İSTANBUL ÜNİVERSİTESİ-CERRAHPAŞA
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

Derin Öğrenme Mimarilerini Kullanarak Mobil Uygulama Üzerinden Katarakt Tespiti

Hazırlayanlar : Fatih Ateş 1306200091

Ensar A. Kurubacak 1306200029

Danışman : Dr. Öğr. Üyesi Muhammed Erdem İsenkul

Haziran - 2024

ÖNSÖZ

Günümüzde teknolojinin hızla evrim geçirmesi ve yapay zekâ alanındaki yenilikler; sağlık uygulamalarında daha etkili, erişilebilir ve özelleştirilebilir çözümlerin ortaya çıkmasına olanak sağlamaktadır. Bu bağlamda, bu araştırma ve geliştirme projesi, derin öğrenme mimarilerini kullanarak mobil uygulama üzerinden katarakt ve üveit hastalığının tespitini hedefleyen önemli bir girişimdir.

Projemizin merkezine alınan odak noktası, kullanıcıların kendi göz sağlıklarını değerlendirmelerine yönelik pratik bir çözüm sunma amacını taşımaktadır. Flutter platformu ve Dart dilinde geliştirilecek olan mobil uygulama, kullanıcı dostu arayüzüyle birlikte katarakt ve üveit hastalıklarının belirtilerini tespit etme sürecini kolaylaştıracak bir araç olmayı amaçlamaktadır. Bunun yanı sıra, uygulama içerisinde kullanıcıların sağlık verilerini güvenli bir şekilde saklayabilmeleri ve geçmiş analiz sonuçlarına kolayca erişebilmeleri de sağlanacaktır. Böylece, kullanıcılar zaman içinde göz sağlıklarındaki değişimleri takip edebilecek ve gerektiğinde bu verileri sağlık profesyonelleri ile paylaşabileceklerdir.

Derin öğrenme algoritmalarının entegrasyonu, projemizin temelini oluşturan unsurlardan biridir. Kullanıcıların mobil cihazları üzerinden hızlı ve etkili bir şekilde katarakt belirtilerini tespit edebilmelerini sağlamak için Python dilinde geliştirilecek olan back-end tarafı, sağlık verilerini işleyecek ve Convolutional Neural Network (CNN) tabanlı yapay zeka modeli ile katarakt tespiti gerçekleştirecektir. Ayrıca, bu projede Google Cloud platformu da kullanılarak veri depolama, model eğitimi ve dağıtımı gibi işlemler daha verimli ve ölçeklenebilir hale getirilecektir.

Bu projenin öncelikli amacı, teknolojinin sağlık sektöründe nasıl kullanılabileceğine dair bir örnek sunarak, katarakt ve üveit hastalığının teşhisinde erken müdahale imkanlarını artırmak ve bu alandaki potansiyeli ortaya çıkarmaktır. Ayrıca, proje kapsamında elde edilen bilgiler ve deneyimler, gelecekte benzer sağlık sorunlarına yönelik yenilikçi çözümler geliştirilmesine de katkı sağlayacaktır.

Bu projede emeği geçen, danışman öğretim görevlisine teşekkürlerimizi sunarız. Ortaya koyduğumuz bu çalışmanın, sağlık teknolojileri alanında önemli bir adım olmasını temenni ediyoruz.

Fatih Ateş 1306200091
Ensar Aydın Kurubacak 1306200029

İÇİNDEKİLER

ŞEKİL LİSTESİ

TABLO LİSTESİ

KISALTMA LİSTESİ

CNN	**:**	Convolutional Neural Network
LOCS	**:**	Lens Opacities Classification System
TPR	**:**	True Positive Rate
FPR	**:**	False Positive Rate
ROC	**:**	Receiver Operating Characteristic
AUC	**:**	Area Under the ROC Curve
TP	**:**	True Positive
FP	**:**	False Positive
TN	**:**	True Negative
FN	**:**	False Negative

ÖZET

DERİN ÖĞRENME MİMARİLERİNİ KULLANARAK MOBİL UYGULAMA ÜZERİNDEN KATARAKT TESPİTİ

Bu çalışmanın ana hedefi, derin öğrenme yöntemlerini kullanarak göz hastalıklarının tespitini mobil uygulama üzerinden geliştirmektir. Yapılacak yapay zeka hesaplamaları için gerekli veriler, çeşitli katarakt, üveit ve normal göz görüntülerinden oluşan veri kümelerinden elde edilecektir. Bu veri kümeleri, sağlıklı bireylerin, katarakt hastalarının ve üveit hastalarının göz görüntülerini içermektedir. Çalışmanın temel amacı, veri kümelerini incelemek ve ardından yapay zeka modelleri oluşturmaktır. İlk olarak, verilerin daha etkili bir şekilde kullanılmasını sağlamak için veri kümesine bazı veri ön işleme ve artırma işlemleri uygulanacaktır. Görsellerin modelin öğrenmesi için uygun hale getirilmesi amacıyla yeniden boyutlandırma, normalizasyon, döndürme, çevirme, renk ayarı ve gürültü azaltma gibi yöntemler kullanılacak ve bu sayede modelin doğruluğu artırılacaktır. Katarakt ve üveit teşhisi, konvolüsyonel sinir ağı (CNN) ve derin artık ağ (DRN) sınıflandırma yöntemleriyle yapılacaktır.

Yapay zeka modeli ve veri kümeleri oluşturulduktan sonra kullanıcı dostu bir mobil uygulama geliştirilecektir. Geliştirme, uygulamanın performansını optimize etmek, güvenliği sağlamak ve ölçeklenebilirliği temin etmek amacıyla bulut bilişim hizmet paketleri ile gerçekleştirilecektir. Bu mobil uygulama ve arka planda geliştirilen yapay zeka modelleri sayesinde işlenen veriler kullanıcıya geri bildirim şeklinde sunulacaktır. Bu geri bildirim ile kullanıcı, bir göz hastalığına sahip olma olasılığı hakkında bilgilendirilecek ve hastalığın erken teşhisini sağlayabilecektir.

Genel olarak, bu proje, yüz tanıma, katarakt ve üveit modelleri oluşturma, görüntü ön işleme ve mobil uygulama ile bulut bilişim hizmetleri arasındaki iletişimi sağlama yoluyla göz hastalıklarının teşhisinde yardımcı olacak bir mobil uygulama geliştirmeyi amaçlamaktadır.

SUMMARY

DETECTING CATARACTS via MOBILE APPLICATION USING DEEP LEARNING ARCHITECTURES

The main goal of this study is to improve the detection of eye diseases using deep learning methods through a mobile application. The necessary data for the artificial intelligence calculations to be carried out in the background will be derived from datasets consisting of various cataract, uveitis, and normal eye images. These datasets include eye images from healthy individuals, cataract patients, and uveitis patients. In this study, the primary aim is to investigate the datasets and subsequently create artificial intelligence models. Initially, some data preprocessing and augmentation will be applied to the dataset to ensure more effective use of the data. Methods such as resizing, normalization, rotation, flipping, color adjustment, and noise reduction will be used to make the images suitable for the model's learning to increase the model's accuracy. The diagnosis of cataract and uveitis using artificial intelligence will be performed with convolutional neural network (CNN) and deep residual network (DRN) classification methods.

Following the creation of the artificial intelligence model and datasets, a user-friendly mobile application will be developed. The development will be carried out with cloud computing service packages to optimize the performance of the application, ensure security, and provide scalability. Thanks to this mobile application and the artificial intelligence models developed in the background, the processed data will be presented to the user in the form of feedback. With this feedback, the user will be informed about the probability of having an eye disease and will be able to ensure early diagnosis of the disease.

Overall, this project aims to develop a mobile application to assist in the diagnosis of eye diseases by using face detection, creating cataract and uveitis models, image preprocessing, and ensuring communication between the mobile application and cloud computing services.

1. GİRİŞ

1.1. Genel Kavramlar

1.1.1. Katarakt Nedir?

Katarakt, merceğin puslu hale gelmesine neden olan bir proteinin birikmesi sonucu oluşur [1]. Katarakt hastalığı; gelişimsel anormallikler, travma, metabolik bozukluklar, genetik, ilaca bağlı değişiklikler, yaş vb. gibi birçok faktörle ilişkilidir [2]. Katarakt, dünya genelinde körlüğün önemli bir sebebi olarak varlığını sürdürmektedir [3]. Dünya çapında 10 milyonun üzerinde insanın ikincil katarakt nedeniyle körlük yaşadığı ve 35 milyonun üzerinde insanın orta veya ciddi görme bozukluğu yaşadığı tahmin edilmektedir [4].

1.1.2. Üveit Nedir?

Üveit, gözün orta tabakasında bulunan ve uvea olarak adlandırılan kısmın iltihaplanmasıdır. Uvea, iris, siliyer cisim ve koroid adı verilen üç ana bölümden oluşur. Bu yapılar, gözün beslenmesi ve ışığın retina üzerine düzgün bir şekilde odaklanmasını sağlayan kan damarlarını içerir. Üveit, bu bölgelerde ağrı, kızarıklık, bulanık görme ve ışığa hassasiyet gibi belirtilere yol açabilir. Üveitin nedenleri arasında enfeksiyonlar, otoimmün hastalıklar, travma ve bazı sistemik hastalıklar bulunabilir. Tedavi edilmezse, üveit ciddi görme kaybına neden olabilir [5, 6].

1.1.3. Kaggle Nedir?

Kaggle, veri bilimi ve makine öğrenimi alanlarında uzmanlaşmış bir çevrimiçi topluluk ve platformdur. 2010 yılında kurulan Kaggle, kullanıcıların veri setlerine erişmesine, çeşitli veri bilim projelerinde iş birliği yapmasına ve makine öğrenimi yarışmalarına katılmasına olanak tanır. Platform, veri bilimcilerin ve analistlerin yeteneklerini geliştirmeleri için bir öğrenme ve paylaşma ortamı sağlar. Google tarafından 2017 yılında satın alınan Kaggle, aynı zamanda Python ve R programlama dilleri için eğitim materyalleri, kod paylaşımı ve veri analiz araçları sunar. Bunun yanı sıra, Kaggle'da kullanıcılar tarafından yüklenen çeşitli veri setlerine erişim sağlanabilir ve bu veri setleri üzerinde projeler geliştirilebilir. Bu özellikler, Kaggle'ı veri bilimi topluluğu için vazgeçilmez bir kaynak haline getirmektedir [7, 8, 9].

1.1.4. Yapay Zeka ve Yapay Zeka Modelleri Nelerdir?

Yapay zekâ, bilgisayar sistemlerine insana benzer zekâ ve düşünce yetenekleri kazandırmayı amaçlayan bir bilim dalıdır. Bu kavram, makinelerin öğrenme, problem çözme, dil anlama ve karmaşık görevleri gerçekleştirme gibi süreçlerde insan benzeri beceriler kazanmasını hedefler. Yapay zekâ, bilgisayarların büyük veri setlerini analiz etme, desenleri tanıma, özerk kararlar alma ve insanlar gibi düşünme yeteneklerini simüle etme amacını taşır.

Yapay zekânın alt dalları arasında makine öğrenimi, derin öğrenme, doğal dil işleme, uzman sistemler ve görüntü tanıma gibi alanlar bulunur. Yapay zekâ, günümüzde birçok sektörde, sağlık, otomotiv, finans ve eğitim gibi, çeşitli uygulamalarda kullanılarak önemli çözümler sunmaktadır. Bu kavram, teknolojik ilerlemenin sınırlarını genişleterek gelecekte birçok alanda daha etkili ve akıllı sistemlerin geliştirilmesine olanak tanıyacaktır.

Yapay zekâ, derin öğrenme algoritmalarını kullanarak, hastaların göz sağlıklarını değerlendirmek ve katarakt belirtilerini tespit etmek konusunda önemli bir araç olarak karşımıza çıkmaktadır. Mobil uygulamamızın kullanıcı dostu arayüzü ve yapay zekâ modelinin doğru sonuçlar elde etme kapasitesi, katarakt tespiti sürecini daha etkili ve hassas bir hale

getirecektir. Bu proje, Yapay zekânın göz sağlığı alanında pratik uygulamalara dönüştürülerek, teknolojinin sağlık sektöründeki olumlu etkilerini gösterme amacını taşımaktadır.

1.1.5. CNN Nedir?

Convolutional Neural Network (CNN), derin öğrenme alanında öne çıkan ve özellikle görüntü tanıma görevlerinde başarıyla kullanılan bir yapay zekâ mimarisidir. Yapay sinir ağlarının evrimleşmiş bir türü olan CNN, veri setlerindeki desenleri tanıyarak ve öğrenerek, karmaşık görevleri gerçekleştirebilme yeteneğine sahiptir.

CNN, genellikle katmanlar halinde yapılandırılmıştır ve her katman, görüntü özelliklerini çıkarmak ve önceki katmanlardan öğrenilen desenleri anlamak için optimize edilmiştir. Temel bileşenleri arasında evrişim (convolution), aktivasyon (activation), havuzlama (pooling) ve tam bağlantı (fully connected) katmanları bulunur. Evrişim katmanları, görüntülerdeki özellikleri vurgular ve sınıflandırma sürecini geliştirir. Aktivasyon fonksiyonları, öğrenilen desenlerin ağırlıklı toplamını hesaplar ve belirli bir eşiği aşanları etkinleştirir. Havuzlama katmanları, boyut azaltma ve hesaplama yükünü azaltma amacıyla görüntülerdeki önemli özellikleri vurgular. Tam bağlantı katmanları, ağın çıktısını oluşturarak sınıflandırma yapar.

CNN modeli, katarakt ve üveit veri setleri içerisinde yer alan etiketlenmiş göz fotoğrafları üzerinde eğitilecek ve test edilecektir. Bu sayede, derin öğrenme algoritmalarının öğrendiği görsel desenlerle katarakt belirtilerini doğru bir şekilde sınıflandırmak mümkün olacaktır. CNN, görüntü işleme ve özellik çıkarma yetenekleriyle projemizin amacına uygun şekilde, kullanıcıların kendi göz sağlıklarını değerlendirmelerine yardımcı olacak bir araç sunacak şekilde tasarlanacaktır. Bu sayede, mobil uygulamamızın katarakt tespiti konusundaki doğruluk oranını artırarak erken teşhise katkıda bulunması hedeflenmektedir.

1.1.6. Performans Metrikleri Nelerdir?

Yapay zeka modellerinin performansını değerlendirmek için çeşitli metrikler kullanılır. Bu metrikler, modelin doğruluğunu, hassasiyetini ve genelleme yeteneğini ölçmede önemli rol oynar. En yaygın kullanılan performans metriklerinden biri doğruluk (accuracy)'tur. Doğruluk, modelin doğru tahminlerinin toplam tahminlere oranıdır. Ancak, doğruluk özellikle dengesiz veri setlerinde yanıltıcı olabilir, bu yüzden başka metrikler de kullanılır. Hassasiyet (precision), doğru pozitif tahminlerin toplam pozitif tahminlere oranıdır ve yanlış pozitiflerin etkisini azaltır. Duyarlılık (recall) ya da tama (sensitivity), doğru pozitif tahminlerin gerçek pozitiflere oranıdır ve yanlış negatiflerin etkisini azaltır. Bu iki metriğin harmonik ortalaması olan F1 skoru, dengeyi sağlamak için kullanılır ve modelin hem hassasiyet hem de duyarlılığını dikkate alır.

Diğer önemli performans metrikleri arasında kesinlik (specificity), ROC eğrisi (Receiver Operating Characteristic Curve) ve AUC (Area Under the Curve) bulunur. Kesinlik, negatif sınıfların doğru tahmin oranıdır ve dengesiz veri setlerinde önemlidir. ROC eğrisi, modelin farklı eşik değerlerindeki performansını gösterir ve AUC, bu eğrinin altındaki alanı ölçer, modelin genel performansını özetler. Kayıp fonksiyonları (loss functions) da yaygın olarak kullanılır; bunlar, modelin eğitim sırasında yaptığı hataları ölçer ve genellikle optimizasyon sürecinde minimize edilmeye çalışılır. Ortalam kare hatası (Mean Squared Error - MSE) ve log-kayıp (log-loss), bu tür kayıp fonksiyonlarına örnektir. Bu metrikler, modelin performansını kapsamlı bir şekilde değerlendirmek ve iyileştirmek için kullanılır [10, 11, 12].

1.1.7. Google Cloud Nedir?

Google Cloud, Google tarafından sunulan bir bulut bilişim hizmetleri paketidir. Bu platform, uygulama geliştirme, veri depolama, veri analizi ve makine öğrenimi gibi birçok bulut

tabanlı hizmeti içerir. Google Cloud, altyapı olarak hizmet (IaaS), platform olarak hizmet (PaaS) ve yazılım olarak hizmet (SaaS) modellerinde geniş bir yelpazede çözümler sunar. Bu hizmetler arasında Google Compute Engine, Google App Engine, Google Kubernetes Engine, ve BigQuery gibi güçlü araçlar bulunmaktadır. Google Cloud, kullanıcıların verilerini güvenli bir şekilde depolamalarına, büyük veri analitiği gerçekleştirmelerine ve ölçeklenebilir uygulamalar geliştirmelerine olanak tanır.

Google Cloud'un sunduğu avantajlardan biri, Google'ın dünya çapındaki veri merkezleri aracılığıyla sağladığı yüksek güvenilirlik ve düşük gecikmeli erişimdir. Ayrıca, Google Cloud'un yapay zeka ve makine öğrenimi alanında sunduğu gelişmiş araçlar, işletmelerin ve geliştiricilerin karmaşık modeller oluşturmasına ve bu modelleri büyük veri setleri üzerinde hızlı bir şekilde eğitmesine yardımcı olur. TensorFlow ve AI Platform gibi araçlar, makine öğrenimi projelerini yönetmeyi ve dağıtmayı kolaylaştırır. Google Cloud, aynı zamanda esnek fiyatlandırma modelleri ve güçlü güvenlik özellikleri ile çeşitli ölçeklerdeki işletmeler için cazip bir seçenek haline gelmiştir [13, 14, 15].

1.1.8. Flask API Nedir?

Flask, Python programlama dili ile yazılmış hafif ve esnek bir web uygulama çatısıdır. 2010 yılında Armin Ronacher tarafından oluşturulan Flask, mikro çerçeve (microframework) olarak bilinir çünkü veri tabanı soyutlaması veya form doğrulaması gibi bir web uygulaması için gerekli olan birçok ortak işlevi içermemektedir. Bunun yerine, geliştiricilere ihtiyaca göre genişletme özgürlüğü tanır. Flask, hızlı bir şekilde web uygulamaları ve API'ler (Application Programming Interface) oluşturmak için idealdir. Minimalist yapısı sayesinde, geliştiriciler, sadece ihtiyaç duydukları bileşenleri ekleyerek projelerini basit ve yönetilebilir tutabilirler.

Flask ile bir API oluşturmak, özellikle RESTful API'ler için oldukça yaygındır. Flask, URL yönlendirme, JSON yanıtları, hata işleme ve HTTP metotları gibi temel web hizmetleri işlevselliklerini sunar. Ayrıca, Flask'ın genişletilebilir yapısı, Flask-RESTful veya Flask-SQLAlchemy gibi genişletmelerin kolayca eklenmesine olanak tanır. Bu araçlar, veri tabanı işlemleri, oturum yönetimi ve daha karmaşık API fonksiyonlarının uygulanmasını kolaylaştırır. Flask ile oluşturulan API'ler, genellikle web ve mobil uygulamalar için veri sağlama ve işleme amaçlı kullanılır. Bu nedenle, Flask, geliştiricilere hızlı ve esnek API geliştirme imkanı sunarak, projelerini kısa sürede hayata geçirme olanağı tanır [16, 17].

1.1.9. Firebase Nedir?

Firebase, Google tarafından sunulan ve mobil ve web uygulamaları geliştirmek için kapsamlı bir dizi araç ve hizmet sağlayan bir platformdur. Başlangıçta 2011 yılında Firebase Inc. tarafından geliştirilen platform, 2014 yılında Google tarafından satın alınmış ve geliştirilmiştir. Firebase, geliştiricilere gerçek zamanlı veritabanı, bulut depolama, kullanıcı kimlik doğrulama, barındırma ve analiz gibi çeşitli hizmetler sunar. Özellikle mobil uygulama geliştirme süreçlerini hızlandırmak ve kolaylaştırmak amacıyla tasarlanmıştır. Gerçek zamanlı veritabanı hizmeti, geliştiricilerin uygulamalarında canlı verileri senkronize etmelerine olanak tanır, böylece kullanıcılar anlık güncellemeleri görebilirler. Firebase, aynı zamanda çeşitli arka uç hizmetleri sağlar. Firebase Authentication, kullanıcı kimlik doğrulama süreçlerini basitleştirir ve sosyal medya, e-posta ve şifre gibi yöntemlerle oturum açma desteği sunar [18].

1.1.10. Flutter Nedir?

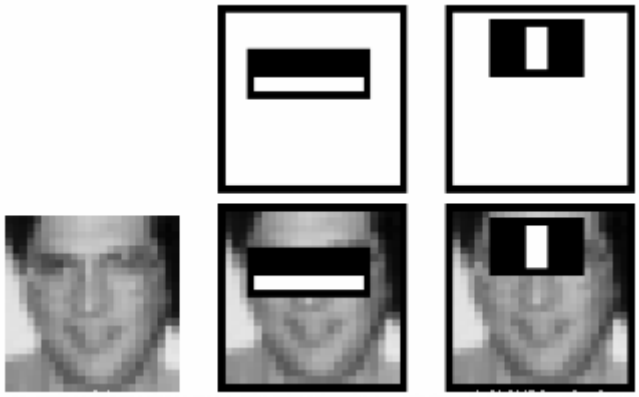
Flutter, Google tarafından geliştirilen açık kaynaklı bir UI (Kullanıcı Arayüzü) yazılım geliştirme kitaplığı ve framework'üdür. Mobil, web ve masaüstü uygulamalar geliştirmek için kullanılan Flutter, hızlı bir şekilde popülerlik kazanmış ve geliştiriciler arasında tercih edilen bir çözüm haline gelmiştir.

Projemizde, Flutter'ı kullanarak mobil uygulama geliştirme sürecini tercih etmemizin temel nedeni, hem kullanıcı dostu bir arayüz tasarlama kolaylığı hem de çeşitli platformlarda sorunsuz çalışabilirlik sağlamaktır.

1.2. Literatür Araştırması

1.2.1. Yüz Tespitine Yönelik Çalışmalar

Viola ve Jones'un 2001 yılında sunduğu "Hızlı Nesne Algılama için Basit Özelliklerin Güçlendirilmiş Kademeli Kullanımı" [19] başlıklı çalışması, nesne algılama için etkili bir yöntem olarak tanınmıştır. Bu makine öğrenimi tabanlı yaklaşımda, bir kaskad fonksiyon, çok sayıda pozitif (yüz içeren) ve negatif (yüz içermeyen) görüntü kullanılarak eğitilir. Daha sonra bu fonksiyon, diğer görüntülerde nesneleri algılamak için kullanılır. Yüz algılama amacıyla, algoritmanın başlangıçta yüz görüntülerine (pozitif görüntüler) ve yüz içermeyen görüntülere (negatif görüntüler) ihtiyacı vardır. Haar özellikleri kullanılarak bu görüntülerden özellikler çıkarılır. Bu özellikler, beyaz ve siyah dikdörtgenler altındaki piksellerin toplamlarının farkı olarak hesaplanır. Bu hesaplamaları hızlandırmak için integral görüntüler kullanılır.



Şekil 1: Göz tespit testi.

Haar özelliklerinin hesaplama verimliliği için, dikdörtgen alanlar genellikle görüntü kenarlarına paralel olacak şekilde yerleştirilir, eğik olmazlar. Ancak, farklı özellikleri ve bir nesnenin ölçek varyasyonlarını yakalamak için çeşitli boyut ve şekillerde dikdörtgenler kullanabiliriz. Haar özelliklerinin temel gücü, üç deseni temsil edebilme yeteneklerinde yatar. Kenarlar, dikey veya yatay olabilirler, çünkü dikdörtgen alanlar bu şekilde yerleştirilmiştir. Farklı görüntü bölgeleri arasındaki sınırları tanımlamak için kullanışlıdır. Çizgiler, görüntüdeki diyagonal kenarlardır. Nesnelerin çizgilerini ve konturlarını tanımlamak için kullanışlıdır. Merkez-çevre özellikleri, dikdörtgen bir bölgenin merkezi ile çevresindeki alan arasındaki yoğunluk değişikliklerini algılar. Belirgin bir şekle veya desene sahip nesneleri tanımlamak için kullanışlıdır. Bu şekilde, Haar özellikleri, nesne algılama işlemlerinde çeşitli şekil ve desenleri etkin bir şekilde temsil eder.

Ancak, hesaplanan özelliklerin çoğu alakasızdır. En iyi özelliklerin seçimi Adaboost yöntemi ile yapılır. Bu yöntemde her özellik tüm eğitim görüntülerine uygulanır ve yüzleri pozitif ve negatif olarak sınıflandırmada en iyi eşik değeri bulunur. En az hata oranına sahip özellikler seçilir. Her görüntü başlangıçta eşit ağırlıkta olup, yanlış sınıflandırılan görüntülerin ağırlıkları artırılır ve süreç tekrarlanır. Son sınıflandırıcı, bu zayıf sınıflandırıcıların ağırlıklı toplamıdır. Yaklaşık 200 özellik ile %95 doğruluk sağlanabilirken, nihai sistemde yaklaşık 6000 özellik bulunur. Yine de, tüm özellikleri her bir pencereye uygulamak verimsizdir. Bu nedenle, Kademeli Sınıflandırıcılar konsepti tanıtılmıştır. Özellikler, kademeli sınıflandırıcılar

olarak gruplandırılır ve bir pencere tüm kademeleri geçerse yüz olarak kabul edilir. Bu yöntem, işlem süresini önemli ölçüde azaltır. Viola ve Jones'un dedektörü, 6000'den fazla özellik ve 38 aşamaya sahip olup, ilk beş aşamada sırasıyla 1, 10, 25, 25 ve 50 özellik bulunur. Ortalama olarak, her alt pencere için 6000'den fazla özellikten sadece 10 tanesi değerlendirilir. Bu yöntem, yüz algılama sürecini hızlandırarak etkin bir hale getirir.

1.2.2. Katarakt Hastalığının Tanısına Yönelik Bilimsel Çalışmalar

"Nuclear_Cataract_Database_for_Biomedical_and_Machine_Learning_Applications" [20], adlı çalışmayı temel alarak katarakt hastalığının tanısında yapay zeka algoritmalarının kullanımını incelemektedir. Araştırmanın ilk amacı, geniş bir veri setinin önemini vurgulamak ve nükleer katarakt tespiti üzerine çalışan araştırmacıların bu veri setlerine erişimini kolaylaştırmaktır (Nuclear Cataract Database, 2022). Veri seti, Meksika şehrindeki bir tıp merkezinden elde edilmiş ve uygun tıbbi protokoller altında uzmanlar tarafından slit lambalarla denetlenmiştir. Bu çalışmada, LOCS III katarakt sınıflandırma sistemine yönelik karşılaştırmalar için bu geniş veri setinin kullanımı incelenmiştir. İkinci amaç, katarakt hastalığının otomatik sınıflandırılması için bir derin evrişimli sinir ağı olan NCC-net'in kullanılmasıdır. Bu ağın performansı, çalışmanın sonuçlarına dayanarak, GoogLeNET gibi bilinen derin mimarilerin aktarım öğrenimi modlarından daha yüksek doğruluk, duyarlılık ve özgüllük değerleri göstermiştir (Nuclear Cataract Database, 2022). Ayrıca, NCC-net mimarisinin Raspberry Pi B+ gibi gömülü sistemlerde de başarılı bir şekilde uygulandığı belirtilmiştir.

"Nuclear_Cataract_Database_for_Biomedical_and_Machine_Learning_Applications" adlı tez çalışmasından esinlenerek, katarakt hastalığının tanısında yapay zeka tekniklerinin kullanılmasının önemini vurgulamakta ve bu alanda yeni bir derin öğrenme modeli olan NCC-net'in performansını incelemektedir.

1.2.3. Üveit Hastalığının Tanısına Yönelik Bilimsel Çalışmalar

Günümüzde tıbbi görüntüleme tekniklerinin hassasiyeti ve derinliklerine olanak sağlayan derin öğrenme (DL) gibi yöntemler, tıbbi teşhis ve hastalık ilerlemesinin izlenmesi için önemli bir araç haline gelmiştir. Göz hastalıklarında da yüksek hassasiyetli görüntüleme teknikleri olan optik koherez tomografi (OCT) gibi tekniklerin kullanımı, tanı ve hastalık ilerlemesinin izlenmesinde kritik öneme sahiptir. Bir çalışmada, Mellak ve ekibi [21], çalışmalarında optik koherez tomografisi (OCT) görüntülerini kullanarak deneysel üveitin nicelendirilmesi için bir makine öğrenme çerçevesi geliştirmişlerdir. Bu çalışmada, hastalığın varlığını doğru bir şekilde tahmin edebilen ve hastalığın farklı evrelerini ayırt edebilen bir sınıflandırma modeli önerilmiştir. Ayrıca, sınıflandırıcının karar verme sürecini açıklamak ve elde edilen sonuçlar hakkında daha derin bir anlayış kazanmak için Grad-CAM görselleştirme tekniği kullanılmıştır. İkinci kısımda, retinada üveitin göstergesi olan ayrılmış parçacıkların tespiti için üç farklı yöntem uygulanmıştır. İlk yöntem tamamen denetimli bir tespit yöntemi iken, ikincisi işaretli nokta işlemi (MPP) tekniğini ve üçüncüsü zayıf denetimli bir segmentasyon yöntemini kullanmaktadır. Segmentasyon modeli, retinanın iki boyutlu dilimlerinde üveitin küçük parçacıklarını segmente etmek için bir temel olarak kullanılmış ve bu da tamamen otomatik bir boru hattı için bir iskelet oluşturmuştur. Retinadaki parçacıkların sayısı hastalığı derecelendirmek için kullanılmıştır ve üç boyutlu (3-D) merkezlerdeki nokta işlem analizi, parçacıkların retinadaki dağılımındaki kümeleme desenlerini göstermektedir.

Bu çalışma, üveitin deneysel nicelendirilmesi için hem sınıflandırma hem de nesne tespiti alanlarında derin öğrenme yöntemlerinin etkin bir şekilde kullanılmasını göstermektedir. Bu bulgular, gelecekte üveit tanısı ve izlenmesinde otomatik ve hassas araçların geliştirilmesine olanak sağlayabilir.

1.3. Literatürdeki Tutarlılık ve Çeşitlilik

Literatürdeki tutarlılık ve çeşitlilik, derin öğrenme ve yapay zeka tekniklerinin kullanımı, görüntü işleme ve ön işleme teknikleri, kullanılan kütüphaneler ve algoritmalar, uygulama alanları gibi temel noktalarda değerlendirilecektir.

Benzerlikler: Literatürde yapılan araştırmalar ve bu tez, derin öğrenme ve yapay zeka tekniklerini kullanarak göz hastalıklarının teşhisi ve takibi üzerine odaklanmaktadır. Her iki alanda da optik koherez tomografisi (OCT) görüntülerinin sınıflandırılması ve nesne tespiti için derin öğrenme algoritmalar kullanılmaktadır. Görüntü işleme ve ön işleme tekniklerinin her iki alanda da önemli olduğu görülmektedir. Literatürdeki araştırmalarda da görüntülerin ölçeklendirilmesi, normalleştirilmesi ve gürültüsünün azaltılması gibi tekniklerin kullanıldığı gözlemlenmektedir. Kullanılan kütüphaneler ve algoritmaların her iki alanda da benzer olduğu görülmektedir. Literatürdeki araştırmalarda OpenCV, Keras ve konvolüsyonel sinir ağları (CNN) gibi kütüphaneler ve algoritmaların kullanıldığı belirtilmektedir. Benzer şekilde, bu tezde de aynı kütüphaneler ve algoritmalar kullanılmaktadır.

Farklılıklar: Uygulama alanları da farklılık göstermektedir. Literatürdeki araştırmalar göz hastalıklarının teşhisi ve takibi üzerine odaklanırken, bu tez mobil uygulamaların geliştirilmesi ve Google Cloud ile iletişimi gibi farklı bir uygulama alanını ele almaktadır.

Bu benzerlikler ve farklılıklar, mevcut literatürdeki bilgiyi anlama ve bu tezin konumunu ve katkılarını belirleme konusunda önemli bir rol oynamaktadır.

2. GENEL KISIMLAR

2.1. Genel Değerlendirme

Bu proje, yapay zeka ve derin öğrenme teknolojilerini kullanarak göz hastalıklarının teşhisini kolaylaştırmayı amaçlayan yenilikçi bir mobil uygulama geliştirmiştir. Uygulama, kullanıcıların göz sağlığını kontrol etmelerini ve erken teşhis sayesinde gerekli tedbirleri almalarını sağlayarak sağlık sektörüne önemli katkılar sunar.

2.2. Kullanılan Algoritmalar

2.2.1. Face Detection

Haar Cascade sınıflandırıcısı, nesne tespiti için kullanılan bir algoritmadır. OpenCV kütüphanesinde bulunan bu algoritma, belirli bir nesnenin (yüz, göz gibi) görüntülerde nasıl göründüğünü öğrenerek onu yeni görüntülerde tespit edebilir.

Haar Cascade sınıflandırıcısı, nesnenin farklı ölçek ve yönlerde nasıl göründüğünü temsil eden bir dizi "özellik" kullanır. Bu özellikler, görüntülerdeki piksellerin yoğunluklarını ve dağılımlarını inceleyerek oluşturulur. Algoritma, önceden etiketlenmiş nesne örnekleri (pozitif örnekler) ve nesnenin olmadığı örnekler (negatif örnekler) içeren bir eğitim seti kullanılarak eğitilmektedir. Eğitim sırasında, algoritma hangi özelliklerin nesneyi doğru şekilde temsil ettiğini ve hangilerinin etkisiz olduğunu öğrenir.

2.2.2. Katarakt/Üveit Modellerinin Oluşturulması

Keras kütüphanesi, yapay sinir ağları (YSA) ve derin öğrenme modellerini oluşturmak için kullanılan popüler bir Python kütüphanesidir. Farklı türde YSA mimarilerini ve katmanları destekleyerek kullanıcıların karmaşık modeller tasarlamalarına ve eğitmelerine olanak tanır.

Bu kütüphanede kullanılan en önemli algoritmalardan bazıları şunlardır: Konvolüsyonel Sinir Ağları (CNN), genellikle görüntü işleme ve bilgisayar görüşünde nesne tanıma, görüntü sınıflandırma gibi görevlerde kullanılır. Tekrarlayan Sinir Ağları (RNN), metin işleme, konuşma tanıma gibi zaman serisi verileriyle ilgili görevlerde kullanılırken, Uzun Kısa Süreli Hafıza (LSTM) ise RNN'lerin bir alt türüdür ve doğal dil işleme, makine çevirisi gibi uzun vadeli bağımlılıkları öğrenmede daha etkilidir. Yoğun katmanlar, sinir ağlarında giriş ve çıkış verileri arasındaki ilişkiyi öğrenmek için kullanılır ve sınıflandırma ve regresyon gibi görevlerde önemli bir rol oynar. Aktivasyon fonksiyonları, sinir ağlarında sinyalleri işleme ve çıktı üretme için kullanılan fonksiyonlardır ve ReLU, sigmoid gibi farklı türleri bulunmaktadır. Kayıp fonksiyonları, modelin tahminleri ile gerçek değerler arasındaki farkı ölçer ve kategorik çapraz entropi gibi çeşitli türleri vardır. Optimizasyon algoritmaları ise modelin ağırlıklarını güncelleyerek kayıp fonksiyonunu minimuma indirmeye çalışır ve Adam, SGD gibi farklı türleri içerir.

Keras, bu algoritmaları ve daha fazlasını kullanarak kullanıcıların geniş bir yelpazedeki yapay zeka modellerini kolayca oluşturmalarına ve eğitmelerine olanak tanır.

2.2.3. Image Preprocessing

Görüntü ön işleme, derin öğrenme modellerinde kullanılan görüntü verilerini modelin öğrenmesi ve doğru tahminlerde bulunması için hazırlama aşamasıdır. Bu aşamada, görüntülerdeki tutarsızlıklar giderilir, boyutlar standart hale getirilir ve modelin öğrenmesini kolaylaştıracak şekilde yeni özellikler eklenir.

Temel ön işleme teknikleri arasında yeniden boyutlandırma, normalleştirme, genişletme, döndürme ve çevirme, renk ayarlama ve gürültü azaltma bulunur. Yeniden boyutlandırma, görüntülerin modelin girdi katmanına uyacak şekilde standart hale getirilmesini sağlar ve modelin daha az bellek kullanmasına olanak tanır. Normalleştirme, görüntü piksellerinin değerlerini 0 ile 1 arasında bir aralığa dönüştürerek farklı aydınlatma koşullarında çekilmiş görüntülerin renkleri arasındaki tutarsızlıkları ortadan kaldırır. Genişletme, tek bir görüntüyü modelin birden fazla örnek olarak yorumlayacağı şekilde boyutlandırır. Döndürme ve çevirme, görüntülerin rastgele döndürülmesi ve çevrilmesi yoluyla veri kümesini yapay olarak genişleterek modelin daha sağlam hale gelmesini sağlar. Renk ayarlama, görüntülerin parlaklığını, kontrastını ve doygunluğunu düzenleyerek renkleri standart hale getirir. Gürültü azaltma ise filtreleme gibi tekniklerle görüntülerdeki gürültüyü azaltır.

Ek ön işleme teknikleri arasında kenar algılama, keskinleştirme, histogram eşitleme ve veri artırma bulunur. Kenar algılama, görüntülerdeki kenarları belirginleştirerek modelin nesneleri daha kolay ayırt etmesini sağlar. Keskinleştirme, görüntülerdeki bulanıklığı gidererek netliği artırır. Histogram eşitleme, görüntülerin kontrastı ve parlaklığını histogram analizi kullanarak ayarlar. Veri artırma ise yapay zeka teknikleri kullanarak veri kümesini yapay olarak genişletir.

2.2.4. Mobil Uygulama-Google Cloud Flask API İletişimi

Google Cloud Platform'da kullanılan bazı algoritmalar, uygulamaların performansını optimize etmek, güvenliği sağlamak ve ölçeklenebilirlik sunmak için kritik roller oynar. Google Cloud Load Balancing, sunucu yükünü dengeleyerek ve otomatik ölçeklendirme sağlayarak hizmetlerin kesintisiz ve verimli çalışmasını sağlar. Google Cloud CDN, içerik dağıtım ağı olarak, kullanıcıya en yakın sunuculardan içerik sunarak gecikmeleri minimize eder ve yükleme sürelerini hızlandırır. Google Cloud Armor, DDoS saldırılarına karşı koruma sağlayarak uygulamaların güvenliğini artırır. Google Cloud Trace, istekleri izlemek ve hataları ayıklamak için kullanılır, böylece performans sorunlarının kaynağını hızlıca tespit eder. Google Cloud Monitoring, sistem performansını izleyerek potansiyel sorunları proaktif bir şekilde yönetir ve müdahale eder.

Mobil uygulamalarda ise belirli algoritmalar, veri güvenliği ve etkin veri aktarımı için kullanılır. HTTPS, veri aktarımını güvenli hale getirmek için şifreleme sağlar, böylece kullanıcı verilerinin gizliliği korunur. JSON, verileri kodlamak ve aktarmak için hafif ve verimli bir format olarak kullanılır, veri değişimi sırasında ağ trafiğini minimize eder. REST API, veriye erişim ve yönetim için standart bir yöntem sunarak uygulamalar arasında veri alışverişini kolaylaştırır ve sistemler arasında uyumluluk sağlar. Bu algoritmalar ve teknolojiler, mobil uygulamalar ve Google Cloud arasında güvenli, hızlı ve güvenilir bir iletişim kurulmasına olanak tanır.

2.3. Sonuç

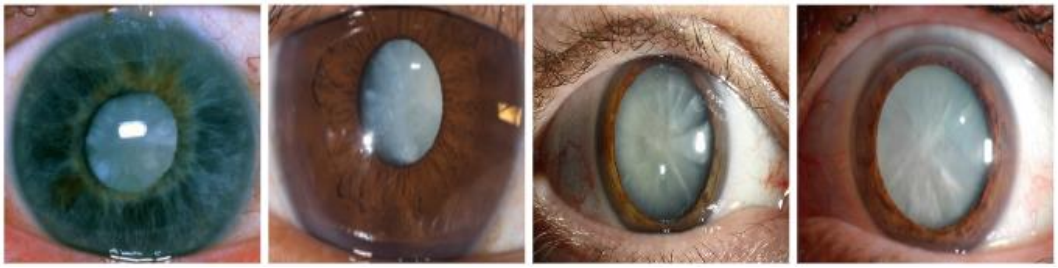
Bu proje, yapay zeka ve derin öğrenme tekniklerini kullanarak göz hastalıklarının teşhisine yardımcı olabilecek bir mobil uygulama geliştirmeyi amaçlamaktadır. Haar Cascade sınıflandırıcısı yüzleri tespit etmek için kullanılırken, Keras kütüphanesi katarakt ve üveit modellerini oluşturmak için kullanılır. Görüntü ön işleme teknikleri, modelin daha yüksek doğruluk elde etmesine yardımcı olmak için kullanılır. Mobil uygulama ve Google Cloud Flask API arasındaki iletişim, güvenli ve verimli bir şekilde sağlanmak için HTTPS, JSON ve REST API gibi algoritmalar ve teknolojiler kullanılır.

3. KULLANILAN ARAÇ VE YÖNTEM

3.1. Veri Seti

3.1.1. Katarakt Veri Seti

Bu projede, katarakt teşhisinde kullanılacak yapay zeka modellerinin eğitimi için Kaggle platformundan elde edilen Cataract dataset [22] ve Cataract [23] veri setleri kullanılmıştır.



Şekil 2: Veri seti katarakt örnekleri.

Şekil 3: Veri seti normal örnekleri.

3.1.2. Üveit Veri Seti

Bu projede, üveit teşhisinde kullanılacak yapay zeka modellerinin eğitimi için Kaggle platformundan elde edilen Eye Disease Dataset [24] ile Roboflow platformundan elde edilen red_eye [25], eye uveitis [26], Uveitis [27] ve Human_pink_eyes_virus [28] veri setleri kullanılmıştır.

Şekil 4: Veri seti üveit örnekleri.

Şekil 5: Veri seti normal örnekleri

3.2. Veri Ön İşleme

Bu aşamada, bulanık, eksik veya hatalı formatta olan resimler tespit edilerek veri setlerinden çıkarılmıştır.

3.2.1. Katarakt Veri Ön İşleme

Düzenleme işlemi sonucunda, başlangıçta 1284 olan resim sayısı 1172'ye düşürülmüştür. Eğitim ve test aşamaları için ayrılan resim sayıları Tablo 1'de belirtilmiştir.

Tablo 1: Katarakt eğitim&test veri setleri.

Eğitim (Katarakt)	Eğitim (Normal)	Test (Katarakt)	Test (Normal)
410	409	177	176
819		353	

3.2.2. Üveit Veri Ön İşleme

Düzenleme işlemi sonucunda, başlangıçta 1598 olan resim sayısı 1428'e düşürülmüştür. Eğitim ve test aşamaları için ayrılan resim sayıları Tablo 2'de belirtilmiştir.

Tablo 2: Üveit eğitim&test veri setleri.

Eğitim (Üveit)	Eğitim (Normal)	Test (Üveit)	Test (Normal)
500	500	214	214
1000		428	

3.3. Katarakt/Üveit Veri Artırma

Veri setlerinin boyutunu ve kalitesini yapay olarak artırmak için çeşitli veri artırma yöntemleri kullanılmıştır. Bu süreç, aşırı uyum (overfitting) sorunlarını çözmeye yardımcı olur ve modellerin eğitim sırasında genelleme yeteneğini artırır. Görüntü artırma işleminde kullanılan ayarlara Tablo 3'de yer verilmiştir.

Rescale işlemi, artırma sürecinde görüntüyü küçültme veya büyütme işlemidir. Rotasyon aralığı, eğitim sırasında rastgele döndürülen görüntülerin derece cinsinden aralığını belirtir, yani 40 derecedir. Width shift, görüntülerin yatay yönde %0.2 oranında kaydırılmasıdır. Height shift, görüntülerin dikey yönde %0.2 oranında kaydırılmasıdır. Ek olarak, 0.2 oranında bir shear range, görüntü açılarını saat yönünün tersine doğru kırpar. Zoom range, görüntüleri %0.2 oranında rastgele yakınlaştırır. Son olarak, görüntüler yatay olarak çevrilir.

Tablo 3: Veri artırma ayarları.

Metot	Ayar
Rescale	1/255
Rotation range	40
Width shift	0.2
Height shift	0.2
Shear range	0.2
Zoom range	0.2
Horizontal flip	True

3.4. Modeller

3.4.1. Katarakt Modeli

Katarakt modeli, ikili görüntü sınıflandırma için tasarlanmış bir derin öğrenme modelidir. Giriş katmanı 224x224 boyutunda ve 3 kanallı (RGB) görüntüler alır. Ardından, sırasıyla 64 filtreli üç evrişim katmanı ve bunları takip eden normalleştirme ve ReLU aktivasyon katmanları gelir. Ortalama havuzlama işlemi ve ardından düzleştirme katmanıyla çıktı tek boyutlu bir vektöre dönüştürülür. Son olarak, 2 sınıf için softmax aktivasyonuyla tam bağlantılı bir katman eklenir.

Model özetine Şekil 6'da yer verilmiştir.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 64)      9408

 batch_normalization (Batch  (None, 224, 224, 64)      256
 Normalization)

 re_lu (ReLU)                (None, 224, 224, 64)      0

 conv2d_1 (Conv2D)           (None, 224, 224, 64)      102400

 batch_normalization_1 (Bat  (None, 224, 224, 64)      256
 chNormalization)

 re_lu_1 (ReLU)              (None, 224, 224, 64)      0

 conv2d_2 (Conv2D)           (None, 224, 224, 64)      36864

 batch_normalization_2 (Bat  (None, 224, 224, 64)      256
 chNormalization)

 re_lu_2 (ReLU)              (None, 224, 224, 64)      0

 average_pooling2d (Average  (None, 112, 112, 64)      0
 Pooling2D)

 flatten (Flatten)           (None, 802816)            0

 dense (Dense)               (None, 2)                 1605634

=================================================================
Total params: 1755074 (6.70 MB)
Trainable params: 1754690 (6.69 MB)
Non-trainable params: 384 (1.50 KB)
```

Şekil 6: Katarakt modeli özeti.

3.4.2. Üveit Modeli

Üveit modeli, ikili görüntü sınıflandırma için tasarlanmış bir derin öğrenme modelidir. Modelin giriş katmanı, 224x224 boyutunda ve 3 kanallı (RGB) görüntüleri alacak şekilde yapılandırılmıştır. İlk olarak, 32 filtreli ve 3x3 boyutunda iki evrişim katmanı (Conv2D) ReLU aktivasyon fonksiyonu ile eklenmiştir. Bunu, çıktı boyutunu yarıya indiren 2x2 boyutunda bir maksimum havuzlama (MaxPooling2D) katmanı takip eder ve aşırı uyumlamayı (overfitting) önlemek için %30 oranında bir dropout katmanı eklenmiştir.

Daha sonra, 64 filtreli ve 3x3 boyutunda iki evrişim katmanı ReLU aktivasyon fonksiyonu ile eklenir. Bu katmanları 2x2 boyutunda bir maksimum havuzlama katmanı ve %30 oranında bir dropout katmanı takip eder. Evrişim ve havuzlama işlemlerinden sonra veriler düzleştirilerek (Flatten) tek boyutlu bir vektöre dönüştürülür.

Son olarak, 16 nöronlu ve ReLU aktivasyon fonksiyonlu bir tam bağlantılı (Dense) katman eklenir ve aşırı uyumlamayı önlemek için bir başka %30 oranında dropout katmanı kullanılır. Çıktı katmanı, 2 sınıfı temsil eden softmax aktivasyon fonksiyonlu bir tam bağlantılı katmandır.

Model özetine Şekil 7'de yer verilmiştir.

3.5. Kullanılan Algoritmalar

3.5.1. Face Detection

face_detection fonksiyonu, Haar Cascade sınıflandırıcısını kullanarak yüzleri ve gözleri algılar. Algoritma öncelikle gri tonlamalı bir görüntü oluşturur ve ardından iki farklı Haar Cascade sınıflandırıcısını kullanarak yüzleri ve gözleri tarar.
Yüz Algılama: haarcascade_frontalface_default.xml dosyasında bulunan yüz sınıflandırıcısı kullanılır. Bu sınıflandırıcı, görüntülerdeki yüzleri farklı ölçek ve yönlerde tespit edebilir.
Göz Algılama: haarcascade_eye_tree_eyeglasses.xml dosyasında bulunan göz sınıflandırıcısı kullanılır. Bu sınıflandırıcı, yüzlerdeki gözleri tespit edebilir.

Algoritma, bir yüz tespit ettikten sonra, o yüz bölgesindeki gözleri de tarar. Her göz için algoritma, gözün parlaklığını kontrol eder ve yeterince parlak değilse göz algılamayı başarısız sayar. Son olarak, algoritma algılanan her göz için bir görüntü dosyası oluşturur ve bu dosyayı bir bulut depolama alanına yükler.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 222, 222, 32)      896

 conv2d_1 (Conv2D)           (None, 220, 220, 32)      9248

 max_pooling2d (MaxPooling2  (None, 110, 110, 32)      0
 D)

 dropout (Dropout)           (None, 110, 110, 32)      0

 conv2d_2 (Conv2D)           (None, 108, 108, 64)      18496

 conv2d_3 (Conv2D)           (None, 106, 106, 64)      36928

 max_pooling2d_1 (MaxPoolin  (None, 53, 53, 64)        0
 g2D)

 dropout_1 (Dropout)         (None, 53, 53, 64)        0

 flatten (Flatten)           (None, 179776)            0

 dense (Dense)               (None, 32)                5752864

 dropout_2 (Dropout)         (None, 32)                0

 dense_1 (Dense)             (None, 2)                 66

=================================================================
Total params: 5818498 (22.20 MB)
Trainable params: 5818498 (22.20 MB)
Non-trainable params: 0 (0.00 Byte)
```

Şekil 7: Üveit modeli özeti.

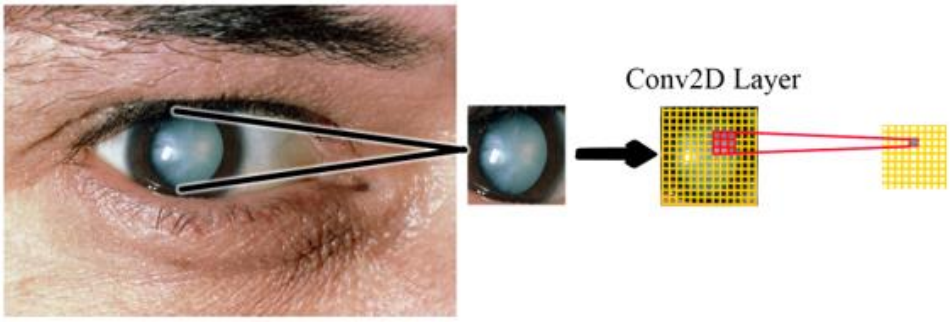
face_detection fonksiyonu, aşağıdaki dönüş değerlerini döndürür:
0: Görüntüde yüz bulunamadı.
1: Görüntüde birden fazla yüz bulundu.
2: Görüntüde yüz bulundu, ancak göz bulunamadı.
3: Görüntüde yüz ve bir göz bulundu.
4: Görüntüde yüz ve birden fazla göz bulundu, ancak en az bir göz yeterince parlak değil.
5: Görüntüde yüz ve iki göz bulundu ve her iki göz de yeterince parlak.

3.5.2. Katarakt/Üveit Modellerinin Oluşturulması

Bu proje kapsamında geliştirilen katarakt ve üveit modelleri; InputLayer, Conv2D, BatchNormalization, ReLU, AveragePooling2D, Flatten, Dense, MaxPooling2D ve Dropout katmanlarından oluşmaktadır.

InputLayer katmanı, genellikle görüntülerde boyut ve kanal sayısını belirtir; örneğin, 224x224 piksel boyutunda ve 3 kanallı bir RGB görüntü beklenir.

Conv2D katmanları, belirli sayıda filtre kullanarak görüntü üzerinde evrişim işlemlerini gerçekleştirirler. Örneğin, ilk Conv2D katmanı genellikle 64 filtre kullanır ve 7x7 boyutunda evrişim çekirdekleri ile işlem yapar. Conv2D katmanları, görüntüdeki özellikleri belirlemek için kullanılır.



Şekil 8: Conv2D katmanı.

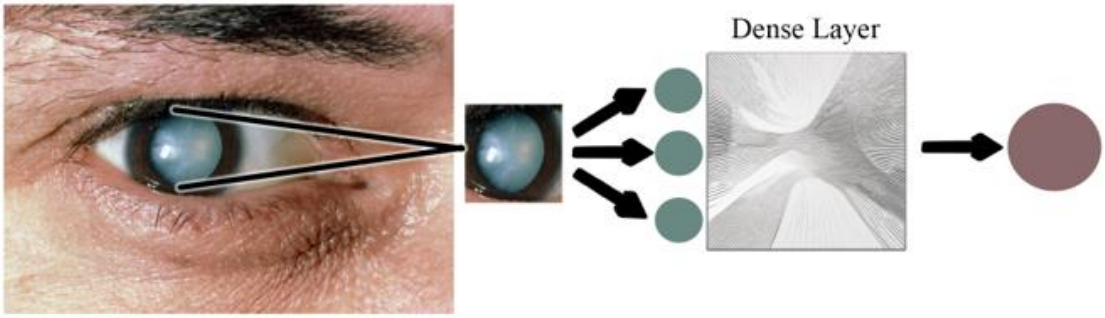
BatchNormalization katmanı, gradyanların daha stabil olmasını sağlar ve ağın daha hızlı ve güvenilir bir şekilde öğrenmesine yardımcı olur.

Aktivasyon fonksiyonlarından biri olan ReLU (Rectified Linear Unit), doğrusal olmayan bir işlevdir. Negatif değerleri sıfıra eşitlerken, pozitif değerleri aynen geçirir. Bu, modelin öğrenme yeteneğini artırır ve aşırı uyum riskini azaltır.

AveragePooling2D katmanı, görüntü boyutunu küçültmek ve özellikleri özetlemek için kullanılır; örneğin, 2x2 boyutundaki bir havuzlama işlemi uygulanarak görüntü boyutu yarıya indirilebilir.

Flatten katmanı, 2B veya 3B yapıdaki verileri tek boyutlu bir vektöre dönüştürerek Dense katmanına geçişi sağlar.

Dense katmanı, sinir ağındaki tüm girdi birimlerini her bir çıktı birimiyle bağlar. Örneğin, 2 sınıflı bir sınıflandırma modelinde 2 çıktı birimi ve softmax aktivasyon fonksiyonu kullanılabilir.



Şekil 9: Dense katmanı.

MaxPooling2D katmanı, özellik haritalarını küçültmek ve önemli özellikleri vurgulamak için kullanılır; örneğin, en büyük değeri alarak görüntüyü küçültmek için 2x2 boyutundaki bir pencere kullanılabilir.

Dropout katmanı, aşırı uyumu azaltmak için kullanılır; burada, belirli bir yüzdeyi (örneğin, %30) rastgele bağlantıları kapatır, böylece modelin genelleme yeteneğini artırır ve aşırı uyumu önler.

3.5.3. Image Preprocessing

preprocess_image fonksiyonunda kullanılan bazı ön işleme algoritmaları şunlardır:
Yeniden Boyutlandırma: cv2.resize fonksiyonu kullanılarak görüntüler istenilen boyuta (224x224 piksel) ölçeklendirilir. Bu işlem, farklı boyutlardaki görüntüleri standart hale getirir ve modelin daha az bellek kullanmasını sağlar.
Normalleştirme: Görüntü piksellerinin değerleri 0 ile 1 arasında bir aralığa dönüştürülür. Bu işlem, farklı aydınlatma koşullarında çekilmiş görüntülerin renkleri arasındaki tutarsızlıkları ortadan kaldırır.
Genişletme: np.expand_dims fonksiyonu kullanılarak görüntülere bir boyut eklenir. Bu işlem, tek bir görüntünün birden fazla örnek olarak yorumlanmasını sağlar.

3.5.4. Mobil Uygulama-Google Cloud Flask API İletişimi

Mobil uygulama ile Google Cloud üzerinde çalışan Flask tabanlı API arasındaki iletişimin teknik detayları ele alınacaktır. Bu iletişim, kullanıcının mobil uygulama üzerinden çektiği göz görüntülerinin işlenerek katarakt tespiti için yapay zeka modeline iletilmesi ve sonuçların geri döndürülmesi sürecini kapsamaktadır.

Mobil uygulama ile Google Cloud arasında veri alışverişi HTTP protokolü üzerinden RESTful API kullanılarak gerçekleştirilir. API, Flask web framework'ü kullanılarak

geliştirilmiş olup, Google Cloud Platform (GCP) üzerinde uygulanmıştır. API'ın işlevleri arasında resim yükleme, resim işleme ve sonuçları geri döndürme işlemleri yer almaktadır.

API geliştirilirken Flask kullanılmıştır. Flask, Python programlama dili ile yazılmış, microframework olarak tanımlanan bir web uygulama çatısıdır.

1. Flask Kurulumu ve Yapılandırması: Flask kütüphanesi, ilgili bağımlılıklar ile birlikte kurulmuştur.
2. API Endpoint'lerinin Tanımlanması:
    - /cataract: POST metodu ile katarakt testi için resim dosyasının yüklenmesini ve işlenmesini sağlar.
    - /uveit: POST metodu ile üveit testi için resim dosyasının yüklenmesini ve işlenmesini sağlar.
3. Model Yükleme ve İşleme Fonksiyonlarının Entegrasyonu: Derin öğrenme modeli TensorFlow/Keras kullanılarak eğitilmiş ve Flask API ile entegre edilmiştir. Model, yüklenen resimleri alır, ön işler ve tahmin sonuçlarını üretir.
4. Google Cloud Platform'da Dağıtım: API, Google App Engine veya Cloud Run üzerinde dağıtılarak ölçeklenebilir ve güvenli bir şekilde erişilebilir hale getirilmiştir. GCP'nin sağladığı avantajlar arasında otomatik ölçeklendirme, yük dengeleme ve güvenlik duvarı yönetimi bulunmaktadır.

Mobil uygulama, API ile iletişim kurarak kullanıcı tarafından yüklenen resimleri API'a göndermelidir. Bu süreç şu adımlardan oluşur:

1. HTTP İsteklerinin Hazırlanması: Mobil uygulama, resim dosyasını ve ilgili kullanıcı bilgilerini bir POST isteği olarak API'a gönderir.
2. İstek Gönderimi ve Yanıt Alma: API, mobil uygulamadan gelen isteği işler, derin öğrenme modeline iletir ve sonucu JSON formatında geri döner.
3. JSON Yanıtının İşlenmesi ve Görüntülenmesi: Mobil uygulama, API'den gelen JSON yanıtını işler ve kullanıcıya uygun bir arayüz ile sonuçları sunar.

Mobil uygulama ile Flask API arasında veri gönderimi ve alımı JSON (JavaScript Object Notation) formatında gerçekleştirilir. JSON, hafif ve insan tarafından okunabilir bir veri değişim formatıdır.

Mobil uygulama ve API arasındaki iletişimin güvenliği, SSL/TLS kullanılarak sağlanır. Bu, veri transferi sırasında iletilen bilgilerin şifrelenmesini ve yetkisiz erişimlerin önlenmesini sağlar. API'a erişim, API anahtarları veya OAuth 2.0 protokolü ile güvence altına alınmıştır. Ayrıca, oluşturulan gizlilik politikaları gereğince kullanıcı verilerinin asla üçüncü taraflarla paylaşılmaması sağlanmıştır.

API'ın performansı ve ölçeklenebilirliği, Google Cloud Platform'un sağladığı altyapı avantajları kullanılarak optimize edilmiştir. Google Cloud Run, API'ın yüküne göre otomatik olarak ölçeklenir, böylece yüksek trafik durumlarında dahi performans kaybı yaşanmaz. Gelen istekler, yük dengeleme algoritmaları ile dağıtılarak API'ın sorunsuz çalışması sağlanır.

Mobil uygulama ile Google Cloud Flask API iletişimi, kullanıcı dostu, güvenli ve ölçeklenebilir bir katarakt ve üveit tespit sistemi oluşturmak amacıyla geliştirilmiştir. Bu iletişim altyapısı, sadece katarakt ve üveit hastalıklarının tespitiyle sınırlı kalmayıp, gelecekte diğer sağlık tespiti uygulamaları için de bir temel oluşturabilir. Örneğin, farklı göz hastalıklarının tespiti, dermatolojik analizler veya diğer medikal görüntü işleme uygulamaları gibi genişletilebilir. Bu esneklik, sistemin uzun vadeli kullanım potansiyelini ve sağlık teknolojileri alanındaki katkısını göstermektedir.

4. SİSTEMİN GERÇEKLENMESİ YA DA BULGULAR

4.1. Performans Değerlendirme Metrikleri

Yapay zeka modellerinin performansını değerlendirmek için kullanılan başlıca performans metrikleri; doğruluk (accuracy), kayıp, Alan Altında Kalmış (Area Under the Curve - AUC), karışıklık matrisi (confusion matrix) ve sınıflandırma raporu (classification report) değerleridir.

Doğruluk (accuracy), modelin doğru tahmin oranını ifade eder. Bu metrik, toplam doğru tahminlerin, tüm tahminlere oranı olarak hesaplanır. Formülü şu şekildedir:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Kayıp (loss), modelin öğrenme sürecinde yaptığı hataların ölçüsüdür. Genellikle, hatanın büyüklüğünü minimize etmek amacıyla optimizasyon algoritmaları tarafından kullanılır.

Alan Altında Kalmış (Area Under the Curve - AUC), ROC eğrisinin altındaki alanı ifade eder. Bu metrik, modelin pozitif ve negatif sınıfları ne kadar iyi ayırt edebildiğini gösterir. AUC değeri 1'e ne kadar yakınsa, modelin performansı o kadar iyidir. Formülü şu şekildedir:

$$AUC = \int_0^1 TPR\, d(FPR)$$

Karışıklık matrisi (confusion matrix), modelin doğru ve yanlış sınıflandırmalarını özetleyen bir tablodur. Bu matris, True Positive (TP), True Negative (TN), False Positive (FP) ve False Negative (FN) değerlerini içerir. Karışıklık matrisi, modelin hangi sınıflarda hata yaptığını ve hangi sınıfları doğru sınıflandırdığını ayrıntılı olarak gösterir.

Sınıflandırma raporu (classification report) ise modelin performansını değerlendiren detaylı bir rapordur. Bu rapor, Precision (kesinlik), Recall (duyarlılık), F1-Score ve Support (destek) gibi metrikleri içerir. Sınıflandırma raporu, her bir sınıf için bu metriklerin değerlerini verir ve modelin hangi sınıflarda daha iyi veya kötü performans gösterdiğini anlamaya yardımcı olur. Bu sayede, modelin genel performansı hakkında kapsamlı bir bilgi edinilir. Formüller şu şekildedir:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

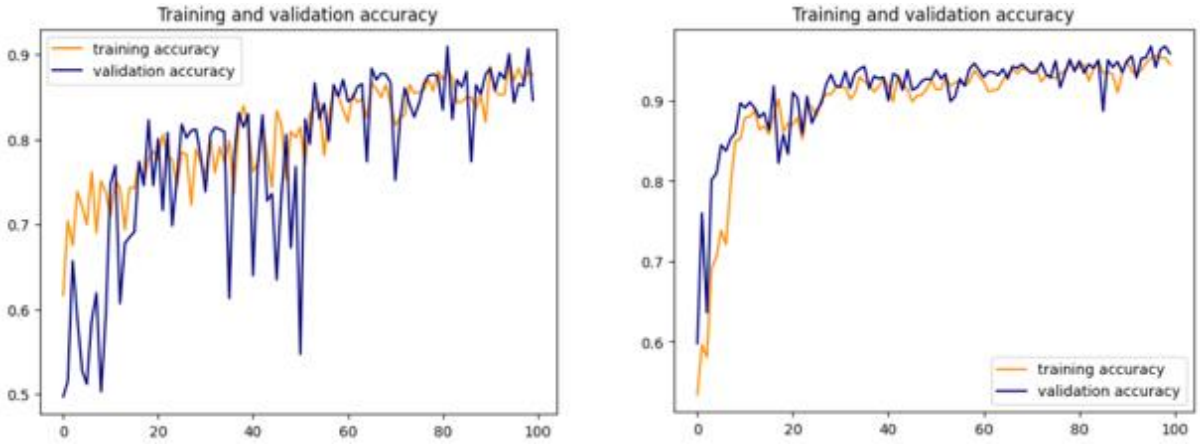
$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4.2. Performans Değerlendirme Sonuçları

4.2.1. Eğitim, Validasyon ve Test Doğruluğu

Katarakt modelinin eğitim aşaması sonucunda elde edilen sonuçlara (Şekil 10) göre, training accuracy değeri %87.67 ve validation accuracy değeri %84.62 olarak belirlenmiştir. Modelin test veri seti üzerinde elde ettiği doğruluk değeri ise %82.72 olarak belirlenmiştir.

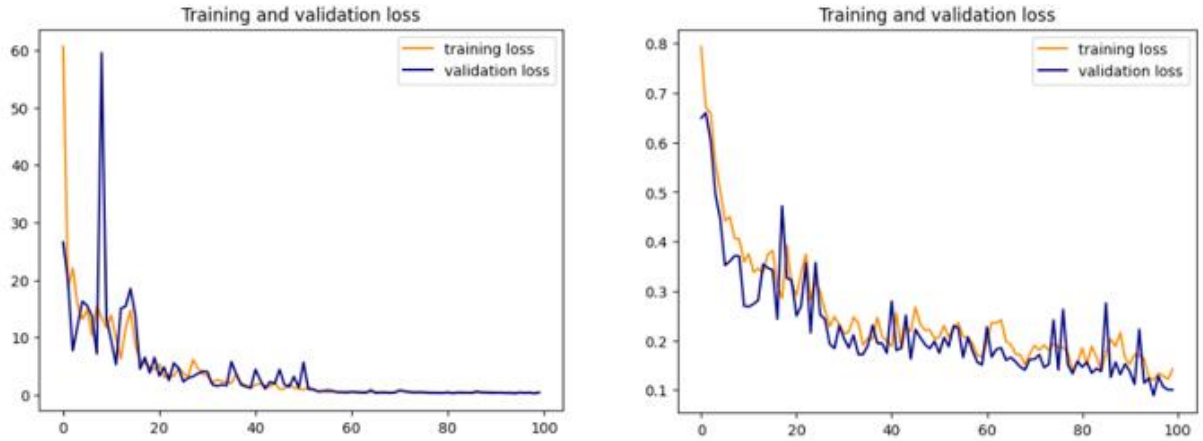
Üveit modelinin eğitim aşaması sonucunda elde edilen sonuçlara (Şekil 10) göre, training accuracy değeri %94.52 ve validation accuracy değeri %95.87 olarak belirlenmiştir. Modelin test veri seti üzerinde elde ettiği doğruluk değeri ise %94.86 olarak belirlenmiştir.



Şekil 10: Katarakt ve üveit modellerinin sırası ile eğitim&validasyon doğrulukları.

4.2.2. Eğitim ve Validasyon Kayıpları

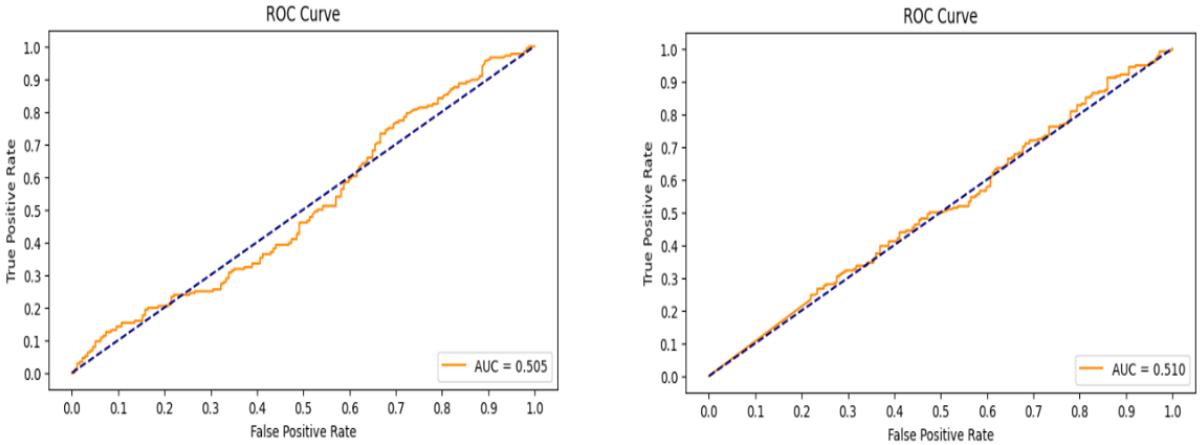
Katarakt ve üveit modellerinin eğitim aşamaları sonucunda elde edilen eğitim ve validasyon kaybı sonuçlarına sırasıyla Şekil 11'de yer verilmiştir.



Şekil 11: Katarakt ve üveit modellerinin sırası ile eğitim&validasyon kayıpları.

4.2.3. ROC Eğrisi ve AUC Değeri

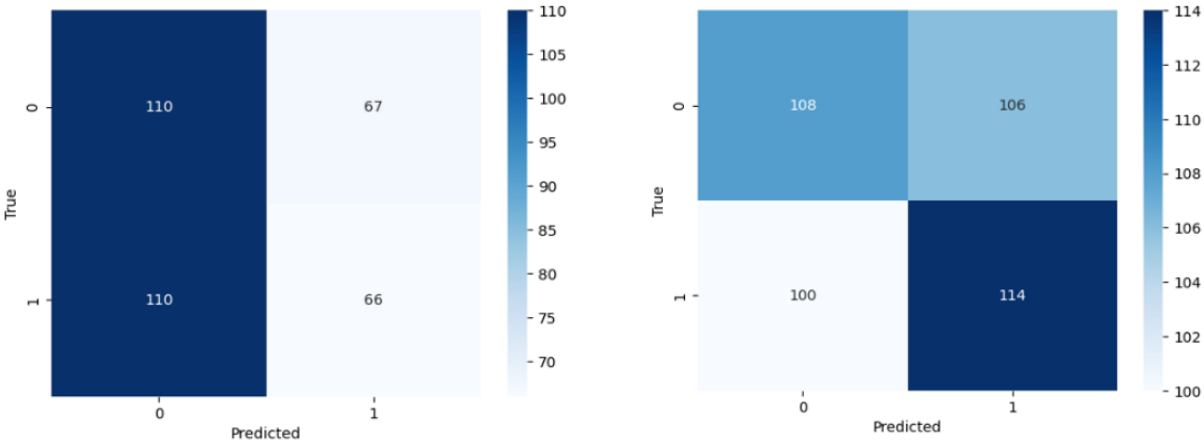
Katarakt ve üveit modellerinin test veri setleri üzerinde elde ettiği ROC eğrisi ve AUC değerlerine sırasıyla Şekil 12'de yer verilmiştir.



Şekil 12: Katarakt ve üveit modellerinin sırası ile ROC eğrisi ve AUC değerleri.

4.2.4. Karışıklık Matrisi

Katarakt ve üveit modellerinin test veri setleri üzerinde elde ettiği karışıklık matrislerine sırasıyla Şekil 13'te yer verilmiştir.



Şekil 13: Katarakt ve üveit modellerinin sırası ile karışıklık matrisleri.

4.2.5. Sınıflandırma Raporu

Katarakt ve üveit modellerinin test veri setleri üzerinde elde ettiği sınıflandırma raporlarına sırasıyla Şekil 14'te yer verilmiştir.

Classification Report:

	precision	recall	f1-score	support
0	0.50	0.62	0.55	177
1	0.50	0.38	0.43	176
accuracy			0.50	353
macro avg	0.50	0.50	0.49	353
weighted avg	0.50	0.50	0.49	353

Classification Report:

	precision	recall	f1-score	support
0	0.52	0.50	0.51	214
1	0.52	0.53	0.53	214
accuracy			0.52	428
macro avg	0.52	0.52	0.52	428
weighted avg	0.52	0.52	0.52	428

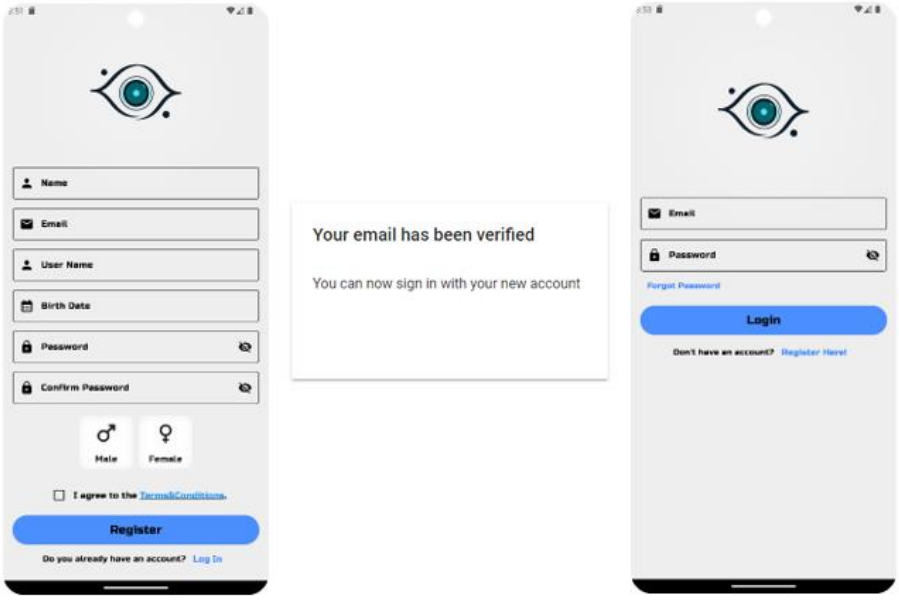
Şekil 14: Katarakt ve üveit modellerinin sırası ile sınıflandırma raporları.

4.3. Kullanım Senaryosu

4.3.1. Kullanıcı Kayıt ve Giriş:

Kayıt: İlk defa uygulamaya giren kullanıcılar, kayıt işlemini gerçekleştirmek zorundadır. Bu işlem için kullanıcı, e-posta adresini girer ve ardından e-posta adresine gönderilen doğrulama bağlantısına tıklayarak hesabını aktif hale getirir.

Giriş: Kayıt işlemi tamamlanan kullanıcılar, e-posta adresleri ve şifreleri ile giriş ekranından uygulamaya erişim sağlayabilir.



Şekil 15: Kullanıcı kayıt ve giriş işlemleri.

4.3.2. Ana Sayfa İşlevleri:

Resim Seçimi: Kullanıcılar, ana sayfada galeriden veya kameradan yüz resimlerini seçebilirler.

Gönderme ve Analiz: Seçilen resim "Send and Analyze" butonuna basılarak Firebase Storage'a kaydedilir.

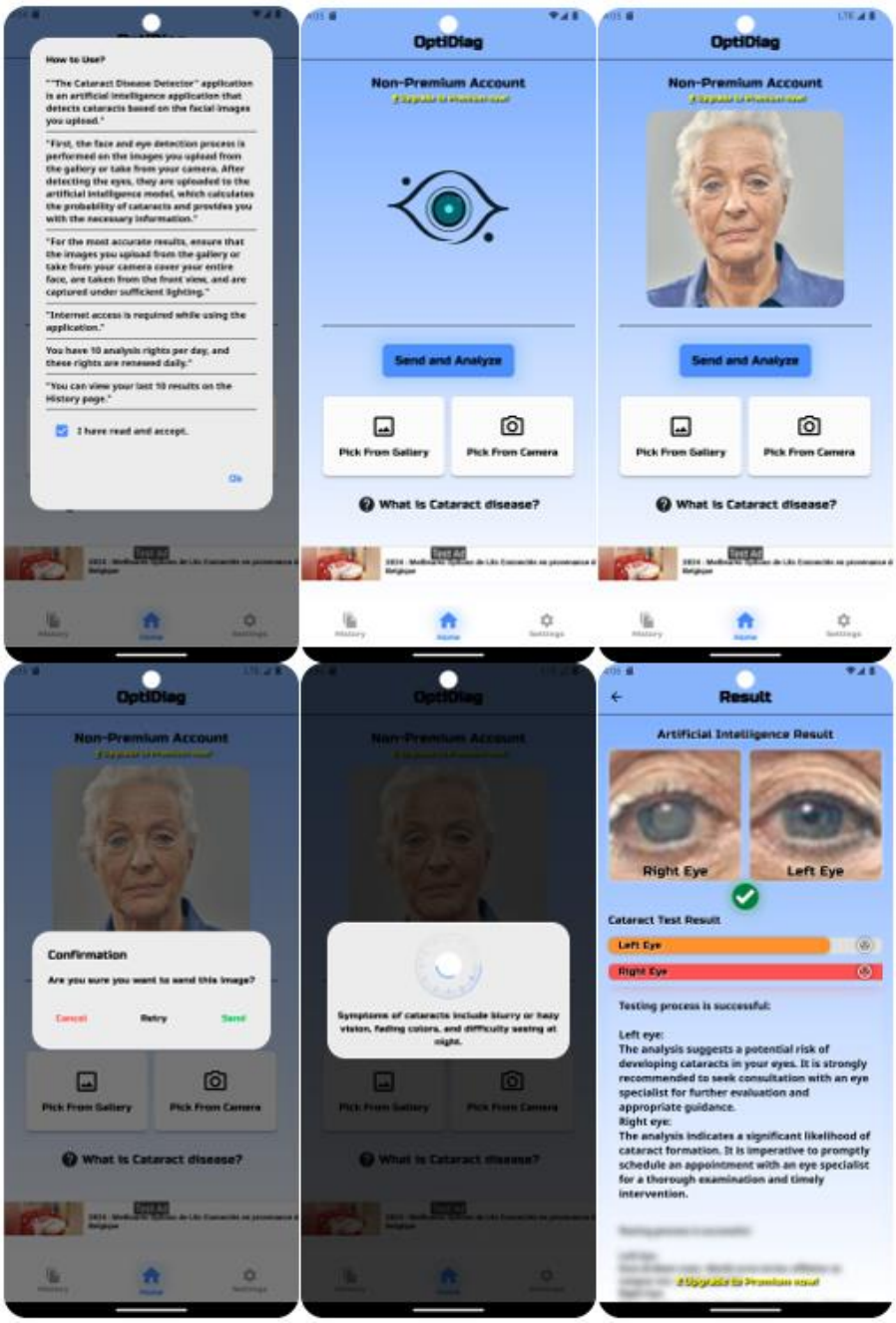
Analiz İşlemi: Firebase Storage'dan alınan resim, Google Cloud üzerinde çalışan Flask API'a gönderilir.

Yüz Tespiti: Flask API, face_detection fonksiyonu ile resimdeki sağ ve sol gözleri tespit eder ve bu görüntüleri yeniden Firebase Storage'a yükler.

Ön İşleme ve Tahmin: Kaydedilen sağ ve sol göz resimleri, predict_image fonksiyonu tarafından geçici bir dosyaya yüklenir ve preprocess_image fonksiyonu ile ön işleme tabi tutulur. İşlemler tamamlandıktan sonra resimler CNN modeline beslenir ve tahmin sonuçları elde edilir.

Karar Oluşturma: make_cataract_decision fonksiyonu, elde edilen sayısal tahmin sonuçlarına göre katarakt ve üveit hastalığına dair bir karar oluşturur.

Sonuçların Kaydedilmesi: Karar ve tahminin sayısal değerleri Firebase Firestore Database'e kaydedilir.



Şekil 16: Resim seçimi, gönderimi, analizi ve sonucun eldesi.

4.3.3. Sonuçlara Erişim:

Tahmin işlemleri sonucunda alınan kararlar, mobil uygulama tarafından erişilebilir durumda olur. Kullanıcılar, ana sayfadaki "History" sekmesine giderek bu bilgilere ulaşabilir ve katarakt veya üveit hastalığına dair risklerini öğrenebilirler.

5. TARTIŞMA VE SONUÇ

Bu çalışmada, katarakt ve üveit hastalıklarının erken teşhisine yardımcı olabilecek bir mobil uygulama geliştirilmiştir. Uygulama, kullanıcıların kendi cihazlarında yüz resimlerini çekerek veya galeriden yükleyerek göz resimlerini otomatik olarak çıkarır ve bu göz resimlerini yapay zeka modelleri ile analiz ederek hastalık riskini tahmin eder.

Uygulamanın başarıları arasında yüksek doğruluk önemli bir yer tutmaktadır. Elde edilen sonuçlar, önerilen derin öğrenme modellerinin eğitim ve test veri setlerinde yüksek doğruluk oranları elde ettiğini göstermektedir. Bu, modellerin katarakt ve üveit hastalıklarını doğru bir şekilde teşhis etme potansiyeline sahip olduğunu gösterir.

Uygulama aynı zamanda kullanıcı dostu bir arayüze sahiptir ve karmaşık işlemler yapmadan katarakt ve üveit riskini değerlendirmeyi kolaylaştırır. Bu kolay kullanım özelliği, kullanıcıların uygulamayı etkin bir şekilde kullanmasını sağlar ve erken teşhis sürecini hızlandırır.

Uygulama, hastalıkların erken evrelerde teşhis edilmesine yardımcı olarak tedavi şansını artırır ve kullanıcıları sağlık konusunda daha bilinçli hale getirir.

Uygulamanın sağladığı uzaktan erişim imkanı önemli bir avantajdır. Tıp uzmanlarına erişimi olmayan kişilere katarakt ve üveit teşhisi imkanı sunarak, sağlık hizmetlerine daha geniş bir erişim sağlar.

Bilinçlendirme de uygulamanın önemli bir etkisidir. Kullanıcıların katarakt ve üveit hastalıkları hakkında bilgi edinmelerine ve erken teşhisin önemini anlamalarına yardımcı olur ve daha sağlıklı bir toplum oluşturmaya katkı sağlar.

Geliştirilmesi gereken alanlar arasında daha fazla veri kullanımı, veri artırma tekniklerinin kullanılması, farklı derin öğrenme mimarilerinin test edilmesi, klinik uygulamaların yapılması, uygulamanın farklı mobil platformlara uyarlama ve kullanıcı dostu arayüzler geliştirme yer alır.

Sonuç olarak, bu çalışma katarakt ve üveit hastalıklarının erken teşhisinde önemli bir rol oynayabilecek bir mobil uygulama sunmaktadır. Gelecekteki geliştirmeler ile birlikte uygulama daha da etkili hale gelerek halk sağlığına önemli katkılar sunabilir. Uygulamanın kolay kullanımı ve uzaktan erişim imkanı sunması, özellikle dezavantajlı gruplara erişimi kolaylaştırabilir ve erken teşhis oranlarını artırabilir.

KAYNAKLAR

[1] D. Allen and A. Vasavada "Cataract and surgery for cataract." BMJ, vol. 333, no. 7559, pp. 128–132, 2006.
[2] Y.C. Liu, M. Wilkins, T. Kim, B. Malyugin, and J.S. Mehta "Cataracts." Lancet, vol. 390, no. 10094, pp. 600–612, 2017.
[3] C.M. Lee and N.A. Afshari "The global state of cataract blindness." Curr. Opin. Ophthalmol, vol. 28, no. 1, pp. 98–103, 2017.
[4] M. Khairallah, R. Kahloun, R. Bourne, H. Limburg, S.R. Flaxman, J.B. Jonas, J. Keeffe, J. Leasher, K. Naidoo, K. Pesudovs, H. Price, R.A. White, T.Y. Wong, S. Resnikoff, and H.R. Taylor "Number of People Blind or Visually Impaired by Cataract Worldwide and in World Regions, 1990 to 2010" Invest. Ophthalmol. Vis. Sci. vol. 56, no. 11, pp. 6762–6769, 2015.
[5] Jabs, D. A., Nussenblatt, R. B., & Rosenbaum, J. T. (2005). Standardization of uveitis nomenclature for reporting clinical data. Results of the First International Workshop. American Journal of Ophthalmology, 140(3), 509-516.
[6] Smith, J. R., & Rosenbaum, J. T. (2002). Management of uveitis: a rheumatologic perspective. Arthritis & Rheumatism: Official Journal of the American College of Rheumatology, 46(2), 309-318.
[7] Simonite, T. (2017). Google Buys Data Science Community Kaggle in Bid to Bolster AI Arm. Wired. Erişim adresi: https://www.wired.com/2017/03/google-buys-data-science-community-kaggle-bid-bolster-ai-arm/
[8] Team, K. (2020). What is Kaggle? Kaggle. Erişim adresi: https://www.kaggle.com/docs
[9] Kashyap, V. (2018). Kaggle: Your Machine Learning and Data Science Community. Towards Data Science. Erişim adresi: https://towardsdatascience.com/kaggle-your-machine-learning-and-data-science-community-a8a745c5566
[10] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45(4), 427-437.
[11] Tharwat, A. (2018). Classification assessment methods. Applied Computing and Informatics.
[12] Powers, D. M. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies, 2(1), 37-63.
[13] "What is Google Cloud?" Google Cloud. Erişim adresi: https://cloud.google.com/what-is-cloud
[14] "Google Cloud Platform." Wikipedia, The Free Encyclopedia. Erişim adresi: https://en.wikipedia.org/wiki/Google_Cloud_Platform
[15] "Introduction to Google Cloud." Google Cloud Documentation. Erişim adresi: https://cloud.google.com/docs/overview
[16] Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
[17] "What is Flask?" Pallets Projects. Erişim adresi: https://flask.palletsprojects.com/en/2.0.x/
[18] "What is Firebase?" Firebase. Erişim adresi: https://firebase.google.com/docs/overview
[19] https://docs.opencv.org/4.x/d2/d99/tutorial_js_face_detection.html
[20] I. Cruz-Vega, H. I. Morales-Lopez, J. M. Ramirez-Cortes and J. De Jesus Rangel-Magdaleno, "Nuclear Cataract Database for Biomedical and Machine Learning Applications," in IEEE Access, vol. 11, pp. 107754-107766, 2023, doi: 10.1109/ACCESS.2023.3312616.
[21] https://opg.optica.org/directpdfaccess/116fc03e-eb7c-4442-a60076a928313bf4_531805/boe-14-7-3413.pdf?da=1&id=531805&seq=0&mobile=no
[22] https://www.kaggle.com/datasets/nandanp6/cataract-image-dataset/data

[23] https://www.kaggle.com/datasets/kershrita/cataract/data
[24] https://www.kaggle.com/datasets/tejpal123/eye-disease-dataset
[25] https://universe.roboflow.com/nguyen-thi-anh-0mljl/red_eye/dataset/1
[26] https://universe.roboflow.com/sona-college-of-technology-pbo8a/eye-uveitis/dataset/1
[27] https://universe.roboflow.com/uni-xhvbo/uveitis/dataset/1
[28] https://universe.roboflow.com/student-gjyci/human_pink_eyes_virus/dataset/1

EKLER

ÖZGEÇMİŞ

Ensar Aydın Kurubacak

2001 yılında İstanbul'un Bağcılar ilçesinde doğdu. İlköğrenimine Kısıklı İlköğretim Okulu'nda başladı ve lise eğitimini Beyoğlu Anadolu İmam Hatip Lisesi'nde tamamladı. Şu anda İstanbul Üniversitesi-Cerrahpaşa'da Bilgisayar Mühendisliği bölümünde eğitimine devam etmekte olup, 4. sınıf öğrencisidir.

Fatih Ateş

2001 İstanbul Fatih doğumlu. İlkokulu, ortaokulu ve liseyi Doğa Okullarında okudu. Şu anda İstanbul Üniversitesi-Cerrahpaşa'da Bilgisayar Mühendisliği bölümünde eğitimine devam etmekte olup, 4. sınıf öğrencisidir.