T.C.
İstanbul
YENİ YÜZYIL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİYOMEDİKAL MÜHENDİSLİĞİ ANA BİLİM DALI

Yapay Zeka İle Melanom Tespiti

YÜKSEK LİSANS TEZİ

Tutku KÜÇÜKERBİR

Tez Danışmanı
Dr. Öğr. Üyesi Sevil ÖZER

İSTANBUL
Aralık 2022

T.C.
İstanbul
YENİ YÜZYIL ÜNİVERSİTESİ
Fen Bilimleri Enstitüsü

KABUL ONAY

Biyomedikal Mühendisliği Ana Bilim Dalı Yüksek Lisans Programı çerçevesinde yürütülmüş olan bu çalıma aşağıdaki jüri tarafından Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Savunma Tarihi:
.....

Tez Danışmanı

Dr. Öğr. Üyesi Sevil ÖZER
İstanbul Yeni Yüzyıl Üniversitesi

Jüri Üyeleri

Dr. Öğr. Üyesi Sevil ÖZER
İstanbul Yeni Yüzyıl Üniversitesi _____

Dr. Öğr. Üyesi Gökalp TULUM
İstanbul Nişantaşı Üniversitesi _____

Dr. Öğr. Üyesi Diyadin CAN
İstanbul Yeni Yüzyıl Üniversitesi _____

ETİK BEYAN

İstanbul Yeni Yüzyıl Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

07 / 12 / 2022

Tutku KÜÇÜKERBİR

Teşekkür

Hayatta emek verdiğim her konuda en az benim kadar emek veren aileme, sonsuz desteğini her zaman hissettiğim Onur Gümüş'e, sabır ve ilgiyle bilgi birikimini benden esirgemeden çalınmama yön veren değerli hocam Dr. Sevil Özer'e sonsuz teşekkür ederim.

İçindekiler

Sembol ve kısaltma Listesi

MM : Malign Melanom

CNN : Konvolüsyonel Sinir Ağları

API : Program Uygulama Arayüzü (Application Programming Interface)

YYM : Yüzeysel Yayılan Melanom

ALMM : Akral Lentijinoz Malign Melanom

LSVRC-2010 : Large Scale Visual Recognition (Büyük Ölçekli Görsel Tanıma)

Vggnet-16 : visual Geometry Group (Görsel geometri grubu)

ABCD : Asimetri,Border,Color,Diameter

LMM : Lentigo melanom

Oneiros : Açık uçlu nöro-elektronik akıllı robot iÇletim sistemi

AUC : Area Under Curve (Eğri Altında Kalan Alan)

ESA : European Space Agency (Avrupa Uzay Ajansı)

GPU : Graphics Processing Unit (Grafik İşlemci Ünitesi)

CPU : Central Process Unit (Merkezi işlem Birimi)

Akiec : Aktinik Keratoz

Bcc : Bazal Hücreli Karsinom

Bkl : Benign keratoz

Df : Dermatofibrom

Mel : Melanom

Nv : Melanositik Nevüs

Vasc : Vasküler Cilt Lezyonu

Şekil ve Grafik Listesi

Tablo Listesi

ÖZET

Yapay Zeka İle Melanom Tespiti

Tutku KÜÇÜKERBİR

İstanbul Yeni Yüzyıl Üniversitesi, Fen Bilimleri
Enstitüsü Yüksek Lisans Tezi, Biyomedikal Mühendisliği Anabilim DalıTez
Danışmanı: Dr. Öğr. Üyesi. Sevil ÖZER
Aralık 2022, 69 sayfa

Melanom, kötü huylu bir cilt kanseri türü olup yayılım hızının yüksek olması sebebiyle ölüm oranı en yüksek cilt kanseridir. Melanom kaynaklı ölümlerin azalması erken teşhis ile mümkün- dür. Günümüzde kullanılan teşhis yöntemlerinde doktorun profesyonelliğine, görüntüleme tekniklerine, patoloji sonucu elde edilen verilerin doktor tarafından yorumlanmasına bağlıdır. Günümüzde yapay zekânın kullanıldığı alanlar her geçen gün artmakta olup, bu alanlardan biri de sağlık sektörüdür. Özellikle görüntü işlemede oldukça başarılı sonuçlar vermesi sebebi ile yapay zekânın bir alt dalı olan derin öğrenme, tıbbi görüntülerin işlenmesinde ve yorumlanmasında sıkça tercih edilmektedir. Bu çalışmanın amacı bugüne kadar yapılan çalışmalardaki doğruluk oranından yüksek oranlara erişen bir yapay zeka modeli oluşturmaktır. Bu doğrultuda PHYTON üzerindeki yapay zeka kütüphaneleri kullanarak, Kaggle üzerinden elde edilen HAM10000 veri seti eğitilmiştir. Veri setini maksimum verimde kullanabilmek adına veri artırım yöntemi uygulanmıştır. Veri sayısı artırılarak veri setinin hem eğitim hem de kontrol amacıyla kullanılması sağlanmıştır. Kullanılan veri seti sadece melanom değil, aynı zamanda 7 farklı cilt lezyonunu ta- nıyabilecek şekilde eğitilmiştir. Tensorflow.jp ile oluşturulan yapay zeka modelinin web uygulaması üzerinde çalışması sağlanmıştır. Uygulama .jpg formatında yüklenen bir cilt lezyonu için3 ayrı tahminde bulunarak, lezyonun türü ve iyi huylu ya da kötü huylu olduğu konusunda %97 başarı oranı göstermektedir.

Anahtar Kelimeler: *Kanser,Melanom Kanseri,Yapay Zeka,Cilt lezyonu, PHYTON,*

ABSTRACT

Detection of Melanoma with Artifical Intelligence

Tutku KÜÇÜKERBİR

Istanbul Yeni Yuzyil University, Science and Engineering Institute

Master Thesis, Biomedical Engineering Department

Supervisor: Dr. Sevil OZER

Dec 2022, 69 pages

Melanoma is a malignant skin cancer and has the highest mortality rate due to its high rate of me tastasis. Reduction of melanoma related deaths is possible with early diagnosis. Diagnostic methods used today place great burdens on physicians and depend entirely on the professionalism of the physician and the interpretation of the imaging techniques and pathology results by the phy- sician. For this reason, in this study, it was aimed to reduce the diagnosis time by making a fasterdiagnosis of cancer patients by doctors. In this direction, the HAM10000 dataset obtained from Kaggle was trained using artificial intelligence libraries on PHYTON. In order to use the data set with maximum efficiency, data augmentation method has been applied. By increasing the number of data, the data set was used for both trainingand control purposes. The data set used was trained to recognize 7 different skin lesions, not just melanoma. The artificial intelligence model created with Tensorflow.jp was enabled to work on the web application. The application makes 3 different predictions for a skin lesion uploaded in .jpg format and shows a %97 success rate in terms of the type of lesion and whether it is benign or malignant.

Keywords: *Cancer, Melanoma, Artificial Intelligence, Skin lesion, PHYTON*

1. GİRİŞ

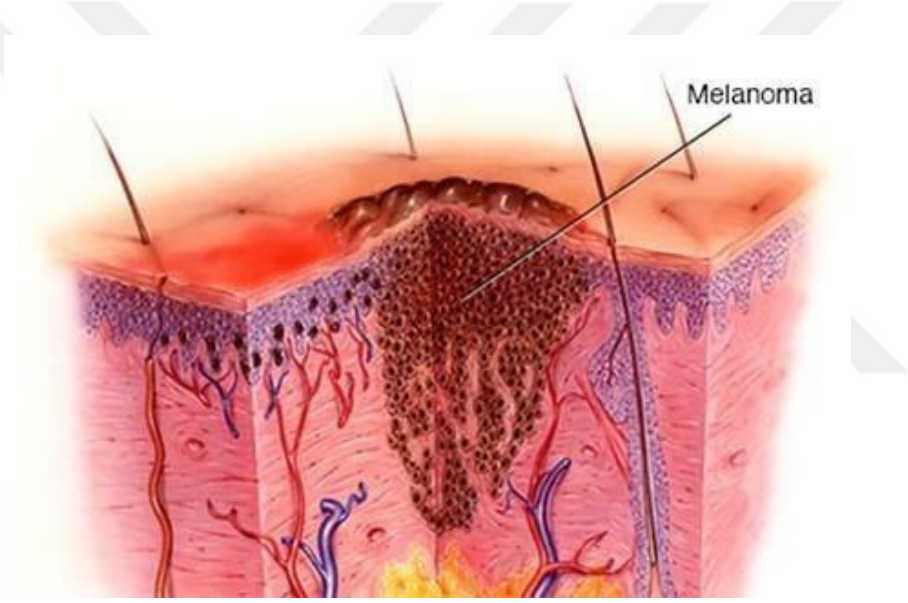
Cilt kanseri, özellikle beyaz tenli insanlarda sıkça görülen bir hastalıktır. Bununla beraber cilt kanserine yakalanma oranları tüm dünyada giderek artmaktadır [1]. Kötü huylu cilt kanseri melanom olarak adlandırılmaktadır. Melanom en ölümcül cilt kanserlerinden biri olmakla beraber dünya genelinde yılda 55.000 kişinin ölümüne neden olmaktadır. Bu tüm kanser türlerinin %0.07'sine tekabül etmektedir. Melanom çevresel ve genetik faktörlere bağlı olduğundan, melanom görülme sıklığı coğrafi konuma göre değişiklik göstermektedir. Cilt kanseri nedeni ile meydana gelen ölümlerin %75'i melanomdan kaynaklanmaktadır [2]. Melanom hızlı yayılım göstermesinden ötürü en çok ölüme neden olan cilt kanseri türü olmasıyla birlikte erken teşhis ile tedavisi mümkündür. Cilt kanseri teşhisi ile ilgili bugüne kadar yapılan çalışmalardan en sık kullanılan biyopsi ile parça alarak, parçanın patolojik incelemesi sonucu doktorların verdiği karardır. Bu yöntem doktorun ve patoloğun deneyimine bağlı olduğundan doğruluk oranı değişkenlik göstermektedir. Teşhiste yüzeysel olarak, deride çıkan lezyonları değerlendirmede kullanılan ABCD yöntemi, 1985'te ortaya atılmıştır. Çıplak gözle uygulanan ABCD yöntemi, (A)symmetry, (B)order, (C)olor, ve (D)iameter ingilizce kelimelerinin ilk harflerinden oluşmaktadır. Günümüzde teknolojinin gelişimi ile kanser teşhisinde yapay zeka uygulamaları oldukça revaçtadır. Genel itibariyle çeşitli tekniklerle hazırlanan doğruluğundan emin olunan verilerin bilgisayara öğretilmesi esasına dayanan bir yöntem ile yapay zeka eğitilir. Doğruluğundan emin olunan veri seti ile eğitilen yapay zeka, gösterilen görüntü üzerinde öğretilen görüntü ile benzerlik kurarak gösterilen görüntünün türünü öğrenebilir. Dermatologların biyopsi alıp patolojik tanısının elde edildiği iyi ve kötü huylu lezyonların dermatoskopik fotoğraflarının olduğu geniş Arşivleri vardır. Bu öğrenmeler sonucunda deri kanseri tanımadaki başarıları çeşitli seviyelerdeki dermatologlarla karşılattırılarak yapay zekanın verimliği değerlendirilmiştir. Yakın zamana yapılan çalışmalarda bilgisayar algoritmaları melanomu tanımada dermatologları üst üste yenmeyi başarmıştır[5]. Bu çalışmanın amacı bugüne kadar yapılan çalışmalardaki doğruluk oranından yüksek oranlara erişen bir sistem tasarlamaktır. Burada yapılmak istenen teşhis koyma işlemi değil, teşhisi kolaylaştırarak melanoma sahip hastaların erken teşhisi konusunda doktorlara destek olarak gelecek dönemde melanom kaynaklı ölüm oranlarının düşmesine yardımcı olacak bir sistemin protatipini gerçekleştirmektir. Çalışmada HAM10000 veri seti kullanılarak yapay zeka, 7 farklı cilt lezyonu için eğitilmiştir. Bu eğitim Phyton üzerinde yapay zeka

kütüphaneleri kullanılarak yapılmıştır. Melanom tespiti üzerine yapılan çalışmalar genel itibariyle sadece melanom tespiti üzerine çalışırken, bu çalışmada 7 farklı cilt lezyonu üzerinden yapay zekanın en yüksek olasılık olmak üzere 3 tahminde bulunması esas alınmıştır. Bu da kanser türünün melanom olmasa dahi iyi huyluya da kötü huylu olmasına karşı daha doğru bir veri vermesini sağlayacaktır. Veri setinde yapılan artırım sayesinde daha büyük bir öğrenim verisi oluşturularak doğruluk oranı yükseltilmiştir. Çalışmada tensorflow ile birlikte keras kütüphaneleri kullanılmıştır. Tezin ikinci bölümünde melanom kanseri hakkında bilgiler verilmiştir. Melanom kanser türleri, melanom kanserinin oluşumu, teşhisine yönelik çalışmalar bu bölümde verilmektedir. Çalışmanın üçüncü bölümü yapay zeka algoritmalarının açıklandığı bölümdür. Yapay sinir ağlarının oluşumu, yapay zeka kütüphaneleri bu bölümde açıklanmıştır. Çalışmanın dördüncü kısmında yapay zeka ile melanom tespitine dair literatürde yapılan çalışmalara yer verilmiştir. Geçmiş çalışmalar ile yapılan çalışma arasındaki farklara tartışma bölümünde yer verilmiştir. Yöntem-metot kısmında çalışmanın amacı, çalışmada kullanılan programlama dili, kütüphaneler ayrıntılı olarak açıklanmıştır. Sonuç bölümünde uygulanan yöntem ve metotlar sonucunda elde edilen çıktılar değerlendirilmiştir.

2. YAPAY ZEKA İLE MELANOM CİLT KANSERİNİN TESPİTİ

2.1 Melanom Nedir ?

Malign melanom cilt dokusunda oluşan ve ölüm ile sonuçlanma ihtimali yüksek olan bir cilt kanseri türüdür. Birçok kanser türünde olduğu gibi erken teşhis edildiğinde hastalığın tedavisi mümkündür. Melanosit kökenli malign kötü huylu bir kanser türüdür. Melanosit hücreleri, deride koyu renkli bir pigment olan melanin üreten hücrelerdir. Melanositler cilde rengini verme görevini üstlenmektedir (Şekil 2.1). Genellikle deride bulunurlar fakat vücudun farklı bölgelerinde de bulunma ihtimalleri vardır. Malign Melanomlar (MM), vücutta melanosit içeren herhangi bir bölümde oluşabilmektedir [6].



Şekil 2.1: Melanoma oluşmuş cilt görüntüsü [7]

Melanom, deri kanserlerinin en az rastlanılan türü olmakla birlikte erken teşhis edilmediği takdirde oldukça tehlikelidir. Deri kanserinden kaynaklanan ölümlerin büyük çoğunluğunun nedeni melanomdur [8]. Melanomun kadınlarda görülme oranı erkeklerde görülme oranından yüksektir. Kadınlarda genellikle bacaklarda meydana gelirken erkeklerde en sık sırtta mey

dana gelmektedir. Melanom görülme oranı coğrafi konuma göre değişkenlik göstermektedir. Coğrafi yayılımın etkin olmasının temel nedeni ultraviyole Işık maruziyeti farklılaşması veya nüfustaki deri pigmentasyon oranının düşüklüğü ile açıklanmaktadır [9].

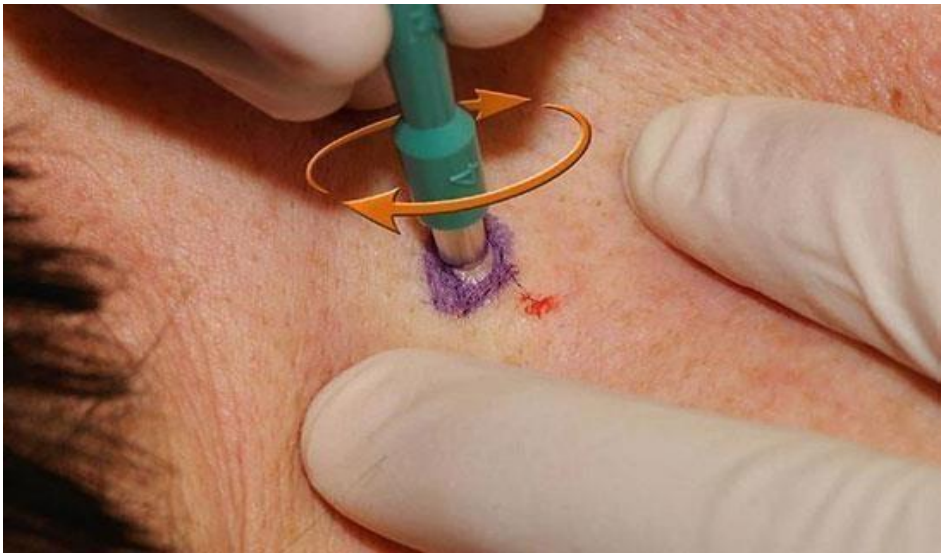
Dünya Sağlık Örgütü (DSÖ)‟nün raporuna göre, her yıl dünya genelinde 48.000 kişi melanom ile ilişkili nedenlerden dolayı hayatını kaybetmektedir[10].

MM‟ların görülme sıklığı tüm dünyada son yıllarda en hızlı artan kanser türü olarak gözlemlenmiştir. Kutanöz maligniteler (KM) arasında bazal hücreli kar- sinom ve skuamöz hücreli karsinoma göre MM‟ler daha az görülmekle birlikte, ölüm hızı en yüksek cilt kanseri türü olduğu ifade edilmektedr. Dünya Sağlık Örgütü verilerine göre dünya genelinde her yıl 160.000 yeni Melanom vakası tespit edilmektedir [11]. MM'u oluşturan tümör kitlesi, diğer tümörlerde de olduğu gibi sadece kanser hücrelerinden değil tümör mikro çevresinden de oluşmakta ve diğer dokulara metaztaz yapmaktadır. Tümörün ilerleme hızı ve buna bağlı metastaz yapması kanser hücrelerinin bu çevre ile etkileşimine bağlıdır [12]. En sık deride görülmekle birlikte diğer dokularda da örneğin gözde konjuntiva ve üveada, mukozalar ve leptomeninkslerde de gelişebilmektedir. Şekil 2.2'de, farklı dokularda gözlenmiş melanom tiplerine yer verilmiştir.

Şekil 2.2: Melanom gözlemlenen doku ve organlar [13]

Kutanöz melanomlar (KM), tüm melanomların %91,2'sini oluşturmaktadır [14]. KM için en büyük risk faktörleri; açık ten rengi, kızıl saç, kişinin ailesinde melanom geçmişinin olması, kronik olarak güneş maruziyeti ve bunun sonucu güneş yanığı öyküsüyle birlikte aşırı oranda solaryum maruziyetidir [15]. KM için tanımlanmış dört ana alt tip mevcuttur. En sık görülen yüzeysel yayılan melanom (YYM) tipidir. Genellikle renk çeşitliğinin görüldüğü kahve-siyah renk değişimi olarak başlayıp plak haline gelir, üzerinde papül veya nodül gelişimi olabilmektedir. İkinci en sıklıkta görülen tipi lentigo malign melanom (LMM) nodülerdir. Özellikle güneş maruziyetinden kaynaklanan LMM, daha çok yaşlı kişilerin yüzlerinde asimetrik kahverengi-siyah lekeyi andıran izler olarak izlenmektedir. Karşılaşılma sıklığı en düşük olan tipi olan akral lentijinöz malign melanom (ALMM) ise en çok tırnakta görülmektedir. (Şekil 2.2).

Melanomun birincil tedavisi cerrahi olup Melanom tespitinde ilk olarak fiziksel muayene yapılmaktadır. Fizik muayene esnasında; deride oluşan lezyonun büyüklüğü, şekli, rengi ve dokusu ile birlikte bu bölgede kanama ya da soyulma olup olmadığı incelenmektedir. Fiziksel muayene sonunda eğer melonomadan şüpheleniliyorsa, şüpheli bölgeden mikroskop altında incelenmek üzere parça alınıp, biyopsisi yoluna gidilmektedir [16]. Deriden biyopsisinin (Şekil 2.3) alınabilmesi için çeşitli metotlar uygulanmaktadır. Hangi yöntemin seçileceği, deri kanserinin vücuttaki yerine ve boyutuna göre değişim göstermektedir. Deriden alınan örnek, mikroskopta doku örneklerini inceleyerek tanı koyma konusunda eğitimli olan bir patoloğa gönderilerek, lezyonun durumu deri hastalıkları ve deri kanserleri konusunda özelleşmiş bir dermotolog ile birlikte değerlendirilmektedir.



Şekil 2.3: Biyopsi için parça alınması [17]

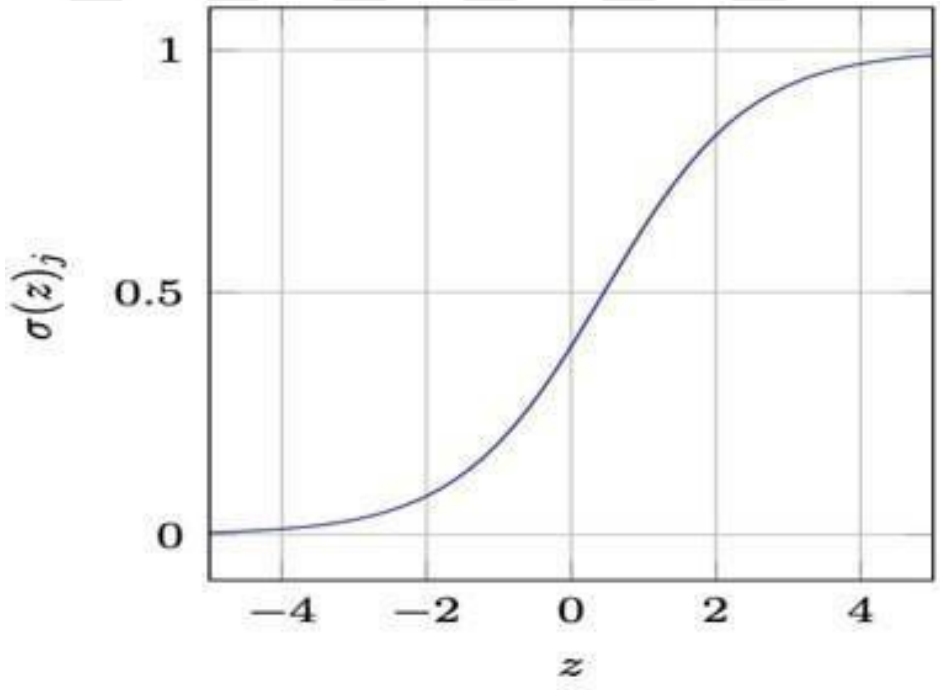
Günümüzde teknolojinin ilerlemesi ile kanser teşhisinde bilgisayar öğrenmesi ve yapay zekâdan destek alınmaktadır. Bunun başlıca nedeni makine öğrenmesi sonucu bilgisayardan alınan verilerin doğruluk oranının oldukça yüksek olmasıdır. Derin sinir ağları, yüksek veri setleri ile eğitildiğinde iyi huylu veya kötü huylu kanser ayrımını kolaylıkla yapabilmektedir. Bu eği- tim sonucunda bilgisayar, iyi huylu veya kötü huylu kanseri doğrusal olma- yan bir yöntem ile öğrenerek sonucu yüksek doğruluk oranı ile tespit ede- bilmektedir. Bu yöntemin en büyük avantajı hızlı sonuç vermesidir. Bu sayede erken teşhis ile kanser hastalarının hayatta kalma oranları yükselmiş olacaktır.

2.2 Yapay Zekâ Algoritması

Günümüzde yapay zekâ birçok alanda oldukça sık kullanılabilir hale gelmiştir. Bu alanların başında sağlık sektörü gelmektedir. Özellikle görüntü işleme konusunda oldukça başarılı sonuçlar veren yapay zekâ uygulamaları gün geçtikçe daha çok tercih edilir duruma gelmiştir. Derin öğrenme yapay zekânın alt dalıdır. Görüntülerin işlenmesi ve yorumlanması hususunda oldukça sık kullanılmaktadır. Tıbbi görüntüleme sürecinde kullanılan derin öğrenme algoritmaları her ne kadar teşhis konusunda uzmanlara yardımcı olsa da günümüzde bu görüntülerin uzmanlar tarafından yorumlanması gerekmektedir. Tek başına yapay zekâ algoritmalarını verdiği sonuçlar teşhis için yeterli değildir. Ancak, yapay zekâ ile otomatik tanı sistemleri oluşturmak artık gelişen teknoloji ve algoritmalar sayesinde her geçen gün ilerleme katetmiştir.

Yapay sinir ağları, bir makine öğrenmesi metodudur. Burada amaç insan öğrenmesini, insan sinir sistemini taklit etmektir. Yapay sinir ağları 3 katmandan oluşmaktadır. Bunlar, giriş katmanı, ara katmanlar ve çıktı katmanlarıdır (Şekil 2.7). Yapay sinir ağlarını oluşturan hücreler birbirine bağlıdır. Bu bağları oluşturan bağlantı noktaları insan sinir hücresindeki sinapslara karşılık gelmektedir. Oluşan bu bağlantılara ağırlık (weight) adı verilmektedir. Şekil 2.7'de görüldüğü gibi, bir yapay sinir hücresinde girdiler weight (ağırlık) ile çarpılır ve tüm değerler toplanır. Ardından bias değeri bu top- lama eklenir. Sonuçta oluşan net girdi aktivasyon fonksiyonundan geçer ve bir çıktı oluşur. Birçok farklı aktivasyon fonksiyonu vardır.

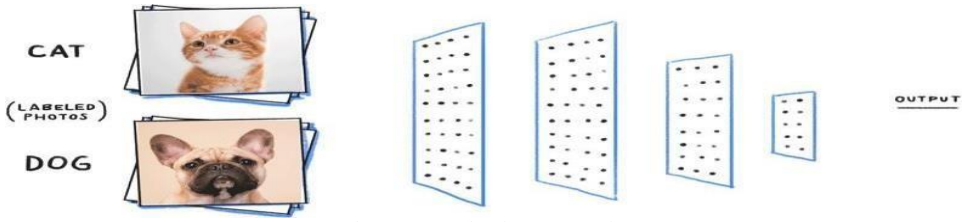
Bu çalışmada ise kullanılan aktivasyon fonksiyonu softmaxtır. Softmax aktivasyon fonksiyonu olasılık işlemlerinde kullanılmaktadır. Fonksiyon, girdileri 0 ile 1 arasında bir değere dönüştürür.(Şekil 2.4).



Şekil: 2.4: Softmax aktivasyon fonksiyonu

Yapay zekânın bu alandaki kullanımından elde edilen başarılı sonuçlar, etmektedir. Buna ek olarak, tıbbi görüntüleme alanında görülen gelişmeler bu alanda çalışmalar yapan radyolog ve teknisyenlerin bu teknolojiyi ne denli benimsediklerinin iyi bir örneğidir. 2019 senesi ile birlikte yapay zekâçalışmaları sağlık alanında oldukça sık kullanılır duruma gelmiştir. Buna karşın tıbbi görüntüleme alanında yapay zekâ kullanımı hala spekülasyona açık ve kesinliği kanıtlanmış değildir [18].

Makine öğrenmesi, insan zekâsını taklit ederek kendi öğrenim algoritmalarını geliştirerek ilerleyen algoritmalar bütünüdür. İnsan öğrenmesi ile makine öğrenmesi benzerlik göstermektedir. İnsan, duyduğu ve gördüğü kavramları algılarken beyin bunu otomatik olarak yapar. Aynı durum makine öğrenmesi uygulamalarında da geçerlidir. Makine, kendisine girilen veri kümelerini özümseyerek, veri kümesini ve gerçekleşmesi beklenen görevi öğrenmektedir. Geleneksel kodlama ve makine öğrenmesi algoritmaları arasındaki fark, bir bebeğin öğrenme süreciyle analoji yapılarak açıklanabilir. Örneğin, hayvanlar hakkında bilgisi olmayan bebeğin öğrenim sürecinin başında bebek, kedi ile köpek kavramını ve ikisinin arasında- ki farkı bilmemektedir. Bu deneyde bebeği; yapay zekâ algoritması olarak, ona farkları öğretecek kişiyi ise, bir bilgisayar mühendisi ya da yazılımcı olarak düşünmek mümkündür. Klasik yapay zekâ uygulamalarına bakılacak olursa; bebeğe öncelikle, kedi ve köpek arasındaki farkı öğretmek yerine, ona kedi ve köpeği ayırt etmesini sağlayacak yönergeler verilmektedir. Bu hem öğretici hem bebek için oldukça zor bir uzun sürece sebep olmaktadır. Ancak bebeğe çok sayıda kedi ve köpek fotoğrafı göstererek yüksek veri sağlayarak, hangisinin kedi hangisinin köpek olduğunu tekrarlamak bebeğin bir noktadan sonra yüksek bir doğruluk ile bu ikisini ayırt etmesini sağlayacaktır.

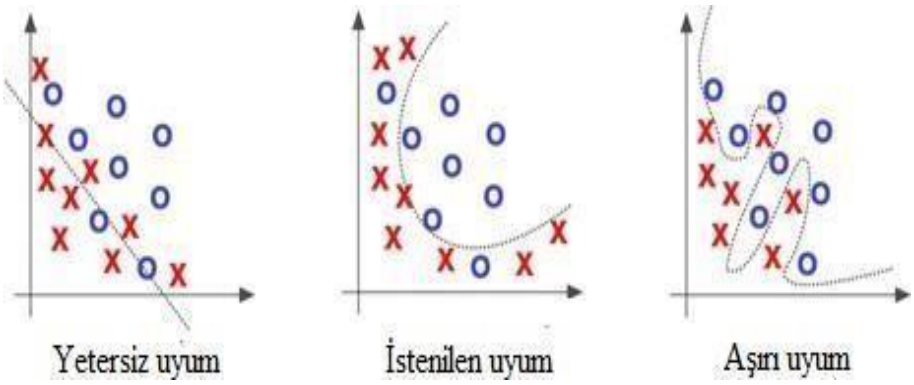


Şekil 2.5: Yapay zeka öğrenme algoritması örneği[19]

Makine öğrenmesi algoritmalarının çalışma prensibi yukarıdaki örnekleme dayanmaktadır. Makineye gösterilen kedi-köpek görüntü setleri ile makinenin kedi ve köpeği öğrenmesi sağlanmaktadır. Bu sadece Şekil 2.4'de olduğu gibi çıkış makine tarafından doğru bulunacaktır. Yapay zekâya kıyasla makine öğrenmesi; daha dar bir alan olarak gözükse de, içerisinde çok fazla algoritma ve yöntem bulunduran bir alan olarak karşımıza çıkmaktadır. Makine öğrenmesi, çeşitli verilerin yazılım bilimciler tarafından bilgisayara öğretilmesi, bilgisayarın gerçek dünya ile etkileşimi sonucunda elde ettiği bu verileri işleyerek karar vermesi, anlaması ve yorumlama yeteneği kazanmasını sağlayan bir araştırma alanını oluşturmaktadır [20].

Öğrenmenin hedeflenen ilk amacı, eğitim seti kullanılarak eğitilmiş olan bir modelin eğitimin sonunda, test veri seti üzerinde başarılı sonuçlar elde etmesini sağlamaktır. Eğitilmiş modelin daha önce görmediği, yani eğitiminde kullanılmamış bir veri üzerindeki başarısı, o modelin genelleme sığasının ne denli yüksek olduğunu gösterir. Modelin eğitiminde, veri sayısının sınırlı olması sonucunda, iki istenmeyen durum ortaya çıkabilir. İlki aşırı uyum, Şekil 2.5'te görüldüğü üzere, modelin eğitim verisi üzerindeki başarıyı test verisi üzerinde gösterememesinden ortaya çıkan bir durum olarak karşımıza çıkar. Yani model eğitim sırasında gösterilen verileri ezberleyerek sadece o veriler üzerinde başarı göstermektedir. Bu durum, modelin eğitim verisini ezberlemesi olarak tanımlanabilir.

İkinci istenmeyen durum olan yetersiz uyum ise, kullanılan modelin eğitim setinde bile iyi sonuçlar vermediği durumlara karşılık gelmektedir [21]. Eğitim verisi dağılımı doğrusal değilken tahmin modeli oluşturmak için doğrusal bir algoritma seçildiği takdirde model doğrusal olmayan ilişkiyi göremediği için doğruluk azalmaktadır. Problemin karmaşıklığı modelin öğrenmek için oluşturduğu algoritmadan karışık olduğunda yetersiz uyum oluşmaktadır. Oluşturulan algoritma eğitim setindeki verilere ulaşmak adına çalıştığında model doğrusal olmayan eğitim setinde yer verilmemiş ya da farklı bir şekli ile yer verilmiş verileri tanıyamaz. Bu durum aşırı uyum problemini ortaya koymaktadır.



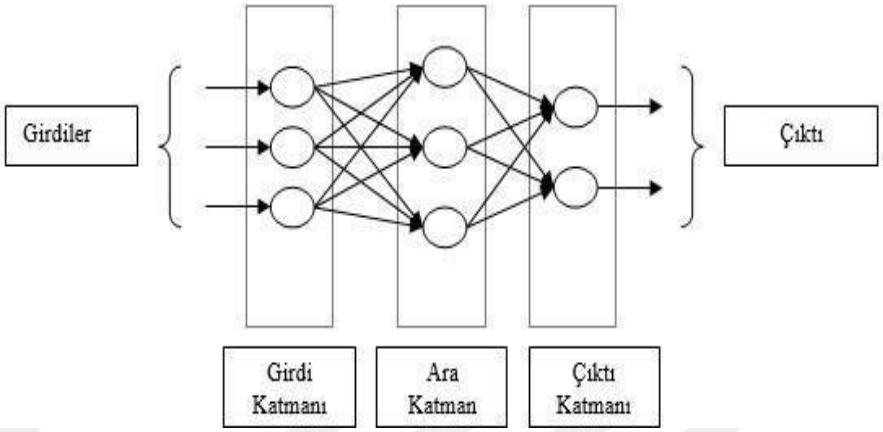
Şekil 2.6: Yetersiz uyum, istenilen uyum, aşırıuyum [22]

2.2.1 Yapay Sinir Ağları

Yapay sinir ağları, insan sinir hücrelerine benzer yapıda olup, makine öğrenmesinde kullanılan bir tür algoritma taslağıdır [23]. Yapay sinir ağları taslağı, bir memeli beyninden esinlenilerek tasarlanmıştır. 2010 yılında Toronto Üniversitesi'nden Alex ve arkadaşları tarafından AlexNet isminde yapay sinir ağı modeli geliştirmiştir [24]. AlexNet'in önemi, 1000 farklı sınıfa ait olan 1.2 milyon görselin yer aldığı ImageNet veri kümesinde eğitilerek, ImageNet LSVRC"nin 2010 yılında gerçekleştirdiği yarışmada, görsel sınıflandırma kategorisinde oldukça yüksek başarı elde etmesidir. Bu durum zorlu problemdeki başarı oranlarının çok yüksek olmasını sağladığından yapay sinir ağlarının gücünü ispat etmiştir.

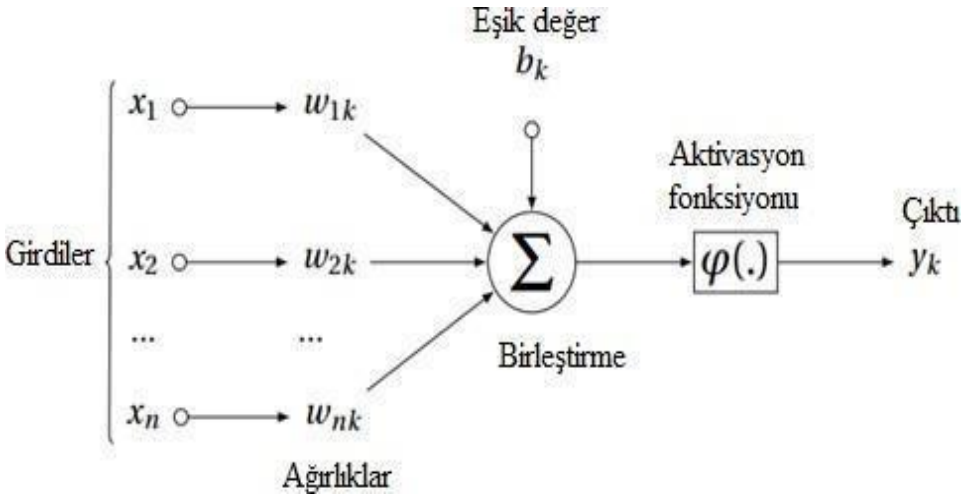
Ancak, ilk yapay sinir hücresini, McCullohg ve Pitts bundan yıllar önce, 1943 senesinde uygulamıştır [25]. Bu çalışmada, girdilerin doğrusal kombinasyonun belli bir eşik değeri geçmesi ile aktive olan nöron yapısı basit bir elektrik devresi ile oluşturulmuştur. McCullogh ve Pitts tarafından yapılan çalışmaların sonucunda ilk algılayıcı ise, Rosenblaatt tarafından 1957 senesinde yapılan bir çalışmada modellenmiştir. Bu çalışmada, ikili sınıflandırma görevleri için tasarlanan modelde, gerçek girdiler ve ağırlıklar, belirlenenöğrenme kuralları ile birlikte görüntü sınıflandırma görevleri için kullanılmıştır. Minsky ve Papert'de 1969 senesinde yapılan bir başka çalışmada da, basit bir sinir ağı mimarisini önermişlerdir.

Girdi katmanı, ara katman ve çıktı olmak üzere üç katmandan meydana gelmektedir. Ancak tek katmanlı sinir ağları öğrenme açısından yeterli olmadığından sonraları gizli katmanlarda eklenmiştir [26].



Şekil 2.7: Girdi, ara katmanlar ve çıktı [27]

Basit bir yapay sinir hücresindeki iletim incelendiğinde (Şekil 2.8); ilk katmandaki algılayıcılar, girdileri ve girdilerin ağırlıklarını dikkate alarak üç basit karar vermektedir. Ardından ikinci katmandaki algılayıcılar, ilk katmandaki algılayıcılardan iletilen kararlar ve ağırlıklar ışığında dört karar vermektedir. Son katmandaki algılayıcılar ise ikinci katmandan gelen dönütler doğrultusunda kararlar vermektedir. Dolayısıyla alınan kararlar çıkışa doğru giderek karmaşık bir hal almaktadır.



Şekil 2.8: Yapay sinir ağı modelinin çalışma şekli[28]

Yapay sinir ağı birçok sayıda yoğun katmanlardan oluşmaktadır. Yani bir katmandaki tüm düğümler, bir sonraki katmandaki bütün düğümlere bağlı olacaktır. Derin öğrenme yöntemlerinin doğrusal olmayan ilişkilerini modellemesi amacıyla, tüm düğümlerde gerçekleşen işlemler doğrusal olmayan bir aktivasyon fonksiyonundan geçirilmektedir.

2.2.2: Derin Öğrenme Kütüphaneleri

Makine öğrenmesi ve derin öğrenme amacıyla çeşitli üniversiteler ve Şirketler tarafından geliştirilmiş, farklı özelliklerde ve derin öğrenme alanında yapılan çalışmaları pratikleştirip kolaylaştıran birçok hazır derin öğrenme kütüphaneleri ve Uygulama Programlama Ara yüzleri (Application Programming Interface-API) bulunmaktadır. Araştırmacının, çalışılacağı konuya uygun kütüphaneleri bilgisayarına kurması gerekmektedir. Bu derin öğrenme kütüphanelerin uygulama ara yüzlerinin her biri farklı işleve sahiptir. Örneğin, Python programlama diline uygun yazılmış bir düzineden fazla derin öğrenme kütüphanesi mevcuttur. Python programlama dilinde kullanılan bazı kütüphaneler, geliştiricisi ve işlevleri Şekil 2.9'de verilmiştir. Bu çalışmada, Keras ve TensorFlow kütüphanelerinden yararlanılmıştır

Kütüphane Adı	Dili	Geliştiricisi	İşlevi
Theano	Python	MILA Lab	Çok boyutlu diziler dâhil, matematik ifadeleri etkili bir şekilde tanımlamayı, en iyilemeyi ve değerlendirmeyi sağlayan bir Python kütüphanesidir.
TensorFlow	Python	Google	Veri akışı grafikleriyle verimli sayısal hesaplamalara olanak tanımaktadır.
Caffe	Python	BVLC	Caffe derin öğrenme yapısı hızlı ve modüler olacak şekilde tasarlanmıştır. Model ve optimizasyonlar kodlama yapılmaksızın ayar dosyası üzerinden yapılabilmektedir.
Keras	Python	Google	Keras kütüphanesi Tensorflow ve Theano'ya üst katman olarak yazılmış, daha kolay model geliştirmeyi sağlayan bir Python kütüphanesidir.
Mxnet	Python	Amazon	Keras gibi yüksek seviye kütüphanedir, bir polyglot (çok dilli) olmasından dolayı, farklı dillerde model paylaşan ekipler için harika çözümler sunmaktadır. Diğer bir avantajı ise dağıtılmış bilgi işlemini desteklemesidir.

Şekil 2.9: Phyton üzerindeki derin öğrenme kütüphaneleri [29]

Yapay zekânın farklı program dilleri ile kodlanması mümkündür. Bu çalışmada, günümüzde yapay zekâ alanında sıkça karşılaşılan bir programlama dili olan Python kullanılmıştır. Python, birçok programlama dillerinin (C ve C++ vb.) aksine derlenmiş bir dil değildir. Python, yorumlanan bir programlama dilidir. Bunun sayesinde daha esnek olup daha düşük program boyutlarına sahiptir. Ayrıca, yorumlayıcılar kaynak kodlarını kendileri yürüttükleri için, kodun kendisi platformdan bağımsızdır ve birçok cihazda kullanılabilir.

Açık kaynak kodlarını içerisinde barındıran TensorFlow, bir derin öğrenme (Deep Learning) kütüphanesidir. TensorFlow, veri akış grafikleri kullanarak sayısal hesaplama yapan açık kaynaklı bir derin öğrenme yazılım kütüphanesidir. Ayrıca, TensorFlow ile derin öğrenme destekli olan yapay zekâ uygulamalarının geliştirilmesinde mümkündür. Buna ek olarak JavaScript kullanımıyla, TensorFlow.js internet tarayıcılarından yapay zekâ ile ilgili çok fazla işlem yapabilmektedir [30]. TensorFlow, 2015 yılında Apache License sürümü adı altında Google Brain ekibi tarafından piyasaya sürülmüştür [31] JavaScript tarafından da desteklenen TensorFlow kütüphaneleriyle, internet tarayıcıları (Browser) aracılığıyla derin öğrenme ve makine öğrenmesi yazılımları geliştirilebilmekte ve kullanılabilmektedir. [32].

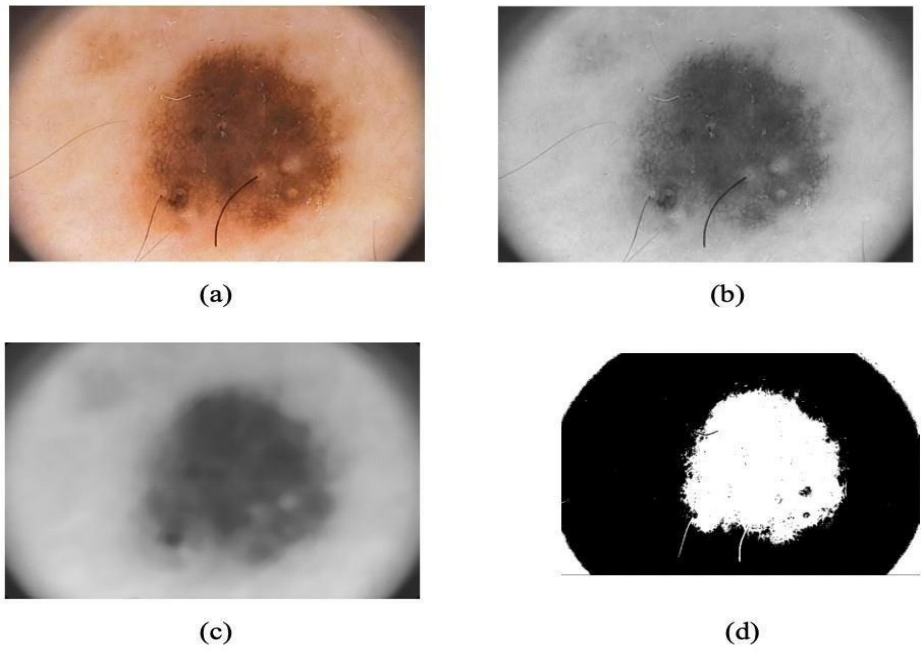
Keras, Python'da yazılmış, yapay zekâ çalınmalarında sıkça kullanılan açık kaynaklı sinir ağı kütüphanesidir. Derin sinir ağları ile hızlı deney yapabilmek için tasarlanan Keras, daha çok kullanıcı dostu, modüler ve genişletile- bilir olmaya odaklanmıştır. ONEIROS (Açık Uçlu Nöro- Elektronik Akıllı Robot İşletim Sistemi) projesinin araştırma sürecinde ortaya çıkmış ve geliştirilmiştir. Ana yazarı ve sürdürücüsü Google mühendisi François Chollet daha sonra, ayrıca XCeption denilen bir derin sinir ağı modelini de geliştirmiştir [33].

Keras ile derin öğrenme modeli oluşturmak oldukça kolaydır. Bu adımlar aşağıdaki sırayladır;

- Eğitim verisi tanımlanır.
- Katmanlar ve model tanımlanır.
- Epoch (Tur sayısı), loss fonksiyonu ve optimizer gibi hiper parametreleri tanımlanır.
- Eğitim verisi ile model beslenir.

3. YAPAY ZEKÂ

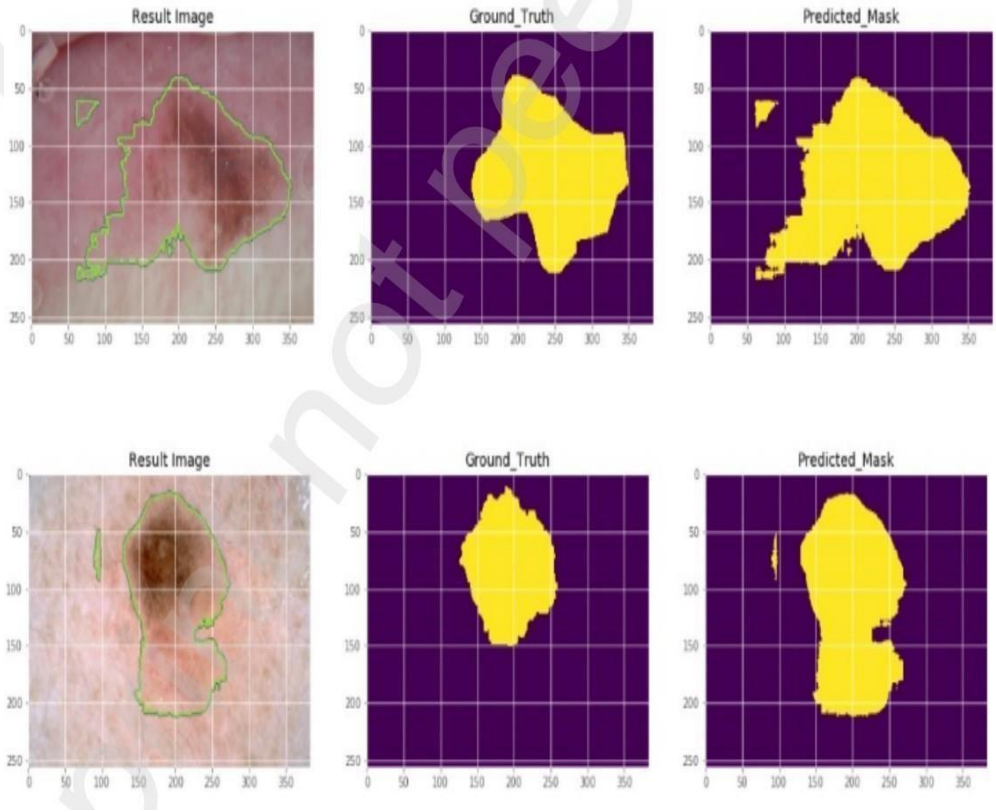
Yapay zekâ ile kanser tespiti, bilgisayar teknolojisinin ve derin öğrenme kavramının ilerlemesi ile günümüzde oldukça popülerleşmiş bir yöntem haline gelmiştir. Farklı veri setleri ile farklı yöntemler kullanılarak farklı öğrenme yöntemleri uygulamak mümkündür. Burada öğrenmenin kalitesini belirleyecek en önemli unsur veri setinin kalitesi olacaktır. Daha önce yapılan çalışmalarda direkt melanom tespiti üzerine yapılan çalışma sayısı oldukça fazladır. Çalışmaların bir kısmı veriseti beslemesi yaparak yapay zekâ öğrenmesi üzerine iken bir kısmı görüntü işleme metotlarını tercih etmektedir. Görüntü işleme ile yapılan melanom tespit çalışmalarında genel itibariyle filtreleme işlemi uygulanmaktadır. Filtreleme sonrası kanserli alanın rengi, çözünürlüğü, boyutu gibi faktörler yapay zekâya tanıtılarak geleneksel kodlama ile günümüz teknolojisinin birleştirilmesine dayalı çalışmalar oldukça fazladır. Şekil 3.1'de bir görüntünün kıl temizleme tekniği sonrası görüntü işleme ile bilgisayar görüşüne uygun hale gelme prosesi adımadım gösterilmektedir. Bu işlem sayesinde görüntü üzerindeki gürültü azaltılmış, görüntü bilgisayarın öğrenebilmesi adına bilgisayar diline evirilmiştir. Bu sayede bilgisayarın görüntüyü öğrenmesi gürültüler, görüntü içerisinden çıkartıldığı için daha kolay bir hale gelmiştir.



Şekil 3.1: Gerçek görüntünün kıl temizleme tekniği sonrası görünürlüğü [34]

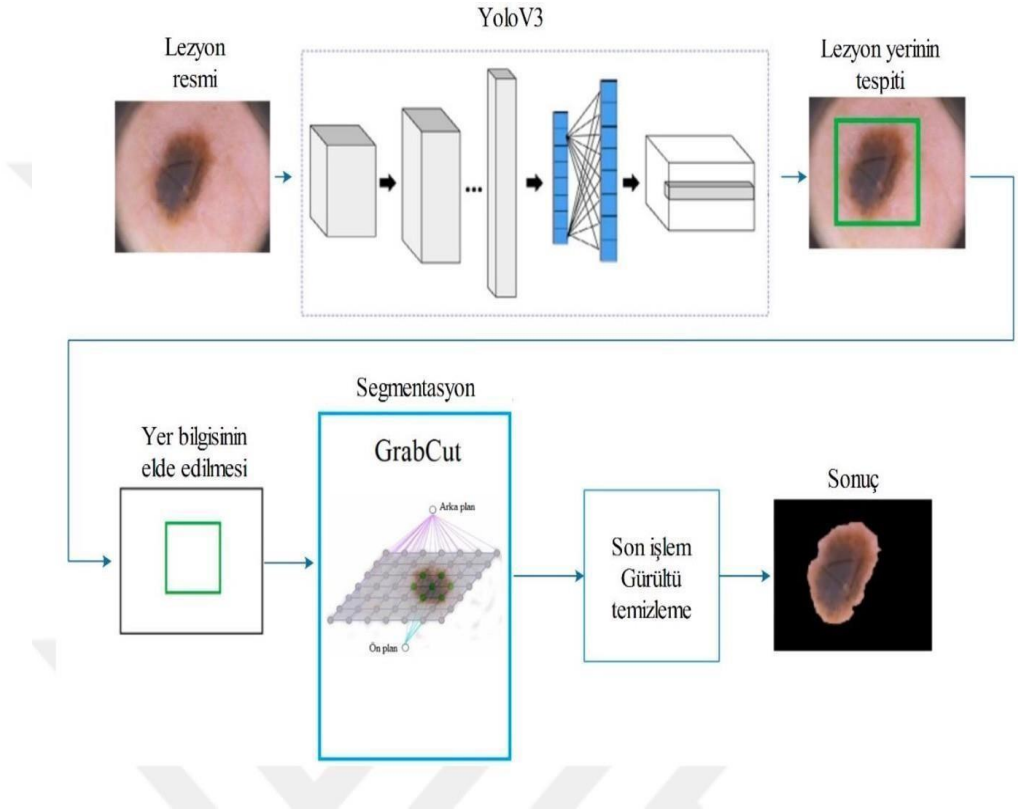
Yapay zekâ ile melanom tespitine dair yapılan son çalışmalarda lezyon görüntüleri alınarak sınıflandırma işlemlerinden geçirilmiştir. Bu sınıflandırmada lezyonun boyutu, rengi, asimetrisi gibi çeşitli parametreler işlenmektedir. Fakat sadece bu parametrelerden yararlanarak yapılan lezyon tahminlerinde doğruluk oranı düşük olacağından çalışmaya makine öğrenimi kullanılarak devam edilmiştir.

Bu çalışmada kullanılan yöntem bir veri setini eğitim, öğrenme ve test amacıyla kullanarak bütün deri görüntüsü üzerinden melanom olan bölgenin çıkartılması esasına dayanmaktadır. Daha önce yapılan filtreleme işlemi sayesinde sistemin melanomu tespit edebilme ihtimali daha yüksek olmaktadır [35].



Şekil 3.2: Asıl görüntüde hastalıklı alan- bilgisa-yarın gördüğü hastalıklı alan [36]

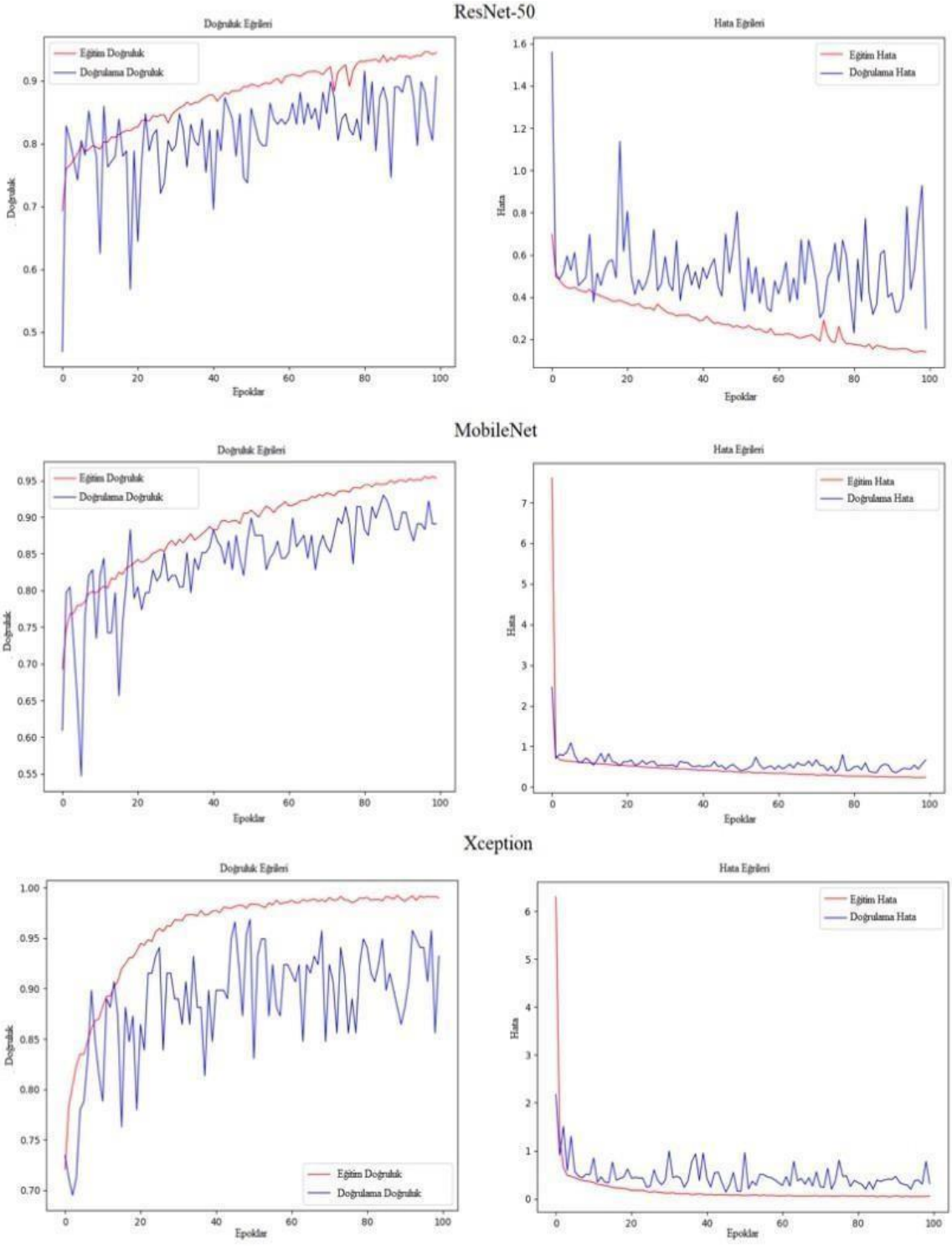
York ve ark. tarafından yapılan bir çalışmada, melanomun tespiti için kaggle üzerinden veri setleri alınmıştır. Ardından kıl filtreleme işlemi uygulanmış ve filtrelenmiş görsellerde kanserli dokunun yeri YOLO modeli ile Şekil 3.2'de gösterildiği gibi tespit edilmiştir. Kanserli doku ile sağlıklı dokuyu ayrıştırma işlemini ve yapay zekânın öğrenimini kolaylaştırmak için YOLOile yer tespiti yapılmıştır. Kanserli olmadığına emin olunan doku görselleri görüntü üzerinden silinerek yapay zekânın iş yükünün sadece kanserli dokuyu tespit etmek olması sağlanmıştır.



Şekil 3.3: Yolo ile tümör yeri tespiti ve grabcutuygulaması [37]

Ardından GRABCUT görüntü işleme algoritması ile segmentasyon işlemi gerçekleştirilmiştir. Önerilen sistemin son aşaması olan sınıflandırma aşamasında segmente edilen görüntüler ile Xception, MobileNet ve ResNet-50 ESA ile eğitim modelleri oluşturulmuştur. Veri seti olarak ISIC kullanılmıştır. ISIC veri setinden alınan 50000‟i sağlan 40000‟i melanoma olmak üzere toplam 90000 veride veri artırma işlemi yapılmıştır. Veri artırım işleminde; belirli açılarla döndürme, rasgele kırpma, parlaklık, kontrast geliştirme, yakınlaştırma gibi işlemler uygulanmıştır. Eğitim sonucunda elde edilen çoğaltılarak elde edilen 50.000 görüntü ile sinir ağları eğitilmiştir. Eğitim sonunda elde edilen (weight) ağırlıklar kaydedilmiştir. İkinci eğitim sürecinde ise modellerin segmente edilmiş görüntüler ile eğitimine bu ağırlıklar kullanılarak başlanmıştır. Eğitim sürecinde öğrenim konusunda verim alınan katmanlar eğitime katılmıştır Fakat modeller tam eğitilmemiş modelin büyüklüğüne göre bazı evrişimsel katmanlar eğitime katılmamıştır. Her bir ağ toplamda 20000 görüntü ile eğitilmiş 1000 görüntü ile doğrulama işlemi gerçekleştirilmiştir [38].

Şekil 3.4’te ResNet-50, Mobile-Net ve Xception modellerde doğruluk oranlarını göstermektedir. Her modelde doğruluk-adım ve hata-adım olmak üzere iki tablo verilmiştir. Bu tablolarda kırmızı renkli çizgi eğitim (train), mavi renkli çizgi doğrulama (valid) fonksiyonunu ifade etmektedir. Eğitim çizgisi ile doğrulama çizgisi birbirine ne kadar yakınsa ve doğruluk oranı ne kadar yüksekse eğitim o kadar başarılı geçmiştir. Aynı Şekilde hata ne kadar düşükse eğitim o kadar başarılı geçmiştir. Şekil 3.4’te görüldüğü üzere MobileNet train ve valid çıktıları birbirine en yakın olup hata parametresi en düşük olan modeldir. Bu durum çalışmada MobileNet kullanma olasılığını yüksek kılmaktadır.



Şekil 3.4: Farklı modellerde doğruluk oranları[39]

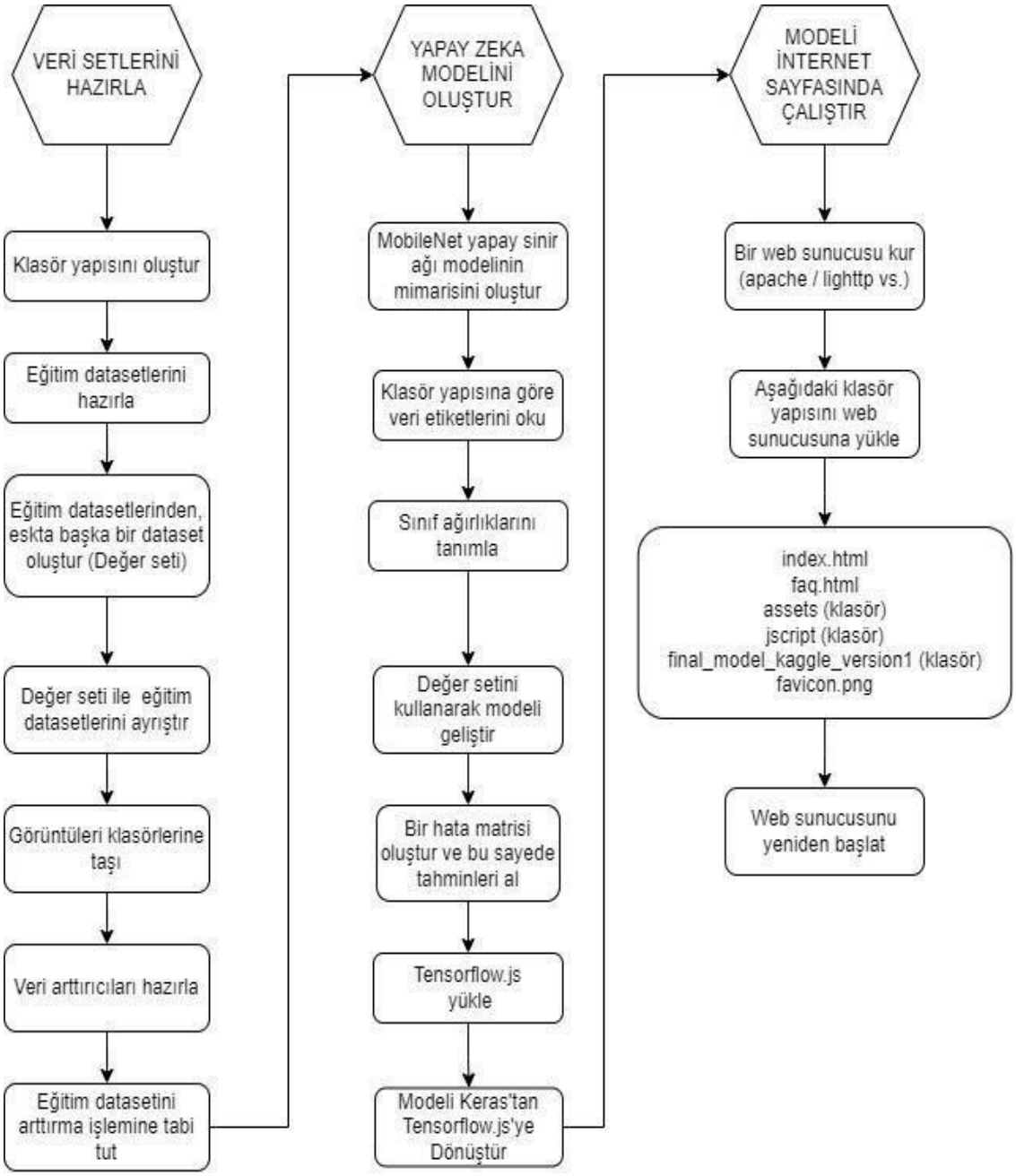
Yapılan bir başka çalışmada HAM10000 data seti kullanılarak, bu data setin eğitiminde farklı yapay sinir ağı modellerinin çıktı verileri gözlenmiştir. Yapay sinir ağları aynı verilerle aynı şartlar altında eğitildiğinde, hangi yapay sinir ağı modelinin öğrenim üzerindeki doğruluk oranının daha yüksek olacağına dair bir çalışma yapılmıştır.

Tablo 3.1: Farklı modellerin doğruluk- hassasiyetskor tablosu [40]

Modeller	Doğruluk	AUC	F-Skor	Kesinlik	Hassasiyet	Hata Oranı
AlexNet*	0,8303	0,8987	0,7866	0,8046	0,7693	0,1696
AlexNet	0,6438	0,7209	0,6545	0,6403	0,6695	0,3561
DenseNet-121*	0,8593	0,9292	0,8231	0,8494	0,7984	0,1406
DenseNet-121	0,7758	0,8526	0,7041	0,7495	0,6640	0,2241
ResNet-18*	0,8661	0,9303	0,8320	0,8590	0,8066	0,1338
ResNet-18	0,7709	0,8453	0,6985	0,7350	0,6655	0,2290
ResNet-34*	0,8751	0,9404	0,8438	0,8761	0,8137	0,1248
ResNet-34	0,8514	0,8980	0,8122	0,8460	0,7811	0,1485
SqueezeNet*	0,7687	0,8106	0,7281	0,7227	0,7336	0,2312
SqueezeNet	0,5690	0,6476	0,6061	0,5972	0,6154	0,4309
VGGNet-16*	0,8695	0,9385	0,8366	0,8602	0,8143	0,1305
VGGNet-16	0,8149	0,8808	0,7619	0,8021	0,7256	0,1850

Tablo 3.1'de farklı modellerin doğruluk, AUC, F-skor, kesinlik, hassasiyet ve hata oranlarına yer verilmiştir. Bir parametre mutlak doğru iken, doğruluk oranı 1 olacaktır. Hata oranı yüksek olan modelin doğruluk oranı düşüktür. AUC değerinin yüksek olması modelin istenilen uyuma yakınlığını ifade etmektedir. F-skor değeri, keskinlik ve duyarlılığın harmonik ortalamasını vermektedir. F- skor değeri yüksek olan modelin hata oranı düşük olacaktır. Bu durumda öğrenim için en doğru modelin VGGNet-16 olduğunu görülmektedir.

4.YÖNTEM METHOD



Şekil 4.1: Sistem Algoritması

Daha önce yapılan çalışmalar ve elde edilen veriler ışığında, bu çalışmada HAM10000 veri seti ile 7 farklı cilt lezyonunun eğitimi PHYTON dilinde keras üzerinden tensorflow kütüphanesi ile sağlanmıştır. Çalışmada weight (ağırlık) değeri melanom üzerine sistemi hassaslaştırmak adına, melanom için özelleştirilmiştir. Sistem melanoma hassaslaştırılmak üzerine tasarlansada çalışmada yapılmak istenen ilk aşama, lezyon tipinin iyi huylu/kötü huylu analizidir. Bu nedenle sistemden 3 farklı tahminde bulunması istenmiştir. Sistemin tahminde bulunması olasılık fonksiyonunu beraberinde getireceğinden aktivasyon fonksiyonu olarak softmax seçilmiştir. Sistem; HAM10000 veri setinin içerdiği bazal hücreli karsinom, dermatofibrom, aktinik keratoz, benign keratoz, melanom, melanositlik nevüs ve vasküler cilt lezyonları için eğitilmiştir. Sistem eğitimi için veri setinden 50 görselde sentetik veri elde etme çalışması yapılmıştır. Her veriden 6000 adet görsel oluşacak Şekilde sistemdeki sentetik veri oluşturma işlemi gerçeklemiştir. Sentetik veriler gerçek verilerin yatay eksende döndürülme, dikey eksende döndürülme, uzaklaştırma, yakınlaştırma, parlaklığını değiştirme, 90 derece döndürme 180 derece döndürme gibi işlemler yaparak elde edilmiştir. Eğitimin ardından eğitimi değerlendirebilmek adına bir hata matrisi elde edilir. Hata matrisi, sistemde eğitime katılan verileri sistemin ne derece öğrendiğini göstermek adına kullanılmaktadır. Eğitim işleminin ardından eğitilmiş sinir hücrelerini bir web uygulamasında kullanabilmek adına tensorflow.jpgfonksiyonu kullanılmıştır.

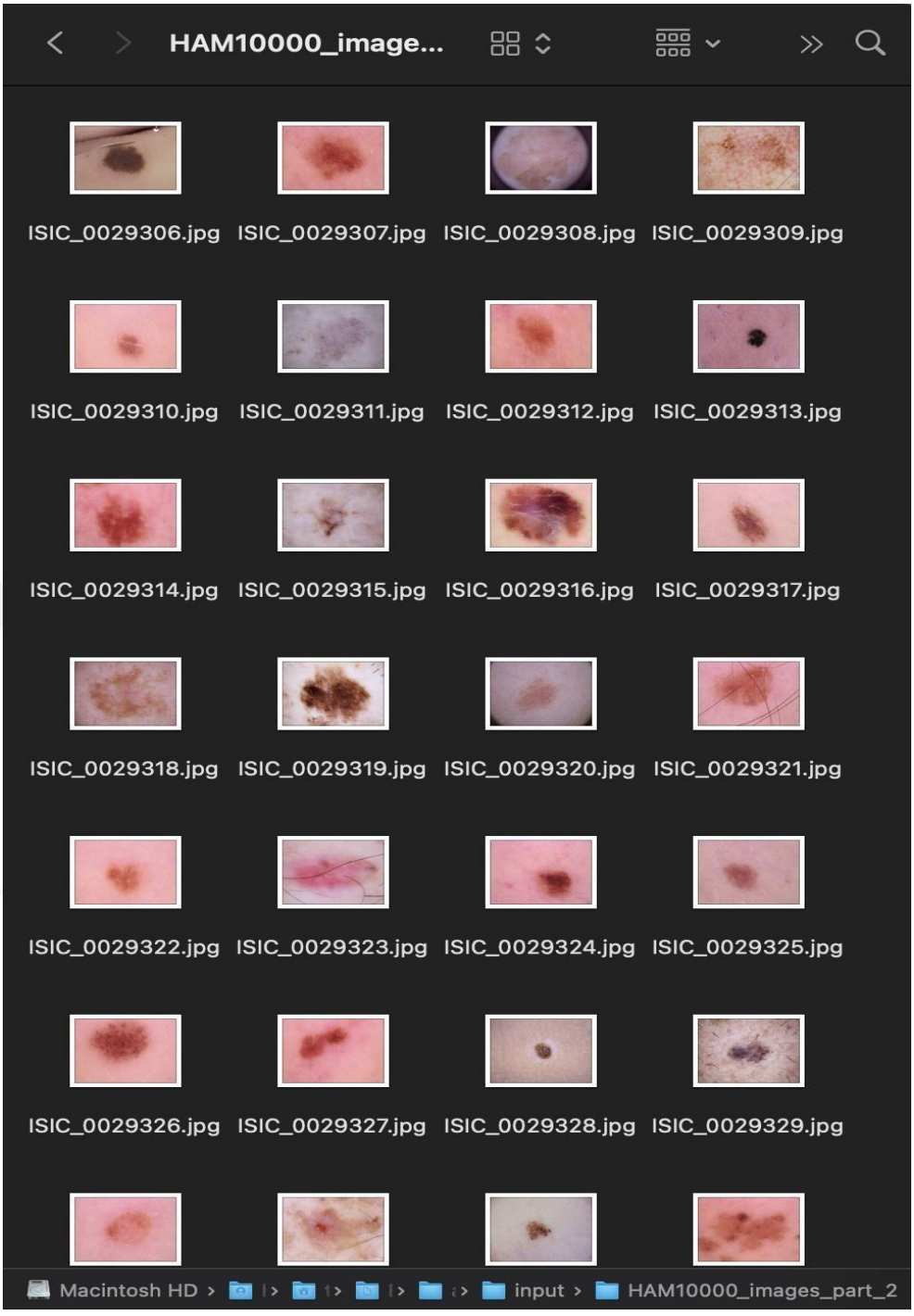
Bu çalışmada kaggle içerisinde bulunan açık kaynak veri setlerinden HAM10000 veri seti kullanılmıştır. Sebebi, 7 farklı lezyon tipini bilgisayar öğrenmesi ile inceleme fırsatı sunmasıdır.

Tablo 4.1: Veri setleri Tablosu

Veri Seti	Lisans	Toplam Görsel	Patolojik Doğruluk Oranı	akiec	bcc	bkl	df	mel	nv	vasc
PH2	Araştırma Eğitim	200	20,5%	-	-	-	-	40	160	-
Atlas	Yok	1024	Bilinmeyen	5	42	70	20	275	582	30
ISIC 2017	CC-0	13786	26,3%	2	33	575	7	1.019	11.861	15
Rosendahl	CC BY-NC 4.0	2259	100,0%	295	296	490	30	342	803	3
ViDIR Legacy	CC BY-NC 4.0	439	100,0%	0	5	10	4	67	350	3
ViDIR Current	CC BY-NC 4.0	3363	77,1%	32	211	475	51	680	1.832	82
ViDIR MoleMax	CC BY-NC 4.0	3954	1,2%	0	2	124	30	24	3.720	54
HAM10000	CC BY-NC 4.0	**10015**	53,3%	327	514	1.099	115	1.113	6.705	142

Tablo 4.1"de farklı veri setlerinin içerdiği toplam görsel sayısı ve farklı cilt lezyonlarına ait görüntü kümelerindeki veri sayısına yer verilmiştir. Çalışmada kullanılan HAM10000 veri setinin kullanım nedenlerinin başında farklı modaliteler tarafından elde edilen ve saklanan farklı popülasyonlardan dermatoskopik görüntüler toplanarak hazırlanmış olması gelmektedir. Veri seti, 10015 farklı görüntüden oluşmaktadır. Dolayısıyla insanın/makinanın tespiti konusunda bize bir fikir sunabilmek adına yeterli içeriğe sahiptir. Bunlardan 6705 tanesi melanositik nevüsler, 1113 tanesi melanom, 1099 tanesı bening keratoz, 514 tanesi bazal hücreli karsinom, 327 tanesi aktinik keratoz, 142 tanesi vasküler cilt lezyonu ve 115 tanesi defibroma olarak sınıflandırılmaktadır.

Şekil:4.2: HAM10000 veri seti örnek bazal hücreli karsinom



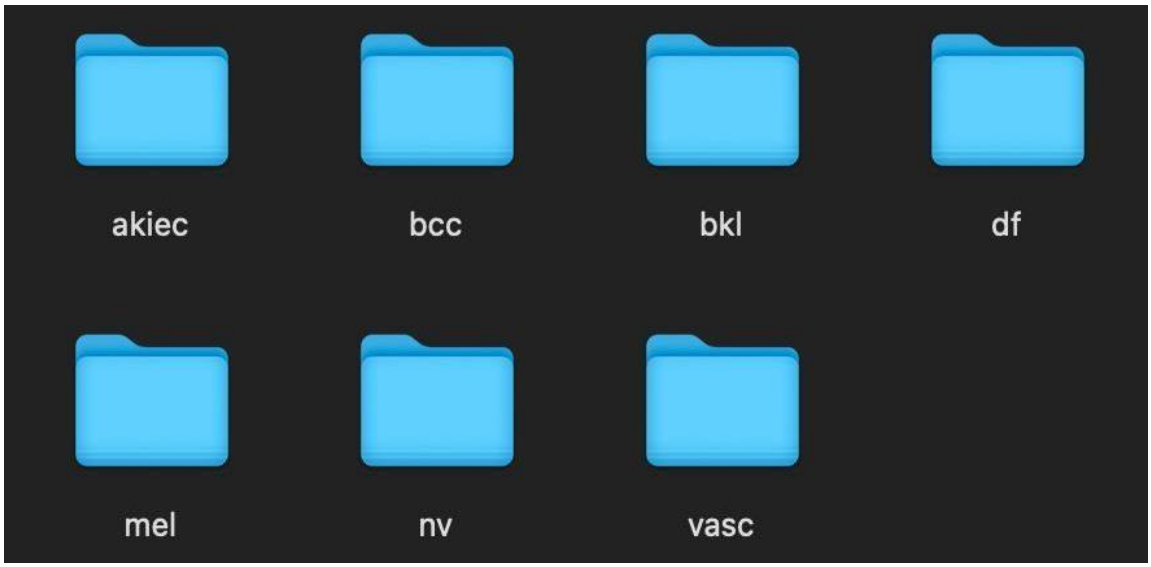
Şekil 4.3: HAM10000 veri seti görselleri

HAM10000_metadata

lesion_id	image_id	dx	dx_type	age	sex	localization
HAM_0000118	ISIC_0027419	bkl	histo	80.0	male	scalp
HAM_0000118	ISIC_0025030	bkl	histo	80.0	male	scalp
HAM_0002730	ISIC_0026769	bkl	histo	80.0	male	scalp
HAM_0002730	ISIC_0025661	bkl	histo	80.0	male	scalp
HAM_0001466	ISIC_0031633	bkl	histo	75.0	male	ear
HAM_0001466	ISIC_0027850	bkl	histo	75.0	male	ear
HAM_0002761	ISIC_0029176	bkl	histo	60.0	male	face
HAM_0002761	ISIC_0029068	bkl	histo	60.0	male	face
HAM_0005132	ISIC_0025837	bkl	histo	70.0	female	back
HAM_0005132	ISIC_0025209	bkl	histo	70.0	female	back

Şekil 4.4: HAM10000 dataset tablosu

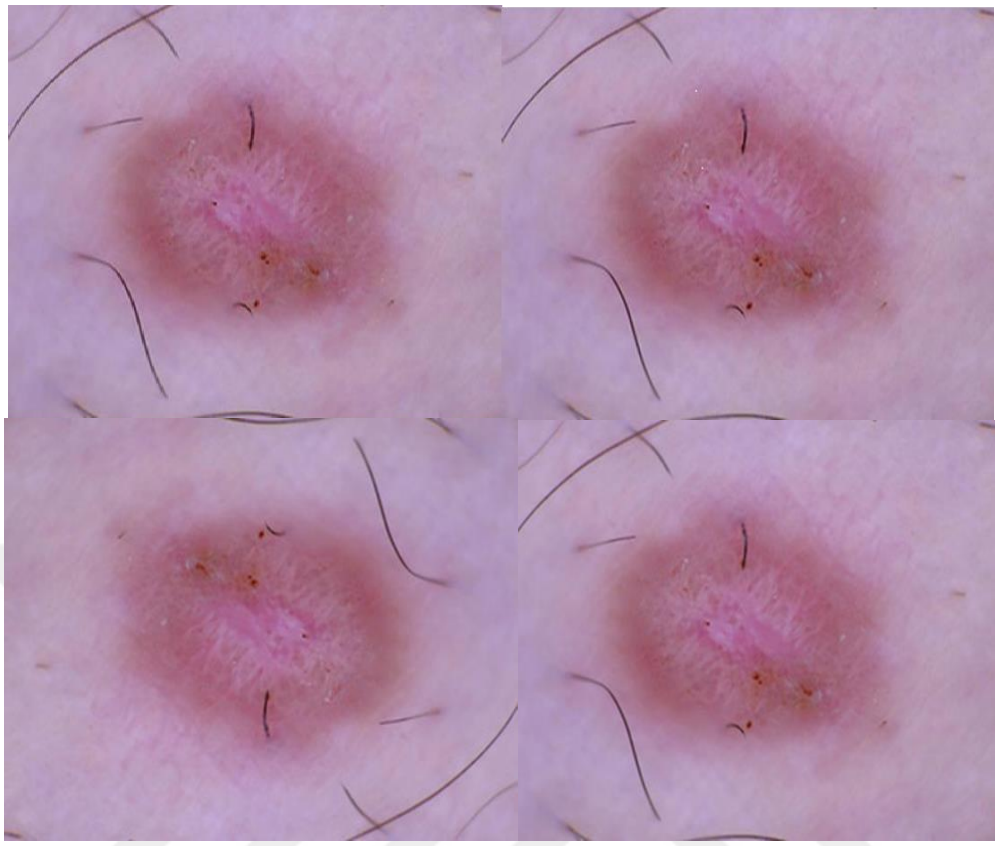
HAM10000 veri setinin dataset tablosunda her bir görselin lezyon türü, kaç yaşında ve hangi cinsiyette bir bireye ait olduğu ve bireyin hangi vücut bölümünden alındığı listelenmiştir



Şekil 4.5: Val_dir ve Train_dir içindeki lezyon alt klasörleri

Çalışmada ilk olarak kullanılacak kütüphaneler tanımlanmıştır. Ardından base_dir adında bir klasör ve bu klasör içerisinde val_dir ve train_dir alt klasörlerini oluşturacak kodlamalar terminal üzerinden yazılır. Val_dir ve train_dir içerisinde de, 7 lezyon türü için alt klasörler oluşur (şekil 4.5). Bu klasörlerin içerisine HAM10000 data setinden, dataset tablosundaki bilgilerdoğrultusunda veriler çekilir.

Veri setinin fazla sayıda veri içermesi modelin iyi eğitilmesi için oldukça önemlidir. Veri sayısının fazla olması ile birlikte veri çeşitliliği de olmak zorundadır. Aksi takdirde ezberleme sorunu ortaya çıkmaktadır. İyi öğrenme beraberinde yüksek başarımı da getirecek, fazla olması model başarımı artımının yanı sıra ezberlemeyi (overfitting) de önleyecektir. Verinin boyutunun az olduğu durumlarda veri setini artırmak için kullanılan yöntemler arasında ilk sırada sentetik veri üretilmesi gelmektedir. Bu çalışmada da sentetik veri üretiminden faydalanılmıştır. Sentetik veri üretirken dikkat edilmesi gereken başlıca unsurlardan biri verileri gerçek veri karmaşasında üretmektir. Bu da ezberlemeyi minimuma indirmek için yapılması gereken işlemlerin başında gelmektedir. Veri setini bu prensibe ne kadar uydurabilirsek, başarım da o oranda artacaktır. Bu anlamda verinin farklı versiyonlarının veri setinde bulunması, veri setinin prensibe uygunluğunu artıracaktır. Bu nedenle verinin farklı varyasyonlarını, farklı boyuttaki görünümleri veri setine dâhil edilerek veri setinin boyutu başarımı artıracak Şekilde büyütülmüştür. Sentetik veri ile verisetinin boyutunun artırılması belli oranda overfitting oluşumunu önlese bile, genelde yetersiz kalmaktadır. Çünkü oluşturulan sentetik veriler, belli miktarda farklılaşma olsa bile mevcut veri seti ile korelasyonunu korumaya devam edecektir. Bu nedenle veri seti artırma yöntemi sırasında alakasız verilerde öğrenim setine eklenerek sistemin görselleri öğrenmek yerine ezberlemesi engellenmiştir. Veri seti artırımı ve eğitimiçin 50 adet görsel HAM10000 veri seti içerisinden seçilmiştir. Ardından bu veriler için döndürme, genişletme, daraltma, yakınlaştırma, yatay çevirme, dikey çevirme içlemleri uygulanarak veri çoğaltma işlemi gerçekleştirilmiştir.



Şekil 4.6: Veri artırım örneği

Veri hazırlığının ardından eğitim aşamasına geçilmektedir. Eğitim aşamasının ilk adımı eğitim parametrelerinin belirlenmesidir.

Öğrenim boyunca ilk 23 adımı sistemin atlaması istenmiştir. Çünkü sistem her katmanda bir önceki katmanda öğrendiği bilginin üstüne yeni bilgiler edinerek eğitilmektedir. Dolayısıyla ilk 23 katmanda öğrendikleri sistemin ham bilgilerini ifade etmektedir. Sistem 3 farklı lezyon tahmini yapacak Şekilde eğitilecektir. Sistemin 7 farklı lezyon için sınıf ağırlığı belirlenmiştir. sınıf ağırlığı, sistemin ağırlık çarpanıdır. Dolayısıyla hangi lezyona ne kadar duyarlı olacağı verilen değerle ilgili olarak değişiklik gösterecektir. Oluşturulan sistemin melanoma duyarlı olması istendiği için melanom class weight diğer lezyonlardan farklı ve büyük seçilmiştir. Melanomun sınıf ağırlık değeri, diğer cilt lezyonlarına göre yüksek tutulmuştur. Bu sayede yapay zekânın öğrenim sırasında melanoma karşı duyarlılığı artırılmıştır. Çalışmada ulaşmak istenen nihai sonuç 7 farklı cilt lezyonu içerisinden sistemin üç farklı tahminde bulunarak lezyonu doğru tespit etmesini sağlamak olduğundan yani sistemin bir olasılık çıktısı vermesi istendiğinden kullanılan fonksiyon Şekil 2.6'da görülen softmax aktivasyon fonksiyonu

olmuştur. Bir yapay sinir hücresinde eğitim, ağırlık ve bias değerlerinin güncellenmesi ile oluşmaktadır. Bias değerini sistemin hata değeri olarak ifade etmek mümkündür. Sistemdeki değişikliklere karşı gösterdiği hassasiyet bias değeri olarak adlandırılmaktadır. Bu parametreler değiştirilerek en optimal öğrenim değerleri elde edilmektedir. Tüm veri setlerinin bir sinir ağı boyunca bir kez gidip gelmesi (tur atması) işlemine epoch adı verilmektedir. Epoch değeri her çalışma için farklılık göstermektedir. Epoch değerinin çok olması ezberlemeye, az olması düşük performansa neden olabileceğinden epoch değeri de çalışma boyunca değiştirilerek sistem için optimum değer bulunmalıdır. Her epoch (adım) da eğitilecek veri boyutu batch size olarak ifade edilmektedir.

Şekil 4.7: Eğitim aşaması

Şekil 4.7"de pembe ile işarelenen kısım Epoch (adım), mavi ile işaretli kısım loss (hata oranı), yeşil ile işaretlenen kısımlar ise yapay zekânın 3 tahmini için doğruluk oranlarını sunmaktadır. Şekilde görüldüğü üzere aşağı doğru devam ettikçe epoch (adım) sayısı artmaktadır.

Adım sayısı arttıkça mavi ile gösterilen loss (hata) değerleri azalmakta, yeşil ile gösterilen 3 lezyon tahmini için doğruluk oranı artmaktadır. Bunun nedeni yapay zekânın her öğrenim adımında bir önceki katmanda öğrendiklerini ekleyerek sonraki adımdaki öğrenme işlemine devam ediyor olmasıdır. Bu sayede öğrenim ileri beslemeli olarak devam edebilmektedir. Burada, veri seti, adım sayısı, sınıf ağırlığı, tur sayısı gibi daha önce tanımladığımız parametreler ile eğitim aşaması başlatılmaktadır. Bu parametreler ışığında yapay zekâ sinir ağı oluşturmaya başlar. Callbacks ile her tur sonunda oluşan yapay sinir ağı kaydedilir. Kaydedilen sinir ağı üzerinden diğer tur başlar. Bu sayede eğitim ileri beslemeli olarak devam eder. Böylece terminal içinde çalışan bir yapay sinir ağı oluşturmuş oluruz fakat bu yapay sinir ağını bir ortamda çalışarak sonuç vermesini sağlayacak değer matrisine dönüştürmemiz gerekmektedir. Yani yapay zekânın çalıma mantığını bize verecek bir matrise ihtiyaç duymaktayız. Bu matrise hata matrisi denmektedir. Hata Matrisi kullanarak yaptığımız classification (sınıflandırma) modeliyle, performansı değerlendirebiliriz. Aynı zamanda hata matrisi yardımı ile farklı metrik değerlerimizi de bulabilir ve model başarımızı yaptığımız işleme göre detaylı şekilde değerlendirebiliriz. Öğrenme sırasında validation loss ile test loss arasındaki fark açılmaya başladığı durumda model ezberliyor (overfitting) ya da gürültüyü öğreniyor (noisy) demektir. Bu durumda validation error artmaya başladığında eğitim durdurulur ve bir önceki adıma geri dönülür. Bir önceki adıma geri dönebilmek için eğitim esnasında her bir öğrenme adımında (epoch) bir önceki adımın verileri saklanmış olmalıdır. Şekil 4.7'de görüldüğü üzere her epochda (adımda) kayıp giderek azalmaktadır. Burada yapmak istediğimiz, 7 farklı cilt lezyonunun arasındaki farkı doğru kavrayarak eğitim sonunda gösterilen lezyon için yapay zekânın üç parametrede benzerlik kurması sağlanarak doğru çıktıyı vermesidir.

Hata matrisi sayesinde terminal üzerinde çalışan yapay zekânın çalıma mantığı bilgisayar diline çevrilmiştir. Yapay zekânın karar verme mekanizmasının verileri bir hata matrisine çevrilmiştir. Bu hata matrisindeki verileri kullanarak bilgisayar programları ile yapay zekâyı taklit ederek karar verebilir ve bunu internette çalışabilir bir koda çevrilebilir bir hale getirmek mümkün kılınmıştır. İnternette çalışabilir bir kod oluşması için bilgisayara tensorflow.js kurulmuştur. Bunu yapmak için terminale *pip* install *tensorflow.js* komutu girilir. Phyton terminaline geçilir. Ardından;

import tensorflowjs as tfjs os.mkdir('tfjs_dir')

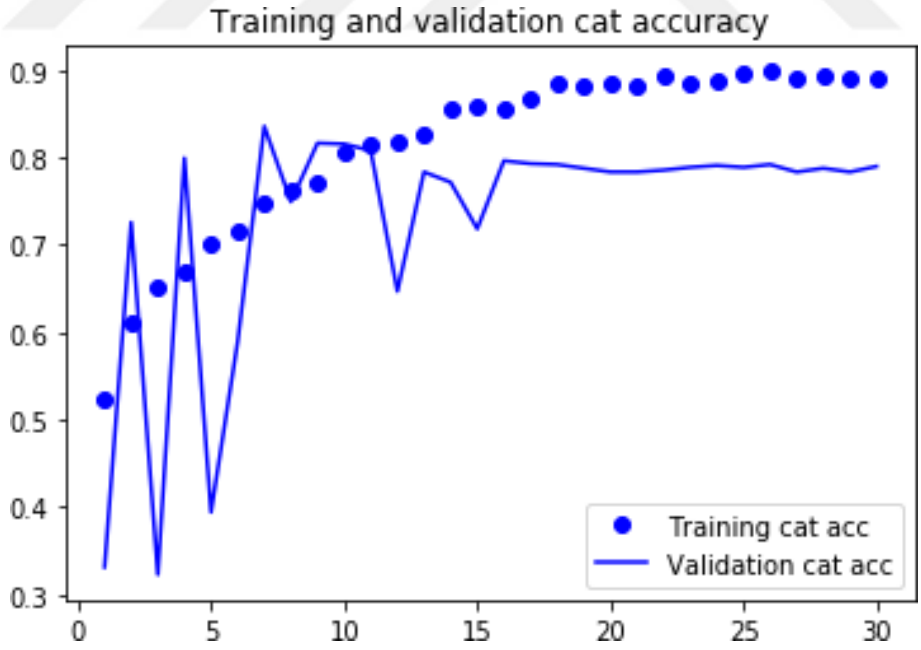
tfjs.converters.save_keras_model(model, 'tfjs_dir')

kodu terminale yazılır.

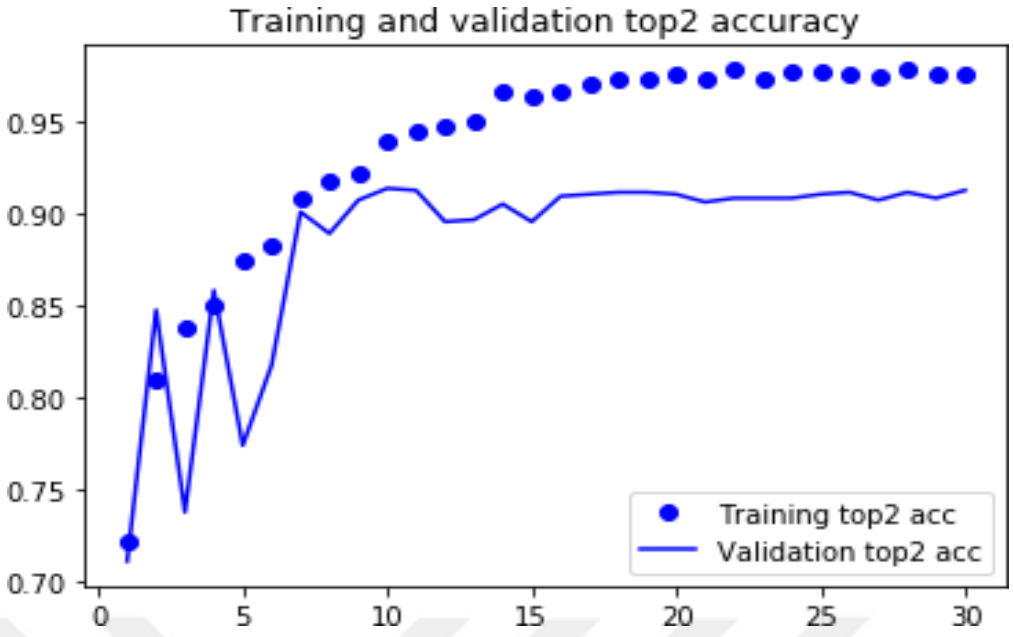
Bu kod ile tensorflow.js kütüphanesi yüklenir ardından tf.js_dir isimli bir klasör açılır. Tensorflow.js kütüphanesi kullanılarak save_keras_model metodu çağırılır. Bu sayede, önceden oluşturulmuş hata matrisi doğrultusunda tarayıcı üzerinde çalışabilen bir yapay zekâ modeli üretilmiş olur. Bu kod ile yapay zekânın arka planda çalışarak ürettiği kodların herhangi bir tarayıcı ile çalıştırılabilir hale gelmesi sağlanmıştır. Böylelikle bu kod dosyası kullanılarak, başka herhangi bir yerde bu yapay sinir ağları kullanılabilir hale gelmiştir.

5. SONUÇLAR

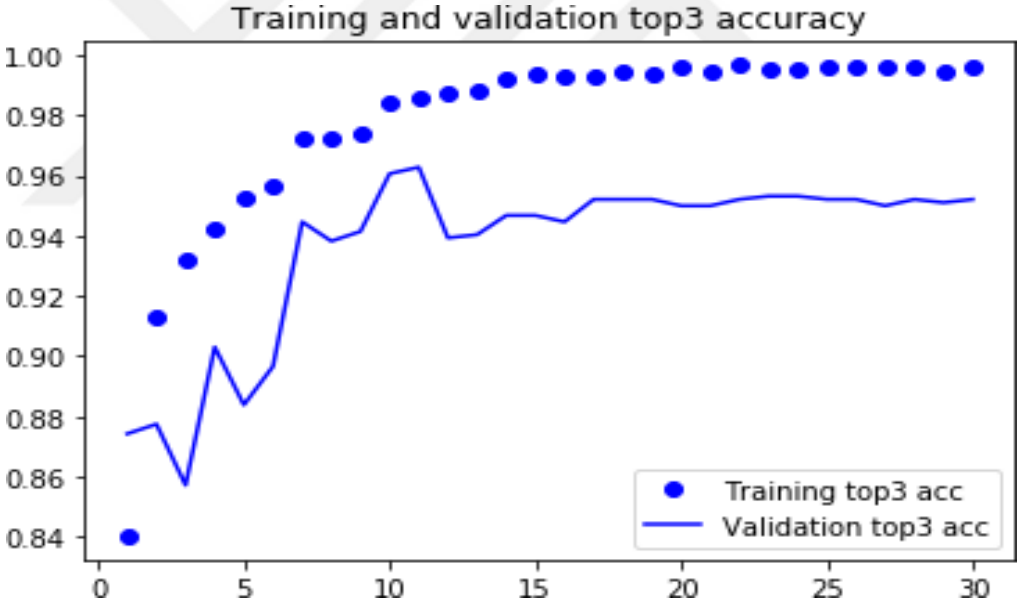
Çalışmaya başlarken hedeflenen nihai sonuç, doktorlar için cilt lezyonuna sahip hastaların verileri üzerinde önceliklendirme yapabilmelerine olanak tanıyacak bir sistem geliştirmektir. Sistemin 7 farklı cilt lezyonunu, gösterilen dataset ile yapay zekâya doğru besleme yaparak ayırt edebilecek bir sistem geliştirmesi planlanmıştır. Sistemin Şekil 5.5'te gösterilen ,jpg formatındaki bir cilt lezyonu için 3 farklı tahminde bulunması sağlanmıştır. Tahminleri yapay zekanın ezberleyerek değil öğrenerek yaptığı Şekil 5.1, Şekil 5.2 ve Şekil 5.3'deki verilerle ortaya konmuştur. Şekil 5.1'de 3 tahmin için de doğruluk oranlarına ver verilmiştir. Training cat acc (eğitim doğruluk oranı) ve validation cat acc (test doğruluk oranı) birbirine ne kadar yakınsa istenilen uyuma o kadar yakın bir çalıma olduğu anlamına gelmektedir. Şekil 5.1, Şekil 5.2 ve Şekil 5.3'te sırasıyla 1, 2 ve 3. tahminler için test ve eğitim verilerinin doğruluk oranlarına yer verilmiştir. Şekillerde, x eksenleri epoch (adım) sayısını, y eksenleri ise doğruluk oranlarını vermektedir. Eğitim doğruluk oranı ile test doğruluk oranları birbirine ne kadar yakın ve adımlar ilerledikçe doğruluk oranları ne kadar yüksekse yapay zekâ eğitimi o kadar başarılı sonuç verir.



Şekil 5.1: 1. Tahmin için doğruluk oranı



Şekil 5.2: 2. Tahmin için doğruluk oranı

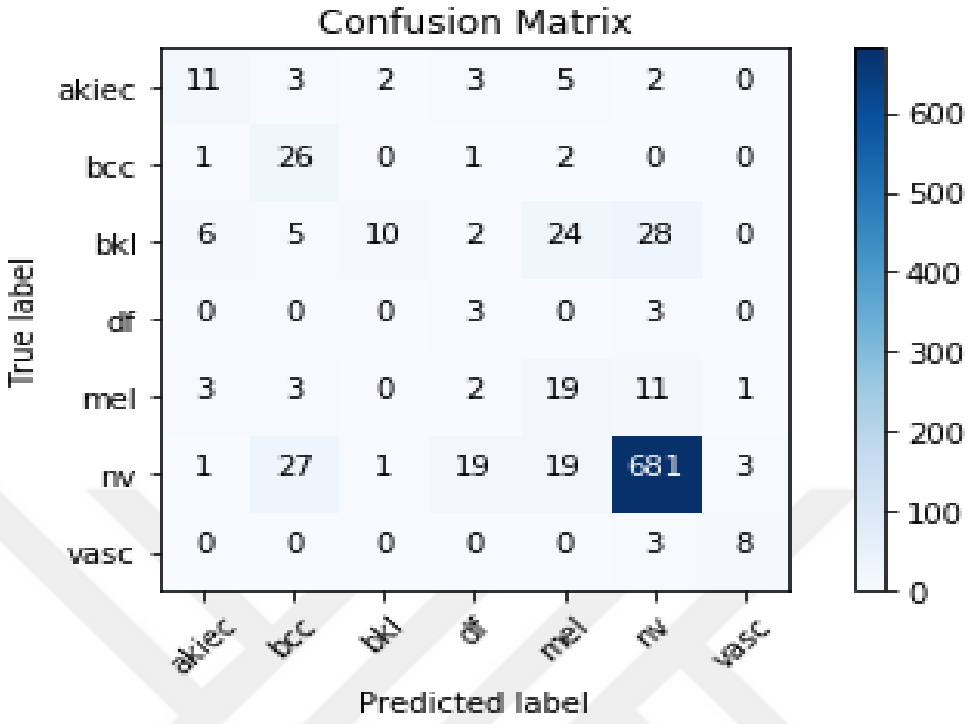


Şekil 5.3: 3. tahmin için doğruluk oranı

HAM10000 veri seti ile öğrenmesi yüksek oranda sağlanan veri setinin öğrenim oranlarını test etmek için doğruluğundan emin olunan veriler kullanılmıştır. Javaspt ile oluşturulan yapay zeka tabanlı web server üzerinden jpg formatında farklı cilt lezyonlarını sisteme yükleyerek 3 farklı tahmin değeri almak mümkün kılınmıştır.

Bu sayede sistem çıktısı doğruluğu 3 parametreli olarak kontrol edilmiştir.

.



Şekil 5.4: Hata matrisi çıktısı

Şekil 5.4'te sistemin validasyonunun hata matrisi çıktısı verilmiştir. Validasyon için kullanılan veri sayısı, eğitim için kullanılan veri sayısının %17'si Akiec (aktinik keratoz), bcc (bazal hücreli karsinom), bkl (bening keratoz), df (dermofibrom), mel (melanom), nv (melanositik nevüs), vasc (vasküler cilt lezyonu)na ait doğruluk oranlarını vermektedir. Bu matris incelendiğinde satırdaki sayıların toplamı lezyona ait toplam görüntü sayısını ifade etmektedir. Burada amaç, satır ve sütundaki aynı lezyon türündeki veri sayısının en yüksek değerde olmasıdır. Bu tahminin doğruluk oranının yüksek olduğu anlamına gelmektedir. Sistemde vasc için 11 veri yüklenmiştir. Sistem 11 veriden 8'ini vasc, 3 adedini nv olarak yorumlamıştır. Sistemin nv için doğruluk oranı %72,72"dir. Aynı Şekilde melanom için toplam görüntü sayısı 39'dur. Sistem bu görsellerin 19'unu melanom olarak doğru tahmin etmiştir.

Sistem 7 farklı lezyon tipi için yüksek oranda doğru sonuçlar verse de kesin sonuç ve tanı için kullanılması mümkün değildir.

Şekil 5.5: Melanom olduğu kesin olan ciltlezyonu görüntüsü

Şekil 5.5'de melanom olduğu kesin olan bir lezyon bulunmaktadır. Yapay zekâ tabanlı sisteme bu görsel yüklendiğinde verdiği çıktı sonuçlar kısmında gözükmektedir. Görüldüğü üzere lezyon görseli yüklendiğinde sistem 3 farklı lezyon tahmini yapmaktadır. Eğitim sonunda yüklenen görüntü üzerindeki başarısı, %0.03'lük bir yanılma payına sahiptir.
Sistem HAM10000 eğitim seti ile eğitildiğinden yüklenen görsele göre doğruluk oranı değişkenlik göstermektedir.

.

Cilt Lezyonu Görüntüsü Yükle

Sonuçlar

melanoma-nedir.jpg
nv, Melanositik Nevüs: 0.998
mel, Melanom: 0.002
bkl, İyi huylu keratoz: 0.000

Şekil 5.6: Melanom olduğu kesin bilinen görselinsistem çıktısı

6. TARTIŞMA

Daha önce melanom tespitine dair yapılmış çalışmalarda genel itibariyle filtreleme işlemi ve görüntü işleme konuları üzerinde durulmuştur. Filtreleme işlemleri gürültüyü azaltmak amacıyla tüm çalışmalarda kullanılabilmektedir. Fakat kişiler laboratuvar ortamında değil kendi imkânları ile sisteme yükleyecekleri. jpg formatındaki görseller ile çıktı alacakları için gürültünün sıfırlanması işlemine bu çalışma tabi değildir. Daha önce yapılan çalışmalar genel itibariyle sadece melanomun tespiti üzerine çeşitli veri setleri kullanılarak tespitte bulunurken HAM10000 veri setini 7 farklı cilt lezyonu üzerine eğiterek yüklenen lezyon görseli için 3 ayrı yüzdesel lezyon ihtimali çıktısı alınmaktadır. Çalışmanın 3 lezyon tahmini yapma sebebi kötü huylu lezyonu türüne bakmaksızın doğru tahmin etmesini sağlamaktır. Bu durum çıktının kötü huylu/iyi huylu lezyonu algılaması konusunda doğruluk oranını yükseltmektedir.

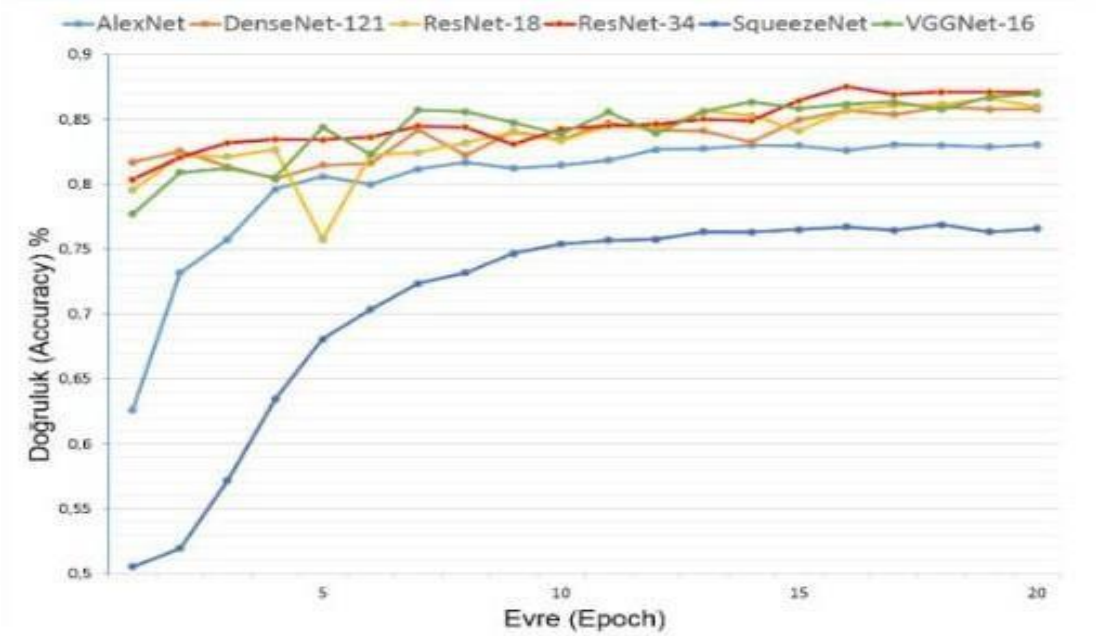
Tablo 4.1'de farklı veri setleri içerisindeki lezyon tiplerinde var olan veri sayıları verilmiştir. HAM10000 veri seti içerdiği 100015 veri sayısı ile çalışmanın eğitim ve test setlerini oluşturmak açısından yeterli veriyi sağlamaktadır. Bununla birlikte eğitim verisini daha doğru sonuçlar elde edebilmek adına sentetik veriler elde etmek için kullanarak her lezyon tipinden 6000 jpg görüntüsü elde edilmiştir. Yapılan çalışmalar göstermektedir ki veri setinin büyüklüğü sistemin eğitimi için oldukça önemlidir.

Şekil 5.4'te hata matrisi incelendiğinde sistemin 7 farklı lezyon tipi için hata matrisi çıktısında da görüldüğü üzere akiec doğruluk oranı %42.30, bcc doğruluk oranı %86.66, bkl doğruluk oranı %13.33, df doğruluk oranı %50, mel doğruluk oranı %48.71, nv doğruluk oranı 68.51 ve vasc doğruluk oranı %72.72 olarak sonuçlanmıştır.

Melanom doğruluk oranı %48.71 olmasına karşın melanom için sistemin yaptığı en yüksek oranlı ikinci tahmin nv olmuştur. Bu da sistemin kötü huylu lezyon türüne hassasiyetinin yüksek olduğu, sistemin amacına uygun olarak çalıştığı anlamına gelmektedir.

Oluşturulan yapay zekâ ile melanom tespiti projesinde hedeflenen sonuç; doktorlara erken teşhis konusunda destek olabilecek bir sistem oluşturmaktı. Bu sistem var olan veri setinin çoğaltılması ile mümkün olan en büyük veri seti ile beslenerek maksimum verim ile oluşturulmuştur.

Sistem, doğru besleme ile en doğru sonucu verdiği noktadadır. Şekil 3.3'te YOLO ve Grabcut kullanılarak kıl temizleme tekniği ile yapılan çalınmadaki doğruluk oranları verilmiştir. Kıl segmentasyon işlemi doğruluk oranını arttırıyor gibi görünse de bu çalışmada .jpg formatında çalışan bir sistem tasarlandığı ve bu sistemin insanların rahatça kullanabilmesi hedeflendiği içinkıl temizleme tekniği kullanmak segmentasyonu arttırırken kişisel kullanımdaki doğruluk oranını düşürecektir. Tablo 4.1'de bulunan diğer veri setleri kullanılarak yapılan çalışmalarda veri seti içerisindeki veri sayısı daha kısıtlı olduğundan sistem eğitiminde ezberleme olma ihtimali yüksek olacaktır. Bu nedenle HAM10000 veri seti ile artırım yaptırarak çalışmak sistemi ezberlemeden uzaklaştırmıştır. Bu durum şekil 5.4'te hata matrisi ile ortaya konmuştur. Literatürde bulunan HAM10000 veri seti ile yapılmış çalışmalara bakıldığında farklı öğrenim parametreleri ile AlexNet, DenseNet121, Resnet118,SqueezeNet ve Vggnet-16 kullanılarak eğitim yapılmıştır. Şekil 6.1'de bu modeller ile yapılan eğitimlerin doğruluk sonuçları verilmiştir. Şekil 5.1'de görüldüğü üzere MobileNet yapay sinir ağı modeli kullanarak yapılan çalışmada doğruluk oranı 0.9 ile 1.0 arasında iken literatürde HAM10000 veri seti ile yapılan farklı bir çalışmada kullanılan diğer yapay sinir ağı modelleri ile doğruluk oranı 0.85-0.9 arasında seyretmektedir.



Şekil 6.1: AlexNet,Denset121,Resnet18,SqueezeNet ve Vggnet-16 için adım-doğruluk oranı[41]

Yapılan eski çalışmalarda görüldüğü üzere, yapay sinir ağı oluşumu için kullanılacak en hızlı veen doğru sonuç veren model MobileNet‟tir. Bu çalışma esnasında MobileNet modelinin kullanım amacı da, literatür kısmında detayları verilen modellerin sonuçlarını en yüksek doğrulukla veren model olmasından ileri gelmektedir.

KAYNAKÇA

1. Haenssle, HA. Fink, C. Schneiderbauer, R. et al. Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists. Ann Oncol 2018; 29: 1836-42.

2. Tschandl. P, Codella. N, Akay. BN et al. Comparison of the accuracy of human readers versus machine-learning algorithms for pigmented skin le- sion classification: an open, web-based,international, diagnostic study. Lan-cet Oncol 2019; 20: 938-47.

3. Dorj U.-O, Lee K.-K, Choi J.-Y and Lee M. "The skin cancer classi- fication using deep convolutional neural network," Multimed. Tools Appl., vol. 77, no. 8, pp. 9909–9924, 2018, doi: 10.1007/s11042-018-5714-1

4. Esteva A, Kuprel B, Novoa RA et al. Dermatologist-level classification of skin cancer with deep neural networks. Nature 2017; 542: 115-8.

5. Dark Skin Turns UV Rays To Heat, Rendering Them Harmless"". çevrimiçi: https://scienceblog.com/74575/dark-skin-turns-uv- rays- heat-rendering-harmless/ (Ekim 2022)

6. Smith R. A., Andrews K. S., Brooks d., Fedewa S. A., Manassaram- Baptiste, Saslow D.,

7. Brawley O. W., and Wender R. C., "Cancer screening in the united states: a review of current american cancer society guidelines and current issues in cancer screening," CA: a cancer journal for clinicians, vol. 68, no. 4, pp. 297–316, 2018.

8. "Medikal Akademi"" (2018) Çevrimiçi: https://www.medikalakademi.com.tr/melanom-nedir-cilt-kanseri-melanomun belirtileri-ve-tedavisi/ (5.10.2022)

9. Yalcin H. (2015) "ÇeĢitli özniteliklerle kötü huylu melanom karakterizasyonu characterization of melanomas using a variety of features," Görsel Zeka Laboratuarı Elektronik ve Haberleşme Mühendisliği Bölümü İstanbul Teknik Üniversitesi

10. “American Cancer Society,” 2018 Çevrimiçi: https://www.cancer.org/cancer/ melanoma-skin- cancer/about/key-statistics(Ekim 2022)

11. Dorj, U.-O., Lee, K.-K., Choi, J.-Y., Lee, M., 2018. The skin cancer classification using deep convolutional neural network. Tools Appl. 77 (Apr. (8)), 9930–9944.

12. Siegel R. L., Miller K. D., and Jemal A. "Cancer statistics, 2019," CA:a cancer journal for clinicians, vol. 69, no. 1, pp. 7-34, 2019.

13. ‟Melanom (Malign Melanom) Çevrimiçi: https://www.derikanseri.org/melanom Erişim Tarihi: 6.10.22

14. Korotkov K. and Garcia H. "Computerized analysis of pigmented skin lesions: a rview," Artificial intelligence in medicine, vol. 56, no. 2, pp. 69- 90, 2012.

15. Chollet F. "Keras," https://github.com/keras-team/keras/tree/master/docs, vol. 25, p. 2017, 2015.

16. Deng J. Dong W, Socher W. "Image- net: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition, 2009, pp. 248-255:

17. ‟Deri Biyopsisi (Punch Biyopsi)‟‟‟ Çevrimiçi: https://drsedaozturk.com/2014/08/23/deri-biyopsisi-punch-biyopsi/Erişim tarihi: 20.10.22

18. Ünver H. M. and Ayan E., "Skin lesion segmentation in dermoscopic images with combination of YOLO and grabcut algorithm," Diagnostics, vol. 9, no. 3, p. 72, 2019.

19. Lopez A. R. , Giro-i-Nieto X., Burdick J , and Marques o, "Skin lesion classification from dermoscopic images using deep learning techniques," in 2017 13th IASTED International Conference on Biomedical Engineering (BioMed), 2017, pp. 49-54: IEEE.

20. ‟Yapay Zeka, Makine Öğrenmesi ve Derin Öğrenme Kavramları Arasındaki Fark Nedir?‟‟‟ Erişim adresi(Ekim 2022) https://evrimagaci.org/yapay-zeka-makine- ogrenmesi-ve-derin- ogrenme-kavramlari-arasindaki-fark-nedir-8889

21. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks, Communications of the ACM 2017; 60(6), 84–90.

22. Brinker TJ, A Hekler ,Enk AH, Berking C, Haferkamp S, A Hausc- hild , Roman C, Berking C. Deep neural networks are superior to dermato- logists in melanoma image classification,European Journal of Cancer 2019; 119, 11–17.

23. Hosny K.M, Kassem MA , Foaud MM . Classification of skin lesions using transfer learning and augmentation with AlexNet, PloS one 2019; 14(5), 217-293. Esteva A, Kuprel B, NovoaRA, Ko J, Swetter SM, Blau HM, Thrun S. Dermatologist-

24. ‟level classification of skin cancer with deep neural networks‟‟‟ Erişim adresi (Kasım 2022) https://www.datasciencearth.com/underfitting- ve-overfitting/level classification of skin cancer with deep neural networks, nature 2017; 542(7639), 115–118.

25.‟Difference between Sigmoid and Softmaxactivationfunction ‟‟‟https://www.nomidl.com/deep-learning/what-is-the-difference-between-sigmoid-and-softmax- activation-function/

26. Kuşkapan. E, Çodur. M, Çodur. M. Y. (2022). Türkiye‟deki Demiryolu Enerji Tüketiminin Yapay Sinir Ağları ile Tahmin Edilmesi. Konya Mühendislik Bilimleri Dergisi, c. 10, s. 1, 72-84,2022

27. „‟Yapay sinir ağları‟‟‟ Erişim adresi: https://www.savasanadam.com/teknoloji/yapay-sinir- aglari/

28. Öztürk M. (Kasım 2020).„‟Phyton kütüphaneleri ve özellikleri‟‟‟ Erişim adresi: https://miracozturk.com/python-kutuphaneleri-ve- ozellikleri/

29. Alsaade. F , Theyazn H. H. Aldhyani , Research Article Developing a Recognition System for Diagnosing Melanoma Skin Lesions Using Artifi- cial Intelligence Algorithms, Hindawi Computational and Mathematical Methods in Medicine Volume 2021, Article ID 9998379, 20 pages https://doi.org/10.1155/2021/9998379

30. Ünver H.M, Ayan. E, Skin Lesion Segmentation in Dermoscopic Ima- ges with Combination of YOLO and GrabCut Algorithm,Diagnostics, Re- ceived: 30 May 2019; Accepted: 8 July 2019; Published: 10 July 2019

31. AYAN. E, Dermoskopik Görüntülerden Melanomanın Derin Evrişimsel Sinir Ağları ile Teşhisi, Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı

32. „"The Skin Cancer Foundation"" Erişim adresi: https://www.skincancer.org/skin-cancerinformation/melanoma#panel1-5

33. Bengio, Y., Learning deep architectures for AI. Foundations and trends in Machine Learning,2 (1), 1- 127, 2009

34. Assaf. Z, Andrew. R. J, Erik S. W, Andres N, Justin C, Ugur E, Brandi L. Cantarel,and Gaudenz Danuser,Interpretable deep learning uncovers cel- lular properties in label-free live cell images that are predictive of highly metastatic melanoma, cell systems

35. Halit. Ç, MobileNetV2 ve MobileNetV3 Tabanlıı Derin Öğrenme Yak-laĢımlarıı ile Cilt Kanserlerinin Sınıflandırılması,3rd International Confe- rence on Applied Engineering and NaturalSciences,July 20-23, 2022, Kon- ya, Turkey

36. Tschandl, P. Rosendahl,C. And Kittler. C. "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmen- ted skin lesions," Sci. data, vol. 5, no. 1, pp. 1–9, 2018

37. Lopez,A. Giro-i Nieto, X. Skin lesion classification from dermoscopic images using deep learning techniques. In 2017 13th IASTED international conference on biomedical engineering (BioMed), pages 49–54. IEEE, 2017

38. Rezvantalab A., Safigholi A, and Karimijeshni S. Dermatologist level dermoscopy skin cancer classification using different deep learning convo- lutional neural networks algorithms. arXiv preprint arXiv:1810.10348, 2018.

39. Tammineni S. & Subramanyam M. V. Early Detection of Skin Cancer Using Melanoma Segmentation technique,Journal of Medical Systems (2019) 43: 190 https://doi.org/10.1007/s10916-019-1334-1

40. Sakeena, M., Riasat, R., Sadiq, A. H., Ahmad, Z., and Riaz, F.,Dermoscopic image analysis using pattern recognition techniques from region of interest (ROI) for detection of melanoma. Int. Conf. Mach. Le- arn. Cybernet. (ICMLC), Ningbo, China 2017:162–168, 2017.

41. Qian S. "MobileNetV3 for Image Classification," in 2021 IEEE 2nd In- ternational Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), 2021, pp. 490–497. doi: 10.1109/ICBAIE52039.2021.9389905.

42. Ergün E, Kılıç K , Derin Öğrenme ile ArtırılmıŞ Görüntü Seti Üzerin- den Cilt Kanseri Tespiti,Black Sea Journal of Engineering and Science,Araştırma Makalesi (Research Article),Cilt 4 - Sayı 4: 192-200 / Ekim 2021,(Volume 4 - Issue 4: 192-200 / October 2021)

EKLER

```
from numpy.random import seed
seed(101)
import tensorflow

tensorflow.random.set_seed(101)

import pandas as pd
import numpy as np
#import keras
#from keras import backend as K

from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.metrics import categorical_crossentropy
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint

import os

from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
import itertools
import shutil
import matplotlib.pyplot as plt

os.listdir('./input')

# Yeni bir dizin oluştur
base_dir = 'base_dir'
os.mkdir(base_dir)


#[TEMEL DİZİN İÇİNDE KLASÖRLER OLUŞTUR]

# şimdi 'base_dir' içinde 7 klasör oluşturuyoruz:

# train_dir
    # nv
    # mel
    # bkl
```

```
    # bcc
    # akiec
    # vasc
    # df

# val_dir
    # nv
    # mel
    # bkl
    # bcc
    # akiec
    # vasc
    # df

# yeni klasörlerin adlarını birleştireceğimiz bir 'base_dir' yolu oluştur
# train_dir

train_dir = os.path.join(base_dir, 'train_dir')
os.mkdir(train_dir)

# # val_dir
val_dir = os.path.join(base_dir, 'val_dir')
os.mkdir(val_dir)


# # Her klasörün içinde her sınıf için ayrı klasörler oluşturuyoruz

# # train_dir içinde yeni klasörler oluştur
nv = os.path.join(train_dir, 'nv')
os.mkdir(nv)
mel = os.path.join(train_dir, 'mel')
os.mkdir(mel)
bkl = os.path.join(train_dir, 'bkl')
os.mkdir(bkl)
bcc = os.path.join(train_dir, 'bcc')
os.mkdir(bcc)
akiec = os.path.join(train_dir, 'akiec')
os.mkdir(akiec)
vasc = os.path.join(train_dir, 'vasc')
os.mkdir(vasc)
df = os.path.join(train_dir, 'df')
os.mkdir(df)

# val_dir içinde yeni klasörler oluştur
```

```
nv = os.path.join(val_dir, 'nv')
os.mkdir(nv)
mel = os.path.join(val_dir, 'mel')
os.mkdir(mel)
bkl = os.path.join(val_dir, 'bkl')
os.mkdir(bkl)
bcc = os.path.join(val_dir, 'bcc')
os.mkdir(bcc)
akiec = os.path.join(val_dir, 'akiec')
os.mkdir(akiec)
vasc = os.path.join(val_dir, 'vasc')
os.mkdir(vasc)
df = os.path.join(val_dir, 'df')
os.mkdir(df)

df_data = pd.read_csv('./input/HAM10000_metadata.csv')

df_data.head()

# bu bize her bir lezyon_id ile kaç tane görüntünün ilişkilendirildiğini söyleyecektir.
df = df_data.groupby('lesion_id').count()

# şimdi, kendisiyle ilişkilendirilmiş yalnızca bir görüntüye sahip olan lezyon_id'leri filtreliyoruz
df = df[df['image_id'] == 1]

df.reset_index(inplace=True)

df.head()

# burada, yinelenen görüntülere sahip olan ve yalnızca bir görüntüye sahip olan lezyon_id'leri
tanımlarız.

def identify_duplicates(x):

    unique_list = list(df['lesion_id'])

    if x in unique_list:
        return 'no_duplicates'
    else:
        return 'has_duplicates'

# lesson_id sütununun bir kopyası olan yeni bir sütun oluştur
df_data['duplicates'] = df_data['lesion_id']
# işlevi bu yeni sütuna uygula
```

```
df_data['duplicates'] = df_data['duplicates'].apply(identify_duplicates)

df_data.head()

df_data['duplicates'].value_counts()

# şimdi kopyaları olmayan resimleri filtreliyoruz
df = df_data[df_data['duplicates'] == 'no_duplicates']

df.shape

# şimdi df kullanarak bir val seti oluşturuyoruz çünkü
# bu görüntülerin hiçbirinin eğitim setinde çoğaltılmış kopyaları olmadığından eminiz
y = df['dx']

_, df_val = train_test_split(df, test_size=0.17, random_state=101, stratify=y)

df_val.shape

df_val['dx'].value_counts()

# Bu küme, val kümesindeki tüm satırlar hariç tutulduğunda df_data olacaktır.

# Bu işlev, bir görüntünün eğitimin mi yoksa değer kümesinin bir parçası mı olduğunu tanımlar.
def identify_val_rows(x):
    # val setindeki tüm lesion_id'lerin bir listesini oluştur
    val_list = list(df_val['image_id'])

    if str(x) in val_list:
        return 'val'
    else:
        return 'train'

# eğitim ve val satırlarını tanımla

# resim kimliği sütununun bir kopyası olan yeni bir sütun oluştur
df_data['train_or_val'] = df_data['image_id']
# işlevi bu yeni sütuna uygula
df_data['train_or_val'] = df_data['train_or_val'].apply(identify_val_rows)

# eğitim satırlarını filtrele
df_train = df_data[df_data['train_or_val'] == 'train']
```

```
print(len(df_train))
print(len(df_val))

df_train['dx'].value_counts()
df_val['dx'].value_counts()

df_data.set_index('image_id', inplace=True)

# İki klasörün her birindeki görüntülerin bir listesini al
folder_1 = os.listdir('./input/ham10000_images_part_1')
folder_2 = os.listdir('./input/ham10000_images_part_2')

# eğitim ve val görsellerinin listesini al
train_list = list(df_train['image_id'])
val_list = list(df_val['image_id'])


# Eğitim görüntülerini aktar

for image in train_list:

    fname = image + '.jpg'
    label = df_data.loc[image,'dx']

    if fname in folder_1:
        # görüntüye giden kaynak yolu
        src = os.path.join('./input/ham10000_images_part_1', fname)
        # görüntüye giden hedef yol
        dst = os.path.join(train_dir, label, fname)
        # görüntüyü kaynaktan hedefe kopyala
        shutil.copyfile(src, dst)

    if fname in folder_2:
        # görüntüye giden kaynak yolu
        src = os.path.join('./input/ham10000_images_part_2', fname)
        # görüntüye giden hedef yol
        dst = os.path.join(train_dir, label, fname)
        # görüntüyü kaynaktan hedefe kopyala
        shutil.copyfile(src, dst)


# Val görüntülerini aktar

for image in val_list:
```

```
    fname = image + '.jpg'
    label = df_data.loc[image,'dx']

    if fname in folder_1:
        # görüntüye giden kaynak yolu
        src = os.path.join('./input/ham10000_images_part_1', fname)
        # görüntüye giden hedef yol
        dst = os.path.join(val_dir, label, fname)
        # görüntüyü kaynaktan hedefe kopyala
        shutil.copyfile(src, dst)

    if fname in folder_2:
        # görüntüye giden kaynak yolu
        src = os.path.join('./input/ham10000_images_part_2', fname)
        # görüntüye giden hedef yol
        dst = os.path.join(val_dir, label, fname)
        # görüntüyü kaynaktan hedefe kopyala
        shutil.copyfile(src, dst)

# her klasörde kaç tane eğitim resmimiz olduğunu kontrol et

print(len(os.listdir('base_dir/train_dir/nv')))
print(len(os.listdir('base_dir/train_dir/mel')))
print(len(os.listdir('base_dir/train_dir/bkl')))
print(len(os.listdir('base_dir/train_dir/bcc')))
print(len(os.listdir('base_dir/train_dir/akiec')))
print(len(os.listdir('base_dir/train_dir/vasc')))
print(len(os.listdir('base_dir/train_dir/df')))


# her klasörde kaç tane val resmimiz olduğunu kontrol et

print(len(os.listdir('base_dir/val_dir/nv')))
print(len(os.listdir('base_dir/val_dir/mel')))
print(len(os.listdir('base_dir/val_dir/bkl')))
print(len(os.listdir('base_dir/val_dir/bcc')))
print(len(os.listdir('base_dir/val_dir/akiec')))
print(len(os.listdir('base_dir/val_dir/vasc')))
print(len(os.listdir('base_dir/val_dir/df')))

# 'nv' sınıfını artırmadığımızı unutma
class_list = ['mel','bkl','bcc','akiec','vasc','df']
```

```
for item in class_list:

    # Bu dizinleri daha sonra sildiğimiz için burada geçici dizinler oluşturuyoruz
    # bir temel dizin oluştur
    aug_dir = 'aug_dir'
    os.mkdir(aug_dir)
    # aynı sınıfın görüntülerini depolamak için temel dizinde bir dizin oluştur
    img_dir = os.path.join(aug_dir, 'img_dir')
    os.mkdir(img_dir)

    # Bir sınıf seç
    img_class = item

    # o dizindeki tüm resimleri listele
    img_list = os.listdir('base_dir/train_dir/' + img_class)

    # Görüntüleri sınıf train dizininden img_dir'e kopyalayın, ör. 'mel' sınıfı
    for fname in img_list:
            # görüntüye giden kaynak yolu
            src = os.path.join('base_dir/train_dir/' + img_class, fname)
            # görüntüye giden hedef yol
            dst = os.path.join(img_dir, fname)
            # görüntüyü kaynaktan hedefe kopyalayın
            shutil.copyfile(src, dst)


    # görüntülerin kendilerine değil, görüntüleri içeren bir dizine işaret et
    path = aug_dir
    save_path = 'base_dir/train_dir/' + img_class

    # Bir veri oluşturucu oluştur
    datagen = ImageDataGenerator(
        rotation_range=180,
        width_shift_range=0.1,
        height_shift_range=0.1,
        zoom_range=0.1,
        horizontal_flip=True,
        vertical_flip=True,
        #brightness_range=(0.9,1.1),
        fill_mode='nearest')

    batch_size = 50

    aug_datagen = datagen.flow_from_directory(path,
```

```
                                save_to_dir=save_path,
                                save_format='jpg',
                                        target_size=(224,224),
                                        batch_size=batch_size)

    # Artırılmış görüntüleri oluşturun ve bunları eğitim klasörlerine ekle

    ############

    # her sınıfta olmasını istediğimiz toplam resim sayısı
    num_aug_images_wanted = 6000

    ############

    num_files = len(os.listdir(img_dir))
    num_batches = int(np.ceil((num_aug_images_wanted-num_files)/batch_size))

    # oluşturucuyu çalıştır ve yaklaşık 6000 artırılmış görüntü oluştur
    for i in range(0,num_batches):

        imgs, labels = next(aug_datagen)

    # ham görüntü dosyalarıyla geçici dizini sil
    shutil.rmtree('aug_dir')

    # Artık her klasörde kaç tane eğitim resmimiz olduğunu kontrol et
# Bu, orijinal görüntüler ve artırılmış görüntüler

print(len(os.listdir('base_dir/train_dir/nv')))
print(len(os.listdir('base_dir/train_dir/mel')))
print(len(os.listdir('base_dir/train_dir/bkl')))
print(len(os.listdir('base_dir/train_dir/bcc')))
print(len(os.listdir('base_dir/train_dir/akiec')))
print(len(os.listdir('base_dir/train_dir/vasc')))
print(len(os.listdir('base_dir/train_dir/df')))

# Her klasörde kaç tane değer resmimiz olduğunu kontrol et

print(len(os.listdir('base_dir/val_dir/nv')))
print(len(os.listdir('base_dir/val_dir/mel')))
print(len(os.listdir('base_dir/val_dir/bkl')))
print(len(os.listdir('base_dir/val_dir/bcc')))
print(len(os.listdir('base_dir/val_dir/akiec')))
print(len(os.listdir('base_dir/val_dir/vasc')))
```

```
print(len(os.listdir('base_dir/val_dir/df')))

# Veri Hazırlığının Sonu
###
============================================================================
============ ###
###
============================================================================
============ ###
# Egitim baslangici

train_path = 'base_dir/train_dir'
valid_path = 'base_dir/val_dir'

num_train_samples = len(df_train)
num_val_samples = len(df_val)
train_batch_size = 10
val_batch_size = 10
image_size = 224

train_steps = np.ceil(num_train_samples / train_batch_size)
val_steps = np.ceil(num_val_samples / val_batch_size)

datagen = ImageDataGenerator(
    preprocessing_function= \
    tensorflow.keras.applications.mobilenet.preprocess_input)

train_batches = datagen.flow_from_directory(train_path,
                                            target_size=(image_size,image_size),
                                            batch_size=train_batch_size)

valid_batches = datagen.flow_from_directory(valid_path,
                                            target_size=(image_size,image_size),
                                            batch_size=val_batch_size)

# Not: shuffle=False, test veri setinin karıştırılmamasına neden olur
test_batches = datagen.flow_from_directory(valid_path,
                                            target_size=(image_size,image_size),
                                            batch_size=1,
                                            shuffle=False)

# mobilenet modelinin bir kopyasını oluştur

mobile = tensorflow.keras.applications.mobilenet.MobileNet()
```

```
mobile.summary()

type(mobile.layers)

# MobileNet'in kaç katmanı var?
len(mobile.layers)

# MODEL MİMARİSİNİ OLUŞTUR

# Yukarıdaki modelin son 5 katmanını hariç tut.
# Bu, global_average_pooling2d_1'e kadar olan tüm katmanları içerecektir
x = mobile.layers[-6].output

# Tahminler için yeni bir yoğun katman oluştur
# 7 sınıf sayısına karşılık gelir
x = Dropout(0.25)(x)
predictions = Dense(7, activation='softmax')(x)

# inputs=mobile.input giriş katmanını seçer
# outputs=predictions yukarıda oluşturduğumuz yoğun katmanı ifade eder.

model = Model(inputs=mobile.input, outputs=predictions)

model.summary()

for layer in model.layers[:-23]:
    layer.trainable = False

# Top2 ve Top3 Doğruluğunu Tanımla

from tensorflow.keras.metrics import categorical_accuracy, top_k_categorical_accuracy

def top_3_accuracy(y_true, y_pred):
    return top_k_categorical_accuracy(y_true, y_pred, k=3)

def top_2_accuracy(y_true, y_pred):
    return top_k_categorical_accuracy(y_true, y_pred, k=2)

model.compile(Adam(lr=0.01), loss='categorical_crossentropy',
              metrics=[categorical_accuracy, top_2_accuracy, top_3_accuracy])

# Her dizinle ilişkili etiketleri al
print(valid_batches.class_indices)
```

```
# Modeli melanomaya daha duyarlı hale getirmek için ağırlıklar ekle

class_weights={
    0: 1.0, # akiec
    1: 1.0, # bcc
    2: 1.0, # bkl
    3: 1.0, # df
    4: 3.0, # mel # Modeli Melanom'a karşı daha duyarlı hale getirmeye çalış.
    5: 1.0, # nv
    6: 1.0, # vasc
}

filepath = "model.h5"
checkpoint = ModelCheckpoint(filepath, monitor='val_top_3_accuracy', verbose=1,
                             save_best_only=True, mode='max')

reduce_lr = ReduceLROnPlateau(monitor='val_top_3_accuracy', factor=0.5, patience=2,
                              verbose=1, mode='max', min_lr=0.00001)

callbacks_list = [checkpoint, reduce_lr]

cifar = tensorflow.keras.datasets.cifar100
(x_train, y_train), (x_test, y_test) = cifar.load_data()
model1 = tensorflow.keras.applications.ResNet50(
    include_top=True,
    weights=None,
    input_shape=(32, 32, 3),
    classes=100,)

loss_fn = tensorflow.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
model1.compile(optimizer="adam", loss=loss_fn, metrics=["accuracy"])

history = model1.fit(x_train, y_train, epochs=30, batch_size=64)

# history = model.fit(train_batches, steps_per_epoch=train_steps, class_weight=class_weights,
validation_data=valid_batches, validation_steps=val_steps, epochs=30, verbose=1,
callbacks=callbacks_list)

# Burada son sonuç kullanılacaktır.

val_loss, val_cat_acc, val_top_2_acc, val_top_3_acc = \
model.evaluate_generator(test_batches,
                         steps=len(df_val))
```

```
print('val_loss:', val_loss)
print('val_cat_acc:', val_cat_acc)
print('val_top_2_acc:', val_top_2_acc)
print('val_top_3_acc:', val_top_3_acc)

# Burada en iyi sonuç kullanılacaktır.

model.load_weights('model.h5')

val_loss, val_cat_acc, val_top_2_acc, val_top_3_acc = \
model.evaluate_generator(test_batches,
                         steps=len(df_val))

print('val_loss:', val_loss)
print('val_cat_acc:', val_cat_acc)
print('val_top_2_acc:', val_top_2_acc)
print('val_top_3_acc:', val_top_3_acc)

# kayıp ve doğruluk eğrilerini göster

import matplotlib.pyplot as plt

acc = history.history['categorical_accuracy']
val_acc = history.history['val_categorical_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
train_top2_acc = history.history['top_2_accuracy']
val_top2_acc = history.history['val_top_2_accuracy']
train_top3_acc = history.history['top_3_accuracy']
val_top3_acc = history.history['val_top_3_accuracy']
epochs = range(1, len(acc) + 1)

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.figure()

plt.plot(epochs, acc, 'bo', label='Training cat acc')
plt.plot(epochs, val_acc, 'b', label='Validation cat acc')
plt.title('Training and validation cat accuracy')
plt.legend()
plt.figure()
```

```
plt.plot(epochs, train_top2_acc, 'bo', label='Training top2 acc')
plt.plot(epochs, val_top2_acc, 'b', label='Validation top2 acc')
plt.title('Training and validation top2 accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, train_top3_acc, 'bo', label='Training top3 acc')
plt.plot(epochs, val_top3_acc, 'b', label='Validation top3 acc')
plt.title('Training and validation top3 accuracy')
plt.legend()

plt.show()

# Test görüntülerinin etiketlerini al.
# Kedilerin ve köpeklerin ayrı klasörlerde olduğunu unutmayın, bu nedenle aşağıdaki kod
# görüntünün bulunduğu klasöre bağlı olarak etiketleri alabilir.
test_labels = test_batches.classes


# bir tahminde bulun
predictions = model.predict_generator(test_batches, steps=len(df_val), verbose=1)

predictions.shape

# Kaynak: Scikit Learn
# http://scikit-learn.org/stable/auto_examples/
# model_selection/plot_confusion_matrix.html#sphx-glr-auto-examples-model-
# selection-plot-confusion-matrix-py


def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')
```

```
    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()


test_labels.shape

# argmax, bir satırdaki maksimum değerin dizinini döndürür
cm = confusion_matrix(test_labels, predictions.argmax(axis=1))

test_batches.class_indices

# Sınıf indekslerinin etiketlerini tanımlayın.
# Bunların yukarıda gösterilen sıraya uyması gerekir.
cm_plot_labels = ['akiec', 'bcc', 'bkl', 'df', 'mel','nv', 'vasc']

plot_confusion_matrix(cm, cm_plot_labels, title='Confusion Matrix')

# Model Oluşturma Sonu
###
============================================================================
============ ###
```