

T.C.
FIRAT ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

DUYGU ANALİZİNE GÖRE UYGUN REKLAM SUNAN
UYGULAMA GOTO

Deniz GÖL
Umut BABALIK
Burak TUNÇEL

Tez Danışmanı
Dr. Öğr. Üyesi Erkan DUMAN

BİTİRME TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

ELAZIĞ – 2024

T.C.
FIRAT ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ

DUYGU ANALİZİNE GÖRE UYGUN REKLAM SUNAN
UYGULAMA GOTO

Deniz GÖL
Umut BABALIK
Burak TUNÇEL

BİTİRME TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Bu bitirme tezi/...../2024 tarihinde, aşağıda belirtilen jüri tarafından oybirliği/oyçokluğu ile başarılı/başarısız olarak değerlendirilmiştir.

(imza)	(imza)	(imza)
Dr. Öğr. Üyesi Erkan DUMAN	Prof. Dr. Burhan ERGEN	Dr. Öğr. Üyesi Erdal ÖZBAY

ÖZGÜNLÜK BİLDİRİMİ

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

Fırat Üniversitesi
Bilgisayar Mühendisliği
23119 Elazığ

09.07.2024

Öğrenci Adı Soyadı

Deniz GÖL

Umut BABALIK

Burak TUNÇEL

BENZERLİK BİLDİRİMİ

Deniz Göl - Umut Babalık - Burak Tunçel

ORJİNAL RAPORU

% **11**
BENZERLİK ENDEKSİ

% **9**
İNTERNET KAYNAKLARI

% **1**
YAYINLAR

% **4**
ÖĞRENCİ ÖDEVLERİ

BİRİNCİL KAYNAKLAR

1	hialgozutok.medium.com İnternet Kaynağı	%4
2	Submitted to Fırat Üniversitesi Öğrenci Ödevi	%2
3	elifmeseci.medium.com İnternet Kaynağı	%1
4	Submitted to Istanbul Aydın University Öğrenci Ödevi	%1
5	www.webtekno.com İnternet Kaynağı	%1
6	docslib.org İnternet Kaynağı	<%1
7	Submitted to Mugla University Öğrenci Ödevi	<%1
8	Gülter, Betül. "Çocuk resimlerindeki nesnelerin derin öğrenme yöntemleriyle tespit edilmesi", İzmir Katip Celebi University (Turkey), 2024 Yayın	<%1
Submitted to ALKEV		

9	Öğrenci Ödevi	<% 1
10	www.coursehero.com İnternet Kaynağı	<% 1
11	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Öğrenci Ödevi	<% 1
12	miuul.com İnternet Kaynağı	<% 1
13	bdigital.ipg.pt İnternet Kaynağı	<% 1
14	repository.ittelkom-pwt.ac.id İnternet Kaynağı	<% 1
15	hdl.handle.net İnternet Kaynağı	<% 1
16	choi98772.blogspot.kr İnternet Kaynağı	<% 1
17	dergipark.org.tr İnternet Kaynağı	<% 1
18	vdocuments.mx İnternet Kaynağı	<% 1
19	Catalbas, Mehmet Cem, Didem Issever, and Arif Gulten. "Morphological feature extraction with local histogram equalization", 2015 23rd	<% 1

Signal Processing and Communications
Applications Conference (SIU), 2015.

Yayın

20	bartubozkurt35.medium.com İnternet Kaynağı	<% 1
21	computer-vision.smartcode.com İnternet Kaynağı	<% 1
22	core.ac.uk İnternet Kaynağı	<% 1
23	medium.com İnternet Kaynağı	<% 1
24	odr.chalmers.se İnternet Kaynağı	<% 1
25	www.brainscape.com İnternet Kaynağı	<% 1
26	www.icondata.org İnternet Kaynağı	<% 1

TEŞEKKÜR

Proje boyunca değerli yönlendirmeleri ve destekleriyle bize rehberlik eden değerli hocamız Sayın Dr. Öğr. Üyesi Erkan DUMAN hocamıza en içten teşekkürlerimizi sunarız.

İÇİNDEKİLER

	<u>Sayfa No</u>
ÖZGÜNLÜK BİLDİRİMİ	III
BENZERLİK BİLDİRİMİ.....	IV
TEŞEKKÜR	VII
İÇİNDEKİLER	I
ŞEKİLLER LİSTESİ.....	IV
ÖZET	VI
KISALTMALAR	VII
ABSTRACT	VIII
1. GİRİŞ	9
2. KULLANILAN TEKNOLOJİLERİN TANITIMI.....	10
2.1. Android Studio Nedir?	10
2.2. Java Nedir?	10
2.3. OpenCV (Open Source Computer Vision Library) Nedir?	10
2.4. Tensorflow Nedir?.....	11
2.5. Kaggle Nedir?.....	12
2.6. Python Nedir?.....	13
2.7. NumPy Nedir?.....	13
2.8. Pandas Nedir?.....	13
2.9. Keras Nedir?.....	14
2.10. CNN (Convolutional Neural Networks) Nedir?	14
2.11. HTML (HyperText Markup Language) Nedir?	14
2.12. CSS (Cascading Style Sheets) Nedir?	15
2.13. JavaScript Nedir?	15
2.14. AWS (Amazon Web Services) Nedir?	15
3. PROJENİN GELİŞTİRİLMESİ.....	17

3.1. Gereksinimlerin Belirlenmesi.....	17
3.2. Yüz Analizi ile Duygu Durumunun Elde Edilmesi İçin Uygun Veri Setinin ve Modelin Belirlenmesi	17
3.2.1. MMA (Multimodal Model-Agnostic) FACIAL EXPRESSION.....	17
3.2.2. MobileNetV2 (Mobile Neural Network Version 2).....	17
3.2.3. Standart Evrişim Katmanı	18
3.2.4. Derinlemesine Ayrılabilir Evrişim Katmanı	19
3.2.5. CNN Çalışma Mantığı.....	19
3.2.5.1. Evrişim Katmanı	20
3.2.5.2. Aktivasyon Katmanı.....	20
3.2.5.3. Ortaklama Katmanı	21
3.2.5.3.1. Ortalama Ortaklama.....	21
3.2.5.3.2. Maksimum Ortaklama	22
3.2.5.4. Flattening Katmanı.....	22
3.2.5.5. Tam Bağlantı Katmanı	23
3.2.5.6. Toplu Normalleştirme Katmanı.....	23
3.2.5.7. Seyreltme Katmanı.....	24
3.3. Modelin Eğitilmesi	24
3.4. Android Uygulamasında Kullanıcı Yüz Analizi ile Duygu Durumu Belirlenmesi	31
3.5. S3 Bucket'ta Görsellerin Depolanması	36
3.6. Uygulamayla S3 Arasındaki Dosya İşlemleri	36
3.7. Site Tasarımı	38
3.8. Yüklenen Reklamların Analizi ve Uygun Reklamın Belirlenmesi	39
3.8.1. Kullanılan Hizmetler	39
3.8.1.1. S3 (Simple Storage Service) Bucket Nedir?	39
3.8.1.2. Rekognition Hizmeti Nedir?	39
3.8.1.3. Lambda Hizmeti Nedir?	39
3.8.1.4. IAM Hizmeti Nedir?	39

3.8.2. S3 Bucket'taki Reklam Görsellerinin Analizi.....	40
3.8.3. IAM Rolüne Verilen izinler	40
3.8.3.1. AmazonRekognitionReadOnlyAccess	40
3.8.3.2. AmazonS3FullAccess	41
3.8.3.3. AmazonS3ObjectLambdaExecutionRolePolicy.....	41
3.8.4. İşlemin Otomatikleştirilmesi	41
3.8.5. Yapılan İşlemlerin Özeti	41
4. SONUÇ.....	43
5. KAYNAKLAR	44
6. ÖZGEÇMİŞ.....	46

ŞEKİLLER LİSTESİ

	<u>Sayfa No</u>
Şekil 3.1. Evrişim katmanları arasındaki farkın gösterimi	18
Şekil 3.2. CNN katmanları.....	19
Şekil 3.3. Çıkış matrisinin hesaplanması.....	20
Şekil 3.4. Aktivasyon fonksiyonları	21
Şekil 3.5. Ortalama ortaklama işlemi.....	22
Şekil 3.6. Maksimum ortaklama işlemi	22
Şekil 3.7. Flattening katmanı	23
Şekil 3.8. Tam bağlantı katmanı	23
Şekil 3.9. Kütüphanelerin projeye dahil edilmesi.....	24
Şekil 3.10. tqdm kütüphanesinin projeye dahil edilmesi	25
Şekil 3.11. Dizilerin oluşturulması	25
Şekil 3.12. a ve b etiketinin sayısının çıktısının alınması	26
Şekil 3.13. Görsellerin piksellerinin normalizasyon işlemi	26
Şekil 3.14. Sözlük tanımlanması.....	26
Şekil 3.15. Eğitim ve test değerlerinin ayarlanması	26
Şekil 3.16. Etiketlerin metine dönüştürülmesi.....	27
Şekil 3.17. Model eğitimi için gerekli kütüphanelerin projeye eklenmesi	27
Şekil 3.18. MobileNetV2 üzerine gerekli katmanların eklenmesi	27
Şekil 3.19. Adam optimizer'ın projeye dahil edilmesi	28
Şekil 3.20. Modelin derlenmesi	28
Şekil 3.21. Modelin en iyi değerlerinin kaydedilmesi	28
Şekil 3.22. Learning Rate parametresini zamanla düşürülmesi	29
Şekil 3.23. Modelin eğitime başlanması.....	29
Şekil 3.24. Ağırlıkların yüklenmesi.....	29
Şekil 3.25. Modelin test verileri üstündeki tahminlerinin kaydedilmesi	30
Şekil 3.26. Eğitilen modeli mobil cihazda kullanabilmek için yapılan dönüştürme işlemi	30
Şekil 3.27. Elimizdeki modelden elde edilen Confusion Matrix	30
Şekil 3.28. F1-Score değerleri tablosu.....	31
Şekil 3.29. Değişkenlerin tanımlanması	32
Şekil 3.30. Eğitilmiş olan modelin yüklenmesi	32
Şekil 3.31. Yüz tespiti.....	33
Şekil 3.32. Yüz tespiti.....	33

Şekil 3.33.	Duygu durumunun belirlenmesi	34
Şekil 3.34.	Belirlenen duygu durumunun S3'te bulunan dosyaya gönderilmesi	35
Şekil 3.35.	Bitmap görüntüsünü derin öğrenme modeli için ByteBuffer formatına dönüştürülmesi	35
Şekil 3.36.	Eğitilen modelin Android uygulamasında yüklenmesi.....	35
Şekil 3.37.	AWS Konfigürasyonu	36
Şekil 3.38.	Android uygulaması ile AWS bağlantısının yapılması	36
Şekil 3.39.	AWS S3 Bucket'a dosya yüklenmesi işlemi	37
Şekil 3.40.	AWS S3 Bucket'a dosya yüklenmesi işlemi	37
Şekil 3.41.	S3 Bucket'tan uygun reklamın yerel depolamaya kaydedilmesi ve kullanıcıya gösterilmesi.....	38
Şekil 3.42.	Kullanıcının reklam yükleyebileceği web sitesinin ekran görüntüsü	39
Şekil 3.43.	Rekognition Hizmetinin Analiz Sonuçlarını Kaydetme	40
Şekil 3.44.	IAM Rolüne Verilen İzinler	40
Şekil 3.45.	Lambda fonksiyonuna eklenen trigger'ın detayları	41

ÖZET

GOTO uygulaması, gelişmiş yüz tanıma teknolojileri ve duygu analizi algoritmaları kullanarak, kullanıcıların yüz ifadelerinden mutluluk, üzüntü, şaşkınlık, öfke gibi duygularını tespit eder. Bu analiz sonucunda, uygulamada tanımlanmış reklamlar arasından kullanıcının mevcut duygusal durumuna en uygun olanı seçilir ve kullanıcıya gösterilir. Örneğin, kullanıcı mutlu bir ruh hali içindeyse, ona neşeli ve olumlu içerikler içeren reklamlar sunulur.

Uygulamanın yapılış amacı ise kullanıcının üzgünken göreceği üzgün içerikli reklamın kullanıcı üzerinde daha vurucu bir etki bırakması ve akılda kalıcılığı artırmasıdır.

Anahtar Kelimeler: TensorFlow, Keras, deep learning, adam optimizer, MobileNetV2, NumPy, OpenCV, pooling layer, flattening layer, CNN, AWS, S3, Lambda, IAM, yüz tanıma, duygu analizi, duygusal durum, reklam seçimi, vurucu etki, akılda kalıcılık

KISALTMALAR

I/O	: Input/Output
IDE	: Integrated Development Environment
IDEA	: Integrated Development Environment Application
TV	: Television
IoT	: Internet of Things
OpenCV	: Open Source Computer Vision Library
BSD	: Berkeley Software Distribution
OS	: Operating System
iOS	: iPhone Operating System
OCR	: Optical Character Recognition
MRI	: Magnetic Resonance Imaging
CT	: Computed Tomography
3D	: Three-Dimensional
API	: Application Programming Interface
CPU	: Central Processing Unit
GPU	: Graphics Processing Unit
TPU	: Tensor Processing Unit
NumPy	: Numerical Python
CNN	: Convolutional Neural Networks
AWS	: Amazon Web Services
S3	: Simple Storage Service
ACL	: Access Control List
IAM	: Identity and Access Management
HTML	: HyperText Markup Language
CSS	: Cascading Style Sheets
SDK	: Software Development Kit
JS	: JavaScript
PNG	: Portable Network Graphics
JPEG	: Joint Photographic Experts Group

ABSTRACT

The “GOTO” application uses advanced facial recognition technologies and emotion analysis algorithms to identify users' emotions such as happiness, sadness, surprise and anger from their facial expressions. As a result of this analysis, among the ads defined in the app, the most appropriate one for the user's current emotional state is selected and shown to the user. For example, if the user is in a happy mood, they are presented with ads that contain cheerful and positive content.

The purpose of the application is that the sad content advertisement that the user will see when the user is sad will have a more striking effect on the user and increase memorability.

Keywords: TensorFlow, Keras, Deep Learning, Adam Optimizer, MobileNetV2, NumPy, OpenCV, Pooling Layer, Flattening Layer, CNN, AWS, S3, Lambda, IAM, Face recognition, emotion analysis, emotional state, emotional state, ad selection, impact, memorability

1. GİRİŞ

Modern çağımızda reklamcılık, dijital dünyanın önemli bir parçası haline gelmiştir. İnternetin ve mobil cihazların yaygın kullanımıyla birlikte reklamlar, bireylerin günlük hayatlarının vazgeçilmez unsurları olmuştur. Bu durum, işletmelerin ve markaların hedef kitlelerine ulaşma ve onlarla etkileşim kurma stratejilerine daha fazla yatırım yapmalarını zorunlu kılmaktadır. Reklamların etkinliği, doğru mesajın doğru kişiye doğru zamanda iletilmesine bağlıdır. Bu noktada kişiselleştirilmiş reklamlar büyük bir önem taşır.

Kişiselleştirilmiş reklamlar, bireylerin ilgi alanlarına, demografik bilgilerine ve çevrimiçi davranışlarına göre özel olarak tasarlanmış içerikler sunar. Bu strateji, geleneksel reklam yöntemlerine kıyasla daha yüksek dönüşüm oranları sağlama potansiyeline sahiptir çünkü hedef kitleye daha uygun ve ilgi çekici içerikler sunulmaktadır. Ancak, bu reklamların başarısı yalnızca kullanıcıların temel bilgilerinin ötesine geçerek duygusal durumlarını da göz önünde bulundurmalıdır.

Bu tezde, görüntü işleme teknikleri kullanılarak kullanıcıların yüz ifadelerinin analiz edilmesi ve bu analizler aracılığıyla kullanıcıların duygusal durumlarının belirlenmesi üzerinde durulacaktır. Yüz tanıma ve duygu analizi, makine öğrenimi ve derin öğrenme algoritmaları kullanılarak gerçekleştirilecektir. Kullanıcıların duygusal durumlarına göre reklam içerikleri sunulacak ve bu sayede reklamların etkinliği artırılacaktır.

Aynı zamanda, reklamların içeriğinde yer alan yüzlerin de duygusal analizinin yapılması, reklamın genel duygusal tonunu belirlememize yardımcı olacaktır. Reklamlarda kullanılan yüz ifadeleri ve duygusal temalar analiz edilerek, bu reklamların kullanıcıların mevcut duygusal durumlarına en uygun şekilde sunulması sağlanacaktır.

2. KULLANILAN TEKNOLOJİLERİN TANITIMI

2.1. Android Studio Nedir?

Android Studio, Google tarafından 16 Mayıs 2013'te düzenlenen Google I/O (Input/Output) etkinliğinde tanıtıldı ve o zamandan beri Android uygulama geliştirme sürecinde kullanılan resmi IDE (Integrated Development Environment) olarak kabul ediliyor. Çeşitli Android cihazlarda çalışabilme yeteneği ile öne çıkan Android Studio, yüksek kaliteli ve verimli uygulamalar geliştirmek için en hızlı araçları sunar.

IntelliJ IDEA (Integrated Development Environment Application) tabanlı olan Android Studio, Android uygulama geliştirmeye yönelik özel olarak tasarlanmıştır. İçeriğinde akıllı bir kod düzenleyici ve hata ayıklayıcı, performans analiz araçları, emülatörler ve daha birçok araç barındırır. Bu özellikler, kullanıcıların ihtiyaçlarını karşılayarak uygulama geliştirme sürecini daha verimli hale getirir.

Android Studio, geniş bir kullanım alanına sahiptir. Mobil uygulamalar, tablet uygulamaları, Android Wear için giyilebilir cihaz uygulamaları, Android TV (Television) için televizyon uygulamaları ve Android Auto için otomotiv uygulamaları geliştirmek için kullanılabilir. Ayrıca, akıllı ev cihazları ve IoT (Internet of Things) uygulamaları gibi daha özel alanlarda da kullanım imkanı sunar. Bu çok yönlülüğü sayesinde, geliştiriciler farklı platformlar ve cihazlar için tek bir IDE kullanarak uygulamalarını oluşturabilirler.

2.2. Java Nedir?

Java, 1995 yılında Sun Microsystems tarafından geliştirilen ve günümüzde Oracle tarafından yönetilen, nesne odaklı ve platform bağımsız bir programlama dilidir. Platform bağımsızlığı, Java kodunun bir kez yazıldıktan sonra, farklı işletim sistemlerinde herhangi bir değişiklik yapılmadan çalıştırılabilmesini sağlar. Bu sayede Java, web uygulamaları, mobil uygulamalar, masaüstü programları, gömülü sistemler ve daha pek çok alanda yaygın olarak kullanılmaktadır [1].

2.3. OpenCV (Open Source Computer Vision Library) Nedir?

OpenCV (Open Source Computer Vision) kütüphanesi, 1999 yılında Intel tarafından başlatılan, şirket ve toplulukların katkılarıyla geliştirilen bir projedir. İlk alfa sürümü 2000 yılında piyasaya sürülmüştür ve başlangıçta C programlama diliyle geliştirilmiştir. Zamanla birçok algoritması C++ diliyle yenilenmiştir. BSD (Berkeley Software Distribution) lisansı altında yayınlanan OpenCV, kullanıcıların projelerinde ücretsiz olarak kullanabilecekleri bir

kütüphanedir, bu da geniş bir kullanıcı tabanına sahip olmasını sağlar. Platform bağımsızlığı sayesinde Windows, Linux, FreeBSD, Android, Mac OS ve iOS (iPhone Operating System) gibi çeşitli işletim sistemlerinde çalışabilir ve C++, C, Python, Java, Matlab gibi dillerle entegre edilebilir.

OpenCV, görüntü işleme ve makine öğrenmesi alanlarında 2500'den fazla algoritma sunar ve yüz tanıma, nesne tespiti, hareket analizi, nesne sınıflandırma, plaka tanıma, 3D görüntü işleme, görüntü karşılaştırma ve optik karakter tanıma OCR (Optical Character Recognition) gibi çeşitli işlemleri kolayca gerçekleştirebilir. Son güncelleme olarak, OpenCV'nin geliştirici firması Itseez'in Intel tarafından satın alındığı ve geliştirme sürecinin Intel çatısı altında devam edeceği duyurulmuştur.

Tıbbi görüntülemelerde kullanılan MRI (Magnetic Resonance Imaging), CT (**Computed Tomography**) ve röntgen gibi görüntülerin işlenmesi ve analiz edilmesi, teşhis koymak için OpenCV'nin sunduğu güçlü araçlardan yararlanır. Robotlar, kameralarından aldıkları görüntüleri işleyerek çevrelerini algılar, nesneleri tanır ve buna göre hareket ederler; bu da OpenCV'nin robotikteki önemini vurgular. Artırılmış gerçeklik uygulamalarında sanal nesnelerin gerçek dünyayla entegre edilmesi ve sanal gerçeklik uygulamalarında 3D (Three-Dimensional) ortamların oluşturulması, kullanıcı ile etkileşime girmesi de OpenCV'nin sunduğu olanaklar arasında yer alır. OpenCV, sağladığı bu çeşitli ve etkili araçlarla, bilgisayarlı görme ve görüntü işlemenin hemen her alanında vazgeçilmez bir kaynak haline gelmiştir [2].

2.4. Tensorflow Nedir?

TensorFlow, Google tarafından geliştirilen ve açık kaynak kodlu olarak sunulan bir derin öğrenme kütüphanesidir. Yapay zeka uygulamaları geliştirmek için yaygın olarak kullanılan bu araç, geniş bir kullanım alanına sahiptir ve çeşitli donanımlarla uyumlu çalışabilir. TensorFlow'un esnekliği, tek bir API (Application Programming Interface) ile CPU (Central Processing Unit), GPU (Graphics Processing Unit) ve TPU (Tensor Processing Unit) gibi farklı donanımlar üzerinde çalışabilmesini sağlar. Bu, çeşitli ihtiyaçlara ve bütçelere uyumlu çözümler sunar. Derin öğrenme modellerini oluşturma ve eğitme konusunda gerekli olan tüm araçları ve işlevleri sağlayarak, karmaşık modellerin geliştirilmesi ve optimize edilmesi için geniş kapsamlı bir altyapı sunar. Ayrıca, TensorFlow'un geniş ve aktif bir kullanıcı topluluğu bulunur. Bu topluluk, çeşitli sorunlara çözüm bulma, kod örnekleri paylaşma ve deneyimlerini aktarma konusunda oldukça etkindir.

TensorFlow, birçok alanda kullanılmaktadır. Görüntü işleme alanında resim sınıflandırma, nesne algılama, görüntü segmentasyonu ve tıbbi görüntüleme gibi görevlerde yüksek doğruluk oranlarına sahip modeller geliştirilmesine olanak tanır. Doğal dil işleme

görevlerinde makine çevirisi, metin özetleme, duygu analizi ve sohbet robotları gibi uygulamalarda etkin bir şekilde kullanılır. Konuşma tanıma uygulamalarında, konuşmayı metne dönüştürme, sesli komut tanıma ve hoparlör tanıma gibi görevlerde doğru ve hızlı sonuçlar elde edilebilir. Tavsiye sistemlerinde ürün, film ve müzik önerisi gibi uygulamalarda kullanıcı davranışlarını analiz ederek kişiselleştirilmiş öneriler sunmak için gelişmiş algoritmalar geliştirilebilir. Ayrıca, anomali tespiti gerektiren sahtecilik ve dolandırıcılık tespiti, ağ güvenliği ve envanteri gibi uygulamalarda da etkin bir şekilde kullanılabilir. TensorFlow'un bu geniş kullanım yelpazesi, onu yapay zeka ve derin öğrenme alanında güçlü ve esnek bir araç haline getirir [3].

2.5. Kaggle Nedir?

Kaggle, veri bilimcileri, makine öğrenmesi uzmanları ve yapay zeka meraklıları için bir araya gelen çevrimiçi bir platform ve topluluktur. Kullanıcılar burada çeşitli yarışmalara katılabilir, veri setlerini keşfedebilir, modellerini paylaşabilir ve diğer veri bilimcilerinden öğrenebilirler. Kaggle, yeni beceriler öğrenmek, portfolyonuzu geliştirmek ve yeni iş fırsatları keşfetmek için harika bir kaynaktır.

Platform, çeşitli yarışmalar düzenleyerek her beceri seviyesinden veri bilimciye hitap eder. Bu yarışmalar, makine öğrenmesi ve veri bilimi ile ilgili çeşitli konuları kapsar ve katılımcılara gerçek dünya problemlerini çözme fırsatı sunar. Ayrıca, Kaggle, geniş bir veri seti kütüphanesi sunar. Bu kütüphane, farklı alanlardan görüntü, metin, ses ve kod gibi veri türlerini içerir, bu da kullanıcıların projeleri için uygun veri kaynakları bulmalarını sağlar.

Kaggle, aktif ve yardımsever bir topluluğa sahiptir. Kullanıcılar, sorularını sormak, başkalarının projelerinden öğrenmek ve geri bildirim almak için toplulukla etkileşimde bulunabilirler. Bu, bilgi alışverişi ve becerilerin geliştirilmesi için büyük bir avantajdır. Ayrıca, kullanıcılar modellerini diğerleriyle paylaşabilir ve geri bildirim alarak kendilerini geliştirme fırsatı bulabilirler. Model paylaşımı, yeni fikirler edinmenin ve yetenekleri geliştirmenin etkili bir yoludur.

Kaggle, iş fırsatları keşfetmek ve işverenlerle bağlantı kurmak için de bir platform sunar. Kullanıcılar, iş ilanlarını görebilir, işverenlerle iletişime geçebilir ve kariyerleri hakkında bilgi edinebilirler. Bu, kariyer gelişimi ve yeni iş fırsatları bulma açısından önemli bir avantajdır.

Kaggle, veri bilimi becerilerini geliştirmek isteyenler için mükemmel bir platformdur. Temel istatistiklerden karmaşık makine öğrenmesi modellerine kadar geniş bir yelpazede beceri kazanma fırsatı sunar. Ayrıca, kullanıcılar katıldıkları yarışmalar ve tamamladıkları projelerle portfolyolarını oluşturabilir, bu da potansiyel işverenlere becerilerini ve deneyimlerini göstermek

için harika bir yoldur. Kaggle, veri bilimi topluluğuyla bağlantı kurmak ve bilgi alışverişinde bulunmak için de ideal bir platformdur. Eğlenceli ve ilgi çekici bir ortamda veri bilimi ve makine öğrenmesi ile ilgili projeler yapmak isteyen herkes için cazip bir platformdur [4].

2.6. Python Nedir?

Python, günümüzde en popüler ve çok yönlü programlama dillerinden biridir. Kolay öğrenilebilir olması, zengin kütüphanelere sahip olması ve çeşitli alanlarda kullanılabilmesi gibi özellikleri ile öne çıkan Python, yeni başlayanlar için ideal bir dil olduğu kadar, deneyimli programcılar için de güçlü bir araçtır. Python, sade ve okunabilir bir söz dizimine sahiptir, bu sayede yeni başlayanlar bile dili hızlı bir şekilde öğrenebilirler. Geniş kütüphane yelpazesi, veri bilimi, web geliştirme, yapay zeka, otomasyon gibi çeşitli alanlarda kullanıcılarına yardımcı olur. Python, web geliştirme, veri bilimi, makine öğrenmesi, yapay zeka, oyun geliştirme, sistem yönetimi gibi birçok alanda kullanılabilen çok yönlü bir dildir. Üstelik, ücretsiz ve açık kaynaklı olması, dile ücretsiz olarak erişim imkanı sunar ve kaynak kodunun incelenmesine olanak tanır. Python'un büyük ve aktif bir topluluğu vardır, bu da sorunlarla ilgili yardım bulmayı, kod örnekleri paylaşmayı ve diğer geliştiricilerle etkileşimde bulunmayı kolaylaştırır.

2.7. NumPy Nedir?

NumPy (Numerical Python), Python programlama dili için geliştirilmiş, bilimsel hesaplama ve veri analizi amacıyla yaygın olarak kullanılan bir kütüphanedir. Çok boyutlu diziler ve matrisler ile çalışmayı kolaylaştıran NumPy, bu veri yapılarına yönelik yüksek düzeyde matematiksel işlemler sunar. NumPy, hesaplamaların hızlandırılması ve verimliliğin artırılması için optimize edilmiştir.

NumPy, bilimsel araştırmalar, mühendislik hesaplamaları, finansal analizler ve veri bilimi gibi çeşitli alanlarda kullanılmaktadır. Araştırmacılar ve veri bilimciler, büyük veri kümeleri üzerinde hızlı ve verimli hesaplamalar yapmak için NumPy'ı tercih ederler. Ayrıca, makine öğrenmesi ve yapay zeka projelerinde de temel bir araç olarak kullanılır. NumPy'nin sağladığı esneklik ve performans, geniş bir uygulama yelpazesinde etkili sonuçlar elde edilmesini sağlar [5].

2.8. Pandas Nedir?

Pandas, Python programlama dili için geliştirilmiş, veri analizi ve veri işleme amacıyla yaygın olarak kullanılan bir kütüphanedir. Pandas, tablo benzeri veri yapıları ve seriler ile çalışmayı kolaylaştırır ve bu veri yapılarına yönelik çeşitli veri manipülasyon ve analiz araçları

sunar. Pandas, veri temizleme, dönüştürme, birleştirme ve görselleştirme işlemlerinde sıklıkla kullanılır. Araştırmacılar, veri analistleri ve veri bilimciler, büyük ve karmaşık veri kümeleri üzerinde hızlı ve verimli işlemler gerçekleştirmek için Pandas'ı tercih ederler. Finans, ekonomi, sosyal bilimler ve biyoinformatik gibi çeşitli alanlarda veri analizi projelerinde kullanılır. Pandas, veri bilimi projelerinde ve makine öğrenmesi süreçlerinde veri hazırlama aşamasında önemli bir rol oynar [6].

2.9. Keras Nedir?

Keras, Python dilinde geliştirilmiş bir yüksek seviye derin öğrenme API'sidir. Derin öğrenme modelleri oluşturmak, eğitmek ve değerlendirmek için kullanılır. François Chollet tarafından geliştirilen bu kütüphane, kullanımı kolay ve modüler yapısıyla öne çıkar.

Keras, görüntü işleme uygulamalarında evrişimli sinir ağları (CNN) gibi modellerle görüntü sınıflandırma, nesne tanıma ve görüntü segmentasyonu gibi görevlerde etkin olarak kullanılır. Özellikle tıbbi görüntüleme, otomotiv endüstrisi ve güvenlik sistemleri gibi alanlarda yaygın olarak tercih edilir.

2.10. CNN (Convolutional Neural Networks) Nedir?

CNN (Convolutional Neural Network), görüntü işleme ve bilgisayarla görme alanında sıkça kullanılan bir yapay sinir ağı türüdür. Görüntülerdeki özellikleri otomatik olarak tanıyıp çıkarabilen katmanlara sahiptir, bu nedenle manuel özellik çıkarımı gerektirmez. Görüntü sınıflandırma, nesne algılama, yüz tanıma, tıbbi görüntü analizi, otonom araçlar için yol ve engel tespiti, video analizi ve artırılmış gerçeklik gibi birçok uygulama alanında kullanılır. CNN'ler, görüntülerdeki uzamsal ve yerel ilişkileri etkili bir şekilde modelleyerek yüksek doğruluk sunar ve derin öğrenme modellerinin performansını artırır [7][8].

2.11. HTML (HyperText Markup Language) Nedir?

HTML, web sayfalarının yapısını oluşturan standart bir işaretleme dilidir. 1990'ların başında Tim Berners-Lee tarafından geliştirilen HTML, internet tarayıcılarının kullanıcıya metin, resim, video ve diğer multimedya öğelerini görüntülemesine olanak tanır. HTML, başlıklar, paragraflar, bağlantılar, listeler ve diğer içerik türlerini tanımlayan çeşitli etiketlerden (tag) oluşur. Bu etiketler, sayfanın içeriğinin yapısını belirler ve tarayıcının bu içeriği doğru bir şekilde yorumlayarak kullanıcıya sunmasını sağlar. HTML, modern web geliştirme dünyasında CSS (Cascading Style Sheets) ve JavaScript gibi teknolojilerle birlikte kullanılarak daha zengin ve dinamik web deneyimleri yaratılmasına imkan tanır [9].

2.12. CSS (Cascading Style Sheets) Nedir?

CSS, web sayfalarının görünümünü ve düzenini kontrol etmek için kullanılan bir stil dilidir. 1996 yılında Håkon Wium Lie ve Bert Bos tarafından geliştirilen CSS, HTML ile birlikte kullanılarak web sayfalarının tasarımını ve görsel sunumunu şekillendirir. CSS, renkler, yazı tipleri, kenar boşlukları, hizalama ve yerleşim gibi görsel unsurları belirlemek için çeşitli kurallar ve özellikler içerir. Bu stil dili, HTML ile içeriğin yapısını ayırarak web geliştiricilerinin web sitelerinin tasarımını daha esnek ve etkili bir şekilde yönetmesine olanak tanır. Ayrıca, CSS ile birden fazla web sayfasına aynı stilleri uygulamak mümkün olduğu için, sitenin bakımını ve tutarlılığını artırır.

2.13. JavaScript Nedir?

JavaScript, web sayfalarına dinamik ve etkileşimli özellikler eklemek için kullanılan yüksek seviyeli, nesne tabanlı bir programlama dilidir. 1995 yılında Brendan Eich tarafından geliştirilen JavaScript, kullanıcı etkileşimlerine tepki verme, içerik değiştirme, form doğrulama ve animasyon gibi işlemleri gerçekleştirmek için yaygın olarak kullanılır. JavaScript, HTML ve CSS ile birlikte çalışarak web sayfalarının statik yapısını dinamik hale getirir ve kullanıcı deneyimini zenginleştirir. Modern JavaScript, geniş kütüphaneler ve çerçeveler (örneğin, React, Angular, Vue.js) aracılığıyla karmaşık web uygulamalarının geliştirilmesini mümkün kılar. Ayrıca, Node.js gibi platformlar sayesinde JavaScript, sunucu tarafında da kullanılarak tam anlamıyla bir bütünsel programlama dili haline gelmiştir [10].

2.14. AWS (Amazon Web Services) Nedir?

AWS (Amazon Web Services), Amazon'un sunduğu bulut bilişim platformudur. İşletmeler ve bireyler için çeşitli bilişim hizmetleri sunar ve bu hizmetler internet üzerinden erişilebilir durumdadır. AWS'nin temel kullanım alanları şunlardır;

- Web uygulamalarının barındırılması ve ölçeklendirilmesi için kullanılır. Büyük web siteleri, e-ticaret platformları ve mobil uygulamalar AWS üzerinde çalışabilir.
- Veri tabanı yönetimi için çeşitli hizmetler sunar. İlişkisel veri tabanlarından NoSQL (Not only SQL) çözümlerine kadar geniş bir yelpazede veri tabanı seçenekleri mevcuttur.
- Büyük miktarda veri depolamak ve bu verilere hızlı erişim sağlamak için S3 gibi nesne tabanlı depolama hizmetleri sunar.
- Yapay zeka ve makine öğrenimi modelleri oluşturmak, eğitmek ve dağıtmak için hizmetler sağlar. Bu alanda Amazon SageMaker gibi yönetilen hizmetler sunar.

- İş sürekliliği ve güvenlik için çeşitli hizmetler sunar. AWS Lambda ile serverless işlevler oluşturulabilir ve AWS IAM ile erişim kontrolleri yönetilebilir.
- AWS'nin sunduğu bu çeşitli hizmetler, işletmelerin IT altyapılarını esnek, güvenilir ve maliyet etkin bir şekilde yönetmelerini sağlar.

3. PROJENİN GELİŞTİRİLMESİ

3.1. Gereksinimlerin Belirlenmesi

Projenin başlangıcında, detaylı bir proje planı oluşturmak büyük önem taşır. Bu plan kapsamında, ihtiyaçlar ve projede çalışacak kişilerin yetkinlikleri göz önünde bulundurularak, projede üstlenecekleri roller belirlenir. Bu aşama, projemizin geliştirilmesi açısından en kritik adımdır.

Bu süreçte, projede ihtiyaç duyulacak teknolojiler, yardımcı yazılımlar ve projenin tamamlanması için öngörülen tarih belirlendi. Ayrıca, geliştirme sürecinde yapılacaklar da planlandı. Seçilen yardımcı yazılımlar, teknolojiler ve sistemlerin kullanımına ilişkin dokümantasyon hazırlandı, proje ekibine sunuldu.

3.2. Yüz Analizi ile Duygu Durumunun Elde Edilmesi İçin Uygun Veri Setinin ve Modelin Belirlenmesi

Projemizin kullanıcının yüz analizi ile duygu durumunun belirlenmesi kısmında veri seti olarak MMA (Multimodal Model-Agnostic) FACIAL EXPRESSION veri seti kullanıldı ve bu veri seti MobileNetV2 derin öğrenme modeli kullanılarak eğitildi.

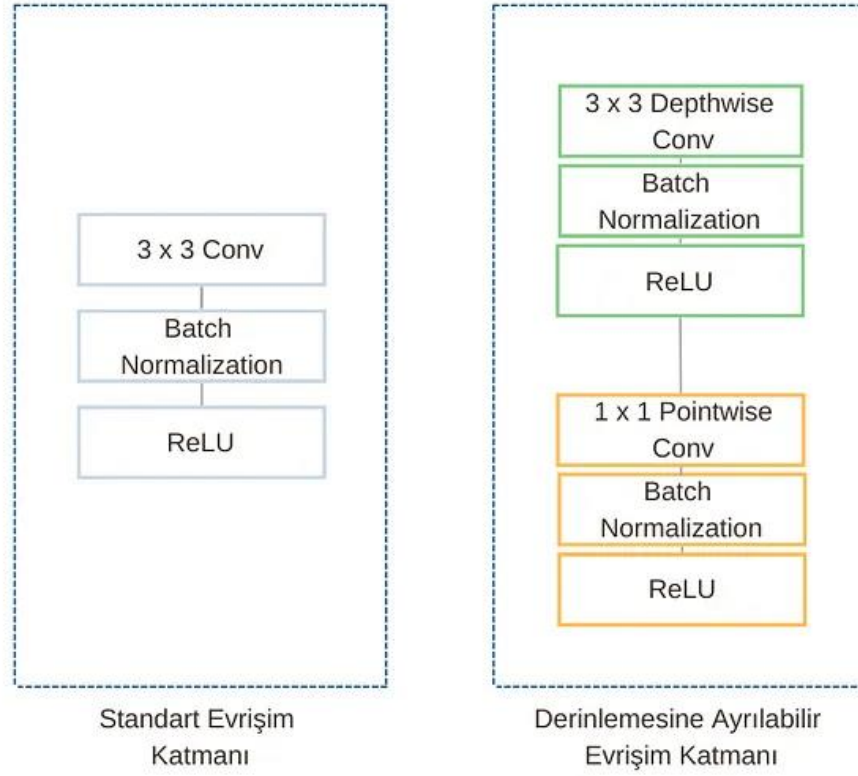
3.2.1. MMA (Multimodal Model-Agnostic) FACIAL EXPRESSION

MMA facial expression veri seti eğitim, doğrulama ve test için ağaç dizinleri içerir. Her dizin, yedi adet yüz ifadesi kategorisine karşılık gelir. Bu ifadeler kızgın, üzgün, mutlu, şaşırmış, korkmuş, iğrenmiş ve doğal bakış ifadeleridir.

Bu veri setini seçmemizin ana sebebi, eğitim için kullanılan fotoğrafların boyutlarının küçük olmasıydı; bu durum mobil uygulamalarda daha optimize çalışmasını sağlamaktadır.

3.2.2. MobileNetV2 (Mobile Neural Network Version 2)

MobileNetV2, Google tarafından 2018 yılında geliştirilen CNN temelli bir mimaridir ve Nesne sınıflandırma ve algılama gibi görevlerde kullanılan diğer ağlarla karşılaştırıldığında, daha küçük boyutlu ve düşük gecikme süreli modellerdir. Küçük boyutları sayesinde mobil cihazlarda derin öğrenme modelleri olarak kullanılabilirler. MobileNet'ler, standart evrişimli katmanlar yerine derinlemesine ayrılabilir evrişimli katmanları tercih eder [11] [12].



Şekil 3.1. Evrişim katmanları arasındaki farkın gösterimi

3.2.3. Standart Evrişim Katmanı

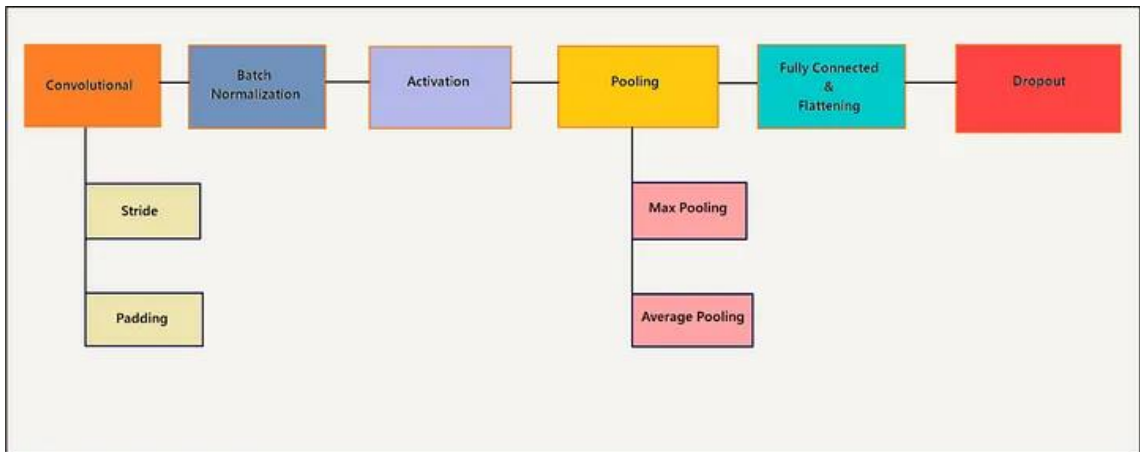
Standart evrişim katmanı, evrişimli sinir ağlarının (CNN) temel yapı taşlarından biridir ve giriş görüntüsüne bir dizi filtre (veya çekirdek) uygulayarak özellik haritaları oluşturur. Bu katman, görüntü üzerindeki belirli özellikleri (kenarlar, köşeler, dokular vb.) tespit etmek için genellikle küçük boyutlu filtreler kullanır. Filtre, giriş görüntüsü üzerinde belirli bir adımla (stride) kaydırılarak, her adımda filtre ve görüntü üzerindeki örtüşen bölge elemanlarının çarpımı alınır ve bu çarpımların toplamı hesaplanır. Bu toplam, özellik haritasının bir noktasını oluşturur. Dolgu (padding) kullanılarak filtrenin görüntü kenarları üzerindeki etkisi korunur ve çıktı boyutları kontrol edilir. Evrişim işlemi sonucunda elde edilen özellik haritası, genellikle bir aktivasyon fonksiyonu (örneğin ReLU (Rectified Linear Unit)) ile işlenir, bu da doğrusal olmayanlık ekleyerek modelin daha karmaşık ilişkileri öğrenmesine olanak tanır. Filtrelerin tüm görüntü üzerinde kaydırılmasıyla elde edilen çıktılar, özellik haritası olarak adlandırılır. Standart evrişim katmanı, giriş görüntüsündeki yerel özellikleri tespit eder ve bu bilgileri daha üst katmanlara ileterek daha karmaşık ve soyut özelliklerin öğrenilmesine katkıda bulunur. Bu katmanlar, derin öğrenme modellerinde görüntü sınıflandırma, nesne algılama ve diğer bilgisayarla görme görevlerinde temel bir rol oynar.

3.2.4. Derinlemesine Ayrılabilir Evrişim Katmanı

Derinlemesine ayrılabilir evrişim katmanı, standart evrişim katmanlarının optimize edilmiş bir varyantıdır ve hesaplama verimliliğini artırmak için kullanılır. Bu katman, standart evrişim işlemini iki ayrı aşamaya böler: derinlemesine evrişim ve nokta evrişimi. İlk aşama olan derinlemesine evrişimde, her filtre yalnızca tek bir giriş kanalında uygulanır, bu da her kanal için ayrı bir özellik haritası üretir. İkinci aşama olan nokta evrişiminde ise, 1x1 boyutunda filtreler kullanılarak tüm kanallar arasında birleştirme yapılır ve nihai özellik haritası elde edilir. Bu iki aşamanın birleşimi, hesaplama maliyetlerini önemli ölçüde azaltır ve modelin hafifliğini artırır. Derinlemesine ayrılabilir evrişim katmanları, standart evrişim katmanlarına kıyasla daha az parametre ve daha düşük işlem maliyeti ile çalıştıkları için mobil ve gömülü cihazlarda yaygın olarak tercih edilir. Bu katmanlar, derin öğrenme modellerinde yüksek doğruluk ve verimlilik sağlarken, daha az bellek kullanımı ve daha hızlı işlem süreleri sunar. Özellikle mobil cihazlar, IoT (Internet of Things) ve gömülü sistemlerde, sınırlı kaynaklarla karmaşık görevlerin gerçekleştirilmesine olanak tanır. Derinlemesine ayrılabilir evrişimler, parametre sayısını azaltır. MobileNet mimarisinde kullanılan derinlemesine ayrılabilir evrişimler ile geleneksel evrişimler arasındaki ana fark, tek bir 3x3 evrişim katmanı yerine, ayrı ayrı 3x3 derinlemesine (depthwise) ve 1x1 noktasal (pointwise) evrişim katmanlarının kullanılmasıdır.

3.2.5. CNN Çalışma Mantığı

CNN'lerde, görüntüler katmanlardan geçirilerek öğrenme işlemi gerçekleştirilir.



Şekil 3.2. CNN katmanları

Temel bir CNN mimarisinde şu katmanlar bulunur: konvolüsyon katmanı, aktivasyon katmanı, havuzlama katmanı, tam bağlantılı ve düzleştirme katmanları. Ayrıca, modelin

performansını optimize etmek için bazen Batch, Normalization ve Dropout gibi katmanlar da eklenir. Bu katmanlar, CNN modellerinin hızını, doğruluğunu ve genel performansını etkiler.

3.2.5.1. Evrişim Katmanı

Evrişim katmanı, giriş görüntüsünden öznitelikler çıkarmak için filtreler kullanır. Bu filtreler, kenar tespiti, köşe algılama ve nesne tanıma gibi görevleri yerine getirir.

CNN'de, görüntüler matris formunda temsil edilir ve işlemler genellikle bu matrisler üzerinde gerçekleştirilir. Matrislerdeki her bir değer, görüntünün piksellerine karşılık gelir.

Bir bilgisayara bir görüntüyü tanımlatırken, aslında o görüntünün piksel değerlerini matematiksel olarak ifade ederiz. Bilgisayar bu matematiksel ifadeleri hafızasında saklar ve daha sonra ilgili işlemleri yapabilir.

$$\text{Çıkış matrisinin boyutu} = \text{Giriş matrisinin boyutu} - \text{Filtre boyutu} + 1$$
$$\text{Formül} = (n-f+1) \times (n-f+1)$$

Şekil 3.3. Çıkış matrisinin hesaplanması

Çıkış matrisinin boyutu belirlendikten sonra, filtre matrisi görüntü matrisinin sol üst köşesine yerleştirilir. İki matrisin kesişen indislerindeki değerler çarpılarak toplanır ve bu işlemle çıkış matrisinin ilk indeksi hesaplanır. Daha sonra filtre matrisi bir adım sağa kaydırılarak bu işlem tekrarlanır. Bu süreç, filtre matrisinin görüntü üzerinde hareket ettiği yolculuğunu tamamladığında çıkış matrisindeki değerler hesaplanmış olur. Bu yöntemle evrişim katmanında gerekli öznitelikler belirlenir. Ayrıca, evrişimli sinir ağlarında birden fazla evrişim katmanı kullanılarak farklı öznitelikler için ayrı ayrı işlemler gerçekleştirilebilir.

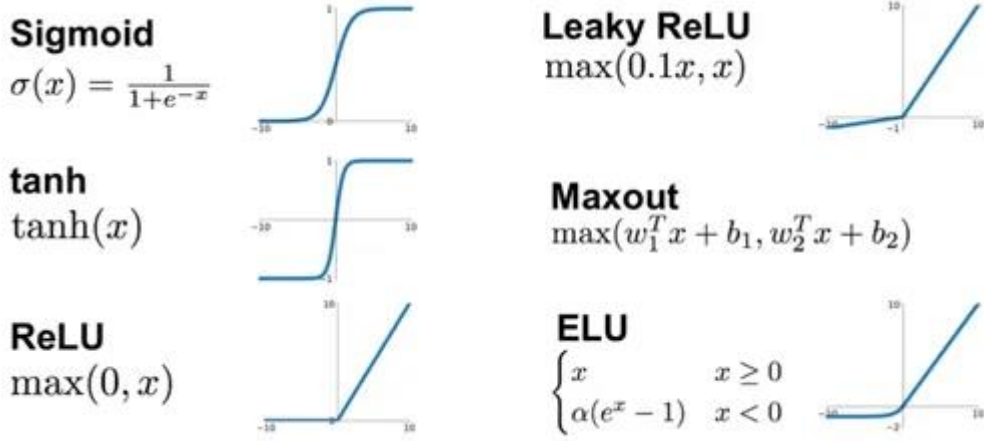
Evrişim işlemi sonrasında, giriş matrisi ile çıkış matrisi arasında boyut farkı oluşabilir. Giriş matrisi ile çıkış matrisinin aynı boyutta olmasını sağlamak için, giriş matrisine piksel ekleme işlemi (dolgulama olarak da bilinir) uygulamak gerekebilir.

3.2.5.2. Aktivasyon Katmanı

Aktivasyon fonksiyonu olmayan bir yapay sinir ağı, basit bir lineer regresyon modelinden farksız olacaktır. Gerçek dünya verileri olarak görüntü, ses, video gibi karmaşık verileri öğretmek için yapay sinir ağları aktivasyon fonksiyonlarına ihtiyaç duyarız.

Aktivasyon işlemi, giriş sinyali üzerinden yapılan doğrusal olmayan dönüşümdür. Bu dönüştürülmüş çıktı, bir sonraki nöron katmanına girdi olarak iletilir.

Genellikle ReLU, tanh, Sigmoid gibi aktivasyon fonksiyonları kullanılır. Hız açısından en iyi sonucu ReLU verdiği için genellikle tercih edilir. Çıkış katmanındaki aktivasyon fonksiyonu ise spesifik probleme göre değişebilir.



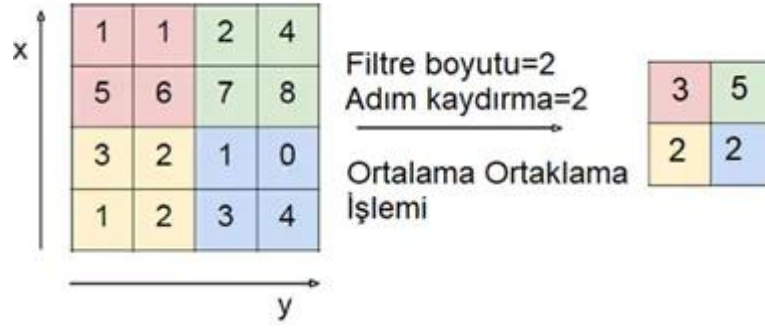
Şekil 3.4. Aktivasyon fonksiyonları

3.2.5.3. Ortaklama Katmanı

Literatürde havuzlama katmanı olarak da bilinen pooling katmanında, boyut indirgeme işlemi yani down-sampling uygulanır. Bu katmanda herhangi bir öğrenme işlemi gerçekleştirilmez. Temel amaç, giriş matrisinin kanal sayısını sabit tutarak genişlik ve yükseklik boyutlarını azaltmaktır. Bu, hesaplama karmaşıklığını azaltmaya yardımcı olur. Bu nedenle, genellikle evrişim katmanından sonra kullanılır. Bu işlemi gerçekleştirmek için aşağıdaki iki yöntem kullanılmaktadır.

3.2.5.3.1. Ortalama Ortaklama

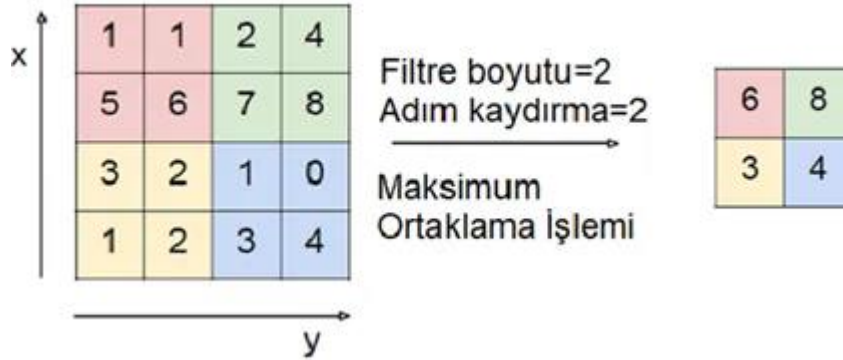
Evrişim işleminden sonra uygulanan ortaklama işleminde, ortalama pooling yöntemi kullanılarak filtrenin görüntü matrisi üzerinde gezinmesi sonucunda kesiştiği noktalardaki piksel değerlerinin ortalaması alınarak boyutu küçültülmüş yeni bir görüntü matrisi oluşturulur. Aşağıdaki örnekte, 4×4 boyutundaki matrise ortalama pooling işlemi uygulandıktan sonra 2×2 boyutundaki yeni görüntü matrisi elde edilmiştir.



Şekil 3.5. Ortalama ortaklama işlemi

3.2.5.3.2. Maksimum Ortaklama

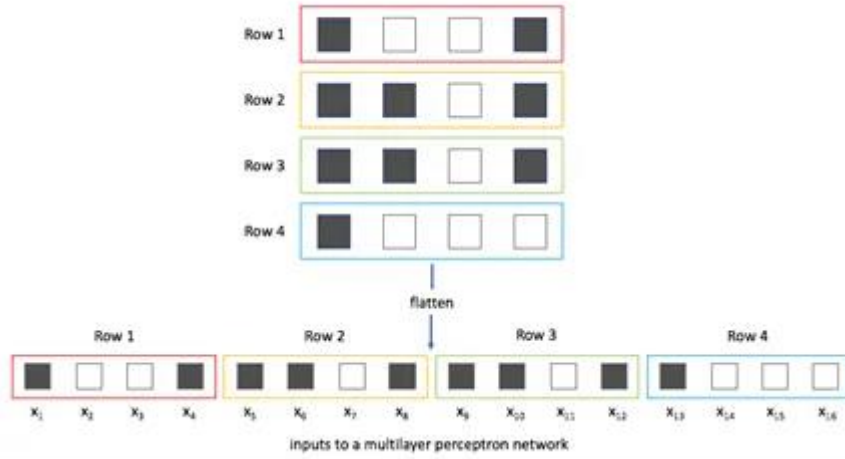
Evrişim işleminden sonra uygulanan ortaklama işleminde, maksimum pooling yöntemi kullanılarak filtrenin görüntü matrisi üzerinde gezindiği her bölgede kesiştiği noktalardaki piksel değerlerinden en büyüğü seçilerek boyutu küçültülmüş yeni bir görüntü matrisi oluşturulur. Aşağıdaki örnekte, 4×4 boyutundaki matrise maksimum pooling işlemi uygulandıktan sonra 2×2 boyutundaki yeni görüntü matrisi elde edilmiştir.



Şekil 3.6. Maksimum ortaklama işlemi

3.2.5.4. Flattening Katmanı

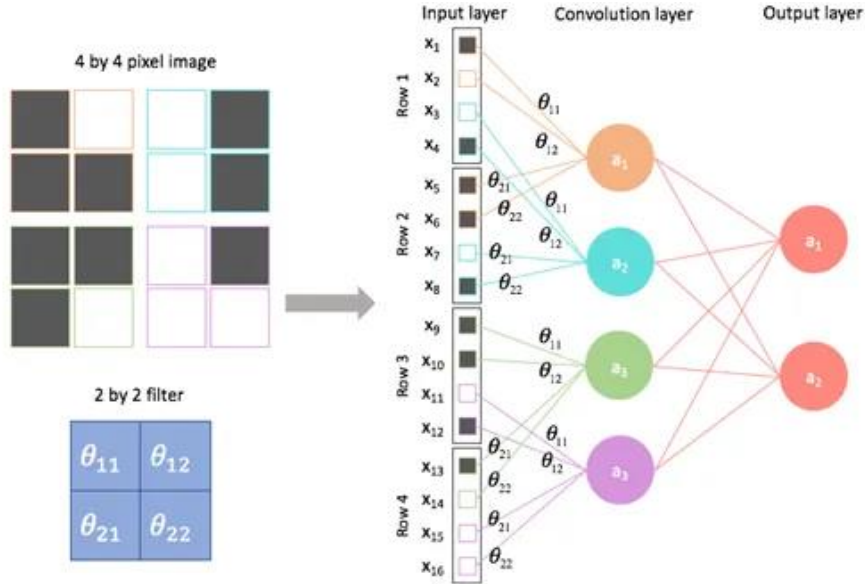
Bu katmana kadar gerçekleştirilen tüm işlemler matrisler üzerinde yapılmıştır. Ancak bir sonraki katmana aktarılabilmesi ve işlenebilmesi için yapay sinir ağlarının gerektiği gibi tek düzlemli bir vektöre dönüştürülmesi gerekmektedir. Bu dönüşüm işlemi, flattening katmanı tarafından gerçekleştirilir.



Şekil 3.7. Flattening katmanı

3.2.5.5. Tam Bağlantı Katmanı

Bu katmanda, flattening katmanı tarafından tek düzlemli hale dönüştürülen vektörler alınarak yapay sinir ağlarına giriş olarak sağlanır. Bu adımla birlikte ilgili öğrenme işlemi süreci başlatılmış olur.



Şekil 3.8. Tam bağlantı katmanı

3.2.5.6. Toplu Normalleştirme Katmanı

Normalleştirme, verileri standartlaştırmak için kullanılan bir ön işleme tekniğidir. Batch Normalization ise bir sinir ağının katmanları arasında yapılan bir normalleştirme işlemidir.

Verileri tümüyle değil, mini gruplar halinde normalleştirerek işlem yapar. Bu yöntem, eğitim sürecini hızlandırır ve daha yüksek öğrenme oranlarını kullanmayı mümkün kılarak öğrenmeyi kolaylaştırır. Genellikle evrişim katmanı ile aktivasyon katmanı arasında yer alır.

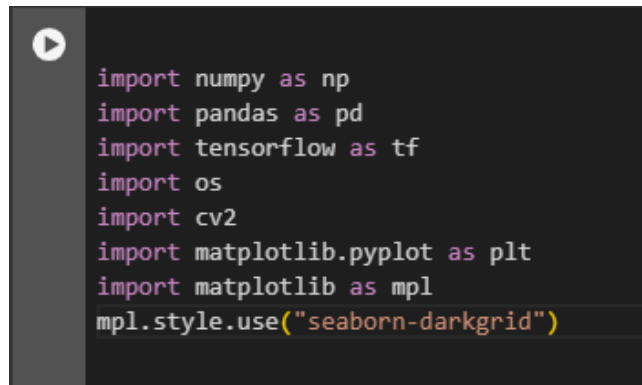
3.2.5.7. Seyreltme Katmanı

Dropout eğitim sırasında rastgele seçilen belirli nöron setlerinin (birimlerin) geçici olarak devre dışı bırakılması demektir. Bu yöntem, modelin overfitting eğilimini azaltmaya yardımcı olabilir. Dropout katmanı kullanılarak modelin genel performansı artırılabilir.

3.3. Modelin Eğitilmesi

Modelin eğitimi, Google Colab üzerinde gerçekleştirilmiştir. Google Colab, bulut tabanlı bir Jupyter defteri ortamı sunarak, Python programlama dilinde makine öğrenimi ve derin öğrenme modellerinin hızlı bir şekilde geliştirilip eğitilmesine imkan tanır. Colab'in sağladığı önemli avantajlar arasında ücretsiz GPU ve TPU kaynaklarına erişim, kullanım kolaylığı ve paylaşım imkanı yer almaktadır.

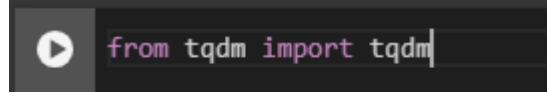
Görüntü işleme projelerinde sıkça kullanılan NumPy, Pandas, TensorFlow ve ne durumda olduğumuzu anlamak için kullandığımız matplotlib gibi temel kütüphaneler projeye dahil edildi. Ayrıca matplotlib kütüphanesini kullanarak grafiklerin arka planı "seaborn-darkgrid" temasıyla ayarlandı, çizdirilen grafikler daha rahat okunabilir hale geldi.



```
import numpy as np
import pandas as pd
import tensorflow as tf
import os
import cv2
import matplotlib.pyplot as plt
import matplotlib as mpl
mpl.style.use("seaborn-darkgrid")
```

Şekil 3.9. Kütüphanelerin projeye dahil edilmesi

tqdm kütüphanesi kullanılarak for döngüsü içinde ilerleme çubuğunu göstererek işlemler takip edildi. Döngü sırasında ne kadar ilerlediğimizi ve işlemlerin mevcut durumunu görsel olarak izleyebilir hale geldik.



Şekil 3.10. tqdm kütüphanesinin projeye dahil edilmesi

Bu kod bloğunda, belirtilen dizindeki alt klasörlerde bulunan görüntü dosyalarını okuyarak bir görüntü ve etiket dizisi oluşturuldu. `image_array` ve `label_array` adında iki liste kullanılarak işlem gerçekleştirildi ve sonunda `image_array`, bir NumPy dizisine dönüştürüldü. Veri yolu (`path`) değişkeni, işlenen görüntülerin bulunduğu dizini temsil ediyor.

Her bir alt klasör için alt dosyalar (`file_sub`) döngüyle işlendi. "neutral" veya "happy" klasörlerindeki ilk 18000 görüntü alındı, diğer klasörlerdeki tüm görüntüler işlendi. Her bir görüntü okunduktan sonra renk formatı BGR'den RGB'ye dönüştürüldü ve `image_array` listesine eklendi. Etiketler (`label_array`), alt klasörün indeks değeri (`i`) olarak atandı. Belirtilen dizindeki görüntü dosyaları okunarak bir görüntü ve etiket dizisi elde edildi.



Şekil 3.11. Dizilerin oluşturulması

Benzersiz etiketleri ve her bir etiketin sayısını bilerek, elimizdeki verilerin ne kadar çeşitli olduğunu ve hangi etiketlerin en yaygın olduğunu anladık. `label_array` dizisindeki benzersiz etiketlerin (a) ve her bir etiketin sayısının (b) çıktısı alındı.


```
a,b=np.unique(label_array,return_counts="True")
a
b
```

Şekil 3.12. a ve b etiketinin sayısının çıktısının alınması

image_array dizisindeki görüntülere normalizasyon işlemi uygulandı. Piksel değerleri 0-255 aralığından 0-1 aralığına normalize edildi. Ardından, label_array listesini bir diziye dönüştürüldü. Bu işlemleri yaparak veri hazırlığını tamamlıyoruz.

```
image_array=np.array(image_array)/255.0
label_array=np.array(label_array)
```

Şekil 3.13. Görsellerin piksellerinin normalizasyon işlemi

label_to_text adında bir sözlük tanımlandı. Bu sözlük, etiketlerin (0-6 arasındaki sayılar) insan okunabilir metin karşılıklarını içeriyor. Bu şekilde her bir etiket sayısı, etiketin karşılık geldiği duygusal ifadeye bağlandı.

```
label_to_text={0:"surprise",1:"fear",2:"angry",3:"neutral",4:"sad",5:"disgust",6:"happy"}
```

Şekil 3.14. Sözlük tanımlanması

train_test_split fonksiyonu kullanılarak image_array ve label_array dizileri train (eğitim) ve test veri setlerine ayrıldı. Eğitim verilerinin %10'u, doğrulama için kullanılmak üzere test setine ayrıldı ve bellek yönetimi yapıldı

```
from sklearn.model_selection import train_test_split
image_array,X_test,Y_train,Y_test=train_test_split(image_array,label_array,test_size=0.1)
gc.collect()
```

Şekil 3.15. Eğitim ve test değerlerinin ayarlanması

Etiketleri metin karşılıklarına dönüştüren bir sözlük tanımlandı. Etiketlerin 0'dan 6'ya kadar olan sayısal karşılıklarını sırasıyla "surprise", "fear", "angry", "neutral", "sad", "disgust" ve "happy" metinlerine çevrildi.

```
label_to_text={0:"surprise",1:"fear",2:"angry",3:"neutral",4:"sad",5:"disgust",6:"happy"}
```

Şekil 3.16. Etiketlerin metine dönüştürülmesi

Keras kütüphanesinden gerekli bileşenler (layers, callbacks, utils, applications, optimizers), Sequential ve Model gibi modellerin tanımlandığı ve mevcut modellerin yüklenip kullanıldığı fonksiyonlar projeye dahil edildi.

```
from keras import layers, callbacks, utils, applications, optimizers
from keras.models import Sequential, Model, load_model
```

Şekil 3.17. Model eğitimi için gerekli kütüphanelerin projeye eklenmesi

Sequential modeli tanımlandı ve ImageNet veri kümesi üzerinde önceden eğitilmiş MobileNetV2 modeli, giriş olarak 48x48 boyutlu RGB görüntüler kabul edecek şekilde import edildi. Önceden eğitilmiş ağırlıklar kullanılarak bu modelin eğitimi aktif hale getirildi. GlobalAveragePooling2D katmanı ile özellik haritalarından global özellikler çıkarılırken, Dropout katmanı ile overfitting riski azaltıldı. Son olarak, 1 adet çıktı sınıfını temsil eden bir Dense katmanı eklendi ve model özetlendi.

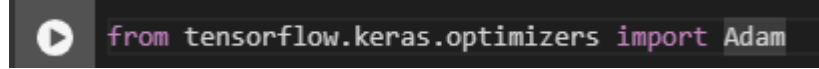
```
model=Sequential()
pretrained_model=applications.MobileNetV2(input_shape=(48,48,3),include_top=False,
                                          weights="imagenet")

pretrained_model.trainable=True
model.add(pretrained_model)
model.add(layers.GlobalAveragePooling2D())
model.add(layers.Dropout(0.3))
model.add(layers.Dense(1))
model.summary()
```

Şekil 3.18. MobileNetV2 üzerine gerekli katmanların eklenmesi

TensorFlow Keras'ın optimizasyon modüllerinden Adam optimizer projeye dahil edildi. Bu optimizer, gradient bazlı optimizasyon yöntemlerinden biridir ve genellikle derin öğrenme

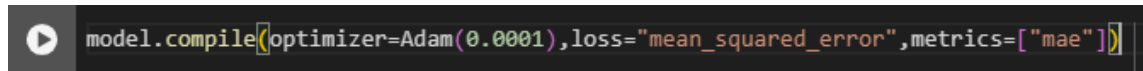
modellerinin eğitiminde kullanılır. Adam optimizer, eğitim sürecinde model parametrelerinin güncellenmesinde momentum ve adaptif öğrenme oranı sağlar [13].



```
from tensorflow.keras.optimizers import Adam
```

Şekil 3.19. Adam optimizer'ın projeye dahil edilmesi

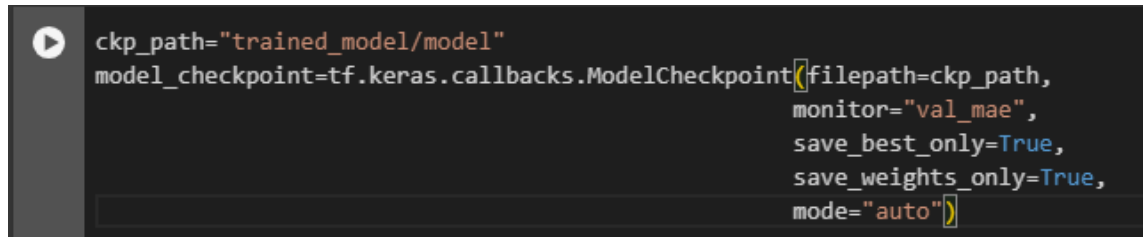
Model derlendi ve Adam optimizer, "mean_squared_error" kayıp fonksiyonu ile "mae" metriği kullanıldı. Adam optimizer'ın öğrenme oranı 0.0001 olarak ayarlandı. Bu ayarlar, modelin performansını optimize etmeye ve hatayı en aza indirmeye yardımcı olur.



```
model.compile(optimizer=Adam(0.0001), loss="mean_squared_error", metrics=["mae"])
```

Şekil 3.20. Modelin derlenmesi

Modelin en iyi ağırlıklarının kaydedilmesi için model kontrol noktası oluşturuldu. ModelCheckpoint callback'i ile en düşük "val_mae" metriğine sahip modelin ağırlıkları "trained_model/model" yoluna kaydedilecek şekilde ayarlandı.



```
ckp_path="trained_model/model"  
model_checkpoint=tf.keras.callbacks.ModelCheckpoint(filepath=ckp_path,  
                                                    monitor="val_mae",  
                                                    save_best_only=True,  
                                                    save_weights_only=True,  
                                                    mode="auto")
```

Şekil 3.21. Modelin en iyi değerlerinin kaydedilmesi

Öğrenme oranını dinamik olarak azaltmak için ReduceLROnPlateau callback'i tanımlandı. Bu callback, belirli bir süre boyunca "val_mae" metriği iyileşme göstermezse, öğrenme oranını %10 oranında düşürür.

```
reduce_lr=tf.keras.callbacks.ReduceLRonPlateau(factor=0.9,  
monitor="val_mae",  
mode="auto",  
cooldown=0,  
patience=5,  
verbose=1,  
min_lr=1e-6)
```

Şekil 3.22. Learning Rate parametresini zamanla düşürülmesi

300 epoch ve 64 batch size değerleri ile modelin eğitimine başlandı. Eğitim sürecinde model kontrol noktası ve öğrenme oranı azaltma callback'leri kullanıldı. Bu süreçte, eğitim ve doğrulama verileri kullanılarak modelin performansı değerlendirildi.

```
EPOCHS=300  
BATCH_SIZE=64  
history=model.fit(image_array,Y_train,  
validation_data=(X_test,Y_test),  
batch_size=BATCH_SIZE,  
epochs=EPOCHS,  
callbacks=[model_checkpoint,reduce_lr])
```

Şekil 3.23. Modelin eğitimine başlanması

300 epoch ile eğitilen modelin performansını değerlendirmemize yardımcı olacak metriklerden bazıları:

loss: 0.0154 - **mae:** 0.0889

val_loss: 1.8961 - **val_mae:** 0.8790

Modelin eğitim sırasında callback'ler kullanılarak modelin en iyi sonuç veren ağırlıklarını yüklemek için `model.load_weights(ckp_path)` kodu kullanıldı. Bu sayede modelin hali kullanıma hazır hale getirildi.

```
model.load_weights(ckp_path)
```

Şekil 3.24. Ağırlıkların yüklenmesi

Test verileri üzerindeki tahminleri almak için model.predict fonksiyonu kullanıldı. Bu işlem, modelin X_test veri kümesi üzerinde tahminler yaparak sonuçları prediction_val değişkenine atar. Böylece modelin test verisi üzerindeki performansı değerlendirilebilir.

```
prediction_val=model.predict(X_test,batch_size=BATCH_SIZE)
```

Şekil 3.25. Modelin test verileri üstündeki tahminlerinin kaydedilmesi

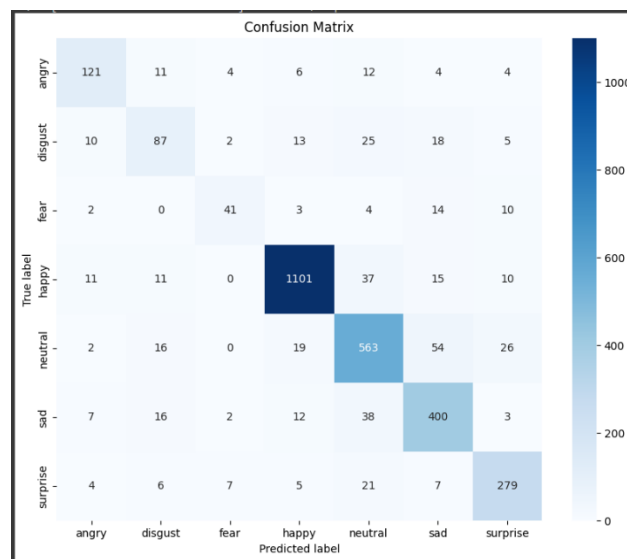
Keras modeli TFLiteConverter kullanılarak TensorFlow Lite formatına dönüştürüldü ve model.tflite dosyasına kaydedildi. Bu işlem, modeli mobil ve gömülü cihazlarda kullanmak için hazır duruma getirdi.

```
converter=tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model=converter.convert()
with open("model.tflite","wb") as f:
    f.write(tflite_model)
```

Şekil 3.26. Eğitilen modeli mobil cihazda kullanabilmek için yapılan dönüştürme işlemi

Confusion Matrix: Modelin tahmin ettiği değerler ile gerçek değerleri karşılaştırır.

Eğitilen Modelin Confusion Matrixi:



Şekil 3.27. Elimizdeki modelden elde edilen Confusion Matrix

F1 - Score : Bir modelin performansını değerlendiren bir metriktir. Sınıflandırma modellerinin performansını değerlendirirken daha dengeli ve güvenilir bir ölçüt sunar. Sadece modelin doğru kararlar verdiği yani true positive ve true negative sonuçlarına bakarak daha keskin bir sonuç elde eder.

Eğitilen modelin F1-Score sonuçları:

F1 - SCORE	
SAD	0.8044
DISGUST	0.5840
ANGRY	0.7926
NEUTRAL	0.9225
FEAR	0.8558
SURPRISE	0.7858
HAPPY	0.8216
ACCURACY	0.8434
MACRO AVG	0.7952
WEIGHTED AVG	0.8414

Şekil 3.28. F1-Score değerleri tablosu

3.4. Android Uygulamasında Kullanıcı Yüz Analizi ile Duygu Durumu Belirlenmesi

Projenin bu kısmında, bir Android uygulaması ile kullanıcı yüz analizi yapılarak duygu durumu belirlenmesi amaçlanmıştır. Uygulama, QR kod tarayıcı olarak hizmet verirken, kamera aracılığıyla kullanıcının yüz ifadelerini analiz eder. QR kod tarama işlemi sırasında, kullanıcının yüzünden 20 farklı sonuç alınır ve en çok tekrar eden duygu durumu belirlenir. Bu duygu durumu her 20 analizden sonra bir metin dosyasına kaydedilir. Böylece uygulama, kullanıcının duygusal tepkilerini gerçek zamanlı olarak izleyip kaydeder.

Bu bölümde, yüz analizi ve duygu tanıma işlemleri için gerekli değişkenler tanımlandı. Ayrıca, duyguların sayısını takip eden sayaçlar ve en sık rastlanan duygu durumu gibi değişkenler belirlendi.

```

public class facialExpressionRecognition {

    private Interpreter interpreter;

    private int INPUT_SIZE;

    private int height = 0;

    private int width = 0;

    private GpuDelegate gpuDelegate = null;

    private CascadeClassifier cascadeClassifier;

    // Emotion recognition variables

    private int[] emotionCounters = new int[7];

    private final String[] emotionLabels = {"Surprised", "Fear", "Angry", "Neutral", "Sad", "Disgust", "Happy"};

    private int readCount = 0;

    private String mostFrequentEmotion = "Neutral"; // Default starting value

```

Şekil 3.29. Değişkenlerin tanımlanması

Bu yapıcı metot, modelin ve sınıflandırıcının yüklenmesini sağlar. Model dosyasının yüklenmesi için gerekli ayarlar yapıldı.

```

facialExpressionRecognition(AssetManager assetManager, Context context, String modelPath, int inputSize) throws IOException {
    INPUT_SIZE = inputSize;
    Interpreter.Options options = new Interpreter.Options();
    gpuDelegate = new GpuDelegate();
    options.addDelegate(gpuDelegate);
    options.setNumThreads(4);
    interpreter = new Interpreter(loadModelFile(assetManager, modelPath), options);
    Log.d("Facial_Expression", "Model is loaded");

    try {
        InputStream is = context.getResources().openRawResource(R.raw.haarcascade_frontalface_alt);
        File cascadeDir = context.getDir("cascade", Context.MODE_PRIVATE);
        File mCascadeFile = new File(cascadeDir, "haarcascade_frontalface_alt");
        FileOutputStream os = new FileOutputStream(mCascadeFile);

        byte[] buffer = new byte[4096];
        int bytesRead;

        while ((bytesRead = is.read(buffer)) != -1) {
            os.write(buffer, 0, bytesRead);
        }
        is.close();
        os.close();
        cascadeClassifier = new CascadeClassifier(mCascadeFile.getAbsolutePath());

        Log.d("Facial_Expression", "Classifier is loaded");

    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

Şekil 3.30. Eğitilmiş olan modelin yüklenmesi

Görüntüdeki yüzleri tespit edildi ve her yüzün etrafına bir dikdörtgen çizildi. Yüzler, gri tonlamaya çevrildi. Haar Cascade yöntemiyle tespit edildi.

```
public Mat recognizeImage(Mat matImage) {
    Core.flip(matImage.t(), matImage, 1);
    Mat grayscaleImage = new Mat();
    Imgproc.cvtColor(matImage, grayscaleImage, Imgproc.COLOR_RGBA2GRAY);
    height = grayscaleImage.height();
    width = grayscaleImage.width();

    int absoluteFaceSize = (int) (height * 0.1);
    MatOfRect faces = new MatOfRect();
    if (cascadeClassifier != null) {
        cascadeClassifier.detectMultiScale(grayscaleImage, faces, 1.1, 2, 2,
            new Size(absoluteFaceSize, absoluteFaceSize), new Size());
    }

    Rect[] faceArray = faces.toArray();
}
```

Şekil 3.31. Yüz tespiti

```
for (Rect face : faceArray) {
    Imgproc.rectangle(matImage, face.tl(), face.br(), new Scalar(0, 255, 0, 255), 2);

    Rect roi = new Rect((int) face.tl().x, (int) face.tl().y,
        ((int) face.br().x) - (int) (face.tl().x),
        ((int) face.br().y) - (int) (face.tl().y));

    Mat croppedRgba = new Mat(matImage, roi);

    Bitmap bitmap = Bitmap.createBitmap(croppedRgba.cols(), croppedRgba.rows(), Bitmap.Config.ARGB_8888);
    Utils.matToBitmap(croppedRgba, bitmap);
    Bitmap scaledBitmap = Bitmap.createScaledBitmap(bitmap, 48, 48, false);
    ByteBuffer byteBuffer = convertBitmapToByteBuffer(scaledBitmap);

    float[][] emotion = new float[1][1];
    interpreter.run(byteBuffer, emotion);
    float emotionValue = emotion[0][0];
    Log.d("Facial_Expression", "Output: " + emotionValue);

    // Update emotion counters for every reading
    updateEmotionCounters(emotionValue);

    Imgproc.putText(matImage, mostFrequentEmotion,
        new Point((int) face.tl().x + 10, (int) face.tl().y + 20),
        1, 1.5, new Scalar(0, 0, 255, 150), 2);
}

Core.flip(matImage.t(), matImage, 0);
return matImage;
}
```

Şekil 3.32. Yüz tespiti

Haar Cascade: Haar Cascade, yüz ve diğer nesnelerin tespiti için kullanılan etkili bir bilgisayarla görme algoritmasıdır. Bu yöntem, farklı bölgelerdeki yoğunluk farklarını belirlemek için Haar özelliklerini kullanır ve hızlı hesaplama için integral görüntü tekniğinden yararlanır. Adaboost algoritması ile zayıf sınıflandırıcıları güçlü sınıflandırıcılara dönüştürerek nesnelerin varlığını belirler. Ardışık aşamalardan oluşan cascade yapısı, nesneleri hızlı ve doğru bir şekilde tespit eder. Haar Cascade, özellikle gerçek zamanlı yüz ve nesne tespiti uygulamalarında yaygın olarak kullanılır.

Analiz edilen her yüz için duygu sayaçları güncellendi ve 20 okuma sonrası en sık rastlanan duyguyu belirlendi. Belirlenen duygu S3'deki emotion.txt dosyasına kaydedildi [14].

```
private void updateEmotionCounters(float emotionValue) {
    if (emotionValue >= 0 && emotionValue < 0.6) {
        emotionCounters[0]++;
    } else if (emotionValue >= 0.6 && emotionValue < 1.2) {
        emotionCounters[1]++;
    } else if (emotionValue >= 1.2 && emotionValue < 2.4) {
        emotionCounters[2]++;
    } else if (emotionValue >= 2.4 && emotionValue < 3) {
        emotionCounters[3]++;
    } else if (emotionValue >= 3 && emotionValue < 4.2) {
        emotionCounters[4]++;
    } else if (emotionValue >= 4.2 && emotionValue < 4.8) {
        emotionCounters[5]++;
    } else {
        emotionCounters[6]++;
    }

    readCount++;
    if (readCount == 20) {
        int maxCount = 0;
        for (int i = 0; i < emotionCounters.length; i++) {
            if (emotionCounters[i] > maxCount) {
                maxCount = emotionCounters[i];
                mostFrequentEmotion = emotionLabels[i];
            }
        }
        for (int i = 0; i < emotionCounters.length; i++) {
            emotionCounters[i] = 0;
        }
        readCount = 0;
        saveEmotionToFile(mostFrequentEmotion);
    }
}
```

Şekil 3.33. Duygu durumunun belirlenmesi

Belirlenen en sık rastlanan duyguyu bir metin dosyasına kaydedildi. Dosya yolunda belirtilen konuma duygular eklendi.

```
private void saveEmotionToFile(String emotion) {
    String filePath = "/sdcard/Download/emotion.txt";
    try {
        FileOutputStream fos = new FileOutputStream(new File(filePath), true); // true to append to the file
        fos.write((emotion + "\n").getBytes());
        fos.close();
        Log.d("Facial_Expression", "Emotion saved to file: " + emotion);
    } catch (IOException e) {
        e.printStackTrace();
        Log.e("Facial_Expression", "Error saving emotion to file: " + e.getMessage());
    }
}
```

Şekil 3.34. Belirlenen duygu durumunun S3'te bulunan dosyaya gönderilmesi

Bitmap görüntü modelin anlayabileceği bir ByteBuffer formatına dönüştürüldü. Görüntü verileri piksel piksel okunarak float değerlerine çevrildi.

```
private ByteBuffer convertBitmapToByteBuffer(Bitmap scaledBitmap) {
    ByteBuffer byteBuffer;
    int sizeImage = INPUT_SIZE; // 48
    byteBuffer = ByteBuffer.allocateDirect(4 * 1 * sizeImage * sizeImage * 3);
    byteBuffer.order(ByteOrder.nativeOrder());
    int[] intValues = new int[sizeImage * sizeImage];
    scaledBitmap.getPixels(intValues, 0, scaledBitmap.getWidth(), 0, 0, scaledBitmap.getWidth(), scaledBitmap.getHeight());
    int pixel = 0;
    for (int i = 0; i < sizeImage; ++i) {
        for (int j = 0; j < sizeImage; ++j) {
            final int val = intValues[pixel++];

            byteBuffer.putFloat((((val >> 16) & 0xFF)) / 255.0f);
            byteBuffer.putFloat((((val >> 8) & 0xFF)) / 255.0f);
            byteBuffer.putFloat(((val & 0xFF)) / 255.0f);
        }
    }
    return byteBuffer;
}
```

Şekil 3.35. Bitmap görüntüsünü derin öğrenme modeli için ByteBuffer formatına dönüştürülmesi

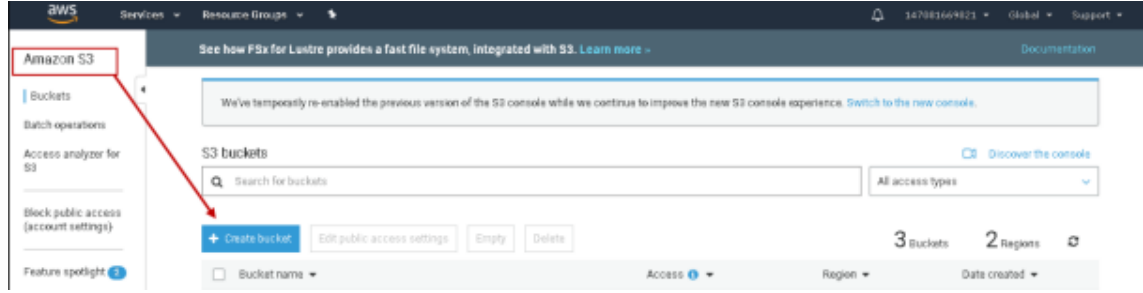
Model dosyası yüklendi.

```
private MappedByteBuffer loadModelFile(AssetManager assetManager, String modelPath) throws IOException {
    AssetFileDescriptor assetFileDescriptor = assetManager.openFd(modelPath);
    FileInputStream inputStream = new FileInputStream(assetFileDescriptor.getFileDescriptor());
    FileChannel fileChannel = inputStream.getChannel();
    long startOffset = assetFileDescriptor.getStartOffset();
    long declaredLength = assetFileDescriptor.getDeclaredLength();
    return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength);
}
```

Şekil 3.36. Eğitilen modelin Android uygulamasında yüklenmesi.

3.5. S3 Bucket'ta Görsellerin Depolanması

AWS yönetim konsolu üzerinden S3 hizmetine giriş yapılarak yeni bir bucket oluşturuldu ve gerekli konfigürasyonlar yapıldı. Bu işlem, AWS'nin sunduğu yönetim araçları kullanılarak gerçekleştirilmişti. Bucket oluşturulması sırasında belirli bir bölge (region) seçildi ve bu bucket için erişim anahtarı (access key) ve gizli erişim anahtarı (secret key) gibi güvenlik ayarları yapıldı.



Şekil 3.37. AWS Konfigürasyonu

3.6. Uygulamayla S3 Arasındaki Dosya İşlemleri

Projenin bu kısmında, uygulamamızdan elde edilen duygu durumu sonucunun S3'e gönderilmesi ve bu sonuca göre uygun reklamın S3'ten uygulamamıza geri gönderilmesi gerçekleştirilmiştir.

Uygulamanın AWS sunucusuyla bağlantısını sağlamak için gerekli olan access key, file path, secret key, region ve bucket name bilgilerini kodlarda gerekli script bölümüne eklendi ve Duygu durumu dosyasının her 10 saniyede güncellenmesi için timer (zamanlayıcı) oluşturuldu.

```
private static final String UPLOAD_FILE_PATH = "/sdcard/Download/emotion.txt";
private static final String BUCKET_NAME = "projewebsite";
private static final String ACCESS_KEY = "AKIAQ3EGSVMIJUDNAHGS";
private static final String SECRET_KEY = "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";
private static final int UPLOAD_TIMER = 10000; // 10 saniye
```

Şekil 3.38. Android uygulaması ile AWS bağlantısının yapılması

Dosya yüklemek için gerekli metod çağrıldı ve dosyanın her 10 saniyede bir güncellenmesi için zamanlayıcı etkinleştirildi.

```

// Handler ve Runnable'ı başlat
handler = new Handler();
uploadRunnable = run() → {
    // Dosyayı yükle
    uploadFileToS3();
    // Belirli bir süre sonra tekrar çalıştır
    handler.postDelayed(this, UPLOAD_TIMER);
};
// İlk yüklemeyi başlat
handler.post(uploadRunnable);
}

```

Şekil 3.39. AWS S3 Bucket'a dosya yüklenmesi işlemi

Amazon S3'ye belirtilen dosyayı yüklemek için gerekli adımları yapıldı. İlk olarak, AWS kimlik bilgileri kullanılarak bir BasicAWSCredentials nesnesi oluşturuldu ve bu bilgilerle bir AmazonS3Client istemcisi Oluşturuldu. Ardından, yeni bir thread oluşturularak dosya yükleme işlemi arka planda başlatıldı. Dosya başarıyla yüklendiğinde, kullanıcı arayüzünde "File uploaded successfully" şeklinde bir bildirim gönderildi; herhangi bir hata durumunda ise, "File upload failed" şeklinde bir hata mesajı kullanıcıya gönderildi.

```

private void uploadFileToS3() {
    BasicAWSCredentials credentials = new BasicAWSCredentials(ACCESS_KEY, SECRET_KEY);
    AmazonS3 s3Client = new AmazonS3Client(credentials);

    new Thread(run() → {
        try {
            File file = new File(UPLOAD_FILE_PATH);
            PutObjectRequest putObjectRequest = new PutObjectRequest(BUCKET_NAME, Upload_File_Name, file)
                .withCannedAcl(CannedAccessControlList.PublicRead);
            s3Client.putObject(putObjectRequest);
            runOnUiThread(run() → {
                Toast.makeText(MainActivity.this, "File uploaded successfully", Toast.LENGTH_SHORT).show();
            });
        } catch (Exception e) {
            e.printStackTrace();
            runOnUiThread(run() → {
                Toast.makeText(MainActivity.this, "File upload failed", Toast.LENGTH_SHORT).show();
            });
        }
    }).start();
}

```

Şekil 3.40. AWS S3 Bucket'a dosya yüklenmesi işlemi

Dosya indirme işlemi, Amazon S3'den belirtilen dosyanın indirilmesi için gerekli adımlar tamamlandı. AWS kimlik bilgileri kullanılarak bir 'BasicAWSCredentials' nesnesi oluşturuldu

ve bu bilgilerle bir `AmazonS3Client` istemcisi başlatıldı. İndirme işlemi, arka planda yeni bir thread içinde gerçekleştirildi; Dosya başarıyla yerel depolama alanına kaydedildi. İşlem başarıyla tamamlandığında, kullanıcıya "File downloaded successfully" bildirimi gönderildi ve indirilen dosya bir `ImageView` içinde gösterildi. Herhangi bir hata durumunda ise, "File download failed" şeklinde bir hata mesajı kullanıcıya gönderildi.

```
private void downloadFileFromS3(String Upload_File_Name) {
    BasicAWSCredentials credentials = new BasicAWSCredentials(ACCESS_KEY, SECRET_KEY);
    AmazonS3 s3Client = new AmazonS3Client(credentials);

    new Thread(run() → {
        try {

            S3Object s3Object = s3Client.getObject(new GetObjectRequest(BUCKET_NAME, Download_File_Name));
            InputStream inputStream = s3Object.getObjectContent();
            File file = new File(getExternalFilesDir(Environment.DIRECTORY_PICTURES), Download_File_Name);
            FileOutputStream outputStream = new FileOutputStream(file);
            byte[] buffer = new byte[1024];
            int length;
            while ((length = inputStream.read(buffer)) > 0) {
                outputStream.write(buffer, 0, length);
            }
            outputStream.close();
            inputStream.close();

            runOnUiThread(run() → {
                Toast.makeText(MainActivity.this, "File downloaded successfully", Toast.LENGTH_SHORT).show();
                ImageView imageView3 = findViewById(R.id.imageView3);
                Bitmap bitmap = BitmapFactory.decodeFile(file.getAbsolutePath());
                imageView3.setImageBitmap(bitmap);
                imageView3.setVisibility(View.VISIBLE);
                imageButton.setVisibility(View.VISIBLE);
            });

        } catch (Exception e) {
            e.printStackTrace();
            runOnUiThread(run() → {
                Toast.makeText(MainActivity.this, "File download failed", Toast.LENGTH_SHORT).show();
            });
        }
    }).start();
}
```

Şekil 3.41. S3 Bucket'tan uygun reklamın yerel depolamaya kaydedilmesi ve kullanıcıya gösterilmesi

3.7. Site Tasarımı

Site tasarımını CSS kullanarak gerçekleştirildi. Butonların ve başlıkların konumları, renkleri ve stilleri aynı şekilde düzenledi. Bu düzenlemeleri style.css ve index.css dosyasında kodlandı ve sitenin tasarım kısmı tamamlandı.

Explore Your Emotions

Please select advertisement.

Upload File

Şekil 3.42. Kullanıcının reklam yükleyebileceği web sitesinin ekran görüntüsü

3.8. Yüklenen Reklamların Analizi ve Uygun Reklamın Belirlenmesi

3.8.1. Kullanılan Hizmetler

3.8.1.1. S3 (Simple Storage Service) Bucket Nedir?

S3 bucket, web siteleri ve uygulamalar için dosya depolamak için kullanılan bir bulut depolama hizmetidir. S3 bucket'lar, Amazon Web Services (AWS) platformunun bir parçasıdır. S3 bucket'lar, ölçeklenebilir, güvenilir ve düşük maliyetlidir. S3 bucket'lar, metin dosyaları, resimler, videolar ve diğer veri türleri dahil olmak üzere çeşitli dosya türlerini depolamak için kullanılabilir [15].

3.8.1.2. Rekognition Hizmeti Nedir?

Amazon Rekognition, Amazon Web Services (AWS) tarafından sunulan bir görüntü ve video analiz hizmetidir. Bu hizmet, makine öğrenimi teknolojilerini kullanarak resim ve videolar içindeki nesneleri, kişileri, metinleri, sahneleri ve etkinlikleri tespit edebilir. Ayrıca yüz analizi ve yüz tanıma yetenekleriyle, yüzlerin kimliklerini doğrulama ve karşılaştırma işlevleri sunar. Rekognition; güvenlik, müşteri etkileşimi, medya analizi ve içerik yönetimi gibi çeşitli alanlarda kullanılarak işletmelerin veri odaklı kararlar almasını ve süreçlerini otomatikleştirmesini sağlar.

3.8.1.3. Lambda Hizmeti Nedir?

Amazon Lambda, AWS tarafından sunulan sunucusuz bilgi işlem hizmetidir. Kendi sunucularınızı yönetmeden kod çalıştırmanızı sağlar. Otomatik ölçeklenebilir ve AWS hizmetleriyle entegre edilebilir, böylece veri işleme, arka plan işlemleri ve API hizmetleri gibi çeşitli görevleri kolayca yönetmenize olanak tanır.

3.8.1.4. IAM Hizmeti Nedir?

Amazon IAM, AWS tarafından sunulan kimlik ve erişim yönetimi hizmetidir. Bu hizmet, AWS kaynaklarına kimlerin erişebileceğini ve bu kaynaklarla neler yapabileceklerini kontrol etmenizi sağlar. Kullanıcıları, grupları ve rolleri yöneterek ayrıntılı izinler tanımlayabilir ve

güvenliği artırabilirsiniz. IAM, AWS hizmetleriyle entegre çalışarak, hesap yönetimi ve erişim kontrolünü sağlama olanağı tanır [16] [17].

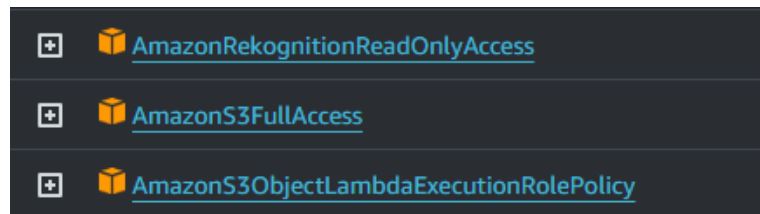
3.8.2. S3 Bucket'taki Reklam Görsellerinin Analizi

Şekil 3.42'de yer alan web sitesi kullanılarak yüklenen reklam görselleri AWS S3 (Simple Storage Service) Bucket'a aktarıldı. Bu görseller duygu durum analizi için AWS (Amazon Web Service)'nin sunmuş olduğu Rekognition adlı hizmete gönderildi. S3 bucket'ta yer alan görseller Rekognition servisine gönderilirken Lambda hizmeti kullanıldı. "DetectFaces" isimli Lambda fonksiyonu oluşturuldu [18] [19].

```
response = client.detect_faces(Image={'S3Object': {'Bucket': bucket_name, 'Name': file_name}}, Attributes=['ALL'])
results = []
for face in response.get('FaceDetails', []):
    emotions = face['Emotions']
    primary_emotion = None
    secondary_emotion = None
    for emotion in emotions:
        if emotion['Confidence'] > 70:
            primary_emotion = emotion
        elif emotion['Confidence'] > 50:
            secondary_emotion = emotion
    if primary_emotion:
        result = {
            "file_name": file_name,
            "emotion": primary_emotion
        }
```

Şekil 3.43. Rekognition Hizmetinin Analiz Sonuçlarını Kaydetme

Lambda servisinde yazılan kodların düzgün çalışabilmesi için gerekli izinler verilmeli, bu işlem AWS'nin IAM hizmetiyle yapıldı. Öncelikle IAM hizmetini kullanabilmek için gerekli rol oluşturuldu ve bu role gerekli izinler verildi.



Şekil 3.44. IAM Rolüne Verilen İzinler

3.8.3. IAM Rolüne Verilen izinler

3.8.3.1. AmazonRekognitionReadOnlyAccess

Amazon Rekognition hizmetinden sadece okuma izni sağlar. Yani Rekognition hizmetinden veri alabilir ancak herhangi bir değişiklik yapamaz.

3.8.3.2. AmazonS3FullAccess

Amazon S3 hizmetine tam erişim sağlar. Bu izinle, S3 üzerinde okuma, yazma, silme ve yönetim dahil her türlü işlemi yapabilirsiniz.

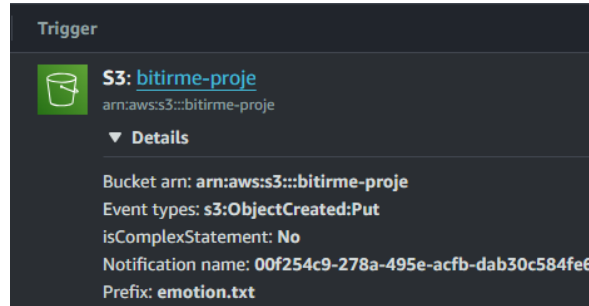
3.8.3.3. AmazonS3ObjectLambdaExecutionRolePolicy

Amazon S3 Object Lambda fonksiyonlarının çalışması için gerekli izinleri sağlar. Bu, Lambda fonksiyonlarının S3 nesneleri üzerinde işlem yapabilmesini mümkün kılar.

3.8.4. İşlemin Otomatikleştirilmesi

Kurulan sistem manuel olarak test edildiğinde başarılı sonuç elde edildi. Bu işlemin otomatik olarak yapılması için DetectFaces isimli Lambda fonksiyonuna trigger (tetikleyici) eklendi.

Telefondan gelen duygu durum analizi sonuçları S3 Bucket içerisindeki “emotion.txt” isimli dosyada depolanıyordu. Eklenen trigger (tetikleyici) “emotion.txt” dosyasına yeni veri geldiğinde, yani “emotion.txt” dosyasına PUT işlemi gerçekleştiğinde, Lambda fonksiyonunu otomatik olarak çalıştıracak şekilde ayarlandı.



Şekil 3.45. Lambda fonksiyonuna eklenen trigger'ın detayları

3.8.5. Yapılan İşlemlerin Özeti

Rekognition servisine gönderilen reklam görsellerinin analizi tamamlandıktan sonra sonuçları JSON formatında kullanıcıya geri döndürdü. Gelen sonuçlardaki “emotion” bölümü lazım olduğundan, Lambda fonksiyonu sadece “emotion” sonuçlarını değerlendirecek şekilde ayarlandı. emotion.txt dosyasının son satırındaki duygu durum değeri ile reklam analizi sonucunda Rekognition hizmetinin tespit ettiği sonuç belirli bir confidence (güven) oranının üstünde eşleştiğinde, uygun olan reklam görseli telefona geri gönderilmek üzere example.jpg dosyasının üstüne yazıldı.

Bütün işlemler tamamlandığında S3 bucket'ta depolanan reklamların analizi, eklenen trigger (tetikleyici) ile emotion.txt dosyasına her yeni veri girişi olduğunda Lambda fonksiyonunu otomatik olarak çalıştırarak Rekognition hizmeti ile tamamlandı. Telefonda gelen duygu durum sonuçlarıyla karşılaştırma yapıp uygun reklam belirlendi. Uygun reklam telefona gönderildi.

4. SONUÇ

Bu proje, kullanıcıların duygu durumlarını analiz ederek onlara uygun reklamları sunma amacını başarıyla gerçekleştirmiştir. Bu sayede, kullanıcıya daha kişisel ve ilgi çekici bir deneyim sunulmakta, reklamların etkisi ve kullanıcı etkileşimi önemli ölçüde artırılmaktadır.

Projede kullanılan duygu analizi algoritmaları, kullanıcıların duygusal tepkilerini yüksek doğrulukla tespit edebilmekte ve bu bilgiler doğrultusunda reklam seçiminde optimize edilmiş sonuçlar elde edilmektedir. Bu durum, reklamların sadece kullanıcıların ilgi alanlarına değil, aynı zamanda duygusal durumlarına da hitap etmesinin ne kadar önemli olduğunu göstermektedir.

Sonuç olarak bu proje, dijital reklamcılık alanında duygu analizi kullanarak kişiselleştirilmiş reklamların etkinliğini artırmada önemli bir adım atmıştır. Kullanıcıların duygu durumlarını göz önünde bulundurarak sunulan reklamlar, hem kullanıcı memnuniyetini hem de reklam verenlerin geri dönüş oranlarını artırmaktadır. Gelecekte, daha gelişmiş duygu analizi algoritmaları ve daha geniş veri setleri ile bu tür kişiselleştirilmiş reklamcılık stratejilerinin daha da yaygınlaşması beklenmektedir. Bu proje, reklamcılığın geleceği için umut verici bir model sunmaktadır.

5. KAYNAKLAR

- [1] Ünal, C., & Bay, Ö. (2009). Java Programlama Dili'nin Bilgisayar Destekli Öğretimi. Bilişim Teknolojileri Dergisi, 2(1), 13-14.
- [2] Pişkin, M. (2018, October 25). OpenCV Nedir? Mesut Pişkin Blog: <https://mesutpişkin.com/blog/opencv-nedir.html> adresinden alındı
- [3] Akgün, D. (2021). A TensorFlow implementation of Local Binary Patterns Transform. MANAS Journal of Engineering, 9(1), 15-21.
- [4] Kaggle'da Yeni Başlayanlar için Sözlük. (2023, Mart 14). CoderSpace: <https://coderspace.io/sozluk/kaggle#:~:text=Kaggle%20veri%20bilimi%20kariyerin%20yeni,takip%20etmelerine%20de%20olanak%20tan%C4%B1r> adresinden alındı
- [5] Durna, M. B. (2019, Ocak 31). Veri Bilimi İçin Temel Python Kütüphaneleri-1 : Numpy. Bilişim Hareketi: <https://medium.com/pisanoeng/veri-bilimi-i-CC%87%C3%A7intemel-python-k%C3%BCt%C3%BCphaneleri-numpy-1-14a8e668b918> adresinden alındı
- [6] Brownlee, J. (tarih yok). Your first deep learning project in python with keras step-by-step. Machine Learning Mastery: <https://machinelearningmastery.com/tutorial-first-neural-networkpython-keras/> adresinden alındı
- [7] Kayalı, N. Z., & Omurca, S. İ. (2021). Konvolüsyonel Sinir Ağları (CNN) ile Çin Sayı Örüntülerinin Sınıflandırması. Computer Science, IDAP-2021 : 5th International Artificial Intelligence and Data Processing symposium, 184-191.
- [8] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 2278-2324.
- [9] Özdener, N. (2008). WEB TASARIMINDA EDITÖR KULLANIMI VE HTML ÖĞRETİMİ. Marmara Üniversitesi Atatürk Eğitim Fakültesi Eğitim Bilimleri Dergisi, 163-175.
- [10] Kabakuş, A. T. (2014). JavaScript için nesne yönelimli programlama yaklaşımları. Erciyes Üniversitesi Fen Bilimleri Enstitüsü Fen Bilimleri Dergisi, 141-147.
- [11] Baydilli, Y. Y. (2021). Polen Taşıyan Bal Arılarının MobileNetV2 Mimarisi ile Sınıflandırılması. Avrupa Bilim Ve Teknoloji Dergisi(21), 527-533. <https://doi.org/10.31590/ejosat.836856>
- [12] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (s. 4510-4520).
- [13] Seyyarer, E., Ayata, F., Uçkan, T., Karci, A. (2020). DERİN ÖĞRENMEDE KULLANILAN OPTİMİZASYON ALGORİTMALARININ UYGULANMASI VE KIYASLANMASI. Computer Science, 5(2), 90-98.
- [14] Yıldız, A., Güney, Z., & Aydın, H. (2023). Performance Evaluation of Face Recognition System (FRS) Developed with Haar Cascade and MongoDB Integration in Recognition of Covered Faces. Bilgisayar Bilimleri Ve Teknolojileri Dergisi, 4(2), 36-45. <https://doi.org/10.54047/bibtcd.1339699>
- [15] Amazon Web Services. (tarih yok). Amazon S3. Amazon Web Services: <https://aws.amazon.com/s3/> adresinden alındı

- [16] Amazon Web Services. (2023, June 25). Getting Started with AWS Identity and Access Management. AWS Documentation: <https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started.html> adresinden alındı
- [17] Amazon Web Services. (2024, June 25). AWS Identity and Access Management (IAM) - Introduction. AWS Documentation: <https://docs.aws.amazon.com/IAM/latest/UserGuide/intro-structure.html> adresinden alındı
- [18] AWS Documentation Team. (2023, June 24). How Amazon Rekognition Works. AWS Documentation: <https://docs.aws.amazon.com/rekognition/latest/dg/how-it-works.html> adresinden alındı
- [19] AWS Documentation Team. (2024). AWS Lambda Documentation. Amazon Web Services (AWS): <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html> adresinden alındı

6. ÖZGEÇMİŞ

Umut BABALIK

Hakkında

Fırat Üniversitesi Bilgisayar mühendisliği öğrencisiyim. İlgili alanlarım arasında gömülü sistemler ve görüntü işleme yer alıyor. Bu konulara olan ilgim, kendimi daha da geliştirmek için çalışmalar yapmamı ve kendimi sürekli güncel tutmamı sağlıyor. Takım çalışmasına uygun olduğumu düşünüyorum. Üniversite öğrenimim boyunca aldığım dersler ve kurslar ile kendimi geliştirmeye devam ediyorum.

İletişim Bilgileri

E-posta: babalikumut1@gmail.com

Eğitim Bilgileri

Lise Öğrenimi: Tevfik İleri Anadolu Lisesi

Lisans Öğrenimi: Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü

Yabancı Diller: İngilizce

İş Tecrübesi

Temmuz 2023 - Ağustos 2023

Bimser - Stajyer

Eylül 2023 – Ekim 2023

Elasoft – Stajyer

Deniz GÖL

Hakkında

Bilgisayar mühendisliği alanında eğitim gören ve kendimi sürekli geliştiren biriyim. Gömülü sistemler, derin öğrenme, oyun programlama ve fotoğrafçılık alanlarında çeşitli projeler geliştirdim. Ayrıca, web ve mobil uygulama geliştirme konularında da deneyim sahibiyim.

İletişim Bilgileri

E-posta: 205260072@firat.edu.tr

Eğitim Bilgileri

Lisans Öğrenimi;

Dokuz Eylül Üniversitesi Fen Fakültesi Fizik 2019-2020

Faculty of Mathematics and Computer Science, University of Lodz 2022-2023

Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü 2019-2024

İş Tecrübesi

EYLÜL 2019- HAZİRAN 2020

Product Photographer - ÖZTÜRK E-TİCARET

HAZİRAN 2023- TEMMUZ 2023

Software Engineer - YAŞAR BİLGİ

EYLÜL 2023- EKİM 2023

Stajyer - Ege Üniversitesi Bilgisayar Mühendisliği Bölümü

Burak TUNÇEL

Hakkında

Bilgisayar mühendisliği öğrencisiyim. Yapay zeka, görüntü işleme ve derin öğrenme gibi konulara ilgi duyuyorum. Çeşitli projeler yaparak kendimi bu alanda geliştiriyorum. Bu konularda çeşitli kurs ve eğitimlere katılarak bilgilerimi güncel tutup yenilikleri takip ediyorum. Takım çalışmalarını seviyorum ve uygun olduğumu düşünüyorum.

İletişim Bilgileri

E-mail: asdesra@gmail.com

Eğitim Bilgileri

Lisans Öğrenimi: Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü

İş Tecrübesi

Stajyer – Deepsis Teknoloji & Yazılım

03/07/2023 – 28/07/2023 | Elazığ, Türkiye