T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BSM 498 BİTİRME ÇALIŞMASI

EVRİŞİMSEL SİNİR AĞI İLE DEEPFAKE TESPİTİ

G181210072 – Baha BÜÇGE

Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ
Tez Danışmanı : (Doç.) Dr. Devrim AKGÜN

2021-2022 Bahar Dönemi

T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

EVRİŞİMSEL SİNİR AĞI İLE DEEPFAKE TESPİTİ

BSM 498 - BİTİRME ÇALIŞMASI

Baha BÜÇGE

Fakülte Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ

Bu tez 06 / 06 / 2022 tarihinde aşağıdaki jüri tarafından oybirliği / oyçokluğu ile kabul edilmiştir.

Doç. Dr. Devrim Akgün	Prof. Dr. Cüneyt Bayılmış	Dr. Öğr. Üyesi Serap Çakar
Jüri Başkanı	**Üye**	**Üye**

ÖNSÖZ

Günümüzde yapay zeka ve derin öğrenme alanlarındaki teknolojilerin gelişmesiyle ve çalışmaların artmasıyla elde edilen kazanımların yanında bu gelişmelerin beraberinde getirdiği sorunlarla da karşı karşıyayız. Bu sorunlardan bir tanesi de Deepfake olarak adlandırdığımız, fotoğraf ve video içeriklerini manipüle ederek insanların sahte görüntülerini elde etmeyi sağlayan bir teknolojinin kötüye kullanımı. Manipüle edilerek oluşturulmuş içerikleri tespit eden bir evrişimsel sinir ağı kullanarak bu sorunun önüne geçilebilir.

İÇİNDEKİLER

ÖNSÖZ……..... iv
İÇİNDEKİLER..... v
SİMGELER VE KISALTMALAR LİSTESİ..... vii
ŞEKİLLER LİSTESİ..... viii
ÖZET..... ix

BÖLÜM 1.
GİRİŞ..... 1
1.1. Projenin Amacı, Kapsamı ve Hedefi..... 1
1.1.1. Projenin amacı ve kapsamı..... 1
1.1.2. Projenin hedefi..... 2
1.2. Daha Önce Yapılmış Çalışmalar..... 2

BÖLÜM 2.
TEKNİK ARKA PLAN..... 3
2.1. Yapay Sinir Ağları..... 3
2.2. Evrişimsel Sinir Ağları..... 4
2.2.1. Evrişim (Konvolüsyon) işlemi..... 4
2.3. Çekişmeli Üretici Ağlar..... 5
2.4. TensorFlow..... 6
2.4.1. TensorFlow.js..... 7
2.5. Keras..... 7
2.6. OpenCV..... 7

BÖLÜM 3.
VERİ SETİ VE YAPAY SİNİR AĞI..... 8
3.1. Veri Seti..... 8
3.2. Yapay Sinir Ağı..... 11
3.3. Elde Edilen Çıktılar..... 12

BÖLÜM 4.
MODELİN İNTERNET ÜZERİNDEN SERVİS EDİLMESİ..... 13
4.1. UI/UX Tasarımları..... 13
4.2. Web Servis Teknik Tasarımı..... 14

BÖLÜM 5.
SONUÇLAR VE ÖNERİLER……..…….... 15

KAYNAKLAR…….. 16
ÖZGEÇMİŞ…….…….. 18

BSM 498 BİTİRME ÇALIŞMASI DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI…… 19

SİMGELER VE KISALTMALAR LİSTESİ

NN	: Sinir Ağı
CNN	: Evrişimsel Sinir Ağı
GAN	: Çekişmeli Üretici Ağ
js	: JavaScript
JSON	: JavaScript Object Notation
AWS	: Amazon Web Services
UI	: Kullanıcı Arayüzü
UX	: Kullanıcı Deneyimi

ŞEKİLLER LİSTESİ

Şekil 1.1.	Deepfake uygulama örneği……	1
Şekil 2.1.	Yapay sinir ağı örneği……...	3
Şekil 2.2.	Evrişimsel sinir ağı örneği……	4
Şekil 2.3.	Konvolüsyon işlemi……	5
Şekil 2.4.	Çekişmeli üretici ağ örneği……	6
Şekil 3.1.	FaceForensics veri örnekleri……	8
Şekil 3.2.	Videolardan sahne kesmek için kullanılan kod örneği……	9
Şekil 3.3.	Fotoğraflardaki kişilerin yüzlerini kırpmak için kullanılan kod örneği……	10
Şekil 3.4.	Modelin görsel şeması……	11
Şekil 3.5	Eğitim ve doğrulama aşamalarının doğruluk oranı grafiği..	12
Şekil 3.6.	Eğitim ve doğrulama aşamalarının kayıp oranı grafiği……	12
Şekil 4.1.	İnternet sitesi fotoğraf yükleme bölümü……...	13
Şekil 4.2.	İnternet sitesi sonuç bildirme bölümü……...	14
Şekil 4.3.	JavaScript'te modelin yüklenmesi…….	14

ÖZET

Anahtar kelimeler: Yapay Sinir Ağları, Evrişimsel Sinir Ağları, Derin Öğrenme, Deepfake Tespiti

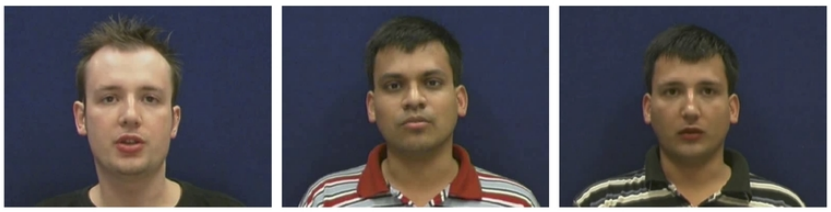
Günümüzde teknoloji artık hayatımızın her alanına temas ediyor. Problemlerimize çözüm bulan, işlerimizi kolaylaştıran teknoloji aynı zamanda karşımıza kötü amaçlı kullanımlar ile de çıkıyor. Bu durumun bir örneği de yapay zekanın ortaya çıkması ve son dönemde oldukça hızlı bir şekilde gelişmesiyle beraber karşımıza çıkan etik problemler. Deepfake teknolojisi ile oluşturulan resim ve videolar, içeriğinde bulundurduğu kişilerin rızası olmadan kişinin söylemlerini çarpıtma, pornografi, kişiyi küçük düşürme ve benzeri amaçlarla kötüye kullanılabiliyor.

Bu noktada, yine yapay zekayı kullanarak bu resimlerin ve videoların bilgisayar ile oluşturulduğunu tespit etmek bu probleme karşı önemli bir çözüm oluşturacaktır. Bu proje ile amaçlanan da bir yapay sinir ağı oluşturarak imaj dosyalarını gerçek ya da sahte olarak kategorilendirebilmektir. Bu amaç doğrultusunda internette yayınlanmış veri setleri kullanılmış, daha önce yapılmış çalışmalar da göz önüne alınarak en uygun model belirlenip gerçeklenmeye çalışılmıştır.

Çalışma sonucunda temel bir derin öğrenme modeli, gerçek dünya örneklerine uygun ancak kısıtlı bir veri seti ile deepfake tespitinde %75 doğruluk oranıyla tespit yapılabileceği sonucuna ulaşılmıştır.

BÖLÜM 1. GİRİŞ

Yapay zeka alanının her gün gelişmesiyle ve teknolojinin internet aracılığıyla herkes tarafından erişilebilir olmasıyla beraber karşımıza önemli bir sorun çıkmaktadır. Deepfake olarak adlandırılan manipüle edilmiş görsel içerikler, donör kişinin yüzünün hedef içerikteki kişinin yüzünün yerine yerleştirmesi ile oluşturulmaktadır.



(a) Original Donor (b) Original Target (c) Face Swapped

Şekil 1.1. Deepfake uygulama örneği

Deepfake teknolojisi siyasetçilerin yanıltıcı görsellerini oluşturmak, ünlülerin yüzlerini cinsel içeriklere yerleştirmek, insanlara şantaj yapmak için görseller oluşturmak gibi kötü amaçlarla kullanılabilmektedir. Bu eylemlerin topluma ve bireylere önemli zararlar vermesinin önüne geçmek için manipüle edilmiş içerikleri tespit ederek sahte olarak etiketlemek gerekmektedir.

Bu proje, sahte içerikleri açığa çıkarmak için derin öğrenme yöntemleri ile bir yapay sinir ağı eğitilmesini ve eğitimi tamamlanmış modeli internet üzerinden yayınlayarak siteye kullanıcılar tarafından yüklenecek içerikleri gerçek ya da sahte olarak etiketlemeyi amaçlamaktadır.

1.1. Projenin Amacı, Kapsamı ve Hedefi

1.1.1. Projenin amacı ve kapsamı

Projenin amacı, Deepfake algoritmaları ile manipüle edilmiş videoları işleyerek bu videoları gerçek ya da sahte olarak kategorilendirebilmektir.

1.1.2. Projenin hedefi

Projenin hedefi, Deepfake ile manipüle edilerek oluşturulmuş videolardan ve manipüle edilmeden önceki hallerinden oluşan bir veri setini kullanarak bir evrişimsel sinir ağını minimum hata payı ile eğitmek ve elde ettiği modeli bir internet sitesi üzerinden kullanıma açmaktır.

1.2. Daha Önce Yapılmış Çalışmalar

Deepfake tespiti üzerine daha önce birçok çalışma yapılmış olup bunların en yaygını AWS, Facebook, Microsoft ortaklığıyla Kaggle'da [1] yayınlanan Deepfake Detection Challenge [2] olabilir. Bu yarışma sonucunda en iyi 5 modelin ortak özellikleri zekice veri arttırma yöntemlerini ve önceden eğitilmiş EfficientNet ağlarını kullanmalarıydı.

Bunun dışında 11 Şubat 2022'de, Dünya'nın farklı yerlerinden akademisyenlerin beraber yayınladığı, hem deepfake oluşturmada hem de tespit etmede kullanılan farklı yöntemleri ve farklı veri setlerini detaylıca inceleyen bir çalışmada [3] görüntü manipülasyon tekniklerinin henüz altın çağına ulaşmadığını ancak yine de manipüle edilmiş görüntüleri tespit etmek için fazlaca efor sarf edilmesi gerektiği sonucuna varılmıştır.

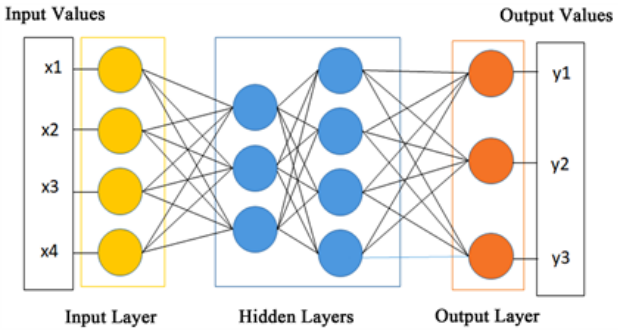
MesoNet adıyla yapılmış başka bir çalışmada [4] ise özellikle videolarda ağız ve göz kısımlarına yoğunlaşmanın ve basit ağlar ile eğitim yapmanın daha verimli sonuçlara yol açtığı belirlenmiştir.

BÖLÜM 2. TEKNİK ARKA PLAN

2.1. Yapay Sinir Ağları

Yapay sinir ağları konsepti, insan beyninin çalışmasını temel almaktadır. Sinir ağları bir girdi katmanı, bir veya daha fazla gizli katman ve bir çıktı katmanı içeren ve girdi katmanından alınan değerleri işleyerek sınıflandıran çok katmanlı ağlardır.

İlk katman olan girdi katmanı verileri alır ve sonraki katmana iletir. İkinci katman olan gizli katman birbirine bağlı nöronlardan oluşur. Her iki nöron arasındaki bağlantıya tanımlı eğitilebilir bir ağırlık bulunmaktadır. Ayrıca her bir nöron bias adı verilen bir girdiye sahiptir.



Şekil 2.1. Yapay sinir ağı örneği

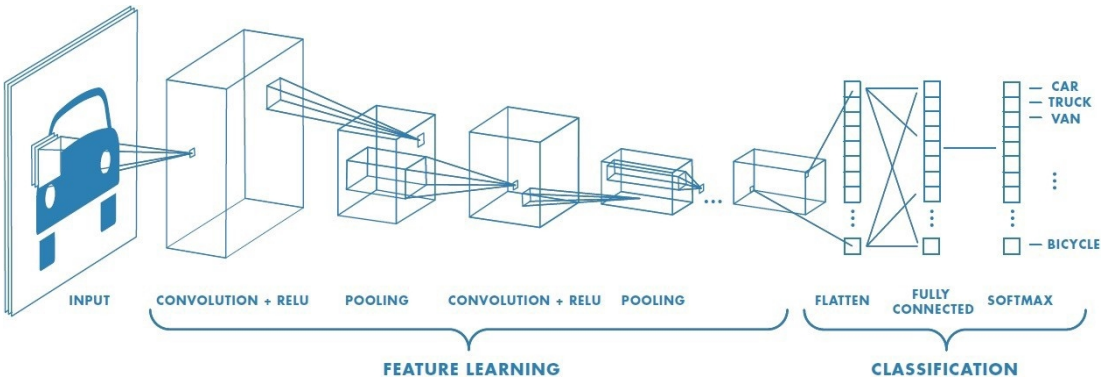
Çıktı katmanında ise model kaç kategoride sınıflandırma yapacaksa o kadar çıktı bulunur. Öğrenme safhasında amaç, yapılan tahmini beklenen sonuç ile kıyaslayıp bir kayıp değeri üretmek ve bu değeri de kullanarak aktivasyon fonksiyonu ile bağlantılar üzerindeki ağırlıkları güncellemektir.

Girdileri işleyerek bir çıktı tahmini üretme işlemi ileri beslemedir. Her katman bir önceki katmandan verileri alarak kendisinden sonraki katmana iletir. Geri besleme ise tahminleri beklenen sonuçlar ile kıyaslayarak elde edilen kayıp değerini kullanarak geriye doğru ağırlıkları güncelleme işlemidir.

2.2. Evrişimsel Sinir Ağları

CNN, yani Evrişimsel Sinir Ağları özel bir tür derin öğrenme mimarisidir. İlk olarak 1979 yılında Kunihiko Fukushima tarafından neocognitron adıyla sunulmasıyla ortaya çıkmıştır. Daha sonra gelişerek günümüze kadar gelmiştir.

Temel bir CNN modelinin yapısı üç tür katman içerir: evrişimsel katman, havuzlama katmanı ve tam bağlı katman.



Şekil 2.2. Evrişimsel sinir ağı örneği

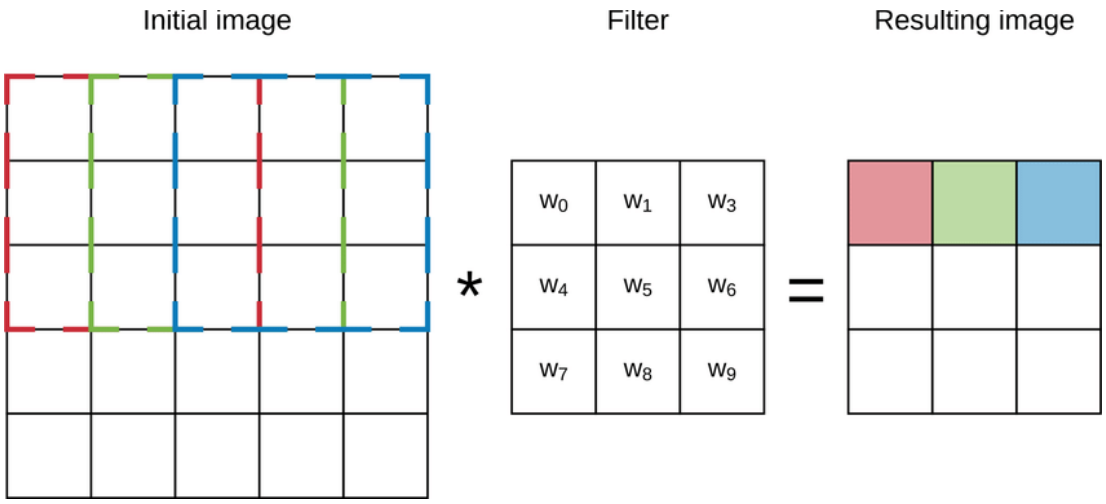
Evrişimsel katmanın amacı özellik çıkarma işlemini gerçeklemektir. Evrişimsel işlemde bir sayı dizisi (çekirdek) özellik haritası çıkarmak için girdi (tensör) boyunca uygulanır. Farklı sayıda çekirdek kullanılarak isteğe bağlı sayıda özellik haritası elde edilebilir. Eğitim sırasında evrişimsel işlem ileri besleme olarak adlandırılır. Geri besleme sırasında ise gradient descent optimizasyon yöntemiyle eğitilebilir parametreler ileri besleme sırasında ortaya çıkan kayıp değerlerine göre güncellenir.

2.2.1. Evrişim (Konvolüsyon) işlemi

Girdi olarak alınan resim iki boyutlu bir tensöre dönüştürülüp konvolüsyon katmanına verildiğinde bu katmanda tanımlanan çekirdekler tensör boyunca uygulandıktan sonra geri besleme ile üzerlerindeki ağırlıkları güncellerler.

Çekirdeğin girdi üzerinde dolaşması sırasında çekirdeğin boyutuna bağlı olarak ortaya çıkan yeni tensörün boyutu küçülecektir. $(N \times N)$ boyutlu bir çekirdek için

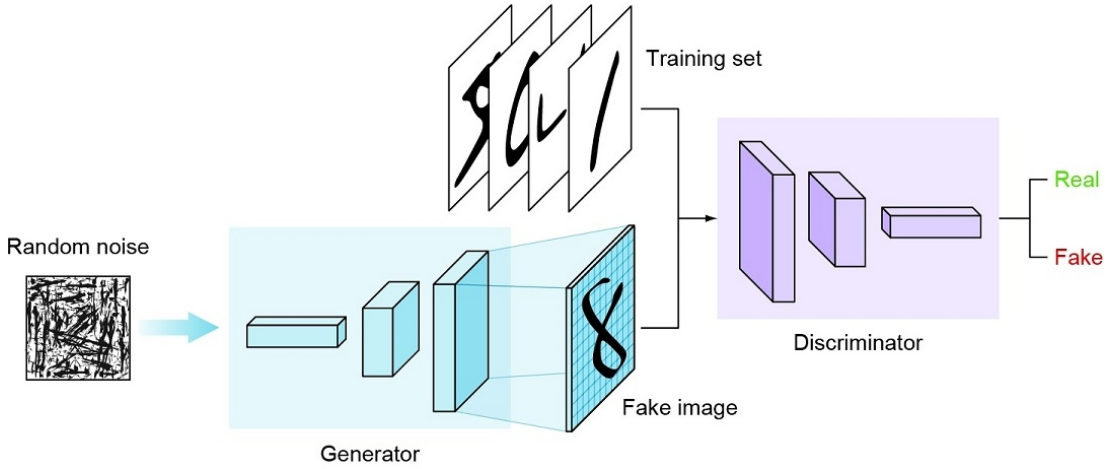
küçülme miktarı *(N – 2)*'dir. Bunun önüne geçmek amacıyla konvolüsyon katmanı eklenirken *padding* parametresi *same* olarak ayarlandığında, girdi tensörü konvolüsyona verilmeden önce tensöre sıfırlardan oluşan sütun ve satırlar eklenir. Böylece sonuca etki edilmeden çıktının boyutunun asıl girdi ile aynı olması sağlanır.



Şekil 2.3. Konvolüsyon işlemi

2.3. Çekişmeli Üretici Ağlar

Çekişmeli üretici ağlar (GAN), üretici modellere belirli bir veri türünden gerçekçi örnekler üretmesini öğretmek için kullanılan bir araçtır [5]. Temelde çekişmeli üretici ağlar iki yapar sinir ağının birleşimidir: Üretici (G) ve ayıklayıcı (D). Bu iki yapay sinir ağı dinamik bir minimaks oyununda birbirleriyle rekabet içindedir. Burada amaç, üretici ağ sahte örnekler oluştururken ayıklayıcı ağın üretilen örneklerin hangilerinin sahte, hangilerinin gerçek olduğunu tespit etmeye çalışmasıdır. Eğer iki model birbiriyle uzun süre rekabet ederse en sonunda epeyce gelişeceklerdir.



Şekil 2.4. Çekişmeli Üretici Ağ Örneği

Üretici ve ayıklayıcı ağların matematiksel minmaks optimizasyonu aşağıdaki gibidir:

$$\begin{aligned} G^* \in & \arg\min\max V(G,D) \\ = & \arg\min\max E_{X\sim P_{data(X)}}\left[\log\left(D(X)\right)\right] \\ & + E_{Z\sim P_{Z(Z)}}\left[1-\log\left(D\left(G(Z)\right)\right)\right] \end{aligned} \qquad (2.1)$$

Burada *Z* üretici girdisini *G(Z)*, *Pz* olasılık dağılımını, *X* üreticinin çıktısını, *D(X)* *X*'in eğitim verisindeki *(Pdata)* kategoriye ait olup olmadığı ihtimalini temsil eder. (Denklem 2.1)

Son zamanlarda tasarımları, kayıpları ve eğitim tekniklerini geliştirmek adına DCGAN [6], WGAN [7], PGGAN [8], BigGAN [9], and Style-GAN [10] gibi çeşitli çekişmeli üretici ağlar üretilmiştir.

2.4. TensorFlow

TensorFlow [11], uçtan uca ve açık kaynak bir makine öğrenimi platformudur. Kapsamlı ve esnek araçlardan, kütüphanelerden ve topluluk kaynaklarından oluşan ekosistemi ile araştırmacıların makine öğrenmesinde son teknolojiyi zorlamasını, geliştiricilerin de kolayca makine öğrenmesi uygulamaları oluşturmalarını ve dağıtmalarını sağlamaktadır.

2.4.1. TensorFlow.js

Tensorflow.js [12], JavaScript için oluşturulmuş bir makine öğrenmesi kütüphanesidir. Direkt olarak internet tarayıcısı üzerinden makine öğrenmesi modelleri geliştirmeyi, Python ile geliştirilmiş modelleri içe aktararak kullanmayı, halihazırda var olan modelleri kendi verileriniz ile tekrar eğitmeyi sağlar.

2.5. Keras

Keras [13], Python'da yazılmış açık kaynaklı bir sinir ağı kütüphanesidir. Derin sinir ağları ile hızlı deney yapabilmek için tasarlanan bu cihaz kullanıcı dostu, modüler ve genişletilebilir olmaya odaklanıyor. Katmanlar, kayıp fonksiyonları, aktivasyon fonksiyonları, optimizörler gibi yapay sinir ağı oluşturmada kullanılan çok sayıda bloku bünyesinde bulundurmaktadır. Standart yapay sinir ağlarının yanında evrişimsel ve yinelemeli sinir ağlarını da desteklemektedir.

2.6. OpenCV

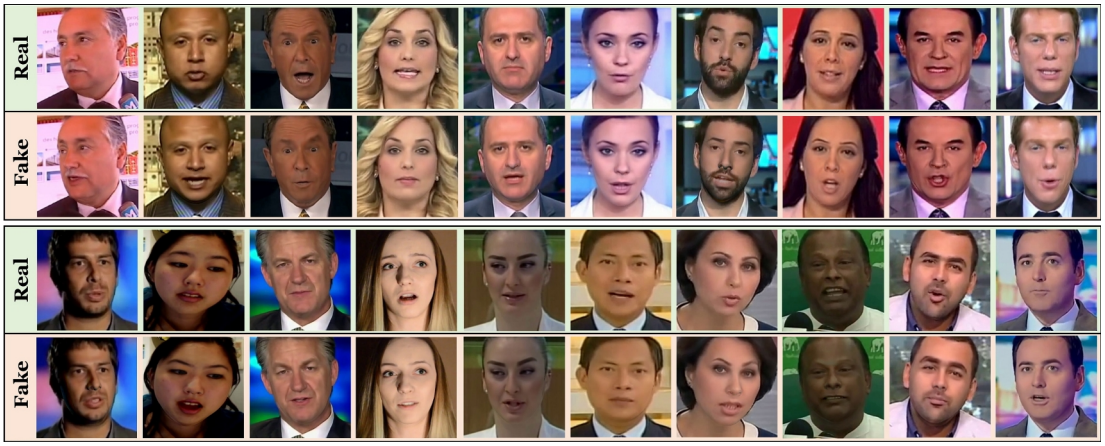
OpenCV [14] (Open Source Computer Vision Library, anlamı Açık Kaynak Bilgisayar Görüsü Kütüphanesi) başlıca görüntü işleme ve bilgisayar görmesi amaçlarına hizmet eden bir Python kütüphanesidir. Bu çalışmada da görüntü işleme işlevi için, özel olarak görüntülerdeki insan yüzlerini tespit etmek ve görüntüyü kırpmak için kullanılmıştır.

BÖLÜM 3. VERİ SETİ VE YAPAY SİNİR AĞI

Etkili bir model oluşturmak için eğer mümkünse çok sayıda iyi örnekten oluşan bir veri seti kullanmak, ezberlemenin ve aşırı uyma'nın (overfitting) önüne geçmek için yöntemler kullanmak ve hiperparametreleri iyi belirlemek önemlidir. Bu bölümde seçilen veri seti, kullanılan yöntemler ve modelin tasarımı incelenecek.

3.1. Veri Seti

Modeli eğitmek için içerisinde oldukça fazla sayıda ve farklı yöntemlerle manipüle edilerek oluşturulmuş videolar olan FaceForensics++ [15] veri seti kullanılmıştır. Python kullanılarak bu veri setinin içerisindeki videoları karelerine ayırıp birkaç karesi seçilerek kaydedildikten sonra yine Python kullanarak seçilen karelerdeki insan yüzleri kesilip veri setinin son hali oluşturulmuştur.



Şekil 3.1. FaceForensics veri örnekleri

FaceForensics++'ten indirilen bin adet videodan oluşan veri setinde bir döngü ile gezerken her videodan üç sahne kesip yeni bir dizine kaydederek resimlerden oluşan bir veri seti oluşturulmuştur.

```
for j in range(1,1001):
    # Opens the Video file
    cap= cv2.VideoCapture(real_videos_dir+str(j)+').mp4')
    frame_count = cap.get(cv2.CAP_PROP_FRAME_COUNT)

    i=0
    while(i <= frame_count):
        i+= round(frame_count / 3)
        for a in range(round(frame_count / 3)):
            ret, frame = cap.read()
        if ret == False:
            break
        cv2.imwrite(dataset_real_dir+str(j)+'_'+str(i)+'.jpg',frame)

    cap.release()
    cv2.destroyAllWindows()

for j in range(1,1001):
    # Opens the Video file
    cap= cv2.VideoCapture(fake_videos_dir+str(j)+').mp4')
    frame_count = cap.get(cv2.CAP_PROP_FRAME_COUNT)

    i=0
    while(i <= frame_count):
        i+= round(frame_count / 3)
        for a in range(round(frame_count / 3)):
            ret, frame = cap.read()
        if ret == False:
            break
        cv2.imwrite(dataset_fake_dir+str(j)+'_'+str(i)+'.jpg',frame)
```

Şekil 3.2. Videolardan sahne kesmek için kullanılan kod örneği

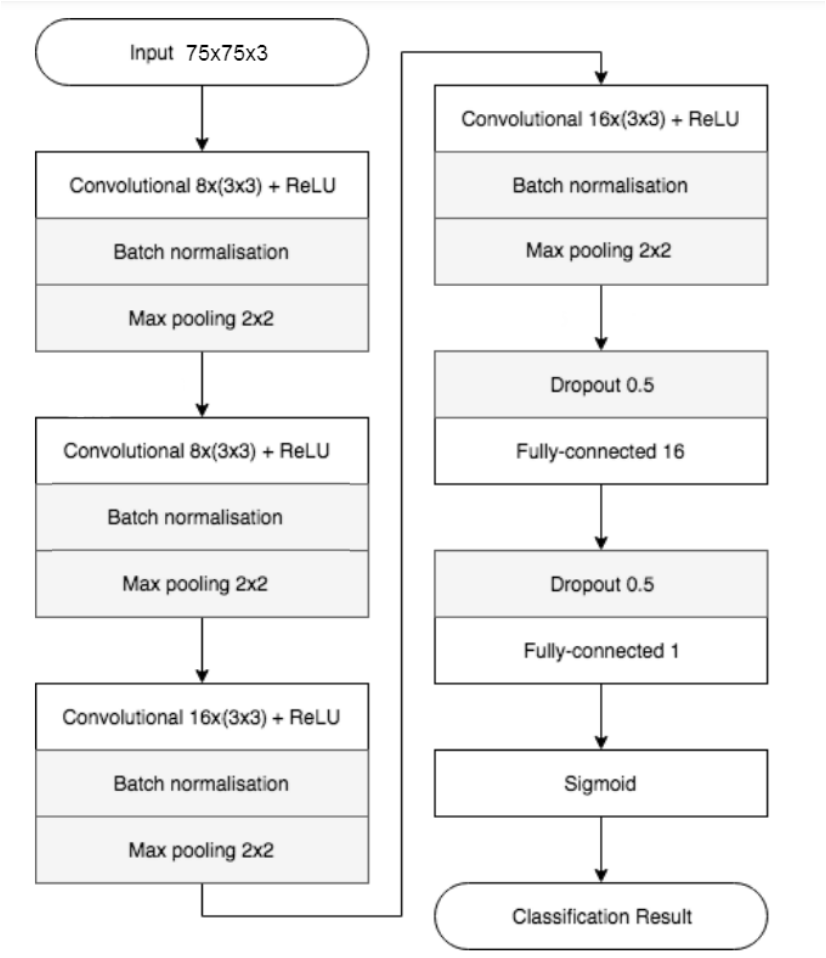
Daha sonra resimlerden oluşturulan yeni dizinde yine bir döngü ile gezinerek her bir fotoğraftaki kişinin yüzü OpenCV kütüphanesinin yardımıyla kırpılıp yeni bir dizine kaydedilmiştir. Bu kırpılmış yüzlerden oluşan son dizindeki veri seti, projede kullanılacak olan veri setidir.

```
for j in range(1,2677):
    # Read the input image
    img = cv2.imread(dataset_real_dir + str(j) + ').jpg')

    # Convert into grayscale
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt2.xml')

    # Detect faces
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)

    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h),
                      (0, 0, 255), 2)

        faces = img[y:y + h, x:x + w]
        cv2.imwrite(dataset_cropped_real_dir + str(j) + '.jpg', faces)

for j in range(1,2677):
    # Read the input image
    img = cv2.imread(dataset_fake_dir + str(j) + ').jpg')

    # Convert into grayscale
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_alt2.xml')

    # Detect faces
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)

    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h),
                      (0, 0, 255), 2)

        faces = img[y:y + h, x:x + w]
        cv2.imwrite(dataset_cropped_fake_dir + str(j) + '.jpg', faces)
```

Şekil 3.3. Fotoğraflardaki kişilerin yüzlerini kırpmak için kullanılan kod örneği

3.2. Yapay Sinir Ağı

Sahnelerden kırpılan yüzler 75 piksele 75 piksel olarak yeniden boyutlandırılıp girdi olarak verilmektedir. Modelde giriş katmanından sonra dört tane konvolüsyon katmanı, iki tane tam bağlı katman ve son olarak çıkış katmanı yer almaktadır. Eğitim sürecini geliştirmek adına konvolüsyonlardan sonra batch normalisation ve max pooling katmanları, tam bağlı katmanda ise dropout kullanılmıştır.

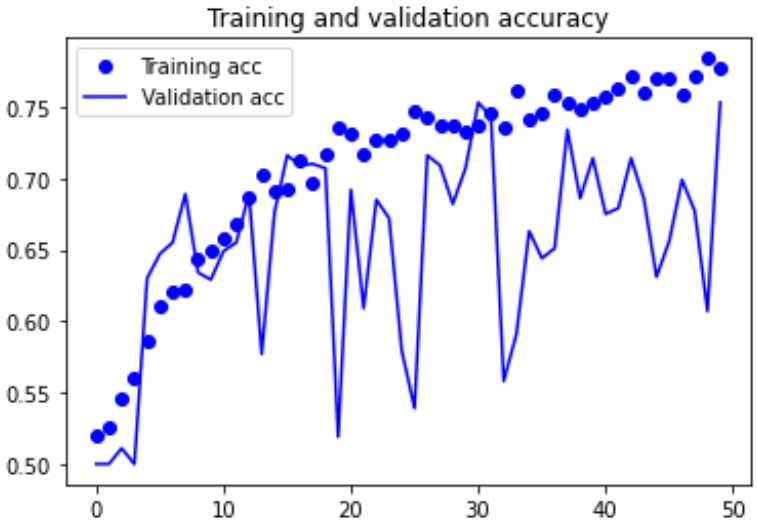


Şekil 3.4. Modelin görsel şeması

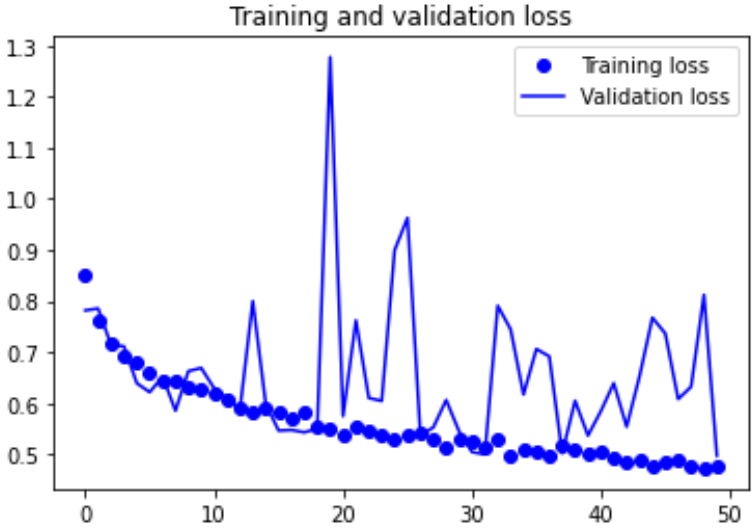
3.3. Elde Edilen Çıktılar

Eğitim sonucunda elde edilen tahmin doğruluk oranı %75 olarak ölçülmüştür. Düşük hacimli bir veri seti, internetten alınmış gerçekçi ama düşük kalitede veriler ve eğitim işlemlerini kısaltmak için basit bir model kullanmanın sonucu olarak bir miktar ezberleme ile karşılaşılmıştır.

Modelin eğitim ve doğrulama sırasında ürettiği çıktılar aşağıdaki gibidir.



Şekil 3.5. Eğitim ve doğrulama aşamalarının doğruluk oranı grafiği



Şekil 3.6. Eğitim ve doğrulama aşamalarının kayıp oranı grafiği

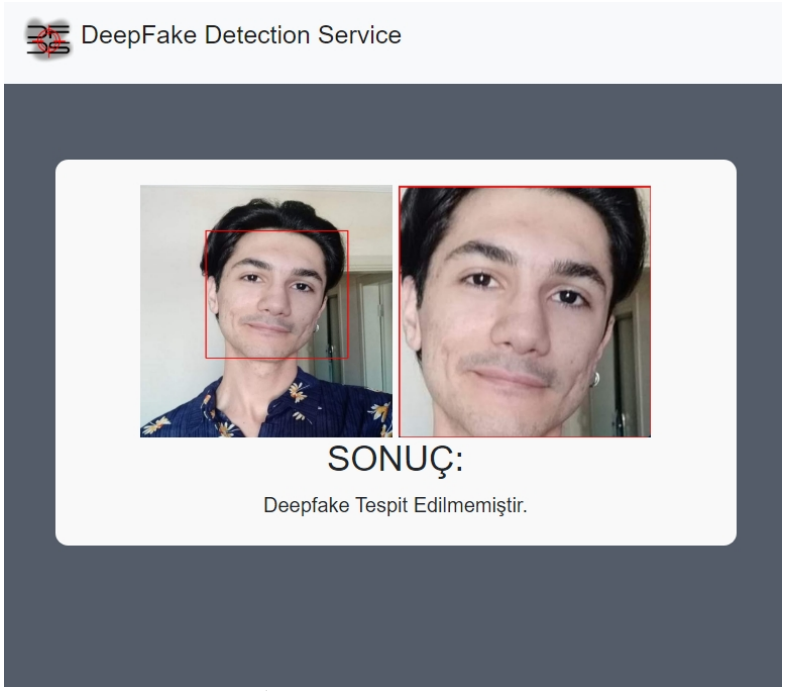
BÖLÜM 4. MODELİN İNTERNET ÜZERİNDEN SERVİS EDİLMESİ

4.1. UI/UX Tasarımları

Başlangıç sayfası, kullanıcıyı basit bir dosya yükleme sayfası karşılayacak ve hızlıca fotoğrafı yükleyip işleme başlamasını sağlayacak şekilde tasarlanmıştır.

Şekil 4.1. İnternet sitesi fotoğraf yükleme bölümü

Daha sonra yüklenen fotoğraf arkaplanda kategorilendirilip sonuç ekrana yazdırılmaktadır.



Şekil 4.2. İnternet sitesi sonuç bildirme bölümü

4.2. Web Servis Teknik Tasarımı

Python ve Keras ile Anaconda üzerinde eğitilen model önce ".h5" dosyası olarak kaydedildikten sonra "tensorflowjs" kütüphanesi yardımıyla JSON dosyası haline getirilip JavaScript dosyasında kullanılmaktadır.

```
let model;
(async function () {
    model = await tf.loadLayersModel("http://localhost/proje/model/model.json");

})();
```

Şekil 4.3. JavaScript'te modelin yüklenmesi

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Bu çalışma sonucunda, Keras ile oluşturulmuş temel bir derin öğrenme modeli ve YouTube videolarından üretilmiş deepfake görüntülerinden oluşan bir veri seti ile deepfake tespitinde alınabilecek sonuçlar elde edilmiştir. Sınırlı veri ve işlem gücü ile eğitilmiş bir modelde %75 oranla deepfake tespiti yapılabilmesi, daha geniş kapsamlı çalışmalarla deepfake'in önemli bir tehdit oluşturmasının önüne geçilebileceğini göstermektedir.

Bu çalışmada elde edilen çıktılar ve daha önce yapılmış çalışmalar gözetilerek deepfake ile manipüle edilmiş görüntülerin tespitinde fotoğraf üzerinden tespitin videodan tespite göre daha başarısız olduğu söylenebilir.

KAYNAKLAR

[1] https://www.kaggle.com, Erişim tarihi: 20.05.2022

[2] https://www.kaggle.com/c/deepfake-detection-challenge, Erişim tarihi: 20.05.2022

[3] A. Malik, M. Kuribayashi, S. M. Abdullahi and A. N. Khan, "DeepFake Detection for Human Face Images and Videos: A Survey," in IEEE Access, vol. 10, pp. 18757-18775, 2022, doi: 10.1109/ACCESS.2022.3151186.

[4] D. Afchar, V. Nozick, J. Yamagishi and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 2018, pp. 1-7, doi: 10.1109/WIFS.2018.8630761.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, ‘‘Generative adversarial nets,’’ in Proc. Adv. Neural Inf. Process. Syst., vol. 27, 2014, pp. 1–9.

[6] A. Radford, L. Metz, and S. Chintala, ‘‘Unsupervised representation learning with deep convolutional generative adversarial networks,’’ 2015, arXiv:1511.06434

[7] A. Creswell and A. A. Bharath, ‘‘Inverting the generator of a generative adversarial network,’’ IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 7, pp. 1967–1974, Jul. 2019.

[8] T. Karras, T. Aila, S. Laine, and J. Lehtinen, ‘‘Progressive growing of GANs for improved quality, stability, and variation,’’ 2017, arXiv:1710.10196

[9] A. Brock, J. Donahue, and K. Simonyan, ‘‘Large scale GAN training for high fidelity natural image synthesis,’’ 2018, arXiv:1809.11096

[10] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, ‘‘Analyzing and improving the image quality of StyleGAN,’’ in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 8110–8119.

[11] https://www.tensorflow.org/, Erişim tarihi: 20.05.2022

[12] https://www.tensorflow.org/js, Erişim tarihi: 20.05.2022

[13] https://keras.io/, Erişim tarihi: 20.05.2022

[14] https://opencv.org/, Erişim tarihi: 20.05.2022

[15] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, ‘‘FaceForensics++: Learning to detect manipulated facial images,’’ in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 1–11

ÖZGEÇMİŞ

Baha Büçge, 04.03.2000'de İstanbul'da doğdu. İlk, orta ve lise eğitimini Kartal'da tamamladı. 2018 yılında Kartal Anadolu Lisesi, Matematik-Fen alanından mezun oldu. 2018 yılında Sakarya Üniversitesi Bilgisayar Mühendisliği Bölümü'nü kazandı.

BSM 498 BİTİRME ÇALIŞMASI
DEĞERLENDİRME VE SÖZLÜ SINAV TUTANAĞI

KONU : EVRİŞİMSEL SİNİR AĞI İLE DEEPFAKE TESPİTİ
ÖĞRENCİLER (Öğrenci No/AD/SOYAD): G181210072 BAHA BÜÇGE

Değerlendirme Konusu	İstenenler	Not Aralığı	Not
Yazılı Çalışma			
Çalışma klavuza uygun olarak hazırlanmış mı?	x	0-5	
Teknik Yönden			
Problemin tanımı yapılmış mı?	x	0-5	
Geliştirilecek yazılımın/donanımın mimarisini içeren blok şeması (yazılımlar için veri akış şeması (dfd) da olabilir) çizilerek açıklanmış mı?			
Blok şemadaki birimler arasındaki bilgi akışına ait model/gösterim var mı?			
Yazılımın gereksinim listesi oluşturulmuş mu?			
Kullanılan/kullanılması düşünülen araçlar/teknolojiler anlatılmış mı?			
Donanımların programlanması/konfigürasyonu için yazılım gereksinimleri belirtilmiş mi?			
UML ile modelleme yapılmış mı?			
Veritabanları kullanılmış ise kavramsal model çıkarılmış mı? (Varlık ilişki modeli, noSQL kavramsal modelleri v.b.)			
Projeye yönelik iş-zaman çizelgesi çıkarılarak maliyet analizi yapılmış mı?			
Donanım bileşenlerinin maliyet analizi (prototip-adetli seri üretim vb.) çıkarılmış mı?			
Donanım için gerekli enerji analizi (minimum-uyku-aktif-maksimum) yapılmış mı?			
Grup çalışmalarında grup üyelerinin görev tanımları verilmiş mi (iş-zaman çizelgesinde belirtilebilir)?			
Sürüm denetim sistemi (Version Control System; Git, Subversion v.s.) kullanılmış mı?			
Sistemin genel testi için uygulanan metotlar ve iyileştirme süreçlerinin dökümü verilmiş mi?			
Yazılımın sızma testi yapılmış mı?			
Performans testi yapılmış mı?			
Tasarımın uygulamasında ortaya çıkan uyumsuzluklar ve aksaklıklar belirtilerek çözüm yöntemleri tartışılmış mı?			
Yapılan işlerin zorluk derecesi?	x	0-25	
Sözlü Sınav			
Yapılan sunum başarılı mı?	x	0-5	
Soruları yanıtlama yetkinliği?	x	0-20	
Devam Durumu			
Öğrenci dönem içerisindeki raporlarını düzenli olarak hazırladı mı?	x	0-5	
Diğer Maddeler			
Toplam			

DANIŞMAN (JÜRİ ADINA): (DOÇ.) DR. DEVRİM AKGÜN
DANIŞMAN İMZASI: