T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DERİN ÖĞRENME MODELLERİNİN HÜCRE VERİ SETİ ÜZERİNDE EĞİTİLEREK KIYASLANMASI VE MOBİL ORTAMA UYARLANMASI

YÜKSEK LİSANS TEZİ

Mehmet YAVUZ

Enstitü Anabilim Dalı : BİYOMEDİKAL MÜHENDİSLİĞİ

Tez Danışmanı : Doç. Dr. Mustafa Zahid YILDIZ

Eylül 2020

T.C.
SAKARYA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DERİN ÖĞRENME MODELLERİNİN HÜCRE VERİ SETİ ÜZERİNDE EĞİTİLEREK KIYASLANMASI VE MOBİL ORTAMA UYARLANMASI

YÜKSEK LİSANS TEZİ

Mehmet YAVUZ

Enstitü Anabilim Dalı : BİYOMEDİKAL MÜHENDİSLİĞİ

Bu tez 02/09/2020 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

JÜRİ	BAŞARI DURUMU
Jüri Başkanı: Doç. Dr. Mustafa Zahid YILDIZ	BAŞARILI
Üye: Dr. Öğr. Üyesi Süleyman UZUN	BAŞARILI
Üye: Dr. Öğr. Üyesi Gökçen ÇETİNEL	BAŞARILI

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Mehmet YAVUZ

02 / 09 / 2020

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar bütün aşamalarında yardımlarını esirgemeyen değerli danışman hocam Doç. Dr. Mustafa Zahid YILDIZ′a teşekkürlerimi sunarım.

Değerli fikir ve tecrübelerinden istifade ettiğim pek kıymetli kardeşim Furkan Yusuf YAVUZ'a şükranlarımı sunarım.

Ayrıca çalışmamı nihayete erdirene kadar maddi manevi destekleri ile yardımcı olup beni cesaretlendiren değerli aileme teşekkür ederim.

İÇİNDEKİLER

KISALTMALAR

BAÜ	: Birmingham Alabama Üniversitesi
DDT	: Dikloro Difenil Trikloroethan
DDÜ	: Düzeltilmiş Doğrusal Ünite (Rectified Linear Unit)
DSÖ	: Dünya Sağlık Örgütü
DVM	: Destek Vektör Makinesi (Support Vector Machine)
ESA	: Evrişimli Sinir Ağı (Convolutional Neural Networks)
OHK	: Ortalama Hata Karesi (Mean Squared Error)
SAİ	: Saniyedeki Aritmetik İşlem (Floating Point Operation Per Second)
UTK	: Ulusal Tıp Kütüphanesi (National Library of Medicine)

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KODLAR LİSTESİ

DERİN ÖĞRENME MODELLERİNİN HÜCRE VERİ SETİ ÜZERİNDE EĞİTİLEREK KIYASLANMASI VE MOBİL ORTAMA UYARLANMASI

ÖZET

Derin öğrenme, günümüzde en çok kullanılan yapay zekâ yöntemidir. Bu tekniğin gelişmesi ile birlikte çeşitli derin öğrenme modelleri tasarlanmış ve otomotiv, reklam, bankacılık, sağlık, ziraat gibi birçok farklı alanda akıllı çözümler üretilmiştir. Özellikle sağlık alanında hastalıkların teşhis ve tedavisinde derin öğrenme teknikleri etkili sonuçlar vermektedir. Bu çalışmada da tedavi için erken teşhisin çok önemli olduğu sıtma hastalığının, ileri derin öğrenme yöntemleri ile hızlı tanı konulması, geliştirilen yöntemin kolay, ucuz ve erişilebilir olması hedeflenmiştir. Zîra DSÖ'nün yayınladığı raporlara göre günümüzde hâlâ her yıl yüz binlerce insanın ölüm sebebi sıtma kaynaklıdır.

İleri yapay zekâ tekniklerinden olan derin öğrenme özetle; oluşturulan modellerin binlerce veriden oluşan veri setleri üzerinde eğitilmesi ve eğitilen model ile eğitim veri setinde olmayan benzer ama farklı verilerin saliselerle ölçülebilecek sürelerde ve mobil gibi yüksek işlemci kapasitesi gerektirmeyen ortamlarda sınıflandırılmasını sağlayan matematiksel algoritmalara dayanan bir yöntemdir.

Bu çalışmada güncel derin öğrenme modelleri, yaygın bir kullanıma sahip olan sıtma (hücre) veri seti üzerinde eğitilmiş ve sonuçlarının başarımları karşılaştırılarak değerlendirilmiştir. 27,558 adet kırmızı kan hücresinden oluşan veri seti üzerinde yapılan eğitimlerde yüksek performansa sahip modeller elde edilmiş ve kıyaslamaları yapılmıştır. Eğitimler, son yıllarda Google tarafından geliştirilen EfficientNet modelleriyle öğrenim aktarımı yapılarak gerçekleştirilmiştir.

Çalışma kapsamında, farklı EfficientNet modellerinin birçok kez farklı parametrelerle otomatik eğitilmesi için parametre setlerini kaydederek döngü içinde çalıştıran bir modül geliştirilmiştir. Bu modül, eğitimler neticesinde alınan çıktıların karşılaştırılmasını kolaylaştırmak amacıyla geliştirilen web uygulamasına bağlanarak bütünleşik çalışan kullanışlı bir araca dönüştürülmüştür. Bu araç sayesinde birçok parametreden oluşan farklı parametre setleri bir seferde eğitime sokulabilmiş ve neticeleri kıyaslanarak en yüksek başarımlı setler tespit edilebilmiştir.

Karşılaştırmalar neticesinde daha yüksek performans elde edilen EfficientNet modeli mobil ortamda çalışacak formata dönüştürülmüş ve dönüştürülen bu mobil model geliştirilen bir mobil uygulamaya yerleştirilmiştir. Geliştirilen mobil uygulama ile model eğitilirken hiç görmediği kırmızı kan hücresi görüntülerinin sağlıklı mı yoksa enfekte mi oldukları tahmini yapılmıştır. Bu sayede sıtma teşhisinin hızlı ve düşük maliyetli olmasının yanında kolay erişilebilir olması amaçlanmıştır.

Anahtar Kelimeler: sıtma, teşhis, kırmızı kan hücreleri, veri seti, yapay zekâ, makine öğrenmesi, derin öğrenme, evrişimli sinir ağları, efficientnet, öğrenim aktarımı

COMPARISION AND MOBILE APPLICATION OF DEEP LEARNING MODELS TRAINED ON BLOOD CELL DATASET

SUMMARY

The innovations in the field of hardware have served artificial intelligence studies to spread rapidly. Deep learning is one of the most used artificial intelligence techniques today. With the development of this technique, various deep learning models have been designed and smart solutions have been produced in many different fields such as automotive, advertising, banking, health, agriculture etc. Especially in the field of health, deep learning technique gives promising results in the diagnosis and treatment of diseases.

One of the purpose of this study is to provide rapid, easy, cheap and accessible diagnosis of malaria which is a vital disease with advanced deep learning methods and mobile technologies. Despite a 28% decline in malaria cases and mortality since 2010, there were approximately 219 million malaria cases (between 203 and 262 million to be exact) and 435,000 malaria deaths globally in 2017, according to the World Malaria Report 2018 which was published by the World Health Organization. The most tragic incident in the World Health Organization African Region, where 93% of all malaria deaths occurred, was that 61% of all deaths were children under the age of 5. It is observed that the most common regions of the parasite causing the disease are Africa, South East Asia and the Eastern Mediterranean, respectively. Its annual cost is estimated to be $12 billion.

After a person is bitten by a malaria-carrying mosquito, the symptoms of the disease are not noticed for a week to a month. During this time, the malaria parasites reproduce in the person's liver before they invade the red blood cells in the bloodstream. Once inside the red blood cells, the parasites continue to multiply and spread the infection. Infected red blood cells eventually rupture, and the patient will show flu-like symptoms such as sweating, high fever, chills and nausea. Incorrect treatment methods cause irreversible damages and deaths in patients due to mixing with other diseases. That is why early diagnosis has vital importance for treatment of malaria disease. Because of these factors our study is highly important.

Generally, the diagnosis is made by scanning the laboratory results and passing them through human control. Difficulties such as specialist training and human-induced errors impose limitations on the diagnosis of the disease through human control of laboratory results. Due to these problems, the use of artificial intelligence and image classification techniques in the diagnosis of malaria comes to the fore.

In recent years, significant gains have been achieved in machine vision systems due to the development of camera sensors and the increase in processing power and high performance models have been created in machine vision applications. At the same time, the decrease in the prices of camera-based systems has prepared the environment for the application of artificial vision in a wide variety of fields.

Improvements in fast computers and algorithms have led to advances in machine learning and sensors. The increase in access to large amounts of data and developments in graphic processors made deep learning methods, one of the artificial intelligence techniques, applicable. Deep learning models appear to solve multivariate and complex problems. Besides, some problems arise with large and unlabeled datasets. Deep learning techniques solve this problem by learning relationships through features and hierarchies.

In recent studies, deep learning models have offered better predictive performance than existing solutions in various fields and purposes in image processing, cybersecurity and internet of things, voice recognition. Under the framework of deep learning, Convolutional Neural Networks, an algorithm based on artificial neural networks, have also been found to be superior to traditional learning algorithms such as K-Nearest Neighbors, Multilayer Perceptron and Support Vector Machine in many aspects of image processing.

As an advanced artificial intelligence technique deep learning briefly is a method based on mathematical algorithms. That enables the models to be trained on datasets consisting of huge amount of data and to classify similar but different data that are not in the training dataset. With a well-trained model, classification process takes only milliseconds. Moreover, in order to use trained models does not require high processor capacities. So this feature makes mobile devices are suitable platforms to run trained deep learning models.

Some difficulties encountered in model training for new problems both interrupt the research and increase the investment cost. Processes such as collecting data for each new problem, extracting data, labeling the data and training the new model again take a serious amount of time in the research process. Often for new problems; It is not possible to collect data and create a model. In order to overcome these problems, Transfer Learning technique was used in this study. Transfer Learning is the application of the knowledge (learned points) acquired during or as a result of model training for a problem solving to another related problem.

Transfer Learning has been found based on the human use of previous experiences in new problems. In the light of these experiences, problems are likely to be solved faster or more effectively. For example; A person who can play an instrument can play another instrument more easily than the first, or a person learning a new language can learn another language more easily. The basics of Transfer Learning are discussed in a study on "learning to learn". Transfer Learning, since 1995; It has attracted more and more attention under different names such as "learning to learn", "lifelong learning", "knowledge / experience transfer", "multitasking learning", "knowledge reinforcement", "cumulative learning".

Transfer Learning is often accomplished by reusing previously trained models for similar problems. A pre-trained model is taken, it is edited for the problem to be applied (e.g., the last layer of the model is changed according to the number of classes) and training is initiated. In this way; More effective results are obtained in a shorter time than creating a model from the beginning. Because the model used was previously trained and shaped for millions of images and thousands of classes. Transferring the learned knowledge provides convenience in new problems and in this way has been instrumental in many successful studies in different fields.

In this study, recently developed deep learning models were trained on a very common dataset. Malaria dataset consists 27,558 red blood cell images which was collected in

Bangladesh. Half of the red blood cell images are parasitized and the other half is uninfected. So the dataset is well balanced which means it is a quite suitable dataset in order to deep learning training. The images were manually labelled by an expert slide reader in Thailand.

Convolutional Neural Networks, which are used extensively in image analysis processes, are multi-layer and feed-forward deep learning techniques. EfficientNet is a new convolutional neural network study developed by Google in 2019, provides significant improvements in accuracy and performance. Some prominent features of EfficienNet models are lightness and high performance. These features were effective in choosing EfficientNet models in this study.

Existing Convolutional Neural Networks models usually focus on increasing the number of layers (depth), width or resolution alone to increase accuracy, while EfficientNet uses the compound scaling method, which is a simple but effective method they have developed, to increase the depth, width and resolution dimensions together at a specified rate.

By using EfficientNet models, high accuracy results up to 98% were obtained in the trainings conducted on dataset and comparisons were made. This is the proof of concept that EfficientNet models can successfully deal with diagnosis of malaria disease.

Within the scope of the study, an automatized deep learning tool has been developed that consists the logging module and the visual comparison module. The logging module provides automatically train of different EfficientNet deep learning models with different parameters. The module takes parameter sets and runs them in a loop. In each loop different models are trained with different parameters. At the end of each loop training results are saved on log files. This module has been transformed into a useful deep learning tool by connecting to the visual comparison module. The visual comparison module is a web application that developed in order to facilitate the comparison of input parameters and of the outputs obtained as a result of the training.

According to comparisons of results of trainings, the model which achieved the highest test performance was selected. By using Tensorflow technologies, selected model has been converted to “tflite” file format which can be used in mobile devices to make predictions on red blood cell images which has never been seen by the model during training. And a mobile application was developed with converted model. With the developed mobile application, it is aimed to make malaria diagnosis fast and low cost as well as easy accessible.

This is the first academic study that trains models with transfer learning technique by using EfficientNet models in in order to diagnosis of malaria disease from parasitized red blood cell images. The fact that the model has a performance of 98% in itself is a successful study, and it is a promising study when examined in the context of malaria diagnosis. As a result of these inferences, it is evaluated that modes built on EfficientNet models with transfer learning technique are suitable for use in the field of medicine and will shed light on future studies.

Keywords: malaria, diagnosis, red blood cells, dataset, artificial intelligence, machine learning, deep learning, convolutional neural networks, transfer learning, artificial neural networks, mobile application, model comparison, efficientnet

BÖLÜM 1. GİRİŞ

Sıtma hastalığı, özellikle Afrika ve Güney Doğu Asya'nın fakir ülkelerinde her yıl on binlerce ölüme sebep olmaktadır. Genellikle laboratuvar sonuçlarının taranarak, insan denetiminden geçirilmesi ile teşhis konulmaktadır. Laboratuvar sonuçlarının insan tarafından denetimi ve hastalığın teşhisi; uzmanlık eğitimi, insan kaynaklı hatalar ve uzman gerektirmesi gibi zorluklar, kısıtlar getirmektedir. Bu problemler sebebiyle sıtma teşhisinde yapay zekâ ile görüntü sınıflandırma tekniklerinin kullanılması ön plana çıkmaktadır.

Son yıllarda yapay görme sistemlerinde; kamera sensörlerindeki gelişme ve işlemci gücünün artması nedeniyle önemli kazanımlar elde edilmiş ve uygulamalarında yüksek performanslı modeller oluşturulabilmiştir. Aynı zamanda, kamera tabanlı sistemlerin fiyatlarındaki düşüş, yapay görmenin çok çeşitli sahalarda uygulanmasına ortam hazırlamıştır (Affonso, Ross, Vieira, & Carvalho, 2017).

Hızlı bilgisayarlar ve algoritmalardaki iyileştirmeler, makine öğrenimi ve algılayıcılarda ilerlemeler sağlamıştır. Büyük miktarda veriye erişimin artması ve grafik işlemcilerindeki gelişmeler ise yapay zekâ tekniklerinden olan derin öğrenme yöntemlerini uygulanabilir kılmıştır (London, Bradski, Coates, Deng, & Shah, 2017). Derin öğrenme modellerinin çok değişkenli ve karmaşık problemleri çözdüğü görülmektedir. Bunun yanı sıra, büyük ve etiketlenmemiş veri setlerinde bazı sorunlar ortaya çıkmaktadır. Derin öğrenme teknikleri, öznitelikler (feature) ve hiyerarşileri üzerinden ilişkileri öğrenerek bu soruna çözüm getirir (Bengio, 2009).

Son yıllarda yapılan çalışmalarda, derin öğrenme modelleri görüntü işleme (Krizhevsky, Sutskever, & Hinton, 2012), siber güvenlik ve nesnelerin interneti (Yavuz, Ünal, & Gül, 2018), ses tanıma (Deng, Hinton, & Kingsbury, 2013) çeşitli alanlarda ve amaçlarda, var olan çözümlere nazaran, daha iyi tahmin performansı sunmuştur. Derin öğrenme çatısı altında, yapay sinir ağları temelli bir algoritma olan Evrişimli Sinir Ağları (ESA)'nın da

yapılan çalışmalarda görüntü işleme konusunda birçok açıdan K-Nearest Neighbors (KNN), Multilayer Perceptron (MLP) ve DVM gibi geleneksel öğrenme algoritmalarından daha üstün olduğu tespit edilmiştir (Chen, Xiang, Liu, & Pan, 2014) (Ferreira & Giraldi, 2017) (Norlander, Grahn, & Maki, 2015).

1.1. Tezin Amacı

Bu çalışmada; sıtma hastalığının teşhisi için enfekte olmuş kırmızı kan hücrelerinin derin öğrenme yöntemi ile tespiti, teşhis modelinin erişilebilirliğini arttırmak için mobil uygulama geliştirilmesi ve model eğitim sürecinin iyileştirilmesi adına bir web uygulamasının geliştirilmesi hedeflenmiştir.

1.1.1. Hastalık ve teşhisi

Çalışmada yapılacak bozukluk tespiti için hedef olarak sıtma hastalığı ile sağlıklı halini yitirmiş kırmızı kan hücreleri seçilmiştir. Hastalıkların, özellikle bu çalışma kapsamında olan sıtma hastalığının, uzmanlar tarafından yapılan teşhisi; yetişmiş uzmana duyulan ihtiyaç, insan kaynaklı sorunlar, teşhis sürelerinin uzun olması ve bununla birlikte teknolojinin hızla gelişmesi sebebiyle yerini bilgisayar destekli çözümlere bırakmaktadır. Bu bağlamda hastalıkla ilgili görüntüler ve derin öğrenme mimarileri ile oluşturulan modeller, hastalık teşhisinde kullanılagelmektedir.

Bu çalışmada sıtmalı hücre görüntülerinin Evrişimli Sinir Ağlarına verilerek modeller oluşturulmuş ve bu modellerin sıtma hastalığı teşhisi üzerindeki performansları karşılaştırılmıştır.

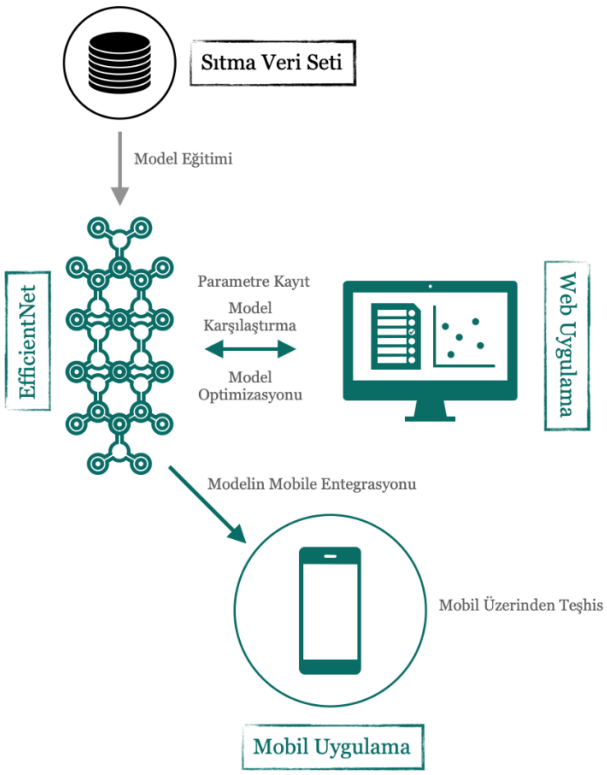
1.1.2. Derin öğrenme ile anomali tespiti ve görselleştirme

Sistemin istenildiği veya tasarlandığı gibi çalışması için istenilmeyen davranışların tespit edilmesi ve gerekli adımların atılması gerekmektedir. Bu sebeple istenmeyen davranışların (anomalilerin) tespit edilmesi sistem teorisinin önemli bir araştırma konusudur. Sistem teorisi de çok geniş bir çalışma alanına sahip olduğu için anomali tespiti her alanda karşımıza çıkmaktadır. Anomali tespiti için çok çeşitli yöntemler geliştirilmiştir. Yapay zekâ destekli anomali tespiti de bunlardan biridir. Son yıllarda etkisini hızla arttıran derin öğrenme kullanılarak, yüksek başarı oranlarıyla anomali

tespiti yapılabilmektedir. Bu çalışmada sıtma parazitinin bulaştığı (enfekte) kırmızı kan hücre görüntülerini içeren veri seti kullanılarak derin öğrenme tabanlı anomali tespiti yapılmıştır. Etiketlenmiş veri seti ile anomaliye özel derin öğrenme modelleri oluşturulmuştur.

Eğitimlerin analiz edilip iyileştirilmesi ihtiyacına binaen model eğitim ortamı ve kayıt tutma sistemi ile bütünleşik çalışan bir web uygulaması geliştirilmiştir. Bu uygulama sayesinde yapılan eğitimlerin başarımları kolaylıkla kıyaslanıp, eğitim girdilerinde gerekli iyileştirmeler yapılarak eğitim başarımları arttırılmıştır.

Hastalık teşhisinde en iyi performansı sağlayan modelin bulunmasına katkı sağlayan web tabanlı bu uygulama grafiklerle desteklenerek modellerin incelenebilirliği ve takip edilebilirliği arttırılmıştır. Oluşturulan modeller; mobil cihazlar üzerinden hızlı tanı koymak, modellerin erişilebilirliğini arttırmak adına, mobil formata çevrilmiştir. Çalışmanın özeti ve çerçevesi Şekil 1.1'de çizilmiştir.



Şekil 1.1 : Çalışmada izlenen yol ve çerçeve özeti.

1.1.3. Çalışmanın mobil uygulama ile desteklenmesi

Eğitilen derin öğrenme modellerinin mobil ortamlara uyarlanması hastalık teşhisini kolaylaştırılması, hızlandırılması ve erişilebilirliğinin arttırılması açılarından önemlidir. Tez kapsamında bu amaca yönelik bir uygulama geliştirilmiştir. Uygulama sayesinde eğitilen modelin pratik bir şekilde kullanılabilir olması sağlanmıştır.

1.2. Literatür Araştırması

2015 yılından itibaren hız kazanan derin öğrenme yöntemlerinin tıbbi alandaki uygulamaları her geçen gün artmaktadır. Bu bağlamda, Birleşik Devletler'de biyomedikal çalışmaların desteklenmesi ve bir çatıda toparlanması adına 1988 yılında NIH (National Institute of Health) tarafından kurulan, NCBI (National Center for Biotechnology Information) hâlihazırda bünyesinde derin öğrenmenin bir kolu olan evrişimli sinir ağı başlığında yaklaşık 20.000 ve derin öğrenme başlığında 15.000'i aşkın makale barındırmaktadır (Affonso, Sassi, & Barreiros, 2015).

Bu bölümde; sıtma hastalığının erken teşhis ve tedavisine katkı sağlamak amacıyla derin öğrenme kullanılan çalışmalar incelenmiştir.

Sıtma hastalığının teşhisi için enfekte olmuş kan hücrelerinin sınıflandırılması amacıyla %97 tutarlılık oranlarına varan DVM gibi birçok geleneksel yöntem kullanılmıştır (Charpe, Bairagi, Desarda, & Barshikar, 2015), (Rosadoa, daCosta, Elias, & Cardoso, 2016). Bu uygulamalar yeterli denebilecek performans değerlerine sahip olsa da birçok ön veriye ve karmaşık ön işlemlere ihtiyaç duymaktadır.

G. Litjens ve ark. (Affonso, Ross, Vieira, & Carvalho, 2017) tıbbi görüntüler üzerinde yapılan 300'ün üzerinde derin öğrenme çalışmasını inceleyerek; bu bağlamdaki zorlukları tanımlamayı ve bu çalışmaların özgün katkılarını belirtmeyi amaçlamışlardır. İncelemeleri sonucunda sıradan evrişimli sinir ağlarına nazaran, soruna özgü inşa edilen mimarilerin daha iyi performans sağladıklarını tespit etmişlerdir. Bunun yanı sıra modelin iyileştirilmesi adına hiperparametrelerin (öğrenme oranı, seyreltme oranı vs.) belirlenmesinde açık bir tarifin bulunmadığı sonucuna varmışlardır. Bu çalışmalarda karşılaşılan en büyük zorluğun; sıklıkla zikredilenin aksine veri seti eksikliği olmadığını,

bu verilerin etiketlenmesi olduğunu belirtmişlerdir. Veri setlerindeki bir diğer sorunun ise sınıflar arası dengesizlik (imbalanced classes) olduğunu da tespit etmişlerdir.

Y. Dong ve arkadaşlarının yaptığı çalışmada (Dong, ve diğerleri, 2017); BAÜ'ye ait sıtma hücre görüntülerini eğitim için işlemden geçirilmiş sonra BAÜ bünyesindeki patoloji uzmanlarına verilmiştir. Her görüntü, en az iki deneyimli patoloji uzmanı tarafından etiketlenmiştir. 1034 adet hastalıklı ve 1531 adet sağlıklı hücre görüntülerini içeren veri setini kullanarak, hastalıklı hücrenin tespiti doğrultusunda dört modeli eğitip performansları karşılaştırılmıştır. Sıtma teşhisi için eğitilen modeller LeNet-5, AlexNet, GoogLeNet, DVM olup sırasıyla; %96.18, %95.79, %98.13, %91.66 başarı oranlarını elde etmişlerdir.

İlerleyen bölümlerde açıklanacak olan ve bu çalışmada kullanılan UTK sıtma veri seti üzerine anomali tespiti amacıyla derin öğrenme kullanılarak birçok çalışma yapılmıştır. S. Kalkan ve arkadaşlarının yaptığı çalışmada (Soner Can Kalkan, Ozgur Koray Sahingoz, 2019) oluşturulan evrişimli sinir ağı modeli bu veri setinde bulunan enfekte kırmızı kan hücresi tespiti için eğitildi. Doğruluk: %97, Kayıp: %7 gibi oldukça iyi eğitim değerlerine erişilen çalışmada, model performansındaki doğruluk %95 iken, kayıp değeri %16'dır. Model eğitim grafiklerinde doğrulama değerlerinin eğitim değerlerine nazaran kötü olması aşırı öğrenme olduğuna dair şüpheleri kuvvetlendirmektedir. Modelin değerlendirmesinde f1 puanı, duyarlık ve hassasiyet değeriyle karışıklık matrisi olmaması dikkat çekmiştir. B. Reddy ve arkadaşlarının yaptığı çalışmada (Reddy & Juliet, 2019) da bu veri seti üzerinde enfekte olmuş ve olmamış hücrelerin sınıflandırması amacıyla ResNet-50 modeli denenmiştir ve %95,91 eğitim doğruluğu, %95,4 doğrulama doğruluğu, %11 eğitim hatası ve %13 doğrulama hatası elde edilmiştir. Eğitim verilerinin iyi çıkmasına karşın karışıklık matrisi gibi model değerlendirme verilerinin olmaması modelin performansı hakkında bilgi vermemekte ve modelin gerçek veriler üzerinde kullanılabilirliğini cevapsız bırakmaktadır. Q. Quan ve ark. (Quan, Wang, & Liu, 2020) da UTK veri setinde bulunan kırmızı kan hücrelerinin sınıflandırması üzerine özel ESA mimarisi geliştirmiş, oluşturdukları modeli kendini birçok alanda ispat eden başka modellerle karşılaştırmışlardır. Çalışmada, ResNet (He, Zhang, Ren, & Sun, 2016) ve DenseNet (Huang, Liu, Maaten, & Weinberger, 2017) modellerinden esinlenerek geliştirilen bu yeni ESA mimarisi "Attentive Dense Circular Net (ADCN)" olarak adlandırılmıştır. ADCN, UTK veri seti ile eğitilmiş ve sıtma hastalığının teşhisi adına

sınıflandırma yapan bir model oluşturulmuş ve %97,5'lik bir f1 puanı elde edilmiştir. Çalışmanın değerlendirme kısmında, sıtma hastalığı ile enfekte olmuş hücrelerin tespiti amacıyla oluşturulmamış modellerin ADCN ile karşılaştırılması bir tutarsızlık gösterse de elde edilen sonuç kayda değerdir. Rajamaran ve ark. (Rajaraman, Jaeger, & Antani, 2019), UTK sıtma veri setini kullanarak enfekte ve sağlıklı kan hücrelerinin sınıflandırılması adına dört adet ESA modeli kullanılmıştır: Özel oluşturulan ESA, VGG-19, SqueezeNet, InceptionResNet-V2. Çalışmada, ImageNet ağırlıkları ile modeller başlatılarak öğrenme aktarımı yapılmış, VGG-19 modeliyle en iyi doğruluk oranı ve f1 puanı (sırasıyla %99,32 ve %99,31) elde edilmiştir. F. Yang ve ark. (Yang, ve diğerleri, 2019) da bu veri seti ile VGG19 modelini kullanıp değişken parametre kümesini daraltarak oluşturdukları bir ESA modeli oluşturmuşlardır. Çalışmada oluşturulan model; VGG-19, AlexNet ve ResNet-50'ye göre daha hızlı eğitilmiş ve %93,4 f-1 puanıyla daha iyi sonuç vermiştir.

J. Hung ve ark. (Hung & Carpenter, 2017) Brazilya'da yaşayan dört sıtma hastasından alınan kan lekelerinin çeşitli işlemlerden geçirilerek yaklaşık 100.000 adet kan hücresi görüntüsünün elde edilmesiyle oluşturulan bir veri setin üzerine çalışmışlardır. Bu çalışma öğrenim aktarımı yapılarak iki aşamada tamamlanmıştır: "faster R-CNN" modeliyle enfekte olmuş kan hücrelerinin tespiti ve AlexNet ile kan hücresi olmayan hücrelerin 7 ayrı tür (trofozoit, şizont, akyuvar vs.) için sınıflandırılması. Tüm modelleri önceden ImageNet ile eğitilmiş ve çalışmadaki veri seti ile değerler düzenlenerek %98'e varan doğruluk oranları elde edilmiştir. Diğer bir öğrenim aktarımı yapan çalışma da Vijayalakshmi ve ark. (A & B, 2020) VGG-DVM modelleriyle öğrenim aktarımı yaparak sıtma hastalığının bulaştığı kan hücrelerinin tespiti amacıyla yapılan çalışmadır. VGG modeli 'imagenet' ağırlıkları ile ilklendirilmiş ve eğitim yapılmıştır. Model performansını değerlendirmek adına %91 f1 puanı elde edilmesi, sıtma teşhisi özelinde tıp alanında öğrenim aktarımının gelecek vadettiğine bir işarettir.

Z. Liang ve ark. (Zhaohui, ve diğerleri, 2016), yarısı enfekte olmuş yarısı sağlıklı 27.578 adet alyuvar hücre görüntüsü içeren veri seti ile 17 katmanlı ESA mimarisini eğiterek enfekte hücre tespiti amacıyla %97'leri yakalayan f1 puanı ve doğruluk değerine sahip ESA modeli oluşturmuşlardır.

X. Ran ve ark. (Ran, Chen, Liu, & Chen, 2017), geliştirdikleri android mobil uygulama üzerinden eş zamanlı nesne tespitini telefonun localinde veya sunucu üzerinden gerçekleştirmeyi hedeflemişlerdir. Çalışmada, ESA mimarisine sahip olan Yolo ve Darknet'in eğitilmesiyle elde edilen modeller kullanılarak mobil ve sunucu üzerinden eş zamanlı nesne tespiti yapılmıştır. Bu çalışmada mobil üzerindeki modelin; mobil ortamın kısıtları sebebiyle daha küçük bir ağa sahip olması ve bundan doğacak eksikliklerin giderilmesi amacıyla mobil, sunucu üzerindeki gerçek ağ ile desteklenmiştir. Çalışmada mobil üzerinde çalışan modelin performansına dair net bir açıklama yapılmadığı için bu çalışmadan, ESA modellerinin mobil üzerindeki performansına dair bilgi alınamamaktadır.

1.3. Hipotez

Derin öğrenme çatısı altında Evrişimli Sinir Ağı modellerinden biri olan EfficientNet, birçok karmaşık veri seti üzerinde başarılı sınıflandırmalar sağladığı gibi enfekte ve sağlıklı kırmızı kan hücresi görüntülerinden oluşan sıtma veri seti ile eğitildiğinde de yüksek yüksek performans değerine sahip sonuçlar verir. Elde edilen yüksek performanslı modeller mobil cihazlarda çalışabilecek şekilde uyarlandığında sıtma teşhisinin hızı ve erişilebilirliği artmış olur.

1.4. Katkılar

Sıtma hastalığına sebep olan enfekte olmuş kırmızı kan hücrelerinin sınıflandırılması adına birçok model ile çalışma yapılmış, fakat bu çalışmadan önce EfficientNet ile yapılmış akademik bir çalışma yoktur. Bu çalışma ile EfficientNet modelleri (B1-B7) kullanılarak sıtma teşhisi amacıyla öğrenim aktarımı yapılmıştır. Sıtma teşhisini başarıyla yapabilecek; %96,1 F-1 puanı ve %96,2 doğruluk oranına sahip bir model oluşturulmuştur. Bu modelin eğitim ve performans değerleri, aynı veri seti ile çalışmış diğer makine öğrenmesi modelleri ile karşılaştırılmıştır.

Model eğitim sırasında; verimin ve görselliğin arttırılması, incelenebilirliğin kolaylaştırılması, parametrelerin, sonuçların kaydedilmesi ve performansın iyileştirilmesi amacıyla web tabanlı bir uygulama geliştirilmiştir. Uygulama github üzerinden derin öğrenme ve özellikle ESA çalışan diğer araştırmacıların istifadesine sunulmuştur.

Sıtma teşhisi için geliştirilen ESA modelinin erişilebilirliğinin arttırılması ve bu sayede hızlı ve erken tanıya olanak sağlanması amacıyla, model mobilde çalışacak hale getirilmiş ve mobil bir uygulama geliştirilmiştir. Oluşturulan model, herhangi bir kuruluma ve eğitime gerek kalmadan, mobil uygulama üzerinden kullanılabilir hale getirilmiştir.

BÖLÜM 2. ÇERÇEVE

EfficientNet ile sıtma hastalığının teşhisi amacıyla; enfekte olmuş kırmızı kan hücresinin tespitini yapacak mimarinin eğitilip modelin oluşturulması, mobile aktarılması ve eğitim süreci verilerinin, oluşan model performansının kaydedilmesi için geliştirilen web uygulamasının anlatımından önce çalışmanın çerçevesinin çizilmesi bu bölümde yapılacaktır. Sırasıyla; veri setinin muhtevasını açıklamak adına sıtma ve teşhisi, bir çatı olarak yapay zekâ, EfficientNet, öğrenim aktarımı ve yardımcı araçlar başlığı altında görselleştirmeyle mobil ortamda derin öğrenmenin uygulanması anlatılacaktır.

2.1. Tıbbi Veriler

Tıbbi veriler iki ana başlıkta incelenebilir; tıbbi görüntüler ve sayısal veriler. Modern tıbbın; analiz, teşhis ve tedavi amacıyla kullandığı en önemli argümanlar bu verilerdir. Sayısal verilere; kan basınç değeri, nabız değeri, karaciğer değerleri vb. örnek verilebilir. Sayısal veriler

Tıbbi görüntüler; tomografi taraması, manyetik rezonans (MR) görüntüsü, nükleer tıp görüntülemesi, ultrason, x-ray, mikroskobik görüntüler vs. olarak sıralanabilir. Bu çalışmada, alınan kan lekeleri görüntüleriyle hazırlanmış veri seti kullanılmıştır.

2.2. Sıtma (Malaria) ve Teşhisi

Sıtma (Malaria), parazit dişi Anopheles sivrisineklerinin ısırıklarıyla insana bulaşan ölümcül bir hastalıktır. DSÖ'nün yayınladığı rapora göre, 2015 yılında tespit edilmiş 214 milyon sıtma vakası ve 438 bin sıtma kaynaklı ölüm olmuştur (WHO, Disease burden of malaria, 2020). 5 yaşından küçük çocukların toplumdaki en savunmasız kesim olduğu ve ölüm oranının % 61 olduğu rapor edilmiştir. Hastalığa sebep olan parazitin en yaygın Afrika'da bulunduğu ve azalarak, sırasıyla Güney Doğu Asya ve Doğu Akdeniz'de olduğu gözlemlenmektedir (WHO, World malaria report, 2017). Ayrıca, sıtma

hastalığının yıllık maliyetinin 12 milyar dolar olduğu tahmin edilmektedir (Mali, Kachur, & Arguin, 2012).

İnsan, sıtma taşıyan bir sivrisinek tarafından ısırıldıktan sonra, bir haftadan bir aya kadar hastalığın belirtileri fark edilmez. Bu süre zarfında sıtma parazitleri, kan dolaşımındaki kırmızı kan hücrelerini işgal etmeden önce kişinin karaciğerinde çoğalır. Kırmızı kan hücrelerinin içine girdikten sonra parazitler çoğalmaya ve enfeksiyonu yaymaya devam eder. Enfekte kırmızı kan hücreleri sonunda yırtılır ve hastada; terleme, yüksek ateş, titreme ve mide bulantısı gibi grip benzeri belirtiler gösterir. Hastalık ilerledikçe, hastanın dalağı ve karaciğeri büyür. Sıtma, anemiye veya sarılığa neden olabilir. Bazı ciddi vakalarda ise beyne saldırır ve nörolojik problemler yaratır (Ashley & White, 2014).

Çok eski bir geçmişe sahip olan sıtma hastalığına karşı günümüze kadar pek çok tedavi yöntemi ve ilaç geliştirilmiştir. Sıtma ile mücadelede, özellik son yüzyıldaki gelişmeler sayesinde hayli yol katedilmiştir. Sıtmaya karşı ilaçların geliştirilmesi, DDT gibi zehirler belli bir süre için tek çözüm olarak görülse de; parazitin ilaca, sivrisineklerin DDT karşı direnç göstermesi, DDT gibi zehirlerin çevreyi kirletmesi gibi sebepler erken teşhis ve tedavinin öneminin tekrar anlaşılması sağlamıştır (Akdur, 2006).

Erken teşhis ve tedavi, birçok hastalık karşısında olduğu gibi, sıtma ile mücadelenin de en etkili yoludur. İnce mikroskobik zar üzerinde kan örneklerinin uzmanlar tarafından incelenmesi, hastalık teşhisinde en güvenilir yöntemdir. Fakat en güvenilir yöntem olmasına rağmen farklı teşhis yöntemlerine yönelmeyi gerektiren zorluklar da vardır: Teşhisi yapacak uzmanın yetkinliği, yanlış teşhis ve akabinde gelen yanlış tedavi, hastalığın ender olduğu ülkelerdeki kaynak sıkıntısı, teşhise harcanan zamanın fazlalığı vs. (K., G., & B., 2003). Sıtma olmadığı halde, koyulan sıtma tanısı sonucu kullanılan ilaçlar sebebiyle karın ağrısı ve bulantı gibi gelen yan etkiler veya sıtma olduğunun anlaşılmaması sonucu kullanılan yanlış antibiyotiklerin sıtmanın şiddetini arttırması, tamamıyla insan tarafından teşhisin yan etkileri olarak verilebilir (MahdiehPoostchi, Silamut, J.Maude, Jaeger, & Thoma, 2018). Bu gibi sebeplerle bilgisayar destekli çözümler önem kazanmaktadır.

Makine öğrenmesi algoritmalarını kullanan bilgisayar destekli teşhis araçları mikroskobik kan lekesi görüntülerini kullanır ve bu sayede klinik teşhiste karşılaşılan zahmeti azaltır. Rajamaran ve arkadaşlarının (Rajaraman, Jaeger, & Antani, 2019)

çalışmasında ortaya koyulan sunumda da görüldüğü gibi bu araçlar hastalığa dair patoloji öznitelikleri ön plana çıkararak klinik teşhiste karar vermeye yardımcı olurlar. Fakat sıtma hastalığına uygulanan çalışmalarda, veri setine özgün optimizasyonlar yapmak için manuel öznitelik çıkarma işlemi yapılmaktadır. Derin öğrenme çalışmaları ile buna son verilmiş baştan sonra otomatize öznitelik çıkarma ve sınıflandırma işlemleri yapılmış, teşhis maliyeti düşürülmüş, güvenilir sonuçlar alınmıştır.

2.3. Yapay Zekâ

Yapay zekâ çalışma alanı (AI) resmi olarak 1956'da New Hampshire, Hannover'deki Dartmouth College'da DARPA destekli bir yaz konferansı ile başladı. 'Yapay Zekâ' terimi 1956 konferansında ortaya çıkmasına rağmen, 1956'dan önce de konuşuluyordu. Alan Turing'in çalışmaları ve Turing Machine bunun bir örneğidir (Bringsjord & Govindarajulu, 2018). Yapay zekâ; bir sistemin harici verileri doğru bir şekilde yorumlayabilme, verileri kullanarak öğrenebilme ve bu öğrenilmiş bilgiyi esnek adaptasyon yoluyla belirli hedeflere ve görevlere ulaşmak için kullanma becerisidir (Kaplan & Haenlein, 2019). Yapay zekâ gelişimi süresince, genellikle birbirleriyle iletişim kuramayan alt bölümlere ayrılmıştır (McCorduck, 2004). Bu alt bölümler; robotik, makine öğrenmesi gibi belirli hedeflere odaklanan teknik hususlara dayanmaktadır (Poole, Mackworth, & Goebel, 1998).

Yapay zekâ çalışmalarının alt bölümü olan makine öğrenmesi, makine tarafından analitik model oluşturmaya yarayan veri analiz metodudur. Makine öğrenmesi konusuna, temelde iki ana iş için çalışılır; makineler tarafından yapılabilecek (ve yapılması tercih edilen) işler, insanlar tarafından yapılamayan veya yapılmaması gereken işler. Makine öğrenmesi ana hatlarıyla; veri toplama, veri temizleme, öğrenme ve problem çözme süreçlerinden oluşur. Problem ile ilgili veriler toplanır, öğrenme sürecine hazır hale getirilir ve öğrenme süreciyle problem çözümü için model oluşturulur. Veri toplama birçok alan için en zor aşamalardan biridir. Yeterli veriyi elde edememe, ilgili alandaki yapay zekâ çalışmalarını sekteye uğratan en önemli sebeplerden biridir. Bunun yanı sıra, veri setindeki verilerin içeriği, kalitesi ve değeri çalışmayı etkileyen önemli özelliklerdendir (Gundersen, Gil, & Aha, 2018). Öğrenme süreci ise, sistemin ortamdaki değişikliği algılayıp adapte olma ve ortamın yeni özelliklerine göre işlevini sürdürmesi olarak tanımlanabilir. Bir sistemin hayatta kalabilmesi için ortamdaki değişikliklere uyum sağlaması gerekmektedir. Makine

öğrenme çalışmaları; bilgisayar bilimi, istatistik ve bilgisayar mühendisliğinden yararlanır veya bu uzmanlıkların ortak çalışma alanıdır. Temelde bu üç alanın çalışmasıyla ortaya çıkmasına rağmen birçok disiplinin ve çalışma alanının sorunlarına çözüm sunar.

Makine öğrenmesiyle, insan açısından anlamsız verilere değer kazandırılıp, bu verilerden anlam çıkartılabilir. Bilgisayar mühendisliği yardımıyla deterministik problemler çözülebilir. Örneğin, bir kalp hastasının sağlığını denetim altında tutmak için yazılan bir yazılım hastanın kalp atışları anormal seviyelere eriştiğinde gerekli adımları atabilir. Ancak, yeterince bilgi sahibi olunmayan yahut çözecek zaman ve/veya gücünün olmadığı pek çok deterministik olmayan problem karşısında bilgisayar mühendisliği yetersiz kalır. İstatistik bu tür problemlere çözüm sağlar. Örneğin, istatistik olmadan insanların üzüntü veya endişelerinin beklenmedik durumlara karşı modellenmesi çok zordur. Özetle makine öğrenmesi, istatistiğin oluşturduğu matematiksel modelleri eğitmek üzere çalışır. Makine öğrenmesi ile programla arasında ince bir fark vardır ve basitçe şöyle tanımlanabilir: Programlama yapılırken gerekli veriler ve veriler üzerinde yapılacak işlem makineye direkt olarak verilir, sonrasında sonuç alınır. Makine öğrenmesi işleminde; veriler ve beklenen sonuçlar makineye verilir, sonrasında bu veriler arasındaki ilişki veya veriler üzerinde yapılması gereken işlemler çıktı olarak alınır. Bu ilişkilerin çıktı olarak alınmasına öğrenme veya model eğitme işlemi denir.

Makine öğrenmesinde eğitimi türleri genel itibariyle üç başlıkta incelenebilir: Denetimli (supervised), denetimsiz (unsupervised) öğrenme ve yarı-denetimli (semi-supervised) öğrenme (Maloof, 2006). Denetimli öğrenmede kullanılan veri seti; girdileri, girdilerin özelliklerini ve girdilerin çıktılarını içerir. Böylece oluşturulan model girdilere göre doğru çıktı üretmeyi öğrenmiştir. Sınıflandırma ve regresyon problemleri için genellikle denetimli öğrenme kullanılır (Alpaydın, 2014). Denetimsiz öğrenmede kullanılan veri seti; girdiler ve girdilerin özelliklerini içerirken, girdilerin çıktılarını içermez. Oluşturulan model girdiler ve özelliklerine bakarak girdiler arasındaki ilişkileri inceler. Kümeleme (clustering) problemleri için genellikle denetimsiz öğrenme kullanılır. Denetimli ve denetimsiz öğrenmenin beraber kullanılarak birbirini tamamladığı eğitimlere de yarı-denetimli öğrenme denir (Shrestha & Mahmood, 2019).

Geleneksel makine öğrenmesi metotları DVM (Acharya, Ng, Tan, & Sree, 2012), sinir ağları (Zhou, Jiang, Yang, & Chen, 2002) gibi çözümleri içerir. Derin sinir ağları ise çoklu katmanlar kurulmasını sağlayıp veri setindeki karmaşık ilişkilerin çözümlenmesinde düşük hata oranlarıyla görüntü analizinde ve işlemede başarı sağlamıştır (Cao, Yuan, He, Zhang, & He, 2018).

2.3.1. Derin öğrenme

Derin öğrenme, yapay sinir ağlarının en önemli ve olgun modelidir. Yapay sinir ağları ile ilgili ilk çalışmalar 1943 yılına kadar uzanır (Kleene, 1956) ve yapay sinir ağlarının bir prototipi olan algılayıcı (perceptron) F. Rosenblatt tarafından 1957'de oluşturulmuştur (Rosenblatt, 1958). 1970'lerde (Werbos & John) hakkında bahsedilmeye başlanan geri yayılım (backpropagation) öğrenme algoritması 1980'den (Lecun, Bengio, & Hinton, Deep learning, 2015) itibaren sinir algoritmalarına uygulanarak büyük etki sağlamıştır. Araştırmacılar son yıllarda gelişen donanım teknolojisiyle daha derin sinir ağlarını eğitebilecek çalışmalar yapmaya başlamışlardır. Donanım alanındaki yenilikler bu çalışmaların hızla yayılmasına hizmet etmiştir (AL-Allaf, AbdAlKader, & Tamimi, 2013).

Sinir ağları beş ayrı sınıfta incelenebilir; ileriye doğru sinir ağları, tekrarlayan sinir ağları, dairesel tabanlı fonksiyon sinir ağları, Kohonen özörgütlemeli (self-organizing) sinir ağları, modüler sinir ağları. İleriye doğru sinir ağlarında, bilgi sadece bir yönde, girdiden çıktıya doğru ilerler. Tekrarlayan sinir ağlarında, ileriye doğru sinir ağlarının aksine, işlem üniteleri bir tür daire formundadırlar. Bir katmanın çıktısı, bir sonraki katmanın girdisi konumundadır. Dairesel tabanlı fonksiyon sinir ağları; girdi, gizli ve çıktı olmak üzere üç katmana sahiptir. Gizli katmanda dairesel tabanlı implemente edilmiştir. Kohonen özörgütlemeli sinir ağları, denetimsiz öğrenmeyi kullanarak modeli kendi kendine organize eder. Son olarak modüler sinir ağları, geniş bir ağı küçük bağımsız ağlara böler. Bu bağımsız ağlar sonradan birleştirilmek üzere farklı görevleri üstlenir (Shrestha & Mahmood, 2019).

Derin öğrenme, sinir ağlarının bir türüdür ve sinir ağ mimarisine sahiptir. Sinir ağlarından farkı ise, derin öğrenme mimarisinde gizli katmanların olmasıdır (Alpaydın, 2014). Bunun yanı sıra derin öğrenme sürecinde model, öğrenme işlemini daha verimli ve

isabetli hale getirecek öznitelikleri kendi çıkartır (Dara & Tumma, 2018). Aynı zamanda, aşırı öğrenme (Tu, 1996) olmaya meyilli modeller, dengeli olmayan veri setleri gibi sorunlara yönelik farklı çözümlerin gerekmesi dezavantajdır. Her modelin her probleme uygulanamaması ise bir başka dezavantaj olarak görülür. "No free lunch theorem" (NFLT) (Wolpert, 1996) diye adlandırılan bu teorem; bir problem için bulunmuş özel bir çözümün başka bir problemde çözümsüz kalması üzerinedir. Buna rağmen derin öğrenme, kanser hücrelerinin tespiti ve kanser hastalığının erken teşhisi için umut verici bir makine öğrenme yöntemidir.

Derin öğrenme sınıflandırma ve tahmin problemlerinde, makine görmesi, örüntü tanıma, görüntü ve ses işlemede oldukça başarılı sonuçlar elde etmiştir (Deng, 2014). Güçlü ekran kartlarının üretilmesi ve yaygınlaştırılmasıyla karmaşık algoritmalar derin sinir ağlarında eğitilebilir hale geldi. Hali hazırda derin öğrenme, büyük verinin ölçeklendirilebilmesi ve işlenebilmesi de düşünüldüğünde yapay zekâ için en son ve etkili teknoloji olarak kabul görmektedir.

Öğrenme algoritmaları, derin öğrenmenin en önemli parçalarındandır. Katman sayılarının artması, öğrenmeyi derin hale getirir. Her katman, probleme dair ayrı bir nokta üzerine odaklanır. Yüz tanıma üzerine yapılan bir çalışmada (Najafabadi, ve diğerleri, 2015), ilk katmanın yüzün şeklini, ikinci katmanın yüzün unsurlarını (göz, burun vs.) ve üçüncü katmanın yüze dair daha detay noktaları tanıdığı, öğrendiği ortaya konulmuştur.

Denetimli öğrenmede genellikle üç adet veri seti kullanılır. Bunlardan ilki eğitim veri setidir ve öğrenmenin denetimli olmasını sağlar. Etiket (label, class) özniteliği beklenen sonuçtur. Sinir ağlarının iç katmanlarındaki ağırlıklar probleme özgü değerlerini bu sonuçlara göre alırlar. İkincisi ise doğrulama (validation) veri setidir. Bu veri seti, öğrenme işleminde ağırlıkların, daha iyi değerler alması adına, ayarlanmasında ve öğrenme işleminin performansının ortaya çıkarılmasında kullanılır. Model eğitim performansı için verilen doğrulama doğruluğu (validation accuracy) ve doğrulama kaybı (validation loss) bu veri seti ile ilişkilidir. Üçüncü ve son veri seti de test veri setidir. Öğrenme işlemi sonucu ortaya çıkan modelin test edilmesinde kullanılır (Şekil 2.1). Öğrenme işlemi sırasında, bir epoch, tüm eğitim veri setinin sinir ağından bir kere tamamen geçmesi anlamına gelmektedir. Sinir ağlarının amacı, ilgili probleme göre en iyi katmanlar arası ağırlıklar setini belirlemektir. En iyi ağırlıkları belirlerken göz önünde

bulundurulması gereken; hesaplama zamanının azaltılması, eğitim ve model performansının arttırılması gibi hedefler derin öğrenmenin zorlukları arasında yer alır.

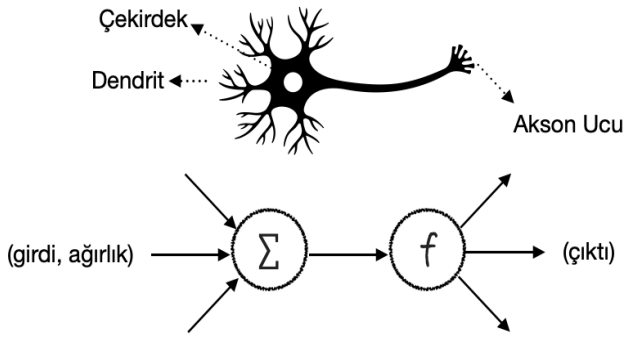
Şekil 2.1 : Denetimli öğrenme eğitimde kullanılan veri setleri ve kullanım alanları.

Sinir hücreleri, öğrenme işleminde en önemli oyuncudur. Bir önceki katmanda bulunan, bir veya daha fazla sinir hücresinden alınan girdi bağlantı ağırlığı ile çarpılıp toplanır (Denklem 2.1). Sonrasında, aktivasyon fonksiyonundan geçirilir ve elde edilen çıktı bir sonraki katmadaki sinir hücresine veya sinir hücrelerine aktarılır (Denklem 2.2).

$$Y = \sum (girdi)\ x\ (ağırlık) + b \tag{2.1}$$

$$çıktı = f(Y) \tag{2.2}$$

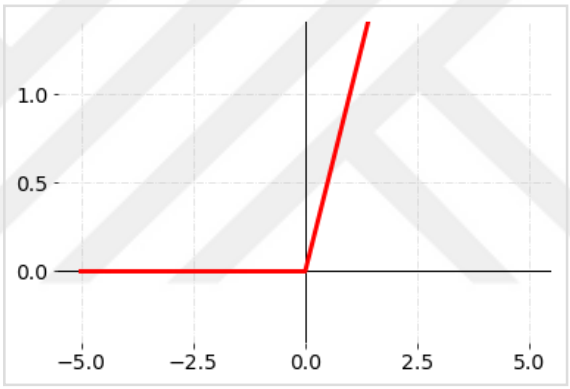
Basitçe, aktarılan çıktı bir sonraki sinir için ateşleme veya sönümleme manasına gelir. Yani bu çıktı bir sonraki nöronu ya etkisiz hale getirir ya da çalışır hale getirir. Ateşleme terimi, beyindeki biyolojik çalışmadan esinlenerek konulmuştur (Şekil 2.2). Yapay sinir ağlarının, ateşleme teriminde olduğu gibi, çalışma prensibi de beyin hücrelerinin çalışma prensibinden esinlenerek tasarlanmıştır (Öztemel, 2003).



Şekil 2.2 : Sinir hücresi ve yapay sinir hücresi benzerliği.

Derin öğrenme uygulamalarında, problemlere göre değişkenlik gösteren çeşitli aktivasyon fonksiyonları kullanılır. Bunlardan bazıları ve en çok tercih edilenleri; DDÜ (Rectified Linear Unit) fonksiyonu (ve türevleri), basamak fonksiyonu, sigmoid fonksiyonu, tanh fonksiyonu olarak sıralanabilir. DDÜ fonksiyonu diğer fonksiyonlara nazaran karmaşık ve çok katmanlı veri setleri üzerinde çalışan derin sinir ağlarında daha etkili ve hızlı öğrenme işlemi sağlamıştır. Ayrıca DDÜ; Sigmoid ve tanh fonksiyonlarına nazaran daha efektif sonuçlar vermiştir (Glorot, Bordes, & Bengio, 2011). DDÜ fonksiyonu basitçe, giriş değeri sıfırdan küçük ise sonucu sıfıra, giriş değeri sıfırdan büyükse sonucu girişe eşitler (2.3), grafiği Şekil 2.3'deki gibidir.

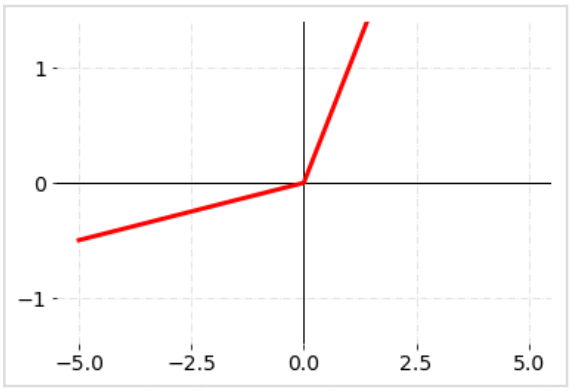
$$f(x) = \begin{cases} x < 0 a.için 0 \\ x \geq 0 a.için x \end{cases} \quad (2.3)$$



Şekil 2.3 : DDÜ Fonksiyon grafiği.

DDÜ'de görülen bir sorun ise Ölen DDÜ Problemidir (Dying ReLU Problem). Bu problem, bazı DDÜ nöronlarının, girdilere bağlı olarak ölmesi/pasif hale geçmesi ve bu halde kalmasıdır (Xu, Wang, Chen, & Li, 2015). Pasif nöronların yapay sinir ağında bulunması ve sayılarının artması, model eğitimini etkiler. Bu sorun, Sızıntı DDÜ ile çözüme kavuşturulmaya çalışılmıştır. Sıfırdan küçük girdi değerleri için sonucun sıfıra eşit olmaması, fonksiyonda sızıntıya sebep olarak, fonksiyon aralığını genişletir (Maas, Hannun, & Ng, 2013). Sızıntı DDÜ fonksiyonu basitçe giriş değeri sıfırdan küçük ise sonucu girişin %1'ine, giriş değeri sıfırdan büyükse sonucu girişe eşitler (2.4), grafiği Şekil 2.4'deki gibidir.

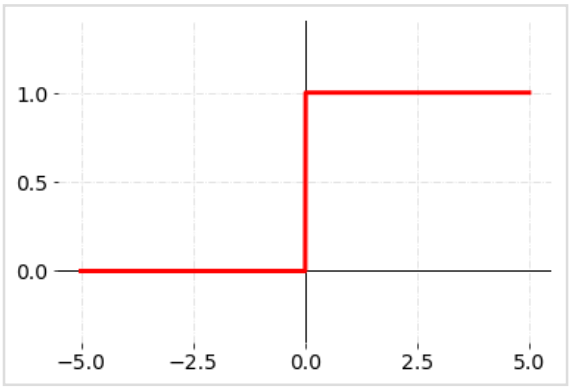
$$f(x) = \begin{cases} x < 0 a\,için ax * 0.01 \\ x \geq 0 a\,için aaax \end{cases} \quad (2.4)$$



Şekil 2.4 : Sızıntı DDÜ Fonksiyon grafiği.

Basamak fonksiyonu, en basit aktivasyon fonksiyonu olarak kabul edilir. Girdi sıfırdan küçük ise çıktı sıfıra, girdi sıfırdan büyük ise çıktı bire eşitlenir (2.5), grafiği Şekil 2.5'deki gibidir.

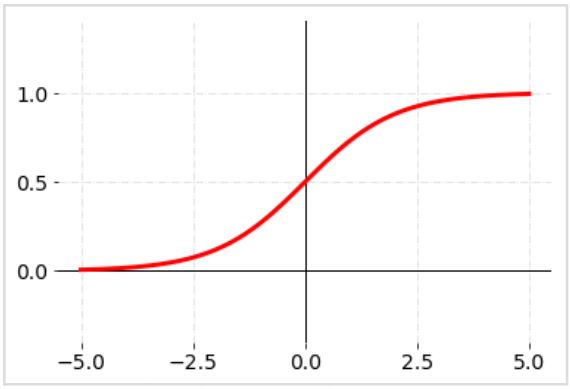
$$f(x) = \begin{cases} x < 0 a\,için a0 \\ x \geq 0 a\,için a1 \end{cases} \quad (2.5)$$



Şekil 2.5 : Basamak Fonksiyon grafiği.

Sigmoid fonksiyonu, basamak fonksiyonunun hiperbolik tanjant formundaki halidir (2.6), grafiği Şekil 2.6'deki gibidir. Basamak fonksiyonundan farkı, problemine göre, daha iyi sınıflandırma sonucu vermesidir:

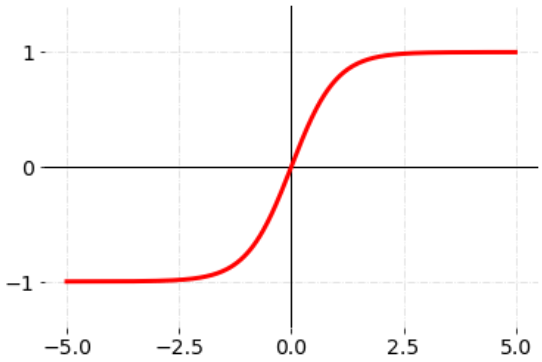
$$f(x) = \rho(x) = \frac{1}{1 + e^{-x}} \tag{2.6}$$



Şekil 2.6 : Sigmoid Fonksiyon grafiği.

Tanh fonksiyonu (2.7) Sigmoid fonksiyonuna, sınırlarının (-1,1) arasında olması dışında, oldukça fazla benzer (Öztemel, 2003). Grafiği Şekil 2.7'deki gibidir.

$$f(x) = tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.7}$$



Şekil 2.7 : Tanh Fonksiyon grafiği.

Sinir ağları, giriş katmanı ve çıkış katmanı arasında yer alan gizli (hidden) katman olarak adlandırılan katmana sahiptir. Bu katmanın sayısının fazlalığı sinir ağları 'derin' hale getirir. Böylelikle derin sinir ağlarındaki öğrenmeye de derin öğrenme denir.

Derin öğrenmedeki diğer bir parametrede seyreltme (dropout) oranıdır. Öğrenme işlemi sırasında, seyreltme oranınca gizli katmanda rastgele seçilmiş sinir hücresi yok sayılır.

Bu hücreler öğrenme işlemi sırasında ağırlıkları ve taşıdıkları fonksiyonla hesaba katılmazlar. Bu işlem, öğrenme sürecinde aşırı öğrenme (overfitting) sorunun azaltılmasına yardımcı olur (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014). Aşırı öğrenme sorunu kısaca şöyle özetlenebilir; eğitilen model eğitim veri setindeki gürültü ve detaylara gerekenden fazla odaklanır ve bu minvalde şekil alır. Eğitim sürecinde performans iyi çıkmasına rağmen, model test veri setinde çok kötü performans sergiler (Hawkins, 2004).

Eğitim süresince, alınan çıktı ve beklenen sonucun arasındaki fark kayıp fonksiyon ile hesaplanır. Bu hesap, eğitimin gerçekleşmesi adına geri bildirim görevi görür. Denklem 2.8'deki Ortalama Hata Karesi (Mean Squared Error) en popüler kayıp fonksiyonlarından biridir:

$$OHK = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{2.8}$$

Diğer bir kayıp fonksiyonu da Çapraz Entropi Hatasıdır (Cross Entropy Loss). Genellikle regresyon ve sınıflandırma problemlerinde kullanılır. Çapraz Entropi Hatası, en büyük olabilirlik kestirimini yapar ve çoğu çalışmaya göre OHK'den daha iyi performans sağlar. Denklem 2.9'da; *C* sınıf sayısını ifade eder:

$$H_p(\mathrm{q}) = -\sum_{c=1}^{C} q(y_c) \cdot \log(p(y_c)) \tag{2.9}$$

Bu çalışmada İkili Çapraz Entropi (Binary Cross Entropy) hata fonksiyonu kullanılmıştır. İkili sınıflandırma yapılan çalışmalarda genellikle İkili Çapraz Entropi hata fonksiyonu kullanılır. Denklem 2.10'daki eşitlik İkili Çapraz Entropi hata fonksiyonuna aittir. Fonksiyonda *y* çıktının ait olduğu sınıfı (enfekte / sağlıklı hücre gibi) ifade ederken, *p(y)* tüm *N* noktaları için çıktının bir sınıfa (bu hastalık teşhisinde genellikle hastalığa dair sınıf olur) ait olduğunu ifade eder. Fonksiyon özetle, sonucun bir sınıfa ait olma olasılığı ile diğer sınıfa ait olma olasılıklarının logaritmik toplamlarını yapıp ortalamasını alarak hatayı hesaplar.

$$H_p(\mathrm{q}) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i)) \quad (2.10)$$

Derin öğrenme, derin katmanlı sinir ağı yapılarıyla sorunu öğrenebilmek adına otomatik olarak etkili özellikler seçmek için geniş bilgi setleri ile karmaşık davranışları özümser (Kim, Choi, & Lee, 2015).

2.3.2. Derin sinir ağları

Derin sinir ağları bünyesinde birçok katman ve nöronu barındırır. Farklı problemlerin çözümü için farklı mimariler geliştirilmiştir. Örneğin Evrişimli Sinir Ağları (Convolutional Neural Networks) genellikle görüntü tanıma ve makine görmesi için kullanılırken Tekrarlayan Sinir Ağları (Recurrent Neural Networks) genellikle zaman dizisi problemlerinde veya tahminlemelerde kullanılır. Öte yandan, sınıflandırma problemi için genel geçer bir model-mimari yoktur, çünkü problem özelinde çalışılan alan birçok etkileyici unsura bağlıdır. En çok kullanılan derin sinir ağı mimarileri; evrişimli sinir ağları, otokodlayıcılar (autoencoders), kısıtlı boltzmann makinesi ve uzun kısa vadeli bellek. Otokodlayıcılar girdiyi öğrenip boyutunu küçültüp ve tekrar orijinal girdiyi oluşturabilecek denetimsiz öğrenimi kullanan bir sinir ağıdır. Kısıtlı boltzmann makinesi de aynı şekilde denetimsiz öğrenimi kullanan yapay bir sinir ağıdır. Uzun kısa vadeli bellek ise tekrarlayan sinir ağlarının bir implementasyonudur ve ilk defa 1997'de yapılmıştır. Bu çalışmada sadece evrişimli sinir ağları detaylıca anlatılacaktır (Shrestha & Mahmood, 2019).

2.3.3. Evrişimli sinir ağları

Görüntü analiz işlemlerinde yoğun olarak kullanılan ESA, çok katmanlı ve ileri beslemeli derin öğrenme teknikleridir. Kedilerin görsel korteksinden esinlenilerek ortaya çıkmıştır (Hubel & Wiesel., 1962). 1990'lı yılların başlarında; ses tanıma, yazı tanıma uygulamaları ile başlayan ESA, görüntü tanıma uygulamalarının yapılması ile kullanımı hızla arttı. 2012 yılında Alex Krizhevsky tarafından ImageNet'in, diğer bir adlandırmayla AlexNet'in, tanıtılmasıyla ESA görüntü tanıma ve sınıflandırma alanında en çok kullanılan mimariler arasında yerini aldı (Zhaohui, ve diğerleri, 2016).

ESA, diğer sinir ağları gibi; ağırlıklara ve önyargılara sahip, aldığı girdilere iç çarpım uygulayan sinir hücrelerine sahiptir. ESA, görüntü işleme amacıyla tasarlanmıştır ve bu doğrultuda kurulan mimari ileride kullanılacak parametre sayısını büyük ölçüde azaltır.

Normal sinir ağları sadece görüntü işleme amacıyla tasarlanmadığı için büyük boyutlu görüntü veri setlerinde, özellikle ölçeklendirme konusunda, yetersiz kalmaktadır. Bir katmadaki sinir hücrelerinin sahip olduğu ağırlık miktarını örneklendirmek gerekirse; veri setindeki girdinin boyutu 16x16x3 boyutunda (16 genişlik, 16 yükseklik, 3 renk) ve ilk katmadaki sinir hücreleri tam bağlı ise, bir sinir hücresi 16*16*3 = 768 adet ağırlığa sahip olacaktır. Fakat veri setindeki girdinin boyutu büyüdüğünde, gerçek görüntülerin boyutları (256x256x3) düşünüldüğünde (256*256*3=196.608 adet ağırlık) ve bunlara eklenecek parametre sayıları da düşünüldüğünde, bu ağırlıklar yönetilemeyecek kadar büyürler. ESA için üç temel katman tipi vardır; evrişimli katmanı, örnekleme katmanı ve tüm girdi-çıktıların birbirine bağlı olduğu tam bağlı katman.

ESA yapısında en önemli katman evrişim katmanıdır. Evrişim katmanındaki filtreler; derinlikleri girdi derinliği ile aynı, genişlik ve yüksekliği girdi genişlik ve yüksekliğinden küçük olacak şekilde boyutlara sahip olurlar. Bu boyutlardaki filtreler, görüntü girdilerinin genişlik ve yükseklikleri boyunca evriştirilir ve girdinin tüm noktaları ile iç çarpıma tabi tutulurlar. Bu sayede, tüm girdi için filtrenin cevabını oluşturacak iki boyutlu bir aktivasyon haritası oluşturulur. Bu aktivasyon haritaları, görüntüdeki anormallikler sebebiyle aktifleşir ve filtrelerin öğrenimini sağlar.

Tam bağlı sinir hücresi; önceki katmanda bulunan tüm sinir hücrelerine bağlanması sebebiyle bu sıfatı alır. Yüksek boyutlarda bunun bir ölçeklendirme sorununa dönüşmemesi için; evrişim katmanındaki filtreler girdinin tüm noktalarına değil, kendi boyutlarındaki yerel bölgelere bağlanırlar. Filtrenin boyutunu belirleyen hiperparametreye algı alanı denir. Filtrenin derinliği, her zaman giriş hacminin derinliğine eşittir. Filtrenin genişliği ve yüksekliği, giriş hacminin genişlik ve uzunluğundan bağımsızdır ama derinlikler eşittir. Örnek olarak giriş hacmi 8x8x4 olur ve filtre 2x2 seçilirse, evrişim katmanındaki her nöron 2*2*4=16 adet bağlantıyı giriş hacmi ile yaparlar. Dolayısı ile normal sinir ağlarında olması beklenen 8*8*4 = 256 adet bağlantı ve ağırlık, ESA'da 16 adet olur.

Evirişim işleminin sonucu oluşan çıktının yapısı üç adet hiperparametre tarafından belirlenir. İlk hiperparametre, çıktının derinliğidir. Bu ayrıca, görüntüdeki farklılıkları algılanması ve ilgili sinir hücrelerinin aktive olması için kullanılmak istenen filtre sayına karşılık gelir. İkinci hiperparametre ise adım kaydırma sayısıdır ve filtrelerin girdi üzerinde evrişim yaparken kaç piksel kayması gerektiğini belirler. Eğer piksel kaydırma sayısı 3 olursa; filtre girdinin tüm yüzeylerini kendi algı alanı kapsamında 3 piksel kayarak tarayarak girdi ile evrişim yapar. Son hiperparametre de sıfır ile doldurmadır. Çıktının çerçevelerini sıfır ile doldurarak, çıktının boyutlarının kontrol edilebilirliğine katkı sağlar.

ESA yapısında diğer bir önemli katmanı ise örnekleme katmanıdır ve genellikle iki evrişim katmanının arasında kullanılır. Aldığı çıktının boyutunu düşürerek; hem ağdaki hesaplama yükünü azaltır hem de parametre sayısını azaltarak aşırı öğrenmeyi kontrol altında tutar. Örnekleme katmanı, girdinin her derinliği için ayrı ayrı, aynı işlemi uygular; belirlenen ölçekte (ki bu genellikle 2x2 olur) örnekleme yaparak girdinin boyutlarını küçültür ve sonraki evrişim katmanı için büyütme operasyonu ile matrise yeni boyut verir. ESA'nın en önemli üçüncü katmanı ise tüm girdi-çıktıların birbirine bağlı olduğu tam bağlı katmadır. Bu da yukarıda tam bağlı sinir hücresinde bahsedildiği gibi; katmada bulunan tüm sinir hücrelerinin bir önceki katmandaki tüm sinir hücreleri ile bağlantı yapmasıdır (Simonyan & Zisserman, 2014).

LeCun ve arkadaşları tarafından ortaya atılan gradyan temelli bir yaklaşım sunan ağ yapısına evrişimli sinir ağı adı verilmiştir. Oluşturulan bu yapay sinir ağına ise LeNet adı verilmiştir (Lecun, Bottou, Bengio, & Haffner, 1998).

Bu sinir ağı birden fazla evrişim katmanı, tam bağlı katman, aktivasyon katmanı, sınıflandırıcı katman, ortaklama katmanı ve bunlara ek katmanlardan oluşan çok katmanlı bir ağdır. Her katmanın kendi fonksiyonunu işletmesiyle sınıflandırıcı katmanda sonuç üretilmektedir.

ESA topolojisi üç ana kavrama dayanmaktadır: Algı alanları, paylaşılan ağırlıklar ve uzamsal veya zamansal örnekleme (Lecun, Bottou, Bengio, & Haffner, 1998). ESA, ağa doğrudan verilen normalize edilmiş görüntüler ile öznitelik çıkarma işlemini ortadan kaldırabilir. Genellikle bir görüntü veri seti yüzlerce piksel içerir. Tüm ağ

düşünüldüğünde, her bir sinir hücresi tüm piksellere bağlanır. Bu nedenle, hesaplama maliyeti ve bellek gereksinimleri karşılanamaz bir hal alır.

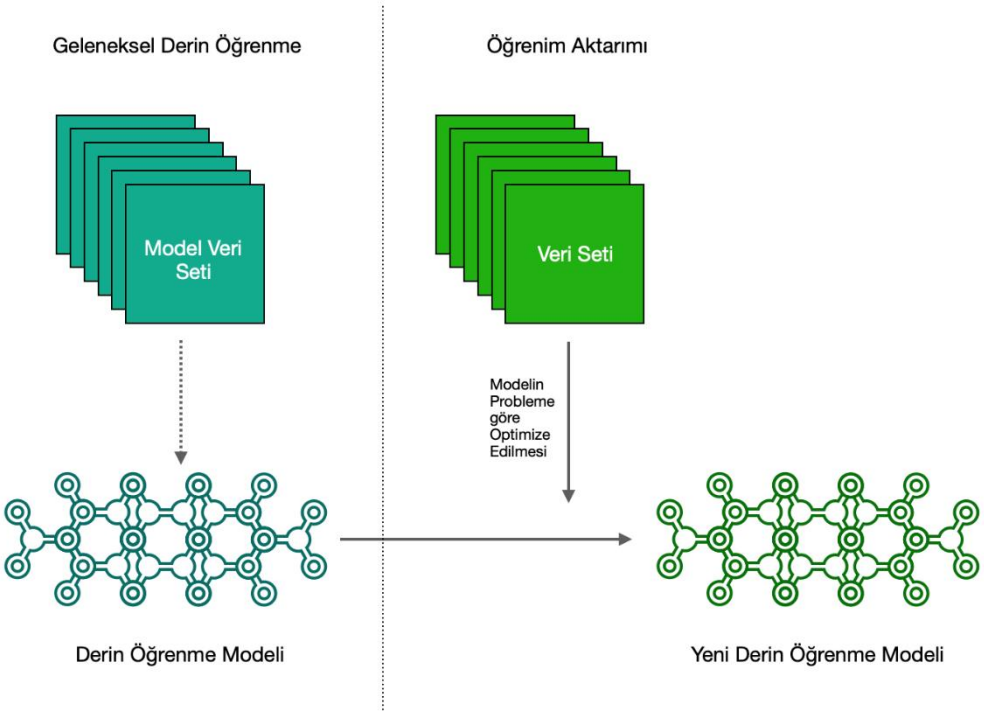
Uzamsal değişmezlik, tüm görüntü pikselleri boyunca paylaşılan ağırlıklardan elde edilir. Alt örnekleme, görsellerdeki bozulmalara karşı değişmezliğe yardımcı olduğu için nesne tanımada önemli bir stratejidir. Kendi doğası gereği görüntü veri kümesinin, güçlü bir uzamsal ilintisi vardır. Bununla başa çıkmak için ESA, algı alanlarını kısıtlar. ESA'nin bir başka özelliği de paylaşılan ağırlıklardır. Bu şekilde, görüntünün farklı yerlerinde bulunan algı alanlarındaki bir piksel seti, öznitelik haritası oluşturan, özdeş ağırlık vektörüne sahiptir. Bu işlem evrişimli dönüşüm olarak düşünülebilir. Her öznitelik haritasının ardından yerel bir alt örnekleme gerçekleştiren bir katman gelir. Böylece ESA alternatif olarak evrişimsel ve alt-örnekleme katmanlarından oluşur (Lecun, Bottou, Bengio, & Haffner, 1998).

2.3.4. Öğrenim aktarımı (transfer learning)

Derin öğrenme şekilleri dört bölüme ayrılabilir: Öğrenim aktarımı, çoklu görev öğrenimi, çok modelli öğrenim ve geleneksel öğrenim. Çoklu görev öğreniminde oluşturulan bir model parametreleri değiştirilerek başka görevler için kullanılır (He, Mao, Guo, & Yi, 2017). Çok modelli öğrenimde, birden fazla model ile bir görev gerçekleştirilmeye çalışılır, bu sayede eğitimde elde edilecek verinin daha zengin olması hedeflenir (Lei, ve diğerleri, 2017). Geleneksel öğrenimde bir model oluşturulur ve eğitim veri seti ile eğitilir. Geleneksel öğrenim; uzun eğitim süresi ve büyük veri seti gerektirmesi gibi bazı zorluklar içermektedir. Bu çalışmada öğrenim aktarımı kullanılmıştır ve aşağıda öğrenim aktarımı anlatılacaktır.

Yeni problemler için yapılan model eğitimlerinde karşılaşılan bazı zorluklar hem araştırmaları sekteye uğratmakta hem de yatırım maliyetini arttırmaktadır. Her yeni problem için veri toplanması, verilerin ayıklanması, verilerin etiketlenmesi ve tekrar yeni modelin eğitilmesi gibi işlemler, araştırma sürecinde ciddi zaman almaktadır. Çoğu zaman yeni problemler için; veri toplanması ve model oluşturulması mümkün olmamaktadır. Öğrenim Aktarımı ise bir problem çözümü için model eğitimi sırasında veya sonucunda edinilen bilginin (öğrenilen noktaların) ilgili başka bir probleme uygulanmasına denir. Örneğin; görüntüdeki elmaları tespit etmek için oluşturulan bir

modelin, görüntüdeki portakalları tespit etmek için kullanılması öğrenim aktarımının bir uygulamasıdır. Elma ile portakalın şekil benzerliğini kullanarak, elma tespit modelinin şekil öğrenimini portakal tespit modeline aktarılmasıdır. Böylece, eğitim sürelerinden önemli tasarruflar yapılmaktadır. Öğrenim aktarımı ile model performansı oranları ciddi miktarda artmakta, eğitim süresi kısalmaktadır (Pan & Yang, 2010). Öğrenme aktarımı ile geleneksel öğrenme ilişkisi Şekil 2.8'de gösterilmiştir.



Şekil 2.8 : Geleneksel derin öğrenme ile öğrenme aktramı ilişkisi.

Öğrenme aktarımı, insanın önceki tecrübelerini yeni karşılaştığı problemlerde kullanmasından yola çıkarak bulunmuştur. Bu tecrübeler ışığında, problemlerin daha hızlı veya daha etkili çözülmesi muhtemeldir. Örnek vermek gerekirse; bir enstrüman çalabilen insan, başka bir enstrümanı ilkine göre daha kolay çalabilir veya yeni bir dil öğrenen bir insan başka bir dili daha rahatlıkla öğrenebilir. Öğrenme aktarımının temelleri "öğrenmeyi öğrenme" üzerine yapılan NIPS-95 çalışmasında tartışılmıştır (Pan & Yang, 2010). Öğrenme aktarımı, 1995 yılından itibaren; "öğrenmeyi öğrenme", "hayat boyu öğrenme", "bilgi/tecrübe aktarımı", "çok görevli öğrenme", "bilgiyi güçlendirme", "birikimli öğrenme" gibi farklı isimler altında giderek fazla dikkat çekmiştir (Thrun & Pratt, 1998).

Öğrenim aktarımı için üç ana husus vardır; neyin aktarılacağı, nasıl aktarılacağı, ne zaman aktarılacağı. Neyin aktarılacağı; bilginin hangi parçasının aktarılacağı ile alakalıdır. Bazı bilgiler eğitildiği probleme özgü üretilmiştir ve az da olsa farklı bir problemde işe yaramayabilir. Neyin aktarılacağına karar verdikten sonra, algoritmanın transfer için geliştirilmesi nasıl aktarılacağı konusu ile alakadır. Ne zamana aktarılacağı konusu bilginin hangi durumda aktarılacağı ile alakalıdır. Örneğin, bir modelin eğitildiği sorun ile transfer edilip uygulanacağı sorun birbirinden tamamen farklıysa bu 'zorlama' transfer hedef kümeye özgü yeni eğitimin performansını negatif yönde etkileyebilir. Bu olumsuz sonuç karşısında oluşan transfere "negatif transfer" adı verilir. Bu gibi hususlarda öğrenim aktarımının faydası yoktur. Neyin aktarılacağı; ağırlıklarının yeniden belirlenmesi ile yapılan örnek tabanlı transfer, öznitelik transferi ve parametre transferi olmak üzere üç durumda incelebilir (Pan & Yang, 2010).

Öğrenim aktarımı sıklıkla önceden eğitilmiş modellerin benzer problemler için tekrar kullanılmasıyla gerçekleştirilir. Önceden eğitilmiş model alınır, uygulanacak problem için düzenlenir (ör: sınıf sayısına göre modelin son katman değiştirilir.) ve eğitim başlatılır. Bu sayede; baştan model oluşturmaya nazaran daha kısa sürede daha etkili sonuçlar elde edilir. Çünkü kullanılan model, öncesinde, milyonlarca görüntü ve binlerce sınıf için eğitilmiş ve şekillendirilmiştir. Öğrenilen bilginin aktarılması yeni problemlerde kolaylık sağlamaktadır ve bu şekilde farklı alanlarda birçok başarılı çalışmaya aracı olmuştur. Do ve arkadaşlarının (Do & Ng., 2006), yazılı metinde sınıflandırma çalışması; Deniz ve arkadaşlarının (Deniz, ve diğerleri, 2018), göğüs kanserinin teşhisi amacıyla yaptıkları çalışma örnek olarak gösterilebilir.

2.3.5. EfficientNet

2019'da Google tarafından geliştirilen yeni bir ESA çalışması olan EfficientNet, doğruluk (accuracy) ve verimlilik (performance) konusunda kayda değer iyileştirmeler sağlar. Yapılan çalışmada sunulan verimlilik modeli, diğer ESA modellerine de uygulanabilir olması ile yeni bir yaklaşım sunar.

Mevcut ESA modelleri doğruluğu arttırmak için genellikle tek başına katman sayısını (derinlik), genişliği veya çözünürlüğü arttırmaya odaklanırken, EfficientNet ise geliştirdikleri basit ama etkili bir metod olan bileşik ölçeklendirme metodunu (compound

scaling method) kullanarak derinlik, genişlik ve çözünürlük boyutlarını belirlenen bir oranda birlikte arttırır.

EfficientNet-B0, AutoML MNAS (Tan, 2018) kullanılarak geliştirilen temel ağdır. Efficient-B1 - B7 modelleri ise temel ağa bileşik ölçeklendirme metodu uygulanarak elde edilir. Özellikle EfficientNet-B7 modeli, en yaygın görsel veri seti olan ImageNet üzerinde 97.1% varan doğruluk elde ederken, mevcut en iyi ESA modeline göre 8.4 kat daha az yer kaplar ve 6.1 kat daha hızlı çıkarım yapar (Tan & Le, 2020). EfficientNet modellerinin en iyi doğruluk (Top-1 Acc.), en iyi 5 doğruluk (Top-5 Acc.), parametre ve SAİ değerlerinin karşılaştırması Tablo 2.1'de yer almaktadır.

Tablo 2.1 : Efficient modellerinin karşılaştırılması.

Model	Top-1 Acc.	Top-5 Acc.	Parametre	SAİ
EfficientNet-B0	77.3 %	93.5 %	5.3 Milyon	0.39 Milyar
EfficientNet-B1	79.2 %	94.5 %	7.8 Milyon	0.70 Milyar
EfficientNet-B2	80.3 %	95.0 %	9.2 Milyon	1.0 Milyar
EfficientNet-B3	81.7 %	95.6 %	12 Milyon	1.8 Milyar
EfficientNet-B4	83.0 %	96.3 %	19 Milyon	4.2 Milyar
EfficientNet-B5	83.7 %	96.7 %	30 Milyon	9.9 Milyar
EfficientNet-B6	84.2 %	96.8 %	43 Milyon	19 Milyar
EfficientNet-B7	84.4 %	97.1 %	66 Milyon	37 Milyar

Derinlik terimi ağlardaki katman sayısı için kullanılır, yüksek derinlik katman sayısının fazlalığını gösterir. Yaygın kanaat daha derin ağların, daha zengin ve karmaşık özellikleri yakalayabilmesi ve yeni verileri üzerinde genelleme yapabilmesidir. Ancak, kaybolan eğim (vanisihing gradient) sorunu nedeniyle daha derin ağların eğitilmesi, görece daha zordur. Atlama bağlantıları (skip connections) ve yığın normalleştirme (batch normalization) gibi çeşitli teknikler eğitim sorununu hafifletmesine rağmen, çok derin ağların doğruluk kazancındaki azalmanın önüne geçemez. Örneğin, ResNet-1000 çok daha fazla katmana sahip olmasına rağmen ResNet-101 ile benzer bir doğruluğa sahiptir (Tan & Le., 2019).

Genişlik ağdaki her bir katmanın büyüklüğü ile ilişkilidir, katmanlardaki nöron sayısı arttıkça ağın genişliği artar. Doğruluğu arttırmak için ağın genişliğini arttırma, genellikle

küçük boyutlu modeller için kullanılır. Daha geniş ağlar daha ayrıntılı özellikleri yakalama eğilimindedir ve eğitilmesi daha kolaydır. Bununla birlikte, aşırı geniş ancak derin olmayan ağlar ile daha üst düzey özelliklerin yakalanması daha zordur (Tan & Le., 2019).

Çözünürlük, derin öğrenmede ağın girdisi olan görsellerin en boy oranı için kullanılan bir terimdir. Yüksek çözünürlüklü görsel girdileri ile daha ince detayları yakalayabilen bir ESA eğitilebilir. Geliştirilen ilk ESA'lar azami 224x224 çözünürlüklü girdilerle eğitilebilirken, güncel ESA modelleri 480x480, 600x600 gibi yüksek çözünürlüklere çıkarak daha yüksek doğrulukta sonuçlar verebilmektedir. Ancak daha yüksek çözünürlüklerde ağ doyuma ulaşıp doğruluk kazanımı yok olmaktadır (Tan & Le., 2019).

Yapılan gözlemler, ESA modellerinde yukarıda açıklanan 3 boyuttan herhangi birini arttırılmasının doğruluk değerini arttırdığı ancak büyük modellerde bu kazanımın kaybolduğu gösterir.

Bileşik ölçeklendirme metodu ile oluşturulan EfficientNet modellerinde kullanılan girdi çözünürlükleri Tablo 2.2'deki gibidir.

Tablo 2.2 : EfficientNet modelleri çözünürlükleri (Fu, 2020).

Model	Çözünürlük
EfficientNet-B0	224
EfficientNet-B1	240
EfficientNet-B2	260
EfficientNet-B3	300
EfficientNet-B4	380
EfficientNet-B5	456
EfficientNet-B6	528
EfficientNet-B7	600

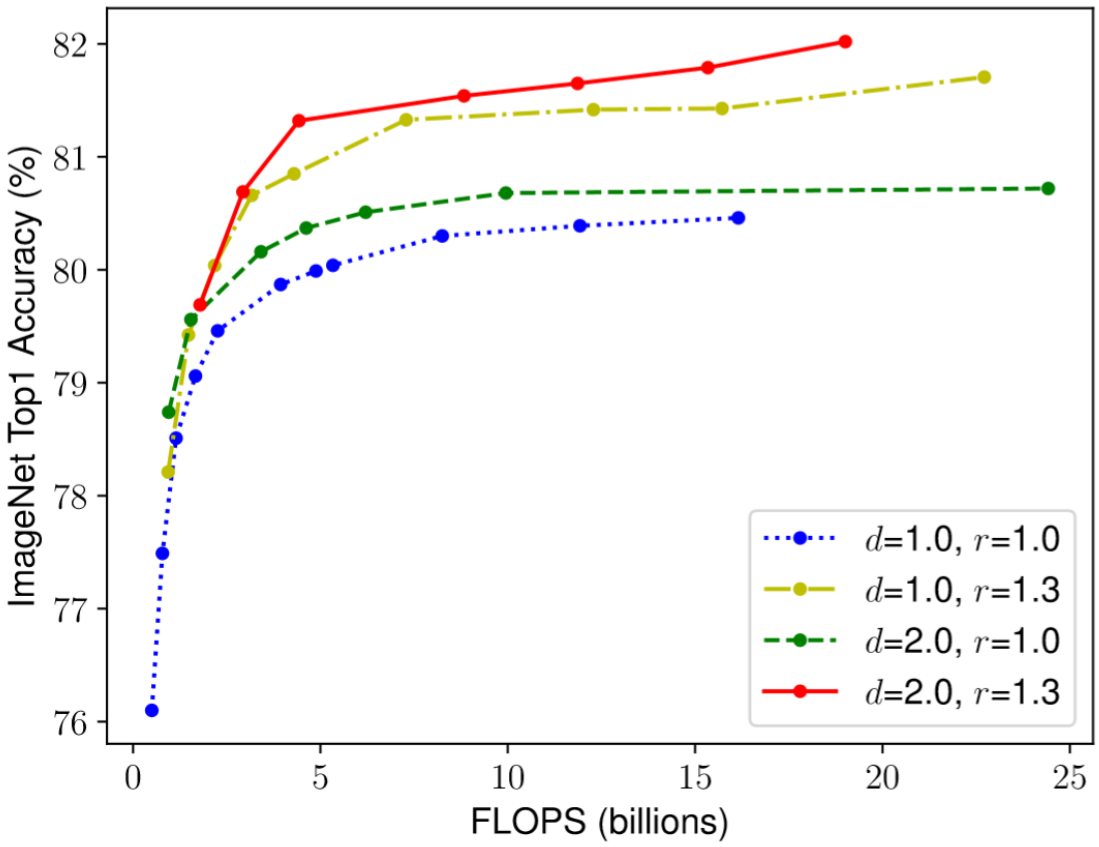
Bileşik Ölçeklendirme Metodu

Birçok ESA eğitimi denemelerinde; derinlik genişlik ve çözünürlük boyutlarının birbirine bağımlı olduğu gözlemlendi. Daha yüksek çözünürlüklü görüntüler için ağ derinliğini artırması, böylece daha büyük alıcı alanlar daha büyük görüntülerde daha fazla piksel

içeren benzer özelliklerin yakalanması yaygın bir kullanımdır. Buna uygun olarak, daha ince ayrıntılı desenler yakalanırken yüksek çözünürlüklü görüntüler kullanıldığın için ağ genişliği de arttırılmalıdır. Bu minvalde, derinlik genişlik ve çözünürlük boyutlarından yalnızca birini ölçeklendirme yerine bu üç boyutu bir metoda göre birlikte ölçeklendirmenin daha yüksek doğrulukta sonuçlar vereceği tespit edilmiştir.

Yukarıdaki yaklaşımı doğrulamak için; genişlik ölçeklendirmesi, Şekil 2.9'da görüldüğü üzere farklı derinlik ve çözünürlükteki ağlarda karşılaştırılmıştır. Derinlik ve çözünürlük sabit tutularak sadece genişlik ölçeklendiğinde doğruluğun hemen doyuma ulaşmasının yanı sıra daha derin ve yüksek çözünürlükte genişlik ölçeklendirmesi aynı aritmetik veri işleme zamanında (FLOPS) çok daha yüksek doğrulukta sonuçlandığı görülmüştür.

Bu sonuçlara göre; daha iyi doğruluk ve verimlilik için, ESA ölçeklendirmesi sırasında ağın genişlik derinlik ve çözünürlük boyutlarının üçü de önemlidir. (Tan & Le., 2019)



Şekil 2.9 : Farklı ESA ağlarının karşılaştırılması. (Tan & Le., 2019, s. 4)

2.3.6. Bozukluk (anomali) tespiti

Anomaliler, verilerdeki normal davranış kavramına uymayan kalıplardır. Verilerde kötü niyetli etkinlik, kredi kartı sahtekârlığı, siber saldırı, terörist faaliyet veya sistemin bozulması gibi çeşitli nedenlerle anomaliler ortaya çıkabilir. Ancak anomalilerde ortak özellik, tüm nedenlerin veriyi inceleyen kişi için "ilginç" olmasıdır. Anomalilerin "ilginçliği" veya gerçek yaşamla ilgisi, anomali tespitinin temel bir özelliğidir. Anomali tespiti, gürültü giderme ve gürültü yerleşimi ile ilgili ancak aynı şey değildir. Gürültü, veri analizine engel teşkil eden ve veriyi inceleyecek kişinin yok etmeye çalıştığı bir olgu olarak tanımlanabilir. Gürültü giderme, veriler üzerinde herhangi bir veri analizi yapılmadan önce istenmeyen nesnelerin kaldırılması; gürültü yerleşimi ise anormal gözlemlere karşı istatistiksel bir model tahmininin aşılanması anlamına gelir.

Bir veriyle ilişkilendirilen etiketler, o verinin normal mi yoksa anormal mi olduğunu gösterir. Etiketleme işlemi genellikle bir uzman tarafından elle (manuel olarak) yapılır. Bu nedenle etiketli eğitim veri setini elde etmek zor ve masraflıdır. Genellikle, olası tüm anormal davranışları kapsayan etiketli verileri elde etmek normal davranışlı verileri elde etmekten daha zordur. Dahası, anormal davranış doğada genellikle dinamiktir, örneğin etiketlenmiş eğitim verisi olmayan yeni tip anomaliler ortaya çıkabilir. Hava trafik güvenliği gibi bazı durumlarda, anormal durumlar felakete dönüşebilir ve felaketler genellikle nadir vuku bulur (Chandola, Banerjee, & Kumar., 2009).

Denetimli anomali tespiti; normal ve anomali sınıfı içeren bir eğitim veri setinin kullanılmasıyla gerçekleştirilir. Bu gibi durumlarda tipik yaklaşım, normal ve anomali sınıfları için bir tahmin modeli oluşturmaktır. Yeni bir veri örneğinin hangi sınıfa ait olduğunu belirlemek için yeni veri, model ile karşılaştırılır. Denetimli anomali tespitinde ortaya çıkan iki ana sorun vardır. İlk olarak, eğitim verilerindeki anormal örnekler, normal örneklere kıyasla çok daha azdır. Dengesiz sınıf dağılımları nedeniyle ortaya çıkan sorunlar veri madenciliği ve makine öğrenimi literatüründe ele alınmıştır (Chawla, Japkowicz, & Kołcz, 2004). İkincisi, özellikle anomali davranışlar için doğru ve isabetli etiketlerin elde edilmesi zordur. Etiketli bir eğitim veri seti elde etmek için normal bir veri setine yapay anomaliler enjekte eden bir dizi teknik önerilmiştir (Steinwart, Hush, & Scove, 2005).

2.4. Yardımcı Araçlar

Derin öğrenme çalışmaları; verimin arttırılması, erişilebilirliğin ve takip edilebilirliğin kolaylaştırılması amacıyla birçok araçla desteklenir. Görselleştirme ile model eğitiminin veriminin arttırılması ve takip edilebilirliğinin kolaylaştırılması hedeflenmekte; mobil uygulama ile modelin erişilebilirliğinin arttırılması hedeflenmektedir.

2.4.1. Görselleştirme

Eğitim süreci; eğitim etkileyen hiperparametrelerin kaydının tutulması ve en iyi hiperparametrelerin belirlenmesi, eğitim sürecinin adım adım izlenmesi ve bu sayede model eğitiminin anlaşılması, kötü eğitimin sebebinin bulunması amacıyla görselleştirilir. F. Hohman ve ark. (Hohman, Kahng, Pienta, & Chau, 2019), yapılan görselleştirme çalışmalarını kapsamlı bir şekilde derlemişlerdir.

2.4.2. Mobil ortamda derin öğrenme

Mobil ortam; hastalığı teşhis edebilecek modele kolay erişimi sağlarken, imkânları sebebiyle de küçük bir ağa sahip modeli bünyesinde barındırabiliyor. Mobil cihazların yaygın olması, eğitilmiş modeller mobile yüklendiği takdirde, modellerin erişilebilirliğini oldukça arttırmaktadır. Örnek vermek gerekirse; ses tanıma sistemi olarak Alexa ve Siri, görüntü işleme olarak ApplePhotoAlbum sıralanabilir. Bir görevi gerçekleştirmek adına oluşturulmuş model, eğitildiği ağın hacmi doğrultusunda mobil cihazda çalıştırılabiliyor. En büyük kısıt olarak araştırmacıların önüne çıkan bu kısıt, sadeleştirilmiş modellerle, küçültülmüş ağlar ile aşılmaktadır. Bu çalışmalara örnek olarak TensorFlow lite verilebilir (Ran, Chen, Liu, & Chen, 2017).

TensorFlow Lite (TensorFlow, 2017), Google tarafından mobil, IoT (Nesnelerin İnterneti) ve gömülü cihazlarda derin öğrenme modellerini eğitmek için geliştirilen bir platformdur. TensorFlow Lite kütüphanesinin sağladığı başlıca avantajlar şöyle sıralanabilir:

- Mobil ortamlardaki kısıtlı kapasite sorununun çözümü için daha az yer kaplar.
- Mobil cihazların işlemcilerinde hızlı çalışabilmesi için optimize edilmiştir.
- Verilerin cihazdan ayrılması gerekmediği için gizliliğin artırılması.

- İnternet bağlantısı gerektirmediğinden güç tüketimini azaltır.
- Sunucuyla iletişim olmadığından gecikme süresini azaltır.

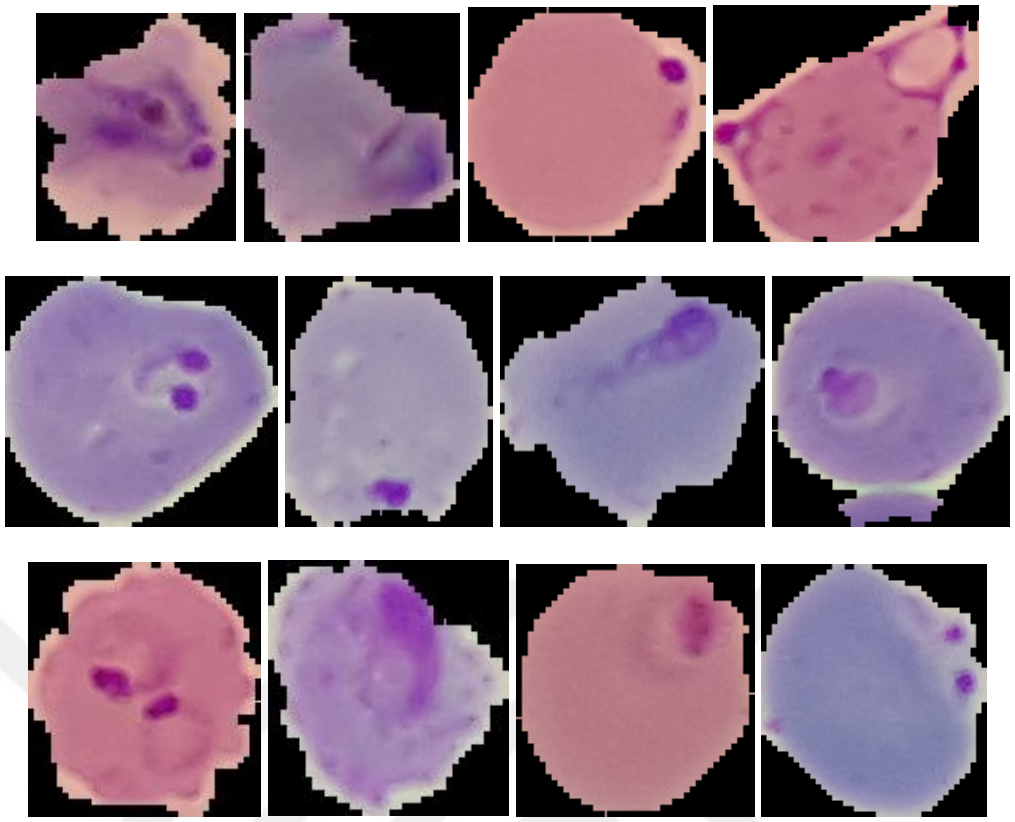
TensorFlow Lite, Python, Java, Swift ve C ++ gibi farklı programlama dilleri için sunduğu API desteği ile mobil cihazlara uygulanabilir. TensorFlow ile eğitilen bir derin öğrenme modelinin mobil cihazlarda kullanılabilmesi için; ilk olarak eğitilen model TensorFlow Lite dönüştürücüsü ile ".tflite" formatına dönüştürülür. Daha sonra derin öğrenme modeli TensorFlow Lite yorumlayıcısı kullanılarak mobil cihazlara dağıtılır. Son olarak model bellek tüketimi ve doğruluğunu optimize etme çalışması yapılabilir.

BÖLÜM 3. UYGULAMA

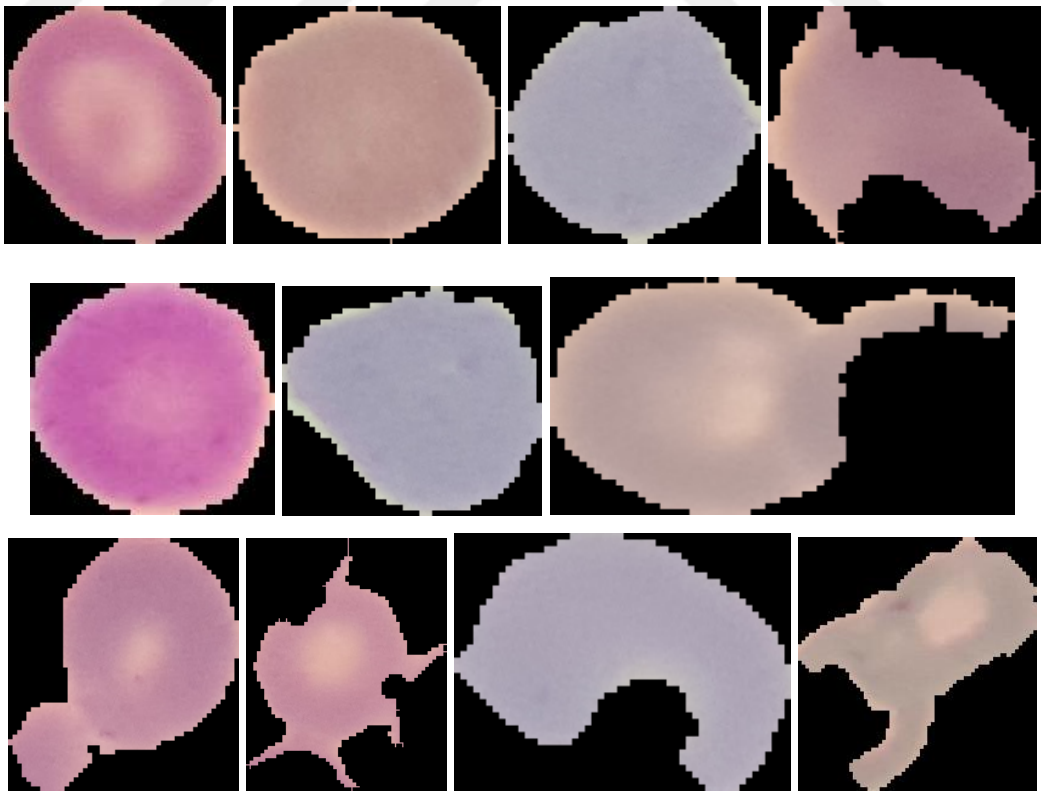
3.1. Veri Seti

Dünya Sıtma Raporu 2018'e göre, 2017 yılında küresel olarak yaklaşık 219 milyon sıtma vakası (tam olarak 203-262 milyon arası) ve 435.000 sıtma ölümü olmuştur. 2010'dan bu yana sıtma vakalarında ve ölüm oranlarında % 28 oranında bir düşüş olmuştur. Tüm sıtma ölümlerinin % 93'ünün gerçekleştiği DSÖ Afrika Bölgesinde en trajik hadise, tüm ölümlerin % 61'ini 5 yaşından küçük çocukların oluşturmasıydı (WHO, World malaria report, 2017).

Araştırmalar sonucu, parazitli kırmızı kan hücreleri görüntülerinden sıtma hastalığı teşhisine imkân veren, kapsamlı ve kaliteli verileri içeren tek veri setinin UTK'nın bünyesinde hazırlanan veri seti olduğu görülmüştür (Jaeger, 2020). Bu çalışmada sıtma hastalığı için kullanılan veri seti için; UTK'nın bir parçası olan Lister Hill Ulusal Biyomedikal İletişim Merkezi'ndeki araştırmacılar, veri eksikliğini azaltmak ve hastalık teşhislerinin doğruluğunu arttırmak için, ışık mikroskobuna bağlanmış akıllı telefonlarda çalışabilecek bir uygulama geliştirmiştir. 150 P. falciparum ile enfekte olmuş ve 50 sağlıklı kişiden Giemsa lekeli ince kan yayma örnekleri toplanmış ve Chittagong Tıp Koleji Hastanesinde (Bangladeş) fotoğraflanmıştır. Akıllı telefon kamerası ile her mikroskobik görüntü için örnek görüntüler alınmıştır. Görüntülere, Tayland'daki Mahidol-Oxford Tropikal Tıp Araştırma Birimi'ndeki bir uzman tarafından manuel olarak açıklamalar eklenmiştir. Kırmızı kan hücrelerini tespit etmek ve ayırmak için seviye-set bazlı bir algoritma uygulanmıştır. Veri seti; eşit sayıda enfekte ve sağlıklı hücre örneklerinden toplam 27,558 hücre görüntüsü içerir. Enfekte ve sağlıklı hücre görüntü sınıfları için hasta kimliği ile hücre eşlemelerini içeren CSV dosyaları hazırlanmıştır. Enfekte sınıfın CSV dosyası 151 hasta girişi içerir (Rajaraman, Jaeger, & Antani, 2019). Veri setindeki sağlıklı ve enfekte kan hücresi örnekleri sırasıyla Şekil 3.1 ve Şekil 3.2'deki gibidir.



Şekil 3.1 : Enfekte kan hücresi örnekleri.



Şekil 3.2 : Sağlıklı kan hücresi örnekleri.

3.2. EfficientNet ile Sıtma Teşhisi

Son yıllarda yapılan uluslararası yarışmalar; EfficientNet modelinin çeşitli versiyonlarının, deri üzerindeki lezyonların (Melanoma) teşhisinde ve sınıflandırılmasında sıkça kullandığını ve yüksek başarımlar elde edildiğini gösteriyor (ISIC 2019, 2020).

Yapılan araştırmada EfficientNet modelinin, Melanoma veri setine göre daha temiz olan sıtma veri seti üzerinde sadece birkaç çalışmada kullanıldığı görüldü. Sıtma veri setinin daha az karmaşık ve temiz hücre görüntülerinden oluşması sebebiyle EfficientNet ile çok daha iyi sonuçlar elde edilebileceği düşünüldüğü için bu çalışmada sıtma veri seti üzerine yoğunlaşılmıştır.

Sıtma hastalığının teşhisi için UTK veri setinde bulunan kırmızı kan hücrelerinde bozukluk tespitine odaklanılmıştır. Bu amaçla, EfficientNet ESA modelleri kullanılarak öğrenim aktarımı yöntemi ile yeni modeller oluşturulmuştur. Bu yeni modeller ile eğitilen ağlar teşhis için kullanılmıştır.

Bu çalışmanın erken dönemlerinde öğrenim aktarımı yöntemi kullanılmadan eğitimler yapılmaya çalışıldı. Ancak sıfırdan oluşturulan modellerin çok derin olması, kullanılan kaynak ihtiyacını arttırdı ve performansı düşürdü. Bu sebeplerden ötürü çalışmada öğrenim aktarımı yöntemi tercih edilmiştir.

3.2.1. Model eğitim ortamının hazırlanması

EfficientNet kullanarak sıtma veri setindeki enfekte olmuş hücreler tespit edilmeye çalışıldı. Model eğitimleri ve diğer geliştirmeler için Google Colab ve Kaggle gibi çevrimiçi ortamlar denendikten sonra, daha hızlı geliştirme yapılabildiği ve geliştirmeler esnasında hata ayıklama imkânı sağladığı için Lenovo Legion Y530 (NVIDIA GTX 1050) modeli bir dizüstü bilgisayarda karar kılındı. Bütün ESA eğitimleri Python programlama dili ile gerçekleştirildi, kullanılan önemli temel kütüphaneler arasında Keras ve Tensorflow sayılabilir. Eğitim ortamının diğer özellikleri Tablo 3.1'de görülmektedir.

Tablo 3.1 : Eğitim ortamı özellikleri.

Eğitim Ortamı	Özellikler
Ekran Kartı	NVIDIA GeForce GTX 1050
İşletim Sistemi	Windows 10
Geliştirme Ortamı	PyCharm 2020.2 (Professional Edition)
Model Paketleri	EfficientNet 1.1.0
	Tensorflow 2.1.0
Kütüphaneler	SciKit Learn 0.23.1
	Numpy 1.18.5

3.2.2. Eğitim modelinin oluşturulması

Yukarıda da belirtildiği gibi veri seti 27.558 hücre görüntüsü içerir. Veri setinden rastgele seçilen 500 tane sağlıklı ve 500 tane sıtmalı hücre görüntüsü, proje klasörü altında "/ veri_seti / test" dizininde konumlandırılmıştır. Diğer hücre görüntüleri proje klasörünün altında "/ veri_seti / train" dizininde; enfekte hücre görüntüleri "Parasitized" klasöründe, sağlıklı hücre görüntüleri "Uninfected" klasöründe olacak şekilde konumlandırılmıştır.

Görsellerin dosya sisteminden çekilip oluşturulacak derin öğrenme modeline aktarılması için; Keras'ın görsel veri türetimini sağlayan "ImageDataGenerator" metodu ile bir nesne oluşturulur (Kod 3.1). Metodun; "validation_split" parametresi ile veri setinin ne kadarının doğrulama için ayrılacağı belirlenir, "rescale" parametresine 1/255 değeri verilerek 0 ile 255 arasında olan renk katsayı değerleri (RGB) 0 ile 1 arasına çekilmek suretiyle normalleştirme işlemi yapılmış olur.

```
data_gen = ImageDataGenerator(rescale=1 / 255., validation_split=0.10)
```

Kod 3.1 : Görsel veri türetici nesne oluşturma.

Oluşturulan nesneden çağırılan "flow_from_directory" metodu ile görseller, derin öğrenme modeline girdi olarak verilebilecek bir sayısal diziye dönüşmüş olur (Kod 3.2). "directory" parametresine eğitim veri setinin bulunduğu dizin girilir. Veri seti "Parasitized" ve "Uninfected" olmak üzere iki sınıftan oluştuğu için "class_mode" parametresi "binary" olarak girilir. "batch_size" parametresi ile modele eğitim esnasında bir seferde kaç görselin verileceği ayarlanır. Görsellerin en boy oranları "target_size" ile

belirlenir. Girdi verisinin doğrulama mı yoksa eğitim için mi olduğu "subset" parametresi ile ayarlanır.

```
input_train = data_gen.flow_from_directory(
    directory="dataset/train",
    class_mode="binary",
    batch_size=16,
    target_size=(224, 224),
    subset="training"
)
```

Kod 3.2 : Görsellerin sayısal diziye dönüştürme.

Aynı nesneden "flow_from_directory" metodu bir kez daha çağırılarak modelin doğrulama girdisi oluşturulur. Bu işlem için sadece "subset" parametresi "validation" olarak değiştirilir.

Eğitim ve doğrulama girdi nesnelerinin oluşturulması gibi test girdi nesneleri de benzer şekilde oluşturulur (Kod 3.3).

```
test_datagen = ImageDataGenerator(rescale=1. / 255)
input_test = test_datagen.flow_from_directory(
    directory= "dataset/test",
    target_size=(224, 224),
    batch_size=1,
    class_mode='binary',
    shuffle=False
)
```

Kod 3.3 : Test girdilerinin ayarlanması.

Derin öğrenme modeli dizisinin ilk modeli olan "efficient_net" nesnesi tanımlanır (Kod 3.4). Tanımlama işlemi için "efficientnet" kütüphanesiden EfficientNet modellerinden birinin methodu çağırılır. "weights" parametresi, 14 milyondan fazla görsel içeren ImageNet verileri üzerindeki eğitimin çıktısı olan ağırlıkları almak için "imagenet" olarak girilir, böylece öğrenme aktarımı (Transfer Learning) sağlanmış olur. Modelin girdi nesnelerini oluşturulurken ayarlanan görsellerin en boy oranları "input_shape" parametresinin ilk iki elemanını oluşturur, 3. eleman ise görsellerin kaç renk kanalından oluştuğunu gösterir. Tüm girdi-çıktıların birbirine bağlı olduğu tam bağlı katmanın modele eklenmesi "include_top" parametresi ile ayarlanır. Örnekleme katmanında en yüksek değeri almak için "pooling" parametresi "max" olarak ayarlanır.

```
efficient_net = efficientnet.model.EfficientNetB3 (
    weights='imagenet',
    input_shape=(224, 224, 3),
    include_top=False,
    pooling='max',
    **kwargs
)
```

Kod 3.4 : EfficientNet nesnesinin tanımlanması.

Keras'ın "Sequential" metodu, katmanlardan oluşan dizi parametreleri ile çağırılarak model oluşturulur (Kod 3.5). İlk katman önceden oluşturulmuş olan "efficient_net" nesnesidir. Model, ikili sınıflandırma yapacağı için son katmanı bir düğümlü, aktivasyon fonksiyonu "sigmoid" olan katmandan oluşur. Sigmoid fonksiyonu, ikili sınıflandırmalarda en iyi performansı sağladığı için tercih edilmiştir. Yukarıda bahsedildiği gibi eğitimdeki aşırı öğrenmeyi engellemek için seyreltme katmanı eklenmiştir.

```
model = Sequential([
    efficient_net,
    layers.Dropout(0.1),
    layers.Dense(units=1, activation='sigmoid')
])
```

Kod 3.5 : Modelin oluşturulması.

Eğitimi başlatmadan önceki son adım olarak model derlenir (Kod 3.6). Derleme metodu, optimizasyon metodu, kayıp fonksiyonu ve ölçüt parametreleri girilerek çağırılır.

```
model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)
```

Kod 3.6 : Modelin derlenmesi.

Model eğitimi "fit" metodu ile başlatılır (Kod 3.7). "fit" metodunun ilk parametresi önceden hazırlanmış eğitim girdisi olan "input_train" nesnesidir. "epochs" parametresi eğitimin kaç döngüden oluşacağını, "steps_per_epoch" parametresi her döngündeki adım sayısını, "validation_steps" parametresi ise her döngüdeki doğrulama adım sayısını belirlemek için ayarlanır. "validation_data" parametresi doğrulama girdisi olarak hazırlanmış "input_val" nesnesini alır. Eğitimin tamamlanmasından sonra "fit"

metodunun döndürdüğü “history” nesnesi, modelin girdilerini, ölçümlerini ve ağırlıklarını içerir.

```
history = model.fit(
    input_train,
    epochs=40,
    steps_per_epoch=10,
    validation_data=input_val,
    validation_steps=10
)
```

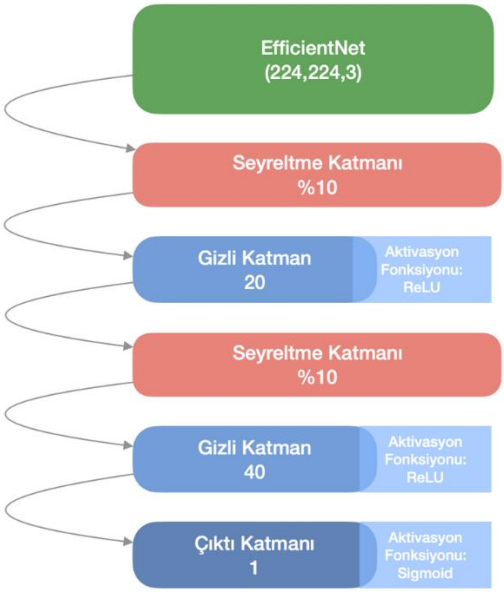
Kod 3.7 : Eğitimin başlatılması.

Eğitilmiş modelin test için ayrılan veriler üzerinde çalıştırıp tahminlerini görmek için “predict” metodu kullanılır, parametre olarak önceden hazırlanmış test girdi nesnesi “input_test” verilir (Kod 3.8). Metot, her test görseli için 0 ile 1 arasında tahminlerin bulunduğu bir dizi döndürür. Karışıklık matrisini elde edebilmek için dizinin değerleri 0 ve 1’e yuvarlanır. Yuvarlanmış değerlerden oluşan dizi “confusion_matrix” metoduna verilerek karışıklık matrisi elde edilir, matrisin web ortamında gösterimi 4. Bölümdeki “Değerlendirme” başlığı altında Şekil 4.5’da gösterilmiştir.

```
predicts = model.predict(input_test)
rounded_predicts = np.round(predicts, 0)
print('Confusion Matrix')
cm = confusion_matrix(input_test.classes, rounded_predicts)
```

Kod 3.8 : Test tahminlerinin yapılıp, karışıklık matrisinin üretilmesi.

EfficientNet-B2 ile oluşturulan öğrenme aktarmalı modelin katmanlarının yapısı Şekil 3.3’deki gibidir.

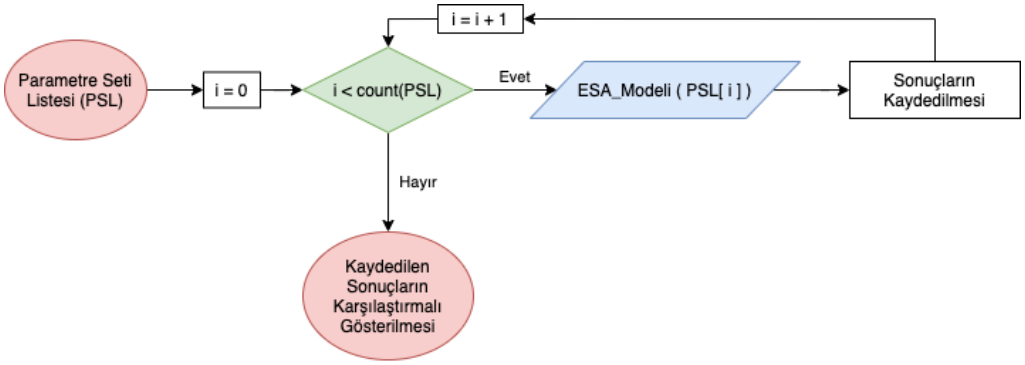


Şekil 3.3 : Model katman yapısı.

3.2.3. Model eğitim sürecinin iyileştirilmesi

Yapılacak eğitim denemelerinin otomatikleştirilmesi için; Python dili ile kodlanılan modül, parametrelerle (epoch sayısı, girdi boyutu, katmanlar vs. gibi) döngüsel olarak çalışabilecek şekilde tekrar kodlandı. Bu sayede her parametre değişikliğinde eğitimi tekrar tekrar çalıştırmak yerine, modüle girdi olarak verilen parametre setleri ile tek seferde onlarca farklı deneme yapılması sağlandı.

Geliştirilen bu modül, aşağıda teferruatlı anlatılmış olan kayıt tutma modülü ile birlikte kullanılarak uçtan uca otomatikleştirilmiş (Şekil 3.4) derin öğrenme aracı oluşturuldu.

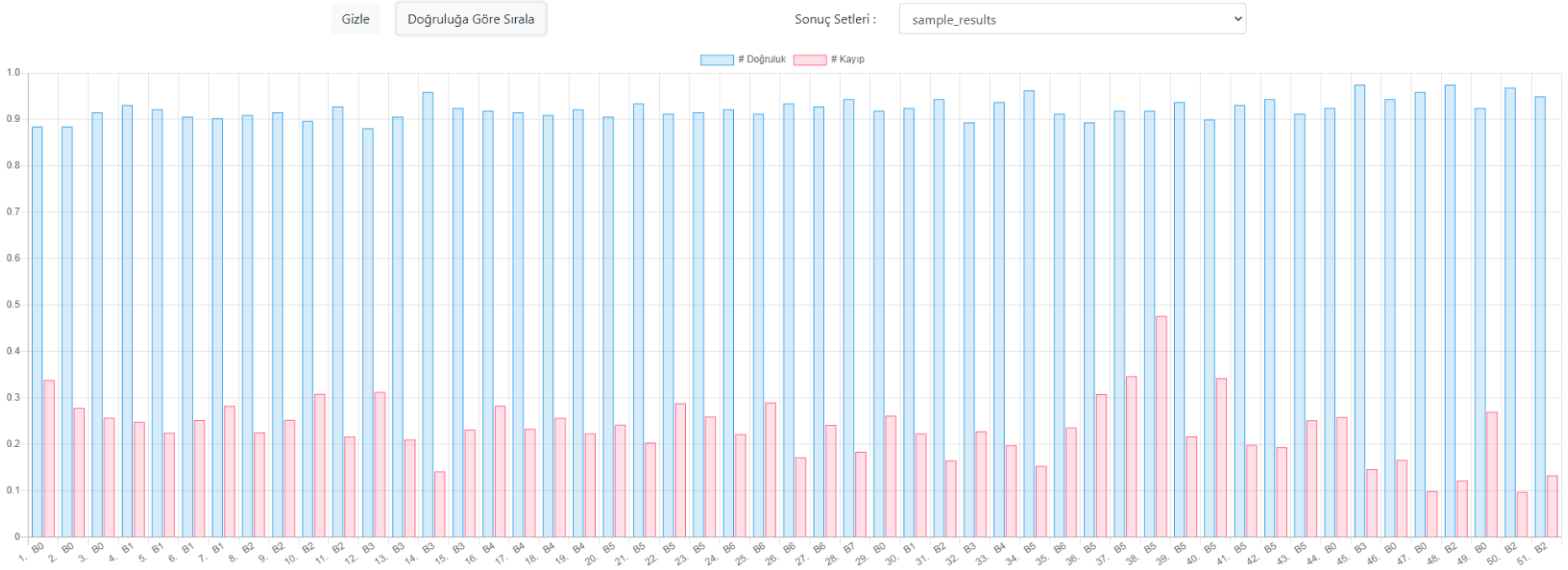


Şekil 3.4 : Uçtan uca otomatikleştirilmiş derin öğrenme aracı.

3.2.4. Kayıt tutma ve görselleştirme modülü

Model eğitimleri için birçok deneme yapıldı. Yapılan deneme sonuçlarını kıyaslamak için mevcut kayıt tutma sistemleri incelendi, ancak kayıtların özetini grafiklerle göstermek, detayları sade bir şekilde listelemek ve kayıt silme gibi gereksinimlere cevap verebilecek uygun bir araç bulunamadı. Bu yüzden ihtiyaçları karşılayabilecek bir kayıt tutma modülü (Yavuz M. , 2020) geliştirildi. Web uygulaması şeklinde geliştirilen kayıt tutma modülü ile verilerin json dosyasına kaydedilmesi ve bu kayıtların HTML, JS, VueJS, CSS ve Python teknolojileri kullanılarak sade pratik bir şekilde gösterilmesi sağlandı.

Denemelerin doğruluk ve kayıp sonuçları, kıyaslamayı kolaylaştıracak yan yana çubuklar halinde tek bir grafikte gösterildi (Şekil 3.5). "Doğruluğa Göre Sırala" butonuna tıklandığında sonuçların doğruluğa göre düşükten yükseğe doğru sıralanması sağlandı, bu özellik sayesinde hangi eğitimden yüksek doğrulukta sonuçlar alındığı kolaylıkla seçilebilir hale geldi. Sonuçların sıralanmış halini Şekil 4.1'de görebilirsiniz. Çubukların alt kısmında denemenin hangi EfficientNet modeli ile gerçekleştirildiği gösterildi. Farklı parametre seti sonuçlarını göstermek için "Sonuç Setleri" seçeneği eklendi. İmleç ile grafikteki çubukların üzerine gelindiğinde denemeye ait doğrulama veya kaybın tam olarak değeri imlecin yanında beliren bir kutucukta gösterildi, böylece çubuklara tıklamadan tam doğruluk ve kayıp değerlerinin öğrenilmesi sağlandı.



Şekil 3.5 : Eğitim sonuçlarının çubuk grafik olarak gösterimi.

Şekil 3.6 : Eğitim girdi ve çıktılarının detaylı gösterimi.

Şekil 3.5'deki çubuklara tıklandığında, denemeye ait "Çözünürlük", "Giriş Ağırlıkları", "Doğrulama Seti", "Katmanlar", "optimizer", "loss", "metrics", "epochs", "steps_per_epoch", "validation_steps" gibi girdi parametreleri ile yapılan eğitim sonucunda elde edilen "acc" (Doğruluk), "val_acc" (Doğrulama Doğruluğu), "loss" (Hata, Kayıp), "validation loss" (Doğrulama Kaybı), "Karışıklık Matrisi" ve "Sınıflandırma Raporu" çıktıları detaylı olarak grafiğin altında listelendi (Şekil 3.6). Bunun yanında her bir eğitim için eğitim boyunca her döngü (epoch) sonunda elde edilen "Doğruluk", "Doğrulama Doğruluğu", "Kayıp" ve "Doğrulama Kaybı" değerlerini gösteren çizgi grafikleri de girdiler ve çıktıların üzerinde gösterildi. Sağ üst kısma sağ ve sol yön okları butonlar eklendi ve butonlar klavyedeki yön tuşlarına bağlandı, bu butonlar sayesinde denemeler arasındaki geçişler kolaylaştırıldı. İstenilmeyen denemelerin silinmesi için bir buton eklendi. Şekil 3.5'deki "Gizle" butonu ile aynı şekildeki büyük

çubuk grafiği gizlendi ve ekranda sadece Şekil 3.6'deki bilgilerin kalması sağladı, bu sayede tıklanılan denemeye ait girdi ve çıktı bilgilerine odaklanılması sağlandı.

3.2.4.1. Kayıt tutma

Eğitim tamamlandıktan sonra "model" nesnesinin barındırdığı verilerle birlikte kıyaslama ve iyileştirme için ihtiyaç duyulan diğer veriler Kod 3.9'da gösterildiği gibi hazırlanıp bir değişkene json formatında atanır. Daha sonra, değişkendeki json verileri dosya sistemine kaydedilir.

```
params = model.history.params
history = model.history.history
results = {
    "acc": [str(round(x, 4)) for x in history['accuracy']],
    "val_acc": [str(round(x, 4)) for x in history['val_accuracy']],
    "loss": [str(round(x, 4)) for x in history['loss']],
    "val_loss": [str(round(x, 4)) for x in history['val_loss']]
}

layers = []
for ly in model.get_config()["layers"]:
    layers.append({
        "name": "" if "name" not in ly["config"] else ly["config"]["name"],
        "units": "" if "units" not in ly["config"] else
ly["config"]["units"],
        "use_bias": "" if "use_bias" not in ly["config"] else
ly["config"]["use_bias"],
        "activation": "" if "activation" not in ly["config"] else
ly["config"]["activation"],
        "trainable": "" if "trainable" not in ly["config"] else
ly["config"]["trainable"],
    })
log = {
    "date": now_str,
    "history": results,
    "input_shape": model.input_shape,
    "layers": layers,
    "params": params,
    "params_count": f"{model.count_params():,d}",
    "loss": model.loss,
    "optimizer": {
        "type": type(model.optimizer).__name__,
        "lr": str(kb.eval(model.optimizer.lr))
    },
    "others": others
}
```

Kod 3.9 : Kaydı tutulacak verilerin hazırlanması.

3.2.4.2. Görselleştirme

Tarayıcılarda kullanıcı tarafında geliştirme yapmak için tasarlanmış olan VueJs kütüphanesi, hızlı çalışması ve pratik geliştirme sağlamasından ötürü tercih edilmiştir. VueJs kullanılarak HTML arayüz kontrolünün sağlandığı kodlar açık kaynak olarak GitHub'a yüklenmiştir (Yavuz M. , 2020).

"Vue" sınıfından oluşturulan "app" nesnesinin "mount" metodunda, istenilen kayıt setini barındıran dosyaya istek yapılarak eğitim kayıtlarının bulunduğu json verisi döndürülmüş olur. Döndürülen veri dizisinden, her eğitimin ayrı ayrı etiket, doğruluk ve kayıp değerleri yeni dizilere atanır. Elde edilen dizilerle "Char.js" kütüphanesinden çekilen "Chart" sınıfı kullanılarak, eğitim setindeki bütün eğitimlerin doğruluk ve kayıp sonuçlarını gösteren çubuk grafik oluşturulmuş olur (Şekil 3.5).

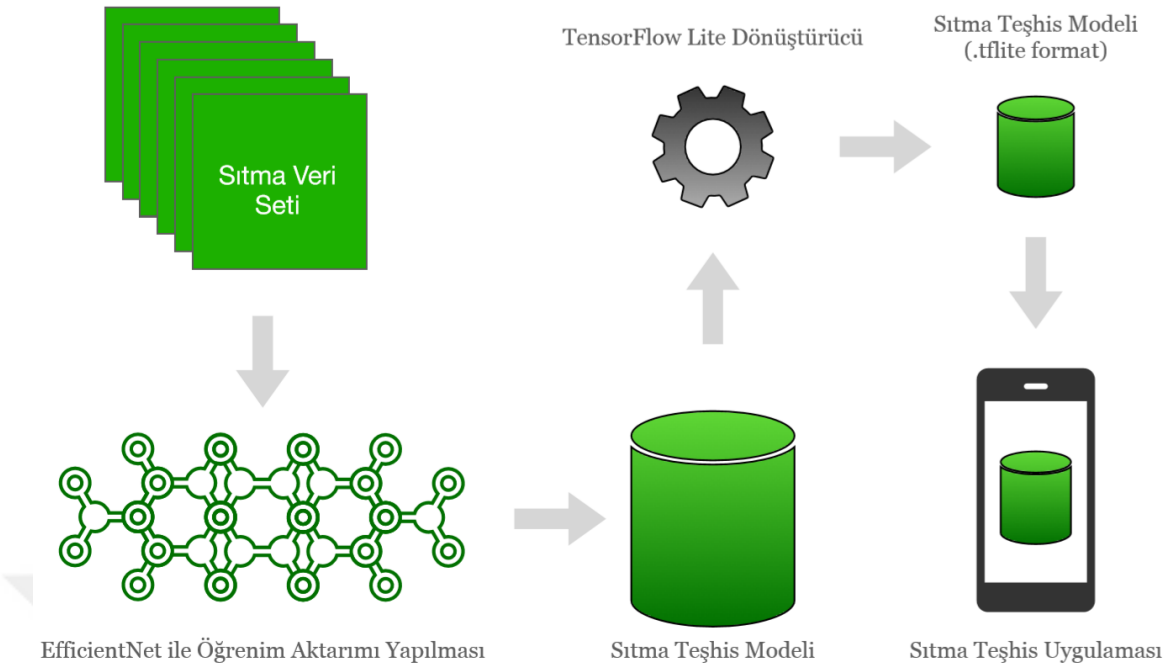
Grafikteki çubuklara tıklatıldığında tıklatılan eğitime ait detayların gösterilmesi için; "chart_main" adıyla oluşturulan "canvas" elamanının "onclick" fonksiyonu oluşturulur. Tıklanılan noktadaki eğitimin veri dizisindeki dizin değeri (index) kullanılarak "list" dizisinden istenilen eğitim çekilir ve "selected_train" değişkenine atanır. VueJs değişkenlerinin HTML kodları arasında şablon sözdizimi (template syntax) olarak kullanımı (Kod 3.10) sayesinde, "selected_train" değişkenindeki veriler pratik bir şekilde çağırılarak sayfada gösterilir (Yavuz M. , 2020).

```
<div id="app">{{ message }}</div>
```

Kod 3.10 : VueJs şablon sözdizimi örneği.

3.3. Mobil Uygulama

Bu çalışmanın son aşaması olan mobil uygulamanın geliştirilmesinden önce, EfficientNet modellerinden öğrenme aktarımı yapılarak oluşturulan yeni ESA modellerinin, sıtma veri seti ile eğitilmesi ile sıtma teşhis modeli oluşturuldu. Oluşturulan modeller, TensorFlow Lite dönüştürücüleri kullanılarak mobil cihazlarda çalışabilecek ".tflite" dosya formatına dönüştürüldü. Bu sayede modeller mobil ortamlarda çalışabilecek şekilde hafif ve hızlı hale getirilmiş oldu. Dönüştürülmüş modeller kullanılarak mobil bir uygulama geliştirildi. Bu sayede uygulamanın, test amaçlı ayrılan kırmızı kan hücresi görselleri üzerinde tahminler yapması sağlandı (Şekil 3.7).

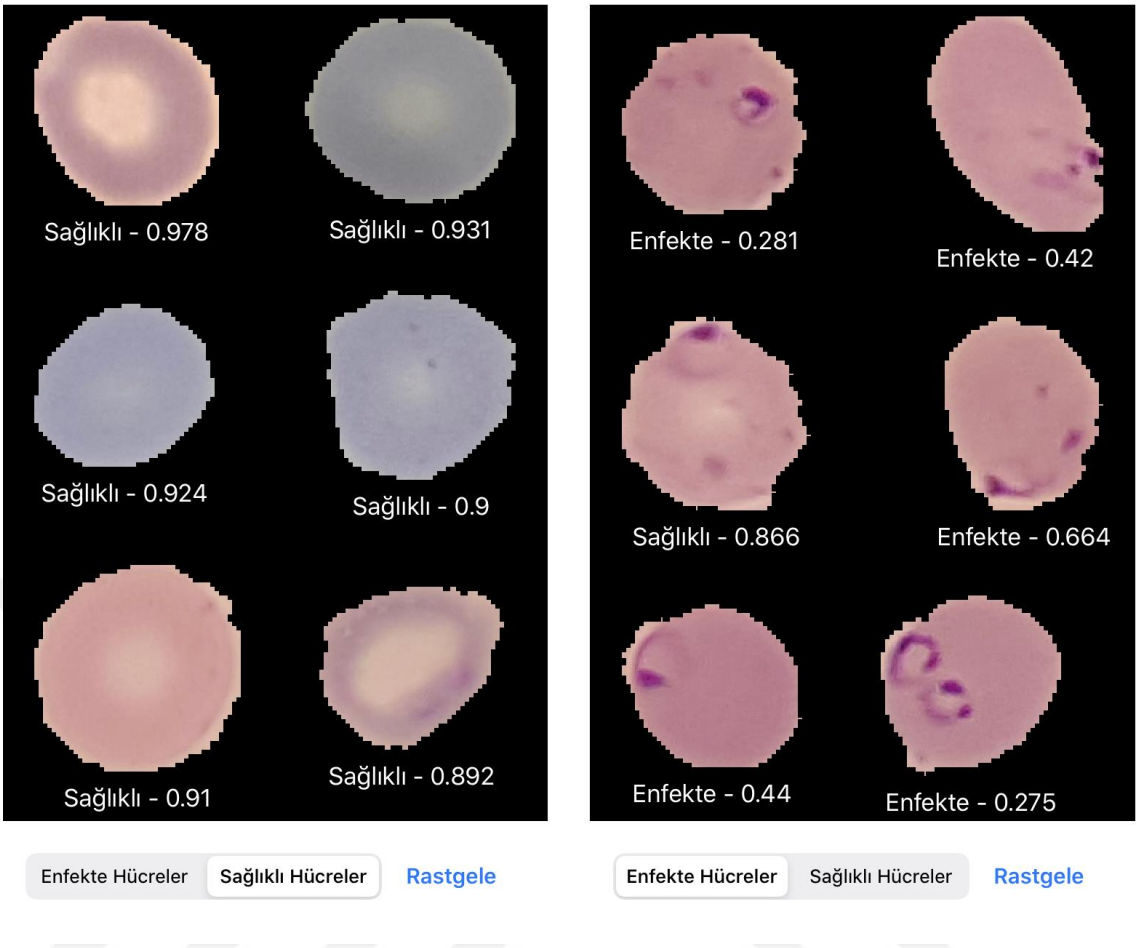


Şekil 3.7: Sıtma modelinin dönüştürülerek mobil ortamda çalıştırılması.

Modelin eğitimi sürecinde ayrılan 500 adet sağlıklı ve 500 adet enfekte olmak üzere 1000 adet görsel, test edilmek için uygulamaya geliştirilme aşamasında projeye yüklenmiştir. Uygulama, arka planda çalışan TensorFlow Lite yorumlayıcısı ile girdi olarak verilen görsellerdeki hücrelerin sağlıklı/enfekte olma tahminini yaparak ekranda gösterir.

Şekil 3.8'de görüldüğü üzere uygulamadaki "Enfekte Hücreler / Sağlıklı Hücreler" geçiş düğmesine tıklanarak enfekte ve sağlıklı hücre görüntüleri arasında geçiş yapılabilir. Kullanıcılar ekrandaki "Rastgele" butonuna tıkladığında, yüklü olan görsellerden 6 tanesi rastgele seçilip TensorFlow Lite yorumlayıcısına gönderilir ve tahminler her bir hücre görüntüsünün altında oranlarıyla birlikte gösterilir.

TensorFlow Lite yorumlayıcısı görsellere 0 ile 1 arasında bir tahmin değeri verir. Eşik değeri olarak belirlenen 0.8'ten küçük olan tahminler "Enfekte", 0.8'ten büyük olanlar "Sağlıklı" olarak değerlendirilir.



Şekil 3.8 : Mobil uygulama ekran görüntüleri.

Uygulama, macOS Catalina işletim sisteminde Xcode 11.6 derleyicisi kullanılarak Swift programlama dili ile yazılmıştır, iOS işletim sistemli mobil cihazlarda çalışmaktadır. Uygulamanın temel işlevlerini sağlayan kod bloğu Kod A.1'deki gibidir. Bu kod bloğunda ekranda görsellerin rastgele çağırılması ile sağlıklı ve enfekte hücre görüntüleri arasında geçiş sağlanır. Ayrıca her bir görsel için Tensorflow Lite yorumlayıcısını kullanarak eğitilmiş modelin görseller için tahminlerini oluşturup görsellerin altında eklendiği kısımdır.

3.4. Bulgular

32x32, 64x64, 96x96 gibi düşük çözünürlükteki görsellerle eğitilen modellerin Tensorflow Lite ile mobil ortama uygun formata dönüştürüldüğünde tahmin doğruluğunda düşüş tespit edilmiştir. Problemin sebebi araştırıldığında; Tensorflow Lite dönüştürücüsünün, modelin mobil cihazlarda daha hızlı çalışabilmesi ve daha az yer

kaplaması amacıyla model boyutunu küçültmesinin, doğruluk değerlerinde düşüşe sebep olduğu görülmüştür.

Problemin çözümü için, modeller daha yüksek çözünürlüklü görseller ile eğitildiğinde, mobile dönüştürülen modelin tahmin doğruluğundaki düşüşün düşük çözünürlükte eğitilen modellere göre daha az olduğu görüldü.

Bunun yanında çözünürlüğü arttırmanın, eğitim sürecinde grafik işlemcisinde tutulan bellek alanını arttırdığı ve yüksek sayıda parametre içeren büyük EfficientNet modelleri ile eğitilmek istendiğinde bellekte oluşan taşmalardan dolayı model eğitimlerinin başarısız olduğu görüldü.

Yapılan araştırmalarda bu problemin çözümü için model girdilerinden "batch_size" parametresinin düşürülmesinin etkili olduğu görülmüştür. Denemelerde 8'e kadar düşürülen "batch_size" parametresi ile her bir eğitim döngüsünde modele verilen görsel girdi sayısının düşürülmesi, EfficientNet-B0, B1, B2 ve B3 modelleri için eğitimi mümkün kılmıştır. Ancak EfficientNet-B4 ve daha büyük modeller için çok daha yüksek grafik işlemci kapasitesine ihtiyaç duyulduğu tespit edilmiştir.

BÖLÜM 4. TARTIŞMA VE SONUÇ

4.1. Eğitim Sonuçları

Eğitimlerde alınan en yüksek doğruluk değeri "2020_08_04__15_54_16" kayıt numaralı denemeye ait olmasına rağmen diğer değerlendirme kriterleri göz önünde bulundurulduğunda test performansı en iyi olan "2020_08_06__11_58_53" kayıt numaralı deneme, bu çalışmanın nihai sonucu olarak seçilmiştir. Bu denemede 224 x 224 çözünürlüğünde görseller girdi olarak kullanılmıştır. Eğitimin yapıldığı modelin yapısı; EfficienNet-B2, 1 seyreltme katmanı, 20 düğümden oluşan DDÜ aktivasyon fonksiyonlu 1 gizli katman ve Sigmoid aktivasyon fonksiyonu katmanlarından oluşur. Eğitim 40 dönemde (epoch) tamamlanmıştır, her dönem 10 eğitim adımı ve 10 doğrulama adımından oluşur. Optimizasyon fonksiyonu olarak "Adam" kullanılmıştır. Eğitim doğrulaması için veri setinin %10'u ayrılmıştır. Veri setinin eğitim, doğrulama ve test için bölümlendirilmesi Tablo 4.1'deki gibidir.

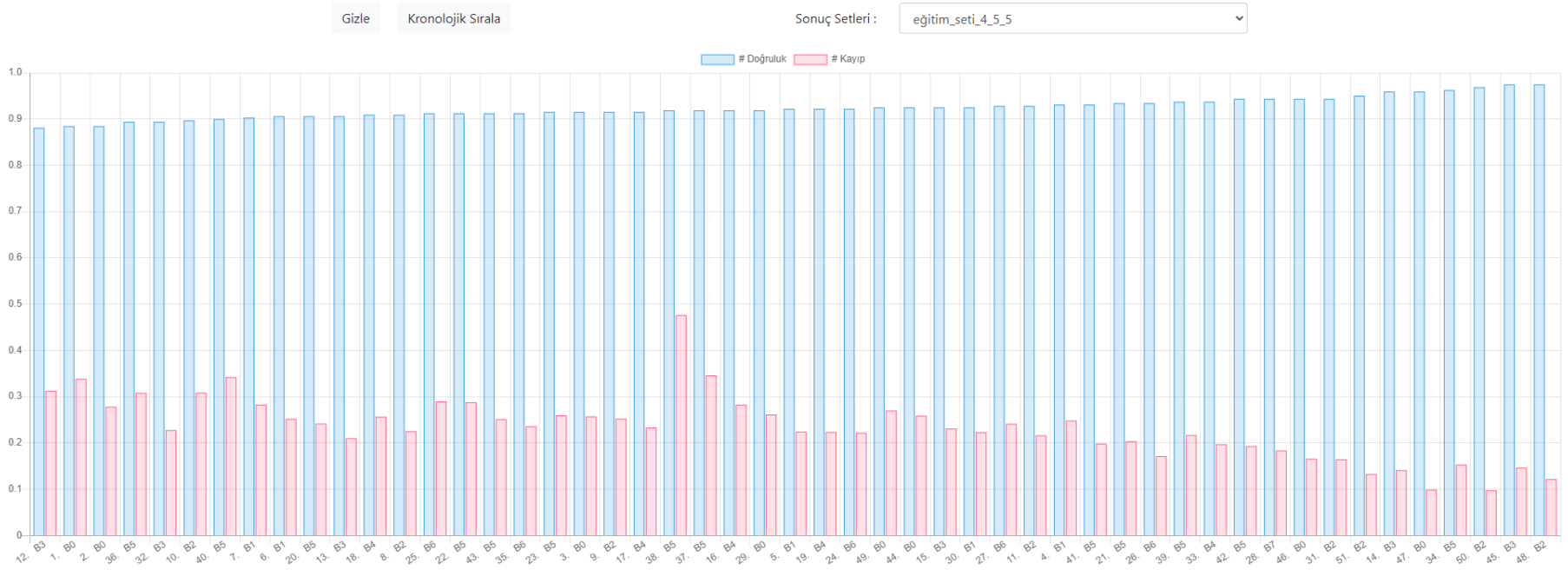
Tablo 4.1 : Veri setinin bölümlendirilmesi.

	Enfekte Hücre Sayısı	Sağlıklı Hücre Sayısı	Toplam
Eğitim	11951	11951	23902
Doğrulama	1328	1328	2656
Test	500	500	1000
Toplam	13779	13779	27558

Çalışmalar neticesinde en yüksek başarımların elde edildiği eğitimlerin sıralamış biçimde bulunduğu deneme seti grafiği Şekil 4.1'deki gibidir. Bu setteki eğitim doğruluk değeri en yüksek ilk 10 eğitimin hangi EfficientNet modeli ile eğitildiği bilgisiyle birlikte doğruluk, doğrulama doğruluğu, kayıp ve doğrulama kaybı sonuçları Tablo 4.2'de yer almaktadır.

Tablo 4.2 : Doğruluk değeri en yüksek ilk 10 eğitim.

Eğitim Kayıt Numarası	EfficientNet Modeli	Doğruluk	Doğrulama Doğrulu	Kayıp	Doğrulama Kaybı
2020_08_04__15_54_16	B2	0.975	0.975	0.121	0.098
2020_07_20__17_07_30	B3	0.975	0.9	0.146	0.256
2020_08_06__11_58_53	B2	0.969	0.969	0.097	0.118
2020_07_16__14_18_45	B5	0.963	0.944	0.153	0.19
2020_08_04__15_45_20	B0	0.959	0.912	0.099	0.223
2020_06_17__17_54_21	B3	0.959	0.925	0.141	0.301
2020_08_06__13_55_55	B2	0.95	0.925	0.133	0.205
2020_07_16__14_00_56	B2	0.944	0.963	0.165	0.102
2020_08_04__12_00_46	B0	0.944	0.888	0.166	0.308
2020_06_18__10_39_14	B7	0.945	0.913	0.183	0.463



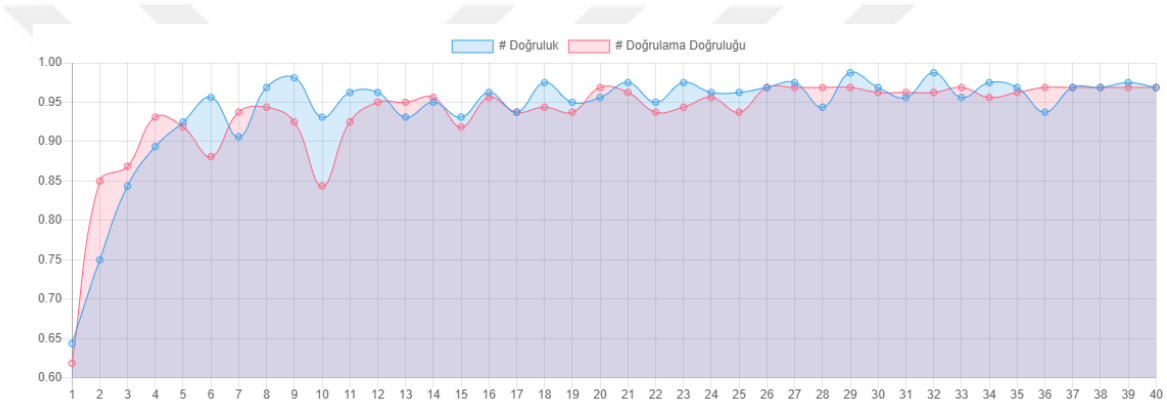
Şekil 4.1 : Eğitim sonuçları.

Modelin eğitimi sonucunda elde edilen doğruluk, doğrulama doğrulu, kayıp ve doğrulama kaybı oranları Tablo 4.3'deki gibidir.

Tablo 4.3 : "2020_08_06__11_58_53" numaralı eğitimin sonuçları.

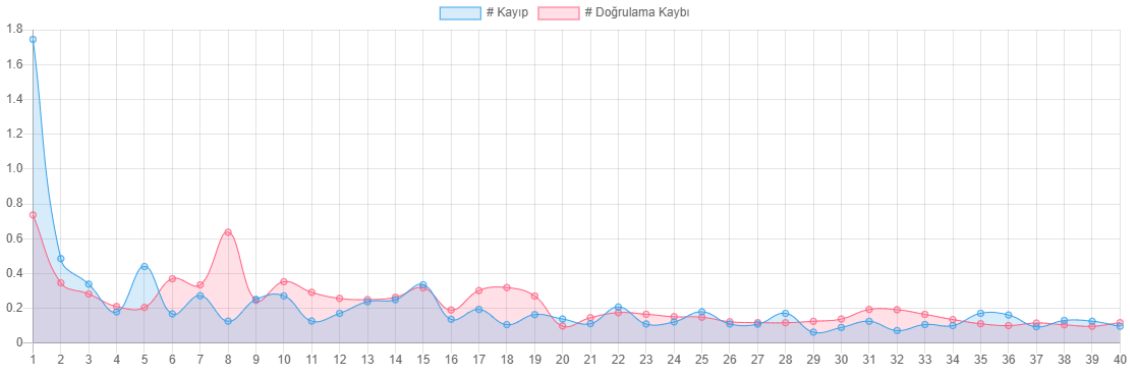
Doğruluk	Doğrulama Doğrulu	Kayıp	Doğrulama Kaybı
0.969	0.969	0.097	0.118

40 döngüden oluşan bu eğitim süresince elde edilen "Doğruluk" ve "Doğrulama Doğrulu" değerleri Şekil 4.2'deki grafikteki gibidir.



Şekil 4.2 : Eğitimin "Doğruluk" ve "Doğrulama Doğrulu" grafiği.

Aynı eğitimin "Kayıp" ve "Doğrulama Kaybı" değerleri Şekil 4.3'daki grafikte gösterilmiştir.



Şekil 4.3 : Eğitimin "Kayıp" ve "Doğrulama Kaybı" grafiği.

4.2. Değerlendirme Kriterleri

4.2.1. Karışıklık matrisi

Karışıklık matrisi, gerçek değerlerin bilinmekte olduğu bir dizi test verisi üzerinde, bir sınıflandırma modelinin performansını değerlendirmek için sıklıkla kullanılan bir tablodur. 100 adet "1" ve 100 adet "0" değerinden oluşan 200 elemanlı bir test verisinin ikili sınıflandırma için karışıklık matrisi örneği Şekil 4.4'deki gibidir.

	Tahmin edilen değer: 1	Tahmin edilen değer: 0	Toplam
Gerçek değer: 1	TP = 90	FN = 10	100
Gerçek değer: 0	FP = 15	TN = 85	100
Toplam	105	95	200

Şekil 4.4 : Karışıklık matrisi örneği.

TP (True Positive) : Gerçek değerin 1 olduğu gibi tahmin edilen değerin de 1 olduğu durumlardır. Örnek matriste görüldüğü üzere tahmin sonucunda 90 adet "1" değerli veri doğru tahmin edilmiştir.

TN (True Negative) : Gerçek değerin 0 olduğu gibi tahmin edilen değerin de 0 olduğu durumlardır. Örnek matriste görüldüğü üzere tahmin sonucunda 85 adet "0" değerli veri doğru tahmin edilmiştir.

FP (False Positive) : Gerçek değerin 0 ancak tahmin edilen değerin 1 olduğu durumlardır. Örnek matriste görüldüğü üzere tahmin sonucunda 15 adet "0" değerli veri yanlış (1) tahmin edilmiştir.

FN (False Negative) : Gerçek değerin 1 ancak tahmin edilen değerin 0 olduğu durumlardır. Örnek matriste görüldüğü üzere tahmin sonucunda 10 adet "1" değerli veri yanlış (0) tahmin edilmiştir.

Karışıklık matrisinden hesaplanan bazı oranlar, TP, TN, FP ve FN'nin birbirleriyle ilişkisini gösterir. Bu terminolojiler denklem 4.1'deki gibidir. Örnek matrise göre hesaplamalar yapıldığında "Toplam" değerinin 200, "Gerçek Pozitifler" değerinin 100 ve "Gerçek Negatifler" değerinin de 100 olduğu görülür.

$$\begin{aligned} Toplam &= TP + TN + FP + FN \\ Gerçek\ Pozitifler &= TP + FN \\ Gerçek\ Negatifler &= TN + FP \end{aligned} \tag{4.1}$$

Doğruluk (Accuracy) : Genel olarak, sınıflayıcının ne sıklıkta doğru tahmin ettiğinin bir ölçüsüdür (Denklem 4.2). "Doğruluk" ölçütü, örnek matrise göre denklem 4.2 ile hesaplandığında 0.87 olarak bulunur.

$$Doğruluk = \frac{TP + TN}{TOPLAM} \tag{4.2}$$

4.2.2. Duyarlılık (recall):

Sınıflayıcının ne kadar gerçek pozitif ya da gerçek negatif değeri doğru tahmin ettiğinin bir ölçüsüdür. İsabet oranı veya hatırlama olarak da bilinir, mümkün olduğu kadar yüksek olmalıdır (Denklem 4.3). "Duyarlılık" ölçütü gerçek pozitif değerler için örnek matrise göre denklem 4.3 ile hesaplandığında 0.90 olarak bulunur.

$$Duyarlılık = \frac{TP}{Gerçek\ Pozitifler} \tag{4.3}$$

4.2.3. Hassasiyet (precision):

Tüm sınıflardan, doğru ve yanlış olarak ne kadar tahmin edildiğinin bir ölçüsüdür. Mümkün olduğu kadar yüksek olmalıdır. Pozitif Tahmin Edici Değer (Positive Predictive Value) olarak da bilinir (Denklem 4.4). "Hassasiyet" ölçütü, örnek matrise göre denklem 4.4 ile hesaplandığında 0.86 olarak bulunur.

$$Hassasiyet = \frac{TP}{TP + FP} \tag{4.4}$$

4.2.4. F-1 puanı (f-1 score):

Duyarlılık (recall) ve hassasiyetin (precision) harmonik ortalamasıdır. Sınıflandırıcının ne kadar iyi performans gösterdiğinin bir ölçüsüdür ve sınıflandırıcıları karşılaştırmakta sıklıkla kullanılır (Denklem 4.5). "F-1 Puanı" ölçütü, örnek matrise göre denklem 4.5 ile hesaplandığında 0.88 olarak bulunur.

$$F1 = 2\,x\,\frac{Hassasiyet\ x\ Duyarlılık}{Hassasiyet + Duyarlılık} \tag{4.5}$$

4.3. Değerlendirme

Şekil 4.5'deki karışıklık matrisinde de gözlemlendiği gibi, "2020_08_06__11_58_53" kayıt numaralı denemede eğitilen modelin testi neticesinde TP, FN, FP ve FN değerleri sırasıyla 488, 12, 26 ve 474 olarak bulunmuştur.

Karışıklık Matrisi (Confusion Matrix)		Tahmin Edilen	
		Enfekte	Sağlıklı
Gerçek	Enfekte	488	12
	Sağlıklı	26	474

Şekil 4.5 : EfficientNet-B2 ile eğitilen modelin karışıklık matrisi.

Modelin performansının görüldüğü sınıflandırma raporu Tablo 4.4'deki gibidir. "Enfekte" sınıfı için değerlendirme kriterleri; hassasiyet 0.949, duyarlılık 0.976 ve F1 puanı 0.963 olarak bulunmuştur. "Sağlıklı" sınıfı için kriterler; hassasiyet 0.975, duyarlılık 0.948 ve F1 puanı 0.962'dir.

Tablo 4.4 : EfficientNet-B2 ile eğitilen modelin sınıflandırma raporu tablosu.

Sınıflandırma Raporu	Hassasiyet	Duyarlılık	F1	Adet
Enfekte	0.9494	0.976	0.9625	500
Sağlıklı	0.9753	0.948	0.9615	500
Doğruluk	-	-	0.962	-

4.4. Kayıt Tutma Sistemi

Kayıt tutma sisteminde kayıtlar, birden fazla ESA eğitimi için oluşturulan parametre setlerinin denenmesi sonucu oluşur. Denemeler neticesinde alınan "Doğruluk" ve "Kayıp" sonuçları tek bir çubuk grafiğinde yan yana gösterildiği gibi grafikteki çubuklara tıklatıldığında grafiğin altında deneme ile ilgili bütün ayrıntıların gösterilmesi eğitim performanslarının kıyaslanmasında ve analizinde büyük kolaylık sağlar.

Toplu eğitim denemelerinin sonuçlarının basit grafik görselleştirmeleri ile analiz edilebilmesi ihtiyacına yönelik geliştirilen kayıt tutma sistemi, özellikle ESA modelleri ile çalışan diğer araştırmacılar istifadesine sunulmak üzere çevrimiçi ortama açık kaynak olarak yüklenmiştir (Yavuz M. , 2020).

4.5. Tartışma

Derin öğrenme son zamanlarda, donanım performansının artması ve önceki senelere nazaran maliyetlerin azalması sebebiyle çokça kullanılmaya başlanmış, kullanım alanı genişlemiştir. Tıp alanında ise; teşhis süresini ve insan kaynaklı hataları azaltması gibi faydalar sebebiyle önem kazanmıştır. Bu alanda, görüntü işlemede elde edilen verimi sebebiyle derin öğrenmenin bir dalı olan ESA'na sıklıkla başvurulur.

Son zamanlarda derin öğrenmenin dezavantajlarından biri olan uzun eğitim sürelerinin önüne geçmek üzere öğrenim aktarımı sıklıkla kullanılmaktadır. Eğitilen bir model başka bir veri seti ile tekrar eğitime tabi tutularak daha verimli bir süreç içerisinde amaca uygun model elde edilmektedir.

Önceki bölümlerde verilen tafsilatlı anlatımlardan sonra yukarıda verilen kısa açıklamalar ışığında, bu çalışmada yapılan başarılar ve katkılar madde madde şu şekilde sıralanabilir:

- Sıtma teşhisi için UTK sıtma veri seti üzerinde bir ESA çeşidi olan EfficientNet modelleri ile öğrenim aktarmı yapılarak modeller eğitilmiştir.
- Eğitim sonucunda; %96,1 F-1 puanına ve %96,2 doğruluk oranına sahip sıtma teşhis modeli oluşturulmuştur.
- Eğitim sonuçları ve model performansı, UTK veri setini kullanan diğer çalışmalar ile karşılaştırılmıştır. (Tablo 4.5)
- Sıtma teşhisinde; erken tanıyı mümkün kılabilecek bir mobil uygulama geliştirilmiştir. Bu mobil uygulama ile oluşturulan sıtma teşhis modelinin erişilebilirliği arttırılmıştır.
- ESA model eğitimi özelinde tüm derin öğrenme model eğitimlerinde kullanılabilecek, model geliştirme ve iyileştirme süreçlerini hızlandıran, eğitim sonuçlarının analizini kolaylaştıran bir web uygulaması geliştirilmiştir.

Parazitli kırmızı kan hücreleri görüntülerinden sıtma teşhis modeli oluşturmada, EfficientNet kullanılan ilk akademik çalışma olmuştur. EfficientNet modelleri, UTK veri seti üzerinde kullanılarak öğrenim aktarımı yapılmıştır. Modelin kendi içerisinde % 96,1 performansa sahip olması bu çalışmanın başarılı olduğunu, Tablo 4.5'e bakarak sıtma teşhisi bağlamında incelendiğinde ise gelecek vaat eden bir çalışma olduğunu göstermiştir. Bu çıkarımlar sonucunda yapılan çalışmanın, EfficientNet modellerinin tıp alanında kullanıma uygun olduğu ve gelecek çalışmalara ışık tutacağı değerlendirilmektedir.

Sıtma hastalığının ve sıtma hastası için erken tanı konmasının önemine önceki bölümlerde değinilmiştir. Bu bağlamda, teşhisin hızlı yapılabilmesi adına geliştirilen mobil uygulama ile oluşturulan sıtma teşhis modelinin erişilebilirliği arttırılmıştır. Mobil uygulama ile herhangi bir altyapı ve yazılım kurulumu gerekmeden sıtma teşhis modeli kullanılabilmektedir. Sıtma teşhisi amacıyla ESA modellerinin mobilde çalıştırılması üzerine yapılan diğer akademik çalışmalardan farklı olarak ilk kez bu çalışmada EfficientNet modellerinden öğrenim aktarımı yöntemi ile oluşturulan modeller kullanılmıştır.

Bu çalışmanın kolaylaştırılması adına geliştirilen web uygulaması; model eğitiminin izlenebilirliğini, görselliğini arttırması, model eğitim parametreleri ve model eğitim sonuçlarını kayıt altına alması gibi özellikleriyle ESA ve diğer derin öğrenme model eğitimlerini kolaylaştırması beklenmektedir. Geliştirilen web uygulaması kaynak kodları; açık kaynak topluğunu da desteklemek adına internet üzerinde depolama servisi veren bir web sitesinde araştırmacıların istifadesine sunulmuştur.

Tablo 4.5 : Sıtma veri seti ile yapılan diğer çalışmalarla karşılaştırma.

Yazar ve Çalışma Yılı	ESA Modeli	Öğrenim Aktarımı	Model Eğitim Doğruluğu	Model Doğrulama Doğruluğu	Model Eğitim Kayıp	Model Doğrulama Kayıp	Model Performans F1 Puanı	Model Performans Doğruluğu
Mehmet Yavuz, 2020 (Bu çalışma)	**EfficientNet**	**Var**	**% 96.9**	**% 96.9**	**% 9.7**	**% 11.8**	**% 96.3**	**% 96.2**
(Soner Can Kalkan, Ozgur Koray Sahingoz, 2019)	Özel ESA	Yok	%97	%95	%7	%16	-	-
(Reddy & Juliet, 2019)	ResNet-50	Yok	%95.91	%95.4	%11.34	%13.01	-	-
(Quan, Wang, & Liu, 2020)	Özel ESA	-	-	-	-	-	%97.7	%97.47
(Yang, ve diğerleri, 2019)	Özel ESA	Yok	-	-	-	-	%93.46	-
(Rajaraman, Jaeger, & Antani, 2019)	VGG-19	Var	-	-	-	-	%99.31	%99.32

4.6. Sonuç

Bu çalışmada ilk defa, parazitli kırmızı kan hücreleri görüntülerinden sıtma hastalığının teşhisi için EfficientNet modelleri ile öğrenim aktarımı yapılarak enfekte / sağlıklı kan hücrelerinin tasnifini yapabilecek modeller oluşturulmuştur. Eğitim doğruluğu % 97 ve performansı % 96'lara varan sıtma teşhis modeli, EfficientNet'in sıtma hastalığı özelinde hastalık teşhisinde tıp alanında kullanılabileceği ortaya konulmuştur. Model eğitim veriminin iyileştirilmesi, performansının ve incelenebilirliğinin arttırılması adına web uygulaması geliştirilmiştir. Bu uygulama ile sıtma teşhis model eğitimi desteklenmiş, sonuçlar ve iyi sonuç veren parametreler kaydedilmiştir. Son olarak, sıtma teşhisi için geliştirilen modelin daha etkili olması için erişilebilirliği, geliştirilen mobil uygulama ile arttırılmıştır. Mobil uygulama ile model, mobil cihazlarda kullanıma hazır hale getirilmiştir.

4.7. Gelecek Çalışmalar

Birçok kan hücresinin bulunduğu mikroskop görüntülerinde segmentasyon yaparak kırmızı kan hücrelerini algılayan bir derin öğrenme modeli geliştirilip, bu tez kapsamında geliştirilen ESA modeli ile birlikte çalışması sağlanabilir. Geliştirilecek bu iki modelden oluşan çalışma, bu tezde kullanılan yöntemlerle mobil ortamda çalışacak bir uygulama haline getirilebilir. Geliştirilecek mobil uygulamanın bulunduğu cihazın mikroskopa bağlandığı, sıtma veri seti görüntüleri toplanırken kullanılan düzenek gibi, bir düzenek kurulabilir. Böylece alınan kan örneğindeki kırmızı kan hücrelerinin tek tek analizi yapıldığı gibi toplu olarak sayımı da yapılabilecek ve kan örneğindeki sıtma paraziti bulunduran kırmızı kan hücresi sayısı/yoğunluğu tespit edilebilecektir. Bu sayede hastalardan alınan kan örneklerinin, mikroskoplara bağlı mobil cihazlarla çekilen görüntüleri, yine aynı mobil cihazlarla saniyeler içinde analiz edilebilecektir. Böyle bir çalışma yapıldığı takdirde, hücre görüntüleri alınır alınmaz enfekte / sağlıklı tahminin yapıldığı bir sistem geliştirilmiş olacak ve sıtma teşhisi son derece pratik bir hale gelecektir.

KAYNAKLAR

A, V., & B, R. K. (2020). Deep learning approach to detect malaria from microscopic images. *Multimedia Tools and Applications*, 15297–15317.

Acharya, U. R., Ng, E. Y., Tan, J.-H., & Sree, S. V. (2012). Thermography based breast cancer detection using texture features and support vector machine. *Journal of medical systems* , 1503-1510.

Affonso, C., Ross, A. L., Vieira, F. H., & Carvalho, A. C. (2017). Deep learning for biological image classification. *Expert Systems with Applications*, 114-122.

Affonso, C., Sassi, R. J., & Barreiros, R. M. (2015). Biological image classification using rough-fuzzy artificial neural network. *Expert Systems with Applications*, 9482–9488.

Agarwal, V. (2020, Ağustos 5). *Complete Architectural Details of all EfficientNet Models | by Vardan Agarwal | Towards Data Science*. Towards Data Science: https://towardsdatascience.com/complete-architectural-details-of-all-efficientnet-models-5fd5b736142 adresinden alındı

Akdur, R. (2006). Sıtma ve Sıtma Salgınları Tarihi. *Bilim Tarihi Araştırmaları*.

AL-Allaf, O. N., AbdAlKader, S. A., & Tamimi, A. A. (2013). Pattern Recognition Neural Network for Improving the Performance of Iris Recognition System. *International Journal of Scientific & Engineering Research*, 661-667.

Alpaydın, E. (2014). *Introduction to machine learning.* MIT press.

Ashley, E. A., & White, N. J. (2014). The duration of Plasmodium falciparum infections. *Malaria Journal*.

Bengio, Y. (2009). *Learning Deep Architectures for AI.* Now Foundations and Trends.

Bringsjord, S., & Govindarajulu, N. S. (2018, July 12). *Artificial Intelligence*. The Stanford Encyclopedia of Philosophy : https://plato.stanford.edu/entries/artificial-intelligence/ adresinden alındı

Cao, W., Yuan, J., He, Z., Zhang, Z., & He, Z. (2018). Fast Deep Neural Networks With Knowledge Guided Training and Predicted Regions of Interests for Real-Time Video Object Detection. *IEEE Acceess*, 8990–8999.

Chandola, V., Banerjee, A., & Kumar., V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 1-58.

Charpe, K., Bairagi, V., Desarda, S., & Barshikar, S. (2015). A Novel Method for Automatic Detection of Malaria Parasite Stage in Microscopic Blood Image. *International Journal of Computer Applications*.

Chawla, N. V., Japkowicz, N., & Kołcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, 1-6.

Chen, X., Xiang, S., Liu, C.-L., & Pan, C.-H. (2014). Vehicle detection in satellite images by hybrid deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters*, 1797-1801.

Dara, S., & Tumma, P. (2018). Feature Extraction By Using Deep Learning: A Survey. *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (s. 1795-1801). IEEE.

Deng, L. (2014). A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing*.

Deng, L., Hinton, G., & Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: an overview. *international conference on acoustics, speech and signal processing* (s. 8599-8603). Vancouver, Canada: IEEE.

Deniz, E., Şengür, A., Kadiroğlu, Z., Guo, Y., Bajaj, V., & Budak, Ü. (2018). Transfer learning based histopathologic image classification for breast cancer detection. *Health information science and systems*.

Do, C. B., & Ng., A. Y. (2006). Transfer learning for text classification. *Advances in Neural Information Processing Systems*.

Dong, Y., Jiang, Z., Shen, H., Pan, W. D., Williams, L. A., Reddy, V. V., . . . Bryan, A. W. (2017). Evaluations of deep convolutional neural networks for automatic identification of malaria infected cells. *IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*.

Ferreira, A., & Giraldi, G. (2017). Convolutional Neural Network approaches to granite tiles classification. *Expert Systems with Applications*, 1-11 .

Fu, Y. (2020, Ağustos 1). *Image classification via fine-tuning with EfficientNet*. Keras: https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/ adresinden alındı

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *International Conference on Artificial Intelligence and Statistics.*, (s. 315-323).

Gundersen, O. E., Gil, Y., & Aha, D. W. (2018). On reproducible AI: Towards reproducible research, open science, and digital scholarship in AI publications. *Ai Magazine*, 56-68.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences* , 1-12.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Conference on Computer Vision and Pattern Recognition (CVPR)* (s. 770–778). IEEE.

He, T., Mao, H., Guo, J., & Yi, Z. (2017). Cell tracking using deep neural networks with multi- task learning. *Image and Vision Computing*, 11.

Hohman, F., Kahng, M., Pienta, R., & Chau, D. H. (2019). Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 2674 - 2693.

Huang, G., Liu, Z., Maaten, L. V., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *Conference on Computer Vision and Pattern Recognition (CVPR)* (s. 4700–4708). IEEE.

Hubel, D. H., & Wiesel., T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 106.

Hung, J., & Carpenter, A. (2017). Applying Faster R-CNN for Object Detection on Malaria Images. *IEEE conference on computer vision and pattern recognition workshops* (s. 5). IEEE.

ISIC 2019. (2020, Nisan 1). The International Skin Imaging Collaboration: https://challenge2019.isic-archive.com/leaderboard.html adresinden alındı

Jaeger, S. (2020, Mayıs 20). *Malaria Datasets | National Library of Medicine*. Lister Hill National Center for Biomedical Communications: https://lhncbc.nlm.nih.gov/publication/pub9932 adresinden alındı

K., M., G., M., & B., G. (2003). The reliability of blood film examination for malaria at the peripheral health unit. *Ethiopian Journal of Health Development*, 8.

Kaplan, A., & Haenlein, M. (2019). Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence. *Business Horizons*, 15-25.

Kim, S., Choi, Y., & Lee, M. (2015). Deep learning with support vector data description. *Neurocomputing*, 111-117.

Kleene, S. C. (1956). *Representation of Events in Nerve Nets and Finite Automata.* Princeton University Press.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 1097-1105.

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 436–444.

Lecun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 396-404.

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE* , 2278-2324.

Lei, H., Han, T., Huang, W., Kuo, J. Y., Yu, Z., He, X., & Lei, B. (2017). Cross-Modal Transfer Learning for HEp-2 Cell Classification Based on Deep Residual Network. *IEEE International Symposium on Multimedia (ISM)* (s. 465–468). IEEE.

London, S., Bradski, G., Coates, A., Deng, L., & Shah, M. (2017, July 7). *Ask the AI experts: What's driving today's progress in AI?* mckinsey: https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/ask-the-ai-experts-whats-driving-todays-progress-in-ai# adresinden alındı

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. icml*.

MahdiehPoostchi, Silamut, K., J.Maude, R., Jaeger, S., & Thoma, G. (2018). Image analysis and machine learning for detecting malaria. *Elsevier Translational Research* , 19.

Mali, S., Kachur, S. P., & Arguin, P. M. (2012). Malaria Surveillance — United States, 2010. *Morbidity and Mortality Weekly Report: Surveillance Summaries*, 17.

Maloof, M. A. (2006). *Machine learning and data mining for computer security: methods and applications.* Springer.

McCorduck, P. (2004). *Machines Who Think.* New York: CRC Press.

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*.

Norlander, R., Grahn, J., & Maki, A. (2015). Wooden knot detection using convnet transfer learning. *Scandinavian Conference on Image Analysis*, 263-274.

Öztemel, E. (2003). *Yapay sinir ağlari.* Istanbul: Papatya Yayincilik.

Pan, S. J., & Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 1345-1359.

Poole, D., Mackworth, A., & Goebel, R. (1998). *Computational Intelligence: A Logical Approach.* New York: Oxford University Press.

pubmed.ncbi.nlm.nih.gov. (2020, 07 10). nih.gov: https://pubmed.ncbi.nlm.nih.gov adresinden alındı

Quan, Q., Wang, J., & Liu, L. (2020). An Effective Convolutional Neural Network for Classifying Red Blood Cells in Malaria Diseases. *Interdisciplinary Sciences: Computational Life Sciences*, 217–225.

Rajaraman, S., Jaeger, S., & Antani, S. K. (2019). Performance evaluation of deep neural ensembles toward malaria parasite detection in thin-blood smear images. *PeerJ 7:e6977*, 16.

Ran, X., Chen, H., Liu, Z., & Chen, J. (2017). Delivering Deep Learning to Mobile Devices via Offloading. *VR/AR Network '17: Proceedings of the Workshop on Virtual Reality and Augmented Reality* (s. 5). LA: Association for Computing Machinery.

Reddy, A. S., & Juliet, D. S. (2019). Transfer Learning with ResNet-50 for Malaria Cell-Image Classification. *International Conference on Communication and Signal Processing.* IEEE.

Rosadoa, L., daCosta, J. M., Elias, D., & Cardoso, J. S. (2016). Automated Detection of Malaria Parasites on Thick Blood Smears via Mobile Devices. *Procedia Computer Science*, 138-144.

Rosenblatt, F. F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 386-408.

Shrestha, A., & Mahmood, A. (2019). Review of Deep Learning Algorithms and Architectures. *IEEE Access*, 53040 - 53065.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *Computer Vision and Pattern Recognition*, arXiv preprint arXiv:1409.1556 .

Soner Can Kalkan, Ozgur Koray Sahingoz. (2019). Deep Learning Based Classification of Malaria from Slide Images. *2019 Scientific Meeting on Electrical-Electronics & Biomedical Engineering and Computer Science (EBBT)* (s. 4). Istanbul, Turkey, Turkey: IEEE.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 1929-1958.

Steinwart, I., Hush, D., & Scove, C. (2005). A classification framework for anomaly detection. *Journal of Machine Learning Research*, 211-232.

Tan, M. (2018, Ağustos 7). *MnasNet: Towards Automating the Design of Mobile Machine Learning Models*. Google AI Blog: https://ai.googleblog.com/2018/08/mnasnet-towards-automating-design-of.html adresinden alındı

Tan, M., & Le, Q. V. (2020, Ağustos 5). *Google AI Blog: EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling*. Google AI Blog:

https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html adresinden alındı

Tan, M., & Le., Q. V. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv*.

TensorFlow. (2017). *TensorFlow Lite*. TensorFlow: https://www.tensorflow.org/lite adresinden alındı

Thrun, S., & Pratt, L. (1998). *Learning to Learn.* Springer Science & Business Media.

Tu, J. V. (1996). Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of clinical epidemiology*, 1225-1231.

Werbos, P., & John, P. (tarih yok). Beyond regression: new tools for prediction and analysis in the behavioral sciences. 1974.

WHO. (2017). *World malaria report*. WHO: http:// www.who.int/malaria/publications/world-malaria-report-2017/report/en/ adresinden alındı

WHO. (2020, January 12). *Disease burden of malaria*. World Health Organization: http://www.who.int/mediacentre/factsheets/fs094/en/ adresinden alındı

Wolpert, D. H. (1996). The lack of a priori distinctions between learning algorithms. *Neural computation*, 1341–1390.

Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv*.

Yang, F., Poostchi, M., Yu, H., Zhou, Z., Silamut, K., Yu, J., . . . Antani, S. (2019). Deep Learning for Smartphone-Based Malaria Parasite Detection in Thick Blood Smears. *IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS*, 1427 - 1438.

Yavuz, F. Y., Ünal, D., & Gül, E. (2018). Deep learning for detection of routing attacks in the internet of things. *International Journal of Computational Intelligence Systems*, 39-58.

Yavuz, M. (2020, Temmuz 25). *mehmetyavuz/visual-comparison: A simple visual comparison tool for results of many CNN training.* Github: https://github.com/mehmetyavuz/visual-comparison adresinden alındı

Zhaohui, L., Powell, A., Ersoy, I., Poostchi, M., Silamut, K., Palaniappan, K., . . . Thoma, G. (2016). CNN-based image analysis for malaria diagnosis. *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (s. 4). IEEE.

Zhou, Z.-H., Jiang, Y., Yang, Y.-B., & Chen, S.-F. (2002). Lung cancer cell identification based on artificial neural network ensembles. *Artificial Intelligence in Medicine*, 25-36.

EKLER

EK A: Kodlar

EK A

```
//  Created by mehmet on 19.07.2020.
//  Copyright © 2020 my. All rights reserved.

import UIKit
class CollectionVC: UIViewController, UICollectionViewDataSource, UICollectionViewDelegate
{
    @IBOutlet weak var collectionView: UICollectionView!
    private var result: Result?
    private var modelDataHandler: ModelDataHandler? =
        ModelDataHandler(modelFileInfo: MobileNet.modelInfo, labelsFileInfo:
MobileNet.labelsInfo)
    let reuseIdentifier = "imgCell" // also enter this string as the cell identifier in the storyboard
    let txtParasitized = NSLocalizedString("Parasitized", comment: "")
    let txtUninfected = NSLocalizedString("Uninfected", comment: "")
    var indexes: [Int] = []
    var all_parasitized: [URL] = []
    var all_uninfected: [URL] = []
    var images : [String] = []
    var labels: [String] = []
    var current_type = NSLocalizedString("Parasitized", comment: "")
    fileprivate func setRandomIndexes() {
        indexes = []
        for _ in 0...5 {
            let number = Int.random(in: 0 ... 500)
            indexes.append(number)
        }
    }
    fileprivate func setDataSource() {
        labels = []
        images = []
        for i in indexes {
            var image:UIImage
            if (current_type == txtUninfected) {
                images.append(all_uninfected[i].lastPathComponent)
                image = UIImage(named: all_uninfected[i].lastPathComponent)!
            }
            else {
                images.append(all_parasitized[i].lastPathComponent)
                image = UIImage(named: all_parasitized[i].lastPathComponent)!
            }
            labels.append(checkCell(image: image))
        }
        collectionView.reloadData()
    }
    fileprivate func getRandomTestImages() {
        setRandomIndexes()
        setDataSource()
    }
    override func viewDidLoad() {
```

Kod A.1 : Swift programlama dili ile mobil ortam geliştirmesi.

```
        if let f = Bundle.main.url(forResource: "Parasitized", withExtension: nil) {
            let fm = FileManager()
            all_parasitized = try! fm.contentsOfDirectory(at: f, includingPropertiesForKeys: nil,
options: [])
        }
        if let f = Bundle.main.url(forResource: "Uninfected", withExtension: nil) {
            let fm = FileManager()
            all_uninfected = try! fm.contentsOfDirectory(at: f, includingPropertiesForKeys: nil,
options: [])
        }
        getRandomTestImages()
    }
    @IBAction func shuffle(_ sender: Any) {
        getRandomTestImages()
    }
    @IBAction func toggle(_ sender: Any) {
        let sc = sender as! UISegmentedControl
        current_type = sc.selectedSegmentIndex == 0 ? txtParasitized : txtUninfected
        setDataSource()
    }
    // MARK: - UICollectionViewDataSource protocol
    // tell the collection view how many cells to make
    func collectionView(_ collectionView: UICollectionView, numberOfItemsInSection section:
Int) -> Int {
        return self.labels.count
    }
    // make a cell for each cell index path
    func collectionView(_ collectionView: UICollectionView, cellForItemAt indexPath:
IndexPath) -> UICollectionViewCell {
        // get a reference to our storyboard cell
        let cell = collectionView.dequeueReusableCell(withReuseIdentifier: reuseIdentifier, for:
indexPath as IndexPath) as! ImgCell
        // Use the outlet in our custom class to get a reference to the UILabel in the cell
        cell.lbl.text = self.labels[indexPath.item]
        cell.image.image = UIImage(named: images[indexPath.item])
        return cell
    }
    // MARK: - UICollectionViewDelegate protocol
    func collectionView(_ collectionView: UICollectionView, didSelectItemAt indexPath:
IndexPath) {
        // handle tap events
        print("You selected cell #\(indexPath.item)!")
    }
    // MARK: - Check Image
    func checkCell(image:UIImage) -> String {
        let attrs = [kCVPixelBufferCGImageCompatibilityKey: kCFBooleanTrue,
kCVPixelBufferCGBitmapContextCompatibilityKey: kCFBooleanTrue] as CFDictionary
        var pixelBuffer : CVPixelBuffer?
        let status = CVPixelBufferCreate(kCFAllocatorDefault, Int(image.size.width),
Int(image.size.height), kCVPixelFormatType_32BGRA, attrs, &pixelBuffer)
        guard (status == kCVReturnSuccess) else { return "" }
        CVPixelBufferLockBaseAddress(pixelBuffer!, CVPixelBufferLockFlags(rawValue: 0))
        let pixelData = CVPixelBufferGetBaseAddress(pixelBuffer!)
        let rgbColorSpace = CGColorSpaceCreateDeviceRGB()
```

Kod A.1 (devam) : Swift programlama dili ile mobil ortam geliştirmesi.

```
        let context = CGContext(data: pixelData, width: Int(image.size.width), height:
Int(image.size.height), bitsPerComponent: 8, bytesPerRow:
CVPixelBufferGetBytesPerRow(pixelBuffer!), space: rgbColorSpace, bitmapInfo:
CGImageAlphaInfo.noneSkipFirst.rawValue)

        UIGraphicsPushContext(context!)
        image.draw(in: CGRect(x: 0, y: 0, width: image.size.width, height: image.size.height))
        UIGraphicsPopContext()
        CVPixelBufferUnlockBaseAddress(pixelBuffer!, CVPixelBufferLockFlags(rawValue: 0))

        // Pass the pixel buffer to TensorFlow Lite to perform inference.
        result = modelDataHandler?.runModel(onFrame: pixelBuffer!)
        var txt = (result?.inferences[0].confidence)! > Float(0.8) ? txtUninfected : txtParasitized
        txt += " - \(round((result?.inferences[0].confidence)! * 1000)/1000)"
        return txt
    }
}
        let context = CGContext(data: pixelData, width: Int(image.size.width), height:
Int(image.size.height), bitsPerComponent: 8, bytesPerRow:
CVPixelBufferGetBytesPerRow(pixelBuffer!), space: rgbColorSpace, bitmapInfo:
CGImageAlphaInfo.noneSkipFirst.rawValue)

        UIGraphicsPushContext(context!)
        image.draw(in: CGRect(x: 0, y: 0, width: image.size.width, height: image.size.height))
        UIGraphicsPopContext()
        CVPixelBufferUnlockBaseAddress(pixelBuffer!, CVPixelBufferLockFlags(rawValue: 0))

        // Pass the pixel buffer to TensorFlow Lite to perform inference.
        result = modelDataHandler?.runModel(onFrame: pixelBuffer!)
        var txt = (result?.inferences[0].confidence)! > Float(0.8) ? txtUninfected : txtParasitized
        txt += " - \(round((result?.inferences[0].confidence)! * 1000)/1000)"
        return txt
    }
}
```

Kod A.1 (devam) : Swift programlama dili ile mobil ortam geliştirmesi.

ÖZGEÇMİŞ

Ad-Soyadı : Mehmet YAVUZ

Doğum Tarihi ve Yeri : 21.02.1992 / SAKARYA

E-posta : mehmetyavuz@subu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans:** 2013, Maltepe Üniversitesi, Mühendislik Fakültesi, Yazılım Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2011 yazında Shenzhen'de HK Smartech firmasında stajyer olarak çalıştı.
- 2012-2013 yıllarında Gebze'de C-Tech Bilişim firmasında yarı zamanlı araştırmacı ve test elemanı olarak çalıştı.
- 2013-2015 yıllarında İstanbul'da Sabim Teknoloji firmasında yazılım geliştirici olarak çalıştı.
- 2015-2017 yıllarında Malezya'da DataMicron Systems firmasında yazılım geliştirici olarak çalıştı.