# Sınıf İçi Maske Kontrolü

### Github Adresi

https://github.com/pythonderin/sinificiMaskeKontrolu

### Grubu oluşturan öğretmenlerin listesi

Fatih ERDAL, Tuba HARUNOĞLU, Serkan SAĞLAM, Evren TEMİZ, Turan TÜRKMEN

# Sınıf İçi Maske Kontrolü Analiz Raporu

### Kısa Özet

Covid-19 bulaşıcı bir hastalıktır. Hastalığın bulaşıcılığını en aza indirmek için en önemli kural hijyen, mesafe ve maskeden oluşmaktadır. Bu projede öğrencilerin sınıf içerisinde bir kamera tarafından izlenerek maske kontrolü yapılacaktır. Öğrencilerden herhangi birisi maskesini çıkarırsa, sistem bir müzik tonuyla uyarı verecektir.

### Problem Tanımı

Maske şart peki maskeyi kim kontrol edecek?

Kalabalık ortamlarda maske kullanılmaması covid-19 virüsünün bulaşma riskini artırır. Sınıflarda öğretmen ders anlatırken sürekli olarak öğrencilerin maskeli olup olmadığını kontrol edemez, bunun için maske kontrolünü otomatik olarak yapacak bir sisteme ihtiyaç vardır. Proje hazırlanması safhasında sadece sınıf içi olarak düşünülse de daha gelişmiş kamera ile cadde ve sokaklarda uygulanabilir şekilde düzenlenebilir. Hatta son yıllarda emniyet güçlerinin aktif olarak kullandığı insansız hava araçlarına ilave edilerek otonom olarak sosyal yaşamda maske takmayan kişileri tespit amacıyla kullanılabilir.

### Analiz Süreci

Ortam olarak VS Code kullanılmıştır. Kütüphane olarak OpenCV, YOLOv3 kullanılmıştır. Modeli eğitmek içinse Google Colab kullanılmıştır. YOLO nesne tespiti için kullanılan bir algoritmadır. Alternatif olarak TensorFlow veya PyTorch kütüphaneleri de kullanılabilir

#### İhtiyaç Analizi

Kalabalık ortamlarda virüs hapşırma, öksürme vb. nedenlerle havada hareket edebilmektedir. Virüs bulaşmış birisinden havaya virüs bırakılmaması ve virüsün bulaşmaması için maske takılması gerekmektedir. Sürekli olarak sınıftaki öğrencilerin maskesinin takılı olup olmadığını kontrol etmek zor bir süreçtir. Bu nedenle sınıftakilerin maske takıp takmadıklarını kontrol eden bir sisteme ihtiyaç duyulmuştur. Öğrenciler her ne kadar sınıf içinde daha kontrollü olsa da özellikle teneffüs zamanında kontrolsüz davranan öğrenciler bulunmaktadır. Okul güvenlik kameralarına entegre edilecek sistem ile daha etkin kullanılabilir.

#### İçerik Analizi

Proje yüz tanıma, maskeli ve maskesiz sınıflandırma, eğitim ve test verilerinden oluşmaktadır.

Proje amacının temelinde maske takmayan öğrencilerin tespit edilmesi ve uyarılması esasına dayanmaktadır. Sınıf ortamında öğretmen ders anlatımı sırasında sırtı sınıfa dönük olmadığı zamanda, öğrencilerin maskeleri uygun şekilde takmadığında veya teneffüste olan öğrencilerin maskelerini çıkarttığı durumlarda okul idare odasında sesli ikaz ve maske takmayan öğrenci sayılarını bildirmesi programın çıktıları olacaktır.

#### Durum Ortam Analizi

Yazılım geliştirilmesi sürecinde kullanılması görüntü işlemek için gereken kütüphaneler OpenCV, tensorflow.

#### Kullanıcı Analizi

Görüntü işlemeye meraklı kendini geliştirmeye yönelik olan herkes projeye erişim sağlayabilir.

# Sınıf İçi Maske Kontrolü Tasarım Raporu

### Kısa Özet

İlk önce videodan yüz algılayan bir model elde edilmesi gerekmektedir. Daha sonra algılanan bu yüz resmi üzerinde maskeli, maskesiz sınıflandırılması yapılması gerekmektedir. Bunun için de TensorFlow ve Keras gibi kütüphaneleri kullanılarak verilerin eğitimi yapılacaktır. Öncelikle internetten indirilen maskeli ve maskesiz resimler train ve test klasörü altında modele verilecek ve modelin bunları tahmin etmesi sağlanarak model eğitilecektir. Model belli bir başarıyı yakaladığında modelin eğitim işi sona erecektir. Eğitim işlemi sona erdikten sonra elde edilen modele bir video veya canlı kamera görüntüsü verilecektir. Verilen görüntüde, model tespit ettiği yüzleri maskeli veya maskesiz olmasına göre yeşil veya kırmızı bir dikdörtgen içerisine alarak “maskeli” veya “maskesiz” yazacaktır. Model yüzde maske olmadığını algıladığında bir uyarı sesi çalacaktır.

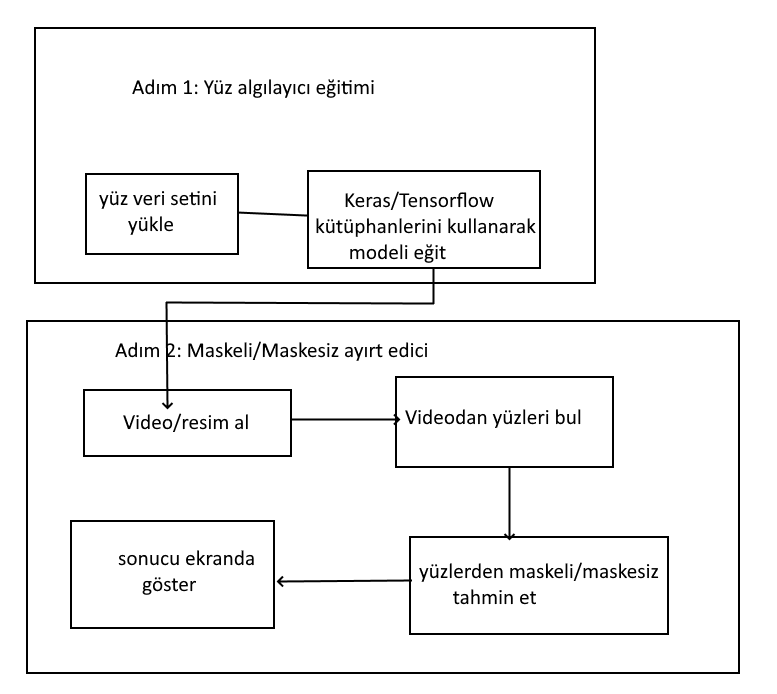
Yukarıda belirtilen model bir sınıf ortamına uygulandığında öğretmen sürekli olarak öğrencilerin yüzlerini kontrol etmesine gerek kalmayacaktır. Öğrencilerden birisi maskesini çıkardığında sistem otomatik olarak uyarı verecektir.

## Veri Tasarımı

## Ara yüz Tasarımı

Kullanıcı ara yüzüne ait tasarımların (Mock Up) yer aldığı bölümdür.

## Kod Tasarımı



## Zaman Çizelgesi

|  |  |
| --- | --- |
| **Yapılacak İşlem** | **Zamanı** |
| Analiz Raporu | 29.11.2020 |
| Tasarım Raporu | 6.12.2020 |
| Rapor gerçekleştirme aşaması | 13.12.2020 |
| Rapor test aşaması ve sunum | 20.12.2020 |
| Rapor githuba yüklenmesi | 23.12.2020 |

# Proje ismi Gerçekleştirme Raporu

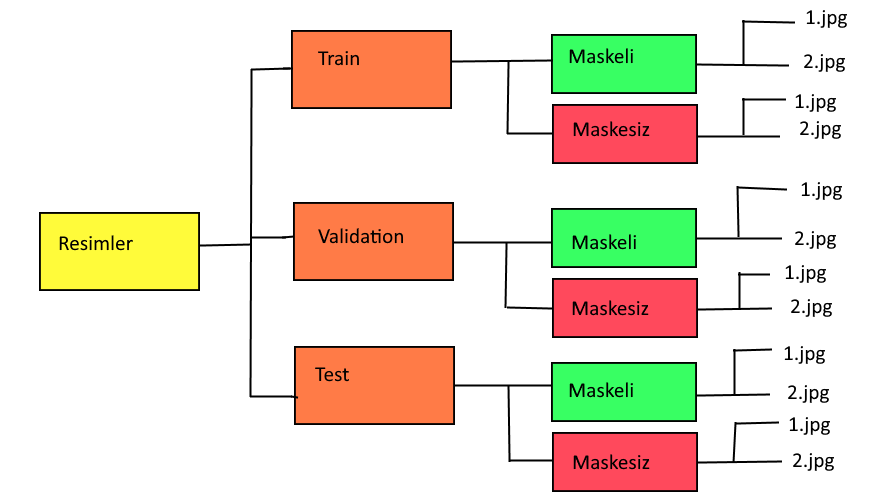
## Karşılaşılan Sorunlar ve Uygulanan Çözümler

Verileri eğitmek için ilk önce kendimize bir model seçmemiz gerektiğini düşündük. İlk olarak yolov4 yapısı ile verileri eğitip modeli test ettik, model test aşamasında video çok ağır çalışınca bu kez farklı bir model üzerine çalışmaya başladık. İkinci olarak TensorFlow üzerinde bir model geliştirip çalıştırdık.

## Proje Bileşenleri ve Görevleri

İlk önce projeyi 3 aşamaya ayırmak gerekiyor.

## 1-Veri toplama

İlk olarak modelimizin eğitilmesi gerekiyor. Bunun için internetten maskeli ve maskesiz yüz resimleri indiriyoruz. Bunun için dosya uzantısı jpg, jpeg veya png olanları indiriyoruz. Uzantısı jpeg veya png olanları jpg uzantısına dönüştürüyoruz. Son olarak bir dizin oluşturulur bu dizinin içerisinde de train, test ve valid dizinleri oluşturulur sonrasında bu dizinlerin de içerisine maskeli/maskesiz dizinleri oluşturarak maskeli olan resimler maskeli dizin içerisine, maskesiz olanlar ise maskesiz içerisine kaydedilir. Burada dikkat edilecek nokta verilerin %80 nini eğitim için harcanır yani elde edilen resimleri %80’i train klasörünün içerisine koyulur. Geri kalanlar ise test ve validation dizinlerine ayrılır. Resimlerin modele hangisinin maskeli, hangisinin maskesiz olduğunun belirtilerek verilmesi gözetimli öğrenmeye örnektir. Ayrıca resimlerin maskeli ve maskesiz sınıflandırılması yapay zekada bir classification modelinin kullanılacağını belirler. 

## 2-Veri Eğitimi (Model Oluşturma)

Veri eğitimi için yüksek performans gpu lu bir bilgisayar kullanılması gerektiği için bize ücretsiz bu hizmeti sunan Google Colab’ı kullandık. Google Colab’da bütün kütüphaneler hazır gelir. Kütüphaneleri import edip kullanmanız yeterlidir.

Bunun için ilk önce resimlerin bulunduğu klasör ziplenerek Google Drive’a atılır. Sonrasında Colab’dan dosya/ drive dan aç denilerek Colab’dan dosyaya erişim sağlanır. Sonrasında dosya unzip yapılır. Diğer gerekli açıklamalar ve python kodları GitHub’daki maskeEgitim.ipynb dosyasında açıklamalarıyla beraber yer almaktadır.

## 3-Modeli Test Etme

Modeli test etmek amacıyla bilgisayarımızda çalıştırılması gerekli olan maske.py dosyasıdır. Bu dosyayı çalıştırmak için Tensorflow 2 veya üzeri bir versiyon, Playsound, Numpy, Time kütüphaneleri yüklü olmalıdır. Modeli test ederken ilk olarak model OpenCV’nin haarcascade\_frontalface\_default.xml dosyası ile kameradaki yüzleri yakalamaya çalışmaktadır, sonrasında ise yakaladığı yüzler üzerinde modeli uygulamaktadır.

## Github Yükleme Süreci

Github için öncelikle bir github hesabı oluşturduk. Sonrasında github içerisinde dosyalarımızı git komutları ile yüklemeye çalıştık. Yükleme esnasında model eğitimi sonucu elde edilen dosyamızın (model.h5) boyutu 263Mbyte olunca github bu dosyanın boyutunun 100Mbytedan fazla olduğunu ve yükleme yapamacağını belirten bir uyarı verdi. Bunun üzerine internetten arama yaptık. Arama sonucu Git Large File Storage (<https://git-lfs.github.com/>) aracı ile dosyayı github repositorisine yükleneceğini tespit etmiş olduk ve github komutları ile bütün dosyalarımızı githuba yükledik.

# Proje ismi Gerçekleştirme Raporu

## Karşılaşılan Sorunlar ve Uygulanan Çözümler

Projenin gerçekleştirme sürecine karşılaşılan sorunlar ve gidermek için uygulanan çözümlerin ifade edildiği bölümler.

## Proje Bileşenleri ve Görevleri

Programa ait dökümantasyonun taslak halininde ortaya çıktığı bölümdür.

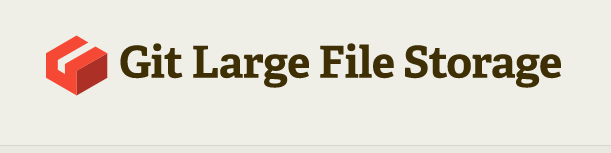
## Github Yükleme Süreci

Yazılım kaynak kodunun github profilinde paylaşılması farklı bir kullanıcı tarafından başka bir platforma yüklenmesi sürecinde yapılması gerekenlerin yer aldığı bölümdür. Şu anda bütün dosyalar <https://github.com/pythonderin/sinificiMaskeKontrolu> adresinde mevcuttur.

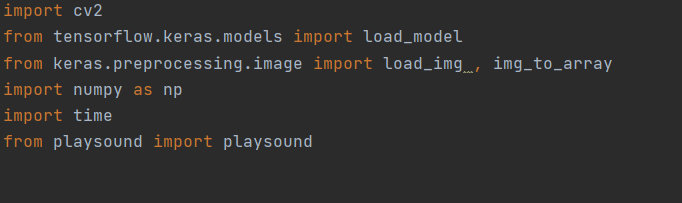
# Proje ismi Test Raporu

## Karşılaşılan Sorunlar ve Uygulanan Çözümler

İlk olarak modeli yolov4 üzerinde eğittik ancak eğitim sonunda çalıştırdığımız dosya bilgisayarı çok fazla kastı. Bu durumda sistem değiştirmeye karar verdik ve opencv kütüphanesinin yüz tespiti için kullandığı haarcascade modelinin tespit ettiği bölge üzerinde maske var mı yok mu tespit etme yoluna gittik.

Diğer bir problemi de eğitim sonucunda elde edilen maske.h5 dosyasının boyutunun 100Mbytedan yüksek olması sebebiyle github modelin upload edilmesine izin vermedi. Sonrasında git large file storage kurulumu yaparak model dosyamızı upload edebildik.

## Test Sürecinde Kullanılan Modüller (Varsa)



Model test edilirken test edilecek bilgisayarda opencv, numpy, time, playsounda ve tensorflow 2 veya kütüphaneleri bilgisayarda kurulu olmalıdır. Model test edilecek dosyanın adı maskeTest.py’dir.

## Değerlendirme Kriterleri

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **KRİTERLER** | Ekip Üyeleri | Eğitmen 1 | Eğitmen 2 | Eğitmen 3 | Ortalama |
| 1. Analiz Raporunun Tamamlanması |  |  |  |  |  |
| 1. Tasarım Raporunun Tamamlanması |  |  |  |  |  |
| 1. Gerçekleştirim Raporunun Tamamlanması |  |  |  |  |  |
| 1. Gantt Diagramı |  |  |  |  |  |
| 1. Arayüz tasarımı |  |  |  |  |  |
| 1. Veri Tasarımı-Sınıf Tasarımı |  |  |  |  |  |
| 1. Kullanıcı Yardım Dökümanı |  |  |  |  |  |
| 1. Programın Çalıştırılması |  |  |  |  |  |
| 1. Yazılım Test Çalışması |  |  |  |  |  |