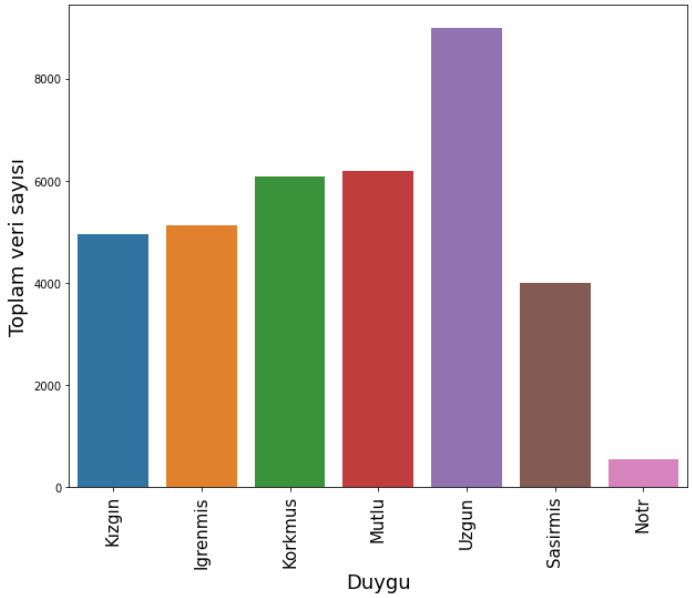
3.2 Veri Seti Araştırması

Veri seti için Kaggle platformundan yararlanılmıştır. Veri seti 35.887 örnek içermektedir. Her bir örneğin boyutu 48x48 piksel olup gri tonlamalıdırlar. Sınıflar mutlu, nötr, üzgün, kızgın, şaşırmış, iğrenmiş ve korkmuş şeklinde 7 tanedir.



Resim 3. Veri setinden bazı örnekler

Aşağıda sınıflarına göre örneklerin miktarının dağılımı gösterilmiştir. Nötr sınıfı hariç örnekler eşit dağılmıştır bu nedenle accuracy başarı ölçümü için kullanılabilecek bir metriktir fakat yine de 4.bölümde karmaşıklık matrisindeki diğer metriklere de bakılacaktır.



Resim 4. Örneklerin sınıflarına göre dağılımları

3.2.1 Veri setinin eğitim ve test verisi olarak ayrılması

Veri setimiz .csv dosyası olduğundan öncelikle python ortamında bu verileri bir dataframe objesi olarak muhafaza ettik. Duygu verilerini (dataframe'in ilk sütunu) Y değişkeni ile, görüntülerimizi oluşturan piksel verilerini (dataframe'in ikinci sütunu) X değişkeni ile göstermiş olduk. Sklearn kütüphanesinin model_selection modülünü kullanarak train_test_split fonksiyonunu programımıza ekleyerek veriyi eğitim ve test verisi olarak ayırmış olduk. Test_size olarak farklı denemelerin sonucu olarak 0.1 değerini tercih ettik ki bu da verisetinin %10'luk bir kısmını oluşturuyor.

3.3 Python Ortamında Model Oluşturma

3.3.1 Kullanılan ortam, dil, kütüphane ve sürümleri

- Ortam olarak ekip çalışmasının rahatlıkla yapılabileceği, değişikliklere her geliştiricinin eş zamanlı ulaştığı Google Collab ortamını tercih ettik.
- Dil olarak Python'ın 3.7 versiyonu kullanılmıştır.
- Aşağıda kullanılan kütüphane ve sürümleri verilmiştir. Kütüphane ve sürümler ayrıca requirements.txt dosyasında bulunmaktadır. Böylece projeyi kendi bilgisayarlarında açmak isteyen kullanıcılar versiyon uyuşmazlıklarına takılmadan doğrudan kullanabileceklerdir.

pandas==1.1.5	seaborn==0.11.2
numpy==1.19.5	matplotlib==3.2.2
oauth2client==4.1.3	pydrive==1.3.1
tensorflow==2.7.0	keras==2.7.0
sklearn==0.0	cv2==4.1.2
PIL==7.1.2	

3.3.2 Modelin oluşturulması, kodu ve şeması

Model oluşturulurken CNN mimarisinde bulunan temel katmanlar kullanılmıştır. Deneysel çalışmalar kısmında da bu katmanların ne işe yaradığı, hangi değerler ile daha başarılı çalıştığı detaylı bir şekilde açıklanmıştır. Modelin şeması kodun altında yer almaktadır.

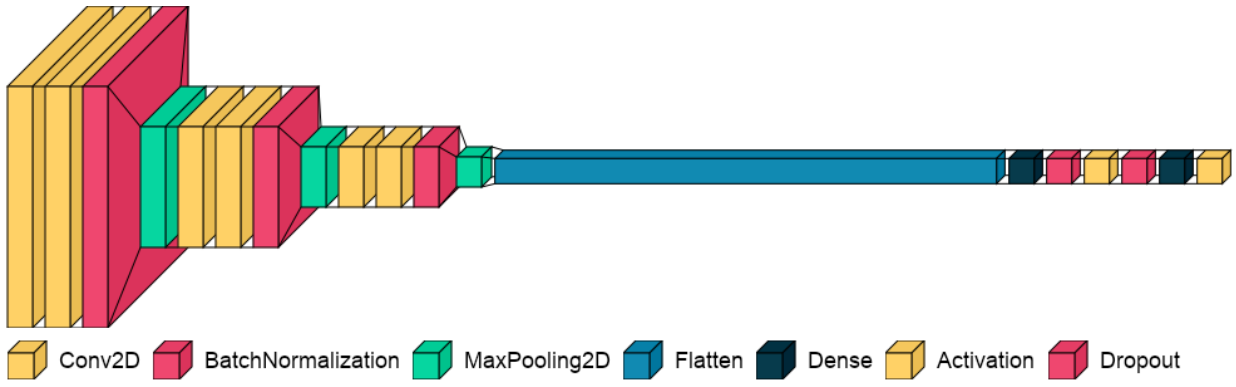
```
def cnn_model(num_layers, filter_size, activation_func, dropout,
optimizer, block_number):
    model = Sequential()
    input_shape = (48,48,1)

    for i in range(block_number):
      model.add(Conv2D(num_layers, filter_size,
input_shape=input_shape,activation=activation_func, padding='same'))
      model.add(Conv2D(num_layers, filter_size,
activation=activation_func, padding="same"))
      model.add(BatchNormalization())
      model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(16))
    model.add(BatchNormalization())
    model.add(Activation(activation_func))
    model.add(Dropout(dropout))
    model.add(Dense(7))
    model.add(Activation('softmax'))

    model.compile(loss='categorical_crossentropy',
metrics=['accuracy'], optimizer=optimizer)

    return model
model=cnn_model(128, (5,5), 'relu', 0.2, 'Adam', 3)
model.summary()
```



3.3.3 Faydalanılan çalışmalar ve farklılıklar

Veri setini aldığımız ortam olan Kaggle'da bu veri seti üzerine yapılmış çalışmalar da bulunmaktaydı. Biz de projeye başlamadan önce bu çalışmaları inceledik ve projemizi nasıl genişletebiliriz diye düşündük.

Benzer çalışmalardan farklı olarak çok detaylı bir parametre seçim süreci gerçekleştirdik. Her bir parametre ile modeli ayrı kurup başarısını grafik üzerinde gösterdik. Ayrıca modelin gerçek dünya üzerindeki başarısını görebilmek için gerçek zamanlı kamera aracılığı ile yüz ifadesi tahmin özelliğini de ekledik.

4. Deneysel Çalışmalar

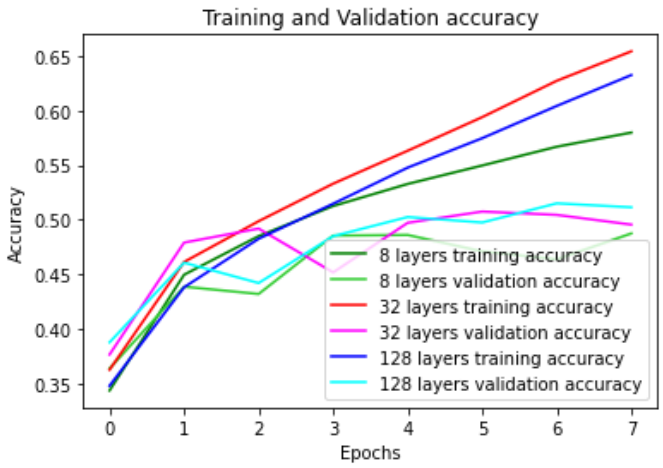
Modelimizi kurarken aşağıdaki özellikleri tercih ettik fakat bu özellikler en başta rastgele belirlenmiş olup modelimizi optimize ettiğimizde değişecektir.

4.1 Katman sayısı

Konvolüsyon (Conv2D) katmanlarında 8, ilk Dense katmanında 16 olarak belirlendi.

Sonuç: Katman sayısı arttıkça öğrenilecek parametre sayısı da artmaktadır. Parametreler veri içindeki benzer desenleri keşfedebilmeyi sağlarlar.

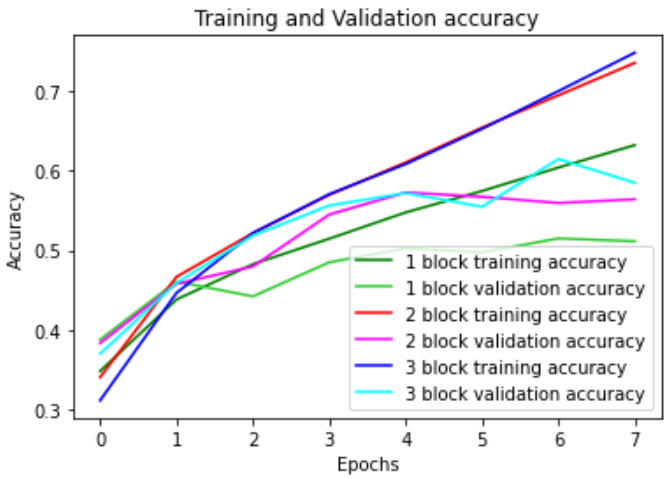
Daha çok parametre ile daha benzer desenlerin keşfedilmesi kolaylaşır. İşlem süresi artsa da grafikten de görüldüğü validation setinin accuracy değeri artmıştır.



Resim 5. Katman sayısı ile accuracy arasındaki ilişki

4.2 Katman (Blok) miktarı

Blok sayısı arttıkça eğitilecek parametre sayısı da artacağından daha başarılı sonuçlar elde edilmiştir.



Resim 6. Blok sayısı ile accuracy arasındaki ilişki

4.3 Filtre boyutu

Filtreler resimde çeşitli özelliklerin taranmasını sağlarlar. Bu özellikler dikey veya yatay çizgiler olabilir. Filtrelerin boyutları 2x2, 3x3, 4x4 gibi kare matris oluşturacak şekilde değerler alır.

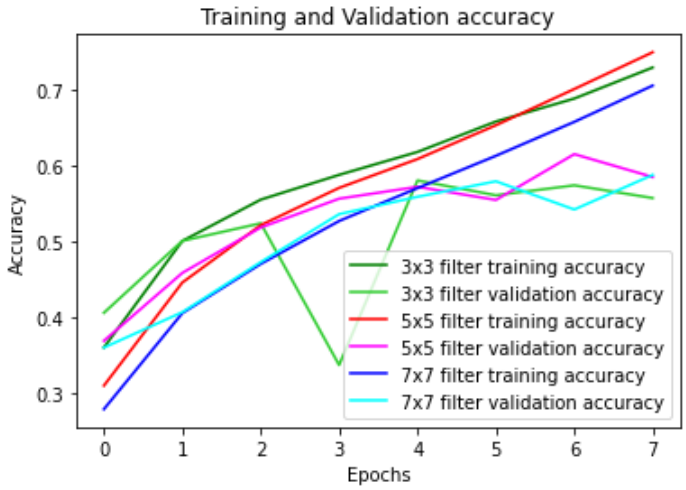
2x2 ve 4x4 filtrelerin kullanılması tercih edilmez çünkü simetrik bir dağılım göstermezler.

Deneme aşamasında 3x3 ve 5x5 filtrelerini karşılaştıracağız.

Sonuç: Filtrelerin boyutu, temel özelliklerin bulunmasında önemli bir rol oynar. Daha büyük boyutlu bir filtre, önemli özellikleri gözden kaçırabilir ve görüntülerdeki temel ayrıntıları atlayabilirken, daha küçük boyutlu bir filtre daha fazla bilgi sağlayarak daha fazla kafa karışıklığına neden olabilir.

Bu nedenle, filtrenin en uygun boyutunun belirlenmesine ihtiyaç vardır. Filtre boyutu model ve çalışılan verisetine bağlı oldukça değişken bir yapı göstermektedir fakat genellikle 3x3 filtreler tercih edilmektedir. Bunun nedeni de merkez ve çevresindeki birim her bir pikseli tarıyor olmasındadır.

Bizim projemizde 7x7 boyutlu filtre en kötü sonucu verirken 5x5 filtre ile en iyi sonucu elde ettik. Hem eğitim hem de test verisinde 5x5 filtre en iyi sonucu verdiğinden modelimizi kurmaya bu filtre ile devam edeceğiz.



Resim 7. Filtre boyutu ile accuracy arasındaki ilişki

4.4 Dropout oranı

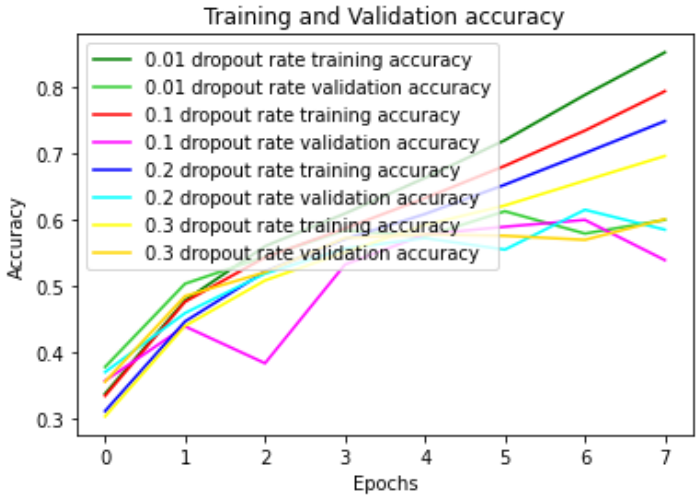
Dropout eğitim sırasında aşırı öğrenmeyi (overfit) engellemek için bazı nöronların işleme dahil edilmemesi demektir. Oranı da ne kadar nöronu işleme dahil etmediğimizi belirtir. İlk modelimizde oran 0.2'dir.

Not: Dropout yalnızca Dense katmanından sonra değil konvolüsyon katmanlarından sonra da yapılabilir. Fakat konvolüsyon katmanlarında overfitting durumunu önlemek için BatchNormalization tercih edilmektedir. Bunun nedeni dropout tekniğinin konvolüsyon katmanlarında daha az etkili olmasındandır.

Sonuç: Dropout ve BatchNormalization gibi yapılar overfit durumunu önlemeyi amaçlar. Bizim modelimizde de overfit durumu geçerlidir çünkü modelimiz eğitim verisinde iyi başarı oranı elde ederken, epoch sayısı artarsa %90 oranına bile çıkarken, test verisinde %60 üstüne çıkmamaktadır.

Dropout oranı azaldıkça eğitim esnasında daha az nöron atılır ve bu nedenle model veriyi daha iyi öğrenip, ezberlemeye yaklaşır.

Modelimizin test verisinde dropout oranının kayda değer bir etkisi bulunmamıştır bu nedenle 0.2 oranı kullanılmaya devam edilecektir.



Resim 8. Dropout oranı ile accuracy arasındaki ilişki

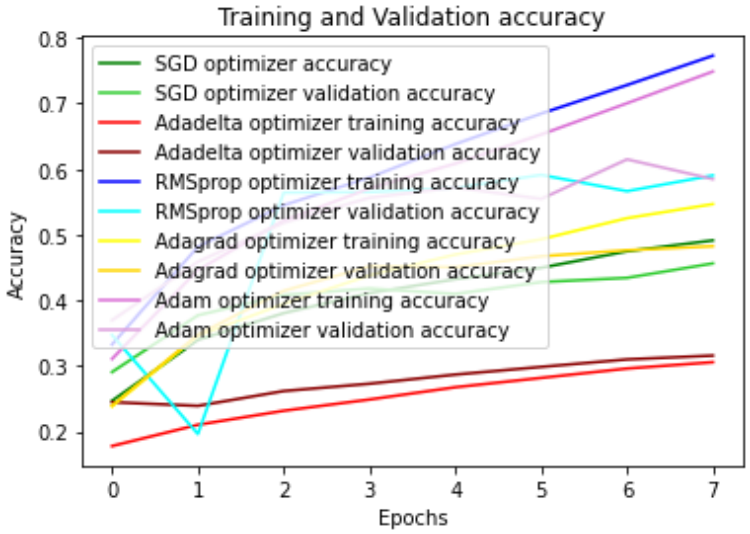
4.5 Optimizer

Optimizer yapay sinir ağlarında weight ve learning rate parametrelerini değiştirip loss değerlerini azaltmayı hedefleyen algoritmalardır. Bu algoritmalardan bazıları aşağıda verilmiştir.

1. Gradient Descent
2. Stochastic Gradient Descent (SGD)
3. Mini Batch Stochastic Gradient Descent (MB-SGD)
4. Adam
5. AdaDelta
6. RMSprop
7. Adaptive Gradient (AdaGrad)

Sonuç: Optimizasyon algoritmaları karşılaştırıldığında RMSprop veya Adam algoritmasının kullanılabileceği belirlenmiştir.

Aradaki fark fazla olmadığından Adam algoritması kullanılmaya devam etmiştir ve model böylece tamamlanmıştır.



Resim 9. Optimizer ile accuracy arasındaki ilişki

4.6 Aktivasyon fonksiyonu

Girdi değerlerine göre yapay sinir ağındaki nöronların üreteceği çıktı değerini belirleyen fonksiyonlardır. Örnekte relu kullanılmıştır.

4.7 Padding

Resmimizin boyutunu ayarlamamızı sağlar. Konvolüsyon katmanlarında uygulanan filtreler sonucu resmin boyutu küçülmektedir. Resmin boyutunu arttırmak için padding işlemi uygulanır. Aşağıda padding türleri verilmiştir.

Valid (No) Padding: Padding işlemi hiç gerçekleşmez. Resim filtre boyutuna oranla küçülür.

Same (Zero) Padding: Output ile input boyutunun aynı olmasını sağlar.

Constant Padding: 2. seçenekteki padding türü resmin etrafına piksel değerlerini 0 olarak eklerken burada kullanıcı istediği padding değerini yerleştirebilir.

Padding işlemi hiç olmazsa resim her bir konvolüsyon katmanından sonra daha çok küçülür ve artık özellik çıkartılamayacak hale gelir, bu nedenle same (zero) padding kullanılması tercih edilmiştir. (Bu parametrede herhangi bir değişiklik yapılmayacaktır.)

4.8 Loss fonksiyonu

Loss değerlerinin hesaplandığı fonksiyondur. Örneğimizde categorical_crossentropy kullanılmıştır. Bizim örneğimiz de sınıflandırma işlemi yaptığından ve sınıflandırmalarda en başarılı loss fonksiyonu bu olduğundan denemelerde bu parametreyi dikkate almayacağız.

Eğitim aşamasında değiştirilebilecek hiperparametreler aşağıda verilmiştir. Fakat hem süre hem de başarı açısından ilk belirlenmiş parametreler uygun olduğundan değiştirilmemelerine karar verilmiştir.

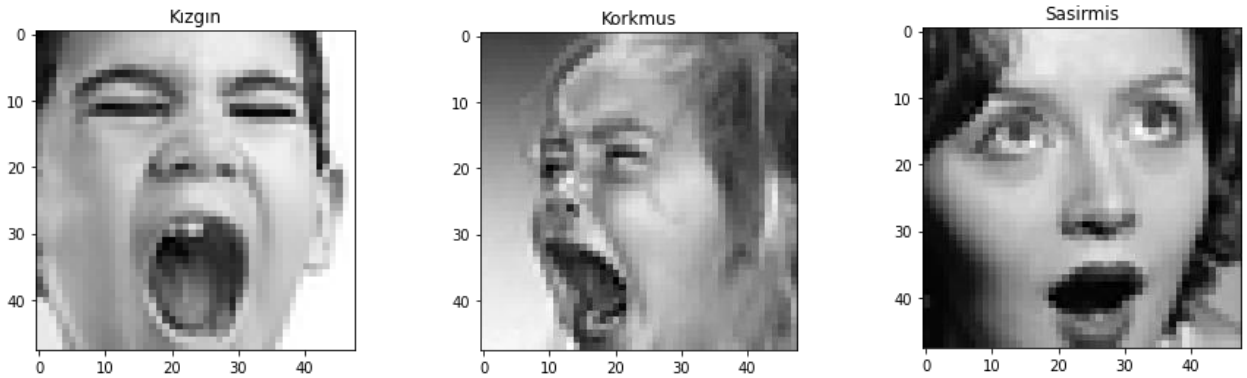
4.9 Batch size

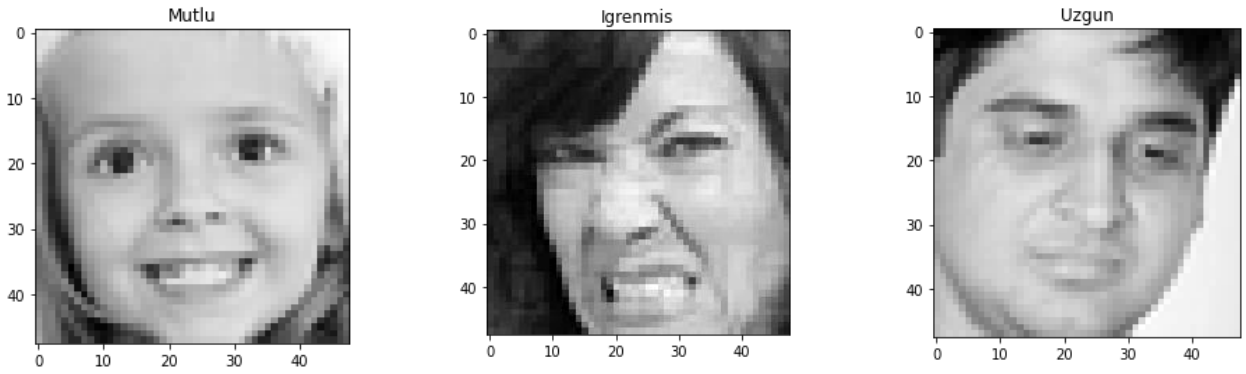
Model parametrelerini güncellemeden önce üzerinde çalışılacak örnek sayısını tanımlayan bir hiperparametredir.

4.10 Epoch

Öğrenme algoritmasının tüm eğitim veri kümesi boyunca çalışacağı sayıyı tanımlayan bir hiperparametredir.

Bir epoch, eğitim veri kümesindeki her örneğin dahili model parametrelerini güncelleme fırsatına sahip olduğu anlamına gelir. Aşağıda modelimizin test verisinden tahmin ettiği bazı örnekler verilmiştir.





Örnek resimlerden de görüldüğü üzere insanların bazı yüz ifadeleri birden fazla duygu barındırabilmektedir. Örneğin korkmuş olarak gösterilen kız resmi veri setinde üzgün olarak geçtiğinden yanlış bir sınıflandırma sayılmıştır fakat kıza bakılınca hem korktuğu hem de üzüldüğü anlaşılabilmektedir. Hem test verisindeki örneklere hem de gerçek zamanlı yüz ifadesi tahminin de görüldüğü üzere modelimiz başarılı bir sonuç vermektedir.

5. Gerçek Zamanlı Yüz İfadesinin Tahmini

Projemizin bu aşamasından sonra, modelin başarısını gerçek zamanlı görüntüler üzerinden ölçme fikrini mantıklı bulduk. Bir derin öğrenme modelinin gerçek zamanlı veriler üzerinde test verilerine yakın değer üretmesi önemli bir husustur, çünkü nihayetinde eğer bu model genel kullanıma sunulacak ise, veriseti dışındaki fotoğraflar veya videolar asıl başarı kıstası olarak sayılmaktadır, ayrıca modelin overfit olup olmama durumunu test etmek için de çok önemli bir yöntemdir.

Projemizi Google Colab ortamında geliştirdiğimiz için, OpenCV'nin hazır webcam fonksiyonlarından maalesef yararlanamadık. Colab'da kodu derleyip çalıştırma işlemi cloud ortamı aracılığı ile sağlandığından, bilgisayarın fiziksel bileşenlerine doğrudan erişimi bulunmamaktadır. Bu konuda bir araştırma yaptıktan sonra [6] numaralı kaynaktaki bilgilere ulaşıp, buradaki hazır kodlar vesilesiyle Colab ortamında webcam çalıştıracak fonksiyon ve araçlara erişmiş olduk.

Webcam bağlantısını sağlayan fonksiyonları tanınmladıktan sonra ise, yine Colab ortamının cloud aracığıyla çalışması dolayısıyla videodaki her bir kareyi (frame) Javascript

objesinden çözümleyecek fonksiyonu projemize ekledik. Bu fonksiyon da önceki paragrafta belirtilen kaynaktan sağlanmıştır.

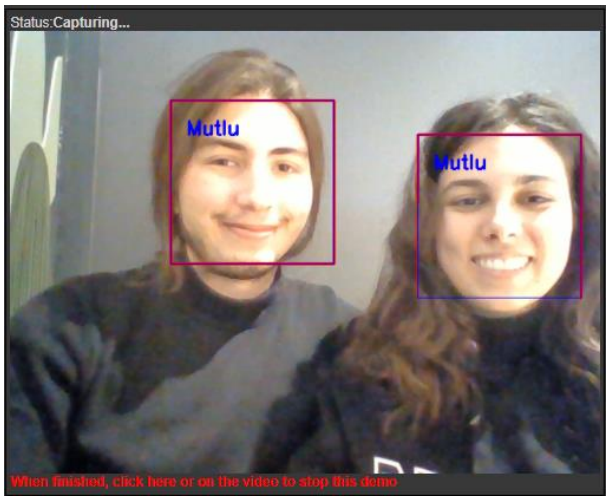
Duygu analizi yapan modelimizi webcam ile kendi yüzlerimizde deneyebilmemiz için, öncelikle bir face detection (yüz tanıma) özelliğine ihtiyaç duyduğumuzdan, [7] numaralı kaynakta belirttiğimiz bir OpenCV fonksiyonu olan CascadeClassifier'I kullandık.

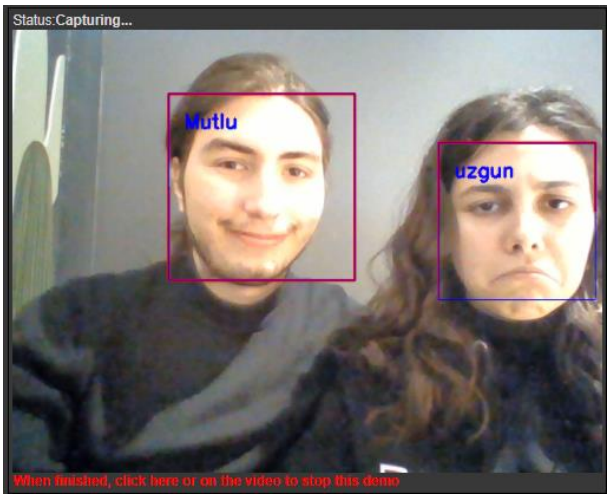
```
face_cascade = cv2.CascadeClassifier(cv2.samples.findFile(cv2.data.haarcas
cades + 'haarcascade_frontalface_default.xml'))
```

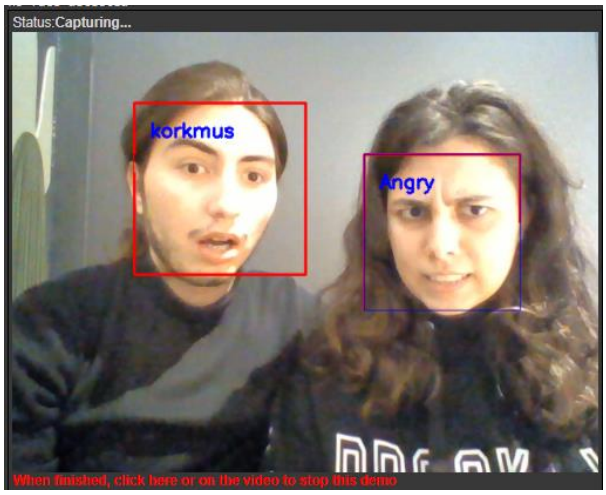
Yukarıda görünen bu kod parçası, 'haarcascade_frontalface_default.xml' dosyasından beslenerek bir görüntüdeki yüzlerin tespiti için kullanılmaktadır. Daha açık ifade etmek gerekirse, yukarıdaki xml dosyası, çeşitli fotoğraflarda bulunan yüz örneklerinin bilgilerini ifade eden bir dosyadır. Biz bu dosyayı kendimize referans alarak pratik bir şekilde yüz tanıma işlemini gerçekleştirebiliyoruz.

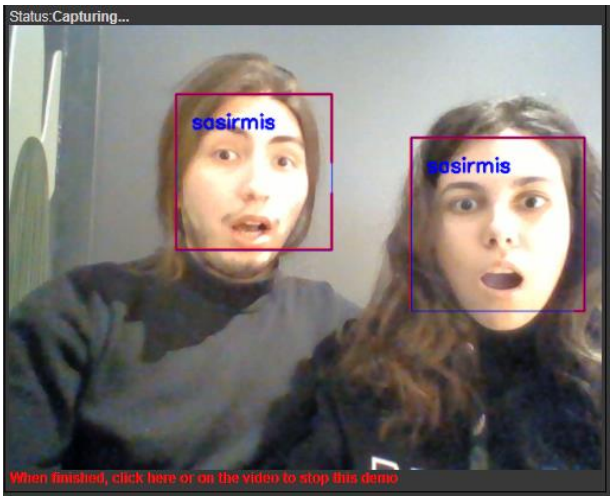
İşlemin son kısmında, sonsuz bir döngü ile webcamden gelen her frame'i gezmek suretiyle önce yüz tanımlaması yapıp, bu yüzleri içeren kapsayıcı çerçevelerin içerdiği görüntüleri de modelimize tahmin verisi olarak yolluyoruz. Modelimiz bu görüntülere göre her bir duygu için belli puanlamalar yapmaktadır, dolayısıyla en ağır basan duyguyu içeren veriyi referans alıp duygumuzu tahminlemiş oluyoruz. Buradaki önemli nokta, kapsayıcı çerçeve (bounding box) içerisindeki görüntünün tasarladığımız modelin input yapısı düzenine getirilmesidir. Yani aldığımız görüntüyü 48x48 piksel formatında ve grayscale yapısında modele tahminletmeye çalışıyoruz.

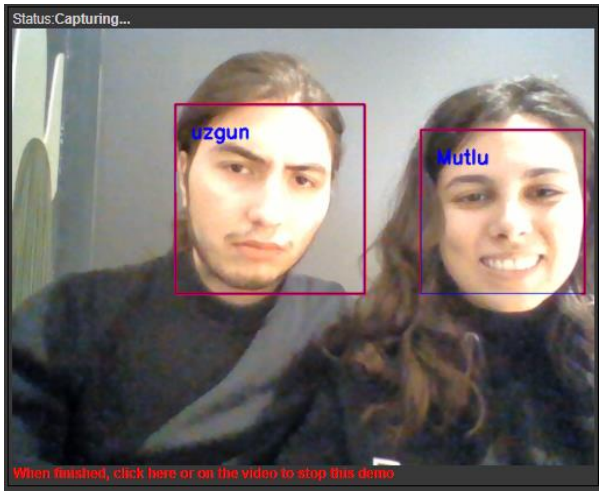
Böylelikle model tasarımı, optimizasyonu, başarı yorumlanması gibi kısımlara ek olarak bir gerçek zaman uygulamasıyla modelimizi mümkün olan çoğu açıdan değerlendirmiş olmaktayız. Gerçek zamanlı duygu analizi için kendi yüzlerimizi içeren örneklere sonraki sayfada yer vermiş bulunuyoruz.

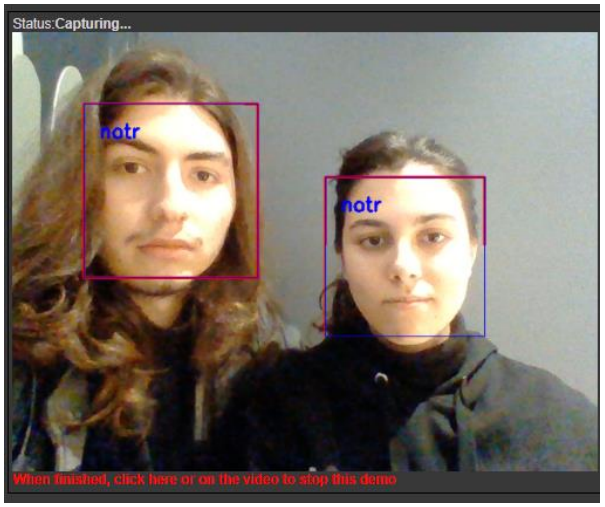

Status:Capturing...
Mutlu
Mutlu
When finished, click here or on the video to stop this demo


Status:Capturing...
Mutlu
uzgun
When finished, click here or on the video to stop this demo


Status:Capturing...
korkmus
Angry
When finished, click here or on the video to stop this demo


Status:Capturing...
sasirmis
sasirmis
When finished, click here or on the video to stop this demo


Status:Capturing...
uzgun
Mutlu
When finished, click here or on the video to stop this demo


Status:Capturing...
notr
notr
When finished, click here or on the video to stop this demo

6. Özdeğerlendirme Tablosu ve İş Bölümü

İstenen Özellik	Puan	Var	Açıklama	Tahmini Puan
1. Image Classification, Object Detection ve Image Segmentation kavramları arasındaki farklar nelerdir?	5	✓	Bu kısım hakkında ara raporda detaylı bir açıklama yapılmıştır. Kavramlar tanıtılıp, örnek üzerinden pekiştirilmiştir.	5
2. Ön Çalışma	5	✓	Bu kısım hakkında ara raporda detaylı bir açıklama yapılmıştır. Proje seçiminde taranan kaynaklardan ve nesne tanımanın kullanım alanlarından bahsedilmiştir.	5
3.1 Konu Belirleme ve Problemin Tanımı	5	✓	Konu yüz ifadesi tanınması olarak belirlenmiştir ve gerçek hayatta kullanılabilecek projelerden bahsedilmiştir.	5
3.2 Veriseti Hakkında Bilgi (Kullanılan veya Oluşturulan)	5	✓	Veri setinin linki, görütülerin özelliği, sınıfların sayısı ve neler oldukları, toplam kaç adet örnek içerdiği detaylı bir şekilde bahsedilmiştir. Ayrıca görselleştirmenin olması adına örneklerden bazıları eklenmiştir.	5
3.3.1 Kullandığınız Ortam	5	✓	Kullanılan ortam, kütüphane ve sürümleri detaylı bir şekilde anlatılmıştır. Sürümlerine proje içerisinden pip komuduyla ulaşılmıştır.	5
3.3.2 Derin öğrenme ile bir tanıma / tespit uygulaması geliştirilmesi	10	✓	Yüz tanıma uygulamasının modeli geliştirilmiş olup, model ve katman sayısı bilgileri kod üzerinde gösterilmiştir. Ayrıca kod üzerinden otomatik şema çıkaran yardımcı kütüphane sayesinde görselleştirme sağlanmıştır.	10
3.3.3 Faydalanılan çalışmalar ve farklılıklar	10	✓	Faydalanılan çalışmalar ve farklılıklardan detaylı bir şekilde bahsedilmiştir.	10
4. Deneysel Çalışmalar	25	✓	Eğitim işlemi detaylı bir şekilde anlatılmıştır, ayrıca parametre değişimleri uygun grafiklerle görselleştirilip seçimlerin neden yapıldığı da anlatılmıştır.	25
5. Özdeğerlendirme tablosu	10	✓	Özdeğerlendirme tablosu her bir maddesinde yeterli açıklamalar ile hazırlanmıştır.	10
6. Kaynakça	5	✓	Proje kapsamında başvurulan tüm kaynaklar kaynakça kısmında yer almaktadır.	5
7. Rapor Düzeni	10	✓	Raporda proje dökumanında istenen düzene uyulmuştur.	10
8. Kapak Sayfası	5	✓	Kapak sayfasında proje dökumanında istenilen tüm bilgiler yer almaktadır.	5

Projemizin iş bölümü kısmı hakkında,

Anıl verisetinin bulunması ve proje amacının uygunluğu kontrolü, verisetinin yazılım ortamına aktarımı ve bu setten eğitim, test verilerinin ayrılması, gerçek zamanlı yüz ifadesi tahmini bölümlerini üstlenmiştir.

Emel ise Modelin oluşturulması ve katman sayılarının, miktarlarının, modelde kullanılan filtre boyutlarının, dropout oranlarının ve optimizer fonksiyonlarının modele etkisinin belirlenmesi ve görselleştirilmesi, projede kullanılan kütüphane versiyonlarının kontrolü ve model şemasının oluşturulması kısımlarını üstlenmiştir.

Modelin seçimi hususunda grup üyeleri olarak öncelikle transfer learning kullanımı üzerinde durduk, fakat sonrasında kendimiz CNN modeli geliştirmeye karar verdik. Model başarısının ölçülmesi hususunda verisetinin validation verilerini kullanmanın yanı sıra, gerçek zamanlı yüz ifadesi tahmini kısmında da modelin katman sayısının ve diğer metriklerin başarıyı nasıl değiştirdiğine dair kazanımlar elde ettik.

Projeyi geliştirirken toplamda 14 saat süre harcanmıştır. Bu süre kapsamında ekip üyeleri eşit zaman harcamışlardır.

7. Kaynakça

[1] Image Classification vs. Object Detection vs. Image Segmentation, https://medium.com/analytics-vidhya/image-classification-vs-object-detection-vs-image-segmentation-f36db85fe81

[2] Object Detection vs Object Recognition vs Image Segmentation, https://www.geeksforgeeks.org/object-detection-vs-object-recognition-vs-image-segmentation/

[3] Object Detection in 2021: The Definitive Guide, https://viso.ai/deep-learning/object-detection/

[4] 83 Most Popular Computer Vision Applications in 2022, https://viso.ai/applications/computer-vision-applications/

[5] The many applications of emotion recognition, https://www.asmag.com/showpost/23883.aspx

[6] Colab Webcam, https://www.youtube.com/watch?v=YjWh7QvVH60

[7] OpenCV: Cascade Classifier, https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

[8] How to visualize deep learning models using visualkeras, https://analyticsindiamag.com/how-to-visualize-deep-learning-models-using-visualkeras/