T.C.
ISPARTA UYGULAMALI BİLİMLER ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ
MEKATRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ZEYTİN YAPRAĞINDAKİ HASTALIKLARIN DERİN ÖĞRENME TEKNİKLERİ KULLANILARAK SINIFLANDIRILMASI

Neşe UYSAL

Danışman
Dr. Öğr. Üyesi Sinan UĞUZ

ISPARTA - 2020

TEZ ONAYI

ZEYTİN YAPRAĞINDAKİ HASTALIKLARIN DERİN ÖĞRENME TEKNİKLERİ KULLANILARAK SINIFLANDIRILMASI

Neşe UYSAL tarafından hazırlanan bu tez çalışması aşağıdaki jüri tarafından Isparta Uygulamalı Bilimler Üniversitesi, Lisansüstü Eğitim Enstitüsü Mekatronik Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

		İmza
Danışman	**Dr. Öğr. Üyesi Sinan UĞUZ** Isparta Uygulamalı Bilimler Üniversitesi	.....
Üye	**Prof. Dr. Hilmi Cenk BAYRAKÇI** Isparta Uygulamalı Bilimler Üniversitesi	.....
Üye	**Dr. Öğr. Üyesi Turgay AYDOĞAN** Süleyman Demirel Üniversitesi	.....

Yukarıdaki Jüri kararı Lisansüstü Eğitim Enstitüsü Yönetim Kurulu'nun ...../...../.... tarih ve...../..... sayılı kararıyla onaylanmıştır.

Prof. Dr. Yusuf UÇAR
Enstitü Müdürü

ETİK BEYANI

Isparta Uygulamalı Bilimler Üniversitesi Lisansüstü Eğitim Enstitüsü tez yazım kurallarına uygun olarak ve bilimsel ahlak ve geleneklere aykırı düşecek bir yol ve yardıma başvurmaksızın 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 ve ortaya çıkan sonuçlarda herhangi bir değişiklik yapmadığımı, bu tezde sunduğum çalışmanın özgün olduğunu, tezimle ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçlara katlanacağımı bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

26/12/2019

Neşe UYSAL

.....

İÇİNDEKİLER

Sayfa

ÖZET

Yüksek Lisans Tezi

ZEYTİN YAPRAĞINDAKİ HASTALIKLARIN DERİN ÖĞRENME TEKNİKLERİ KULLANILARAK SINIFLANDIRILMASI

Neşe UYSAL

Isparta Uygulamalı Bilimler Üniversitesi
Lisansüstü Eğitim Enstitüsü
Mekatronik Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Sinan UĞUZ

Konvolüsyonel sinir ağı modellerinin bitki yaprak görüntüleri üzerinde hastalık ve zararlıların tanı ve teşhisi konusunda oldukça başarılı olduğu bilinmektedir. Bu tez çalışmasında zeytin bitkisine ait 3 farklı yaprak türü konvolüsyonel sinir ağı modeli ile sınıflandırılmıştır. Veri seti 3400 adet zeytin yaprak görüntüsünden oluşmaktadır. Halkalı leke hastalığı, zeytin yaprak pasakarı zararlısı ve sağlıklı yaprakların oluşturduğu 3 farklı sınıf bulunmaktadır. Farklı optimizasyon algoritmaları ve iterasyon sayılarının sınıflandırma başarısına etkisi, Konvolüsyonel Sinir Ağları (CNN) yardımı ile gözlemlenmiştir. Tez çalışmasında kullanılan CNN modelleri Google Colab bulut servisinde yer alan Tesla GPU'lar ile eğitilmiştir. Anaconda (Spyder) IDE platformunda python programlama dili ile yazılmıştır. Eğitilen modelin web uygulaması Flask kütüphanesi yardımıyla oluşturulmuştur.

Tez çalışmasında önerilen CNN modeli 100 iterasyon üzerinden Adamoptimizasyon algoritması ile %84 doğruluk oranı elde etmiştir. Transfer öğrenme yöntemi kullanılarak, VGG16 ve VGG19 gibi gelişmiş CNN modelleri oluşturulan veri seti ile eğitilmiştir. Karşılaştırmalı deney sonuçlarına göre %88 doğruluk oranı ile VGG16 modeli en iyi sonucu vermiştir. Tez çalışmasında önerilen CNN modeline, Keras kütüphanesi ImageDataGenarator sınıfı ile veri arttırım işlemi uygulanmıştır. Veri arttırım yöntemi uygulanan bu model Adam optimizasyon algoritması kullanılarak 100 iterasyonda eğitilmiştir. Veri arttırım işleminin sonucunda önerilen CNN modeli %94 doğruluk oranı ile en iyi sınıflandırma başarısını elde etmiştir.

Anahtar Kelimeler: Derin öğrenme, Konvolüsyonel sinir ağı, Zeytin bitkisi hastalık ve zararlıları, Transfer öğrenme, Veri arttırım yöntemi, Optimizasyon algoritmaları

2020, 71 sayfa

ABSTRACT

M.Sc. Thesis

CLASSIFICATION OF OLIVE LEAF DISEASES USING DEEP LEARNING TECHNIQUES

Neşe UYSAL

Isparta University of Applied Sciences
The Institute of Graduate Education
Department of Mechatronic Engineering

Supervisor: Asst. Prof. Dr. Sinan UĞUZ

Convolutional neural network models are known to be very successful in diagnosing diseases and pests on plant leaf images. In this thesis, three different leaf types belonging to olive plant were classified with convolutional neural network model. The data set consists of 3400 olive leaf images. There are 3 different classes Olive Peacock Spot disease, Aculus Olearius and healthy leaves. The effect of different optimization algorithms and iteration numbers on the classification success was observed with the help of Convolutional Neural Networks (CNN).CNN models used in the thesis were trained with Tesla GPUs in Google Colab cloud service. Anaconda (Spyder) is written in python programming language on the IDE platform. The web model of the trained model was created with the help of the Flask framework.

The proposed CNN model achieved an accuracy rate of 84% with the Adam optimization algorithm over 100 iterations. Using the transfer learning method, advanced CNN models such as VGG16 and VGG19 were trained with the generated data set. According to comparative test results, VGG16 model gave the best result with 88% accuracy. The CNN model, which was proposed in the thesis study, was applied to the dataaugmentation process with ImageDataGenarator class of Keras framework. This model, which is applied dataaugmentation method, was trained in 100 iterations by using Adam optimization algorithm. As a result of the dataaugmentation process, the proposed CNN model achieved the best classification success with 94% accuracy rate.

Key Words: Deep learning, Convolutional neural network, Olive plant diseases and pests, Transfer learning, Data augmentation method, Optimization algorithms

2020, 71 pages

TEŞEKKÜR

Tezimin yürütülmesinde desteğini ve emeğini hiçbir zaman esirgemeyen tez danışmanım sayın Dr. Öğr. Üyesi Sinan UĞUZ'a, çalışma sürecinde bilgi ve tecrübesiyle bana desteklerinden dolayı sayın Dr. Öğr. Üyesi Ufuk ÖZKAYA'ya teşekkürlerimi sunarım.

Tezimin her aşamasında beni yalnız bırakmayan aileme sonsuz sevgi ve saygılarımı sunarım.

Neşe UYSAL
ISPARTA, 2020

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

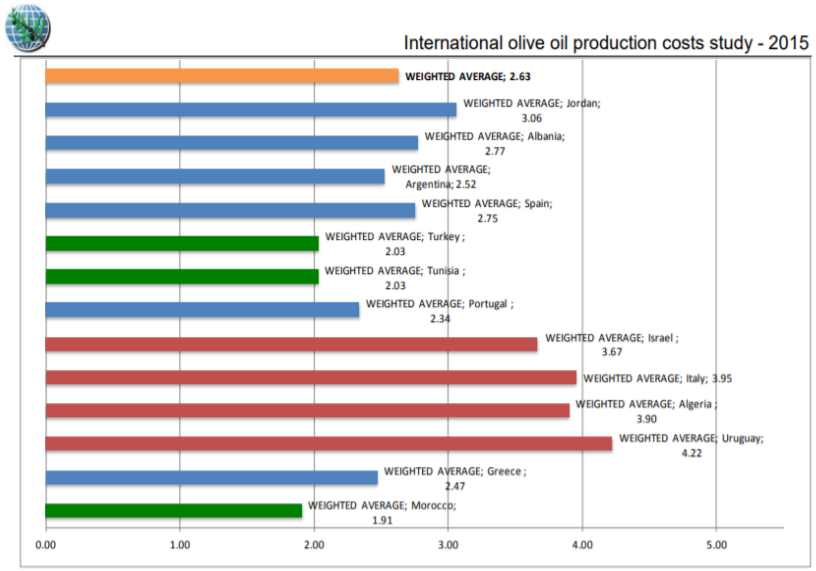
Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neurl Network
EC2	Elastic Compute 2 Cloud
FC	Fully Connected
GPU	Graphics Processing Unit
IDE	Integrated Development Environment
KNN	K Nearest Neighborhood
ReLu	Rectified Linear Units
RGB	Red Green Blue
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine

1. GİRİŞ

Zeytin ülkemizde Ege, Marmara ve Akdeniz gibi belli bölgelerde yetiştirilmektedir. Türkiye'nin önemli tarımsal ihraç ürünlerinden olan zeytin ve zeytinyağı, ülkemiz potansiyeli dikkate alındığında, tarım sektörümüz için dünyada rekabet gücü olan ürünlerdendir (Gümrük ve Ticaret Bakanlığı, 2018). 2017 istatistikleri incelendiğindeTürkiye sofralık zeytin üretiminde dünyada sayılı ülkeler arasında yer almaktadır (Gümrük ve Ticaret Bakanlığı, 2018). Ayrıca Şekil 1.1'de 2015 yılında ülkelerin 1 kg zeytin yağı üretebilmek için harcamaları gerek maliyet tablosu Euro üzerinden verilmiştir. Grafiği incelediğimizde 1 kg zeytinyağı üretim maliyetinin dünya ortalaması kilogram başına 2.63 euro olarak görülmektedir. Ancak Türkiye dünya ülkeleri ortalamasından daha düşük bir maliyet ile zeytinyağı üretimi yapabilmektedir (Anonim, 2015).



Şekil 1.1. Dünyada ülkeler bazında 1 kg zeytinyağı üretim maliyeti (Anonim, 2015)

Ülkemiz ekonomisi açısından da önemli olan bu bitki türüne ait pek çok hastalık ve zararlı türleri mevcuttur (Anonim, 2017a). Bitki hastalıkları, küresel boyutta tarımsal üretimin verim ve kalitesinde ortak bir tehdit oluştururken aynı zamanda üretim maliyetlerinin de önemli bir kısmının sorumluluğunu üstlenmektedir (Maa vd., 2018). Bitki hastalık ve zararlılarını optik gözlem yoluyla tanıma süreci genellikle zaman alıcı, zahmetli ve özneldir. Hastalık hasarı değerlendirme ve tedavisinin çoğu, ziraat mühendisleri rehberliğinde, sahadaki çiftçiler tarafından yapılmaktadır (Maa vd., 2018).

Bu tez çalışması ile zeytin bitkisine ait ülkemizde yaygın görülen hastalık ve zararlı türlerinin derin öğrenme yöntemi kullanılarak sınıflandırılması gerçekleştirilmiştir.

1.1. Zeytin Hastalık ve Zararlıları

Bu tez çalışmasında oluşturulan verisetinde yer alan zeytin bitkisine ait bir çeşit hastalık ve bir çeşit zararlı türü ile sağlıklı yapraklar Çizelge 1.1'de örnek resimler ile gösterilmiştir.

Çizelge 1.1. Zeytin yapraklarına ait sınıflandırma yapılan örnek resimler

Sağlıklı	Halkalı Leke Hastalığı	Zeytin Yaprak Pasakarı
		
		
		

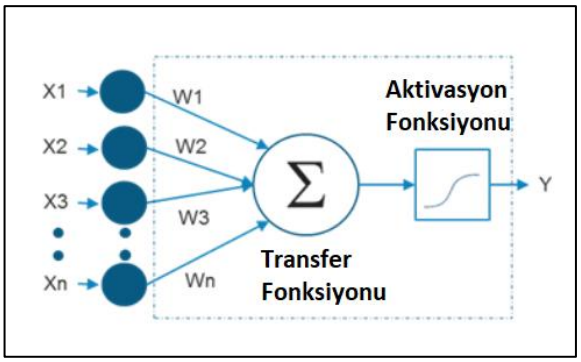
Zeytinde halkalı leke bir hastalık türü olarak tanımlanmaktadır. Halkalı leke hastalığı zeytin ağacında genellikle yaprakların dökülmesine yol açarken ağacın meyvesinin de erken dökülmesine ve verimin düşmesine neden olur (Anonim, 2017a). Zeytin akarı ve zeytin yaprak pasakarı olarak ifade edilen zararlı tür, zeytin yapraklarında ve meyve üzerinde şekil bozuklukları oluşturmaktadır (Anonim, 2017a). Belirttiğimiz bu hastalık ve zararlı türler; Ege, Akdeniz ve Marmara bölgelerinde görülmektedir. Tez çalışmamızda bitkiye ait yaprak görüntüleri Ege bölgesindeki Denizli ilinde Gemlik türü olarak adlandırılan zeytin ağaçlarından toplanmıştır. Bölgenin geçim

kaynaklarından olan zeytin yetiştiriciliği bu hastalık ve zararlılardan olumsuz etkilenmektedir. Tunç ve Onoğur'un çalışmasında, halkalı leke hastalık türünün zeytin bitkisi üzerinde %40'a varan oranlarda meyvelerin dökülmesine sebep olduğu ifade edilmiştir (Tunç ve Onoğur, 2013). Ayrıca zeytin bitkisine olan zarar oranı en az %15-20 iken en fazla %50-90 seviyelerinde olabildiği belirtilmiştir (Tunç ve Onoğur, 2013). Tez çalışmasında ki araştırmalardaPamukkale İlçe Tarım Müdürlüğüziraat mühendisleri de halkalı leke hastalığının bitkide verimi düşürdüğünü ifade etmişlerdir.

Zeytin akarı ve pasakarı olarak bilinen zararlı türü, zeytin yapraklarından beslenerek yapraklarda şekil ve renk bozukluğuna yol açmakta ve yaprak üzerinde deformasyonlara, kıvrılmalara neden olmaktadır (Anonim, 2016b). Meyve oluşum dönemlerinde de meyve üzerinde beslenmesiyle şekil bozukluğuna neden olurken zeytinyağı asitlik oranını arttırmaktadır (Anonim, 2016b). Bu durum zeytinyağı kalitesini de olumsuz etkilemektedir.

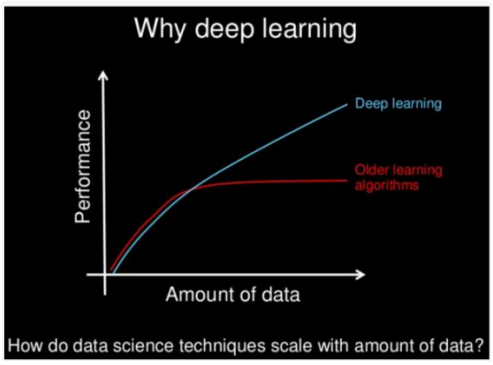
1.2. Derin Öğrenme

Derin öğrenme biyolojik nöronların, matematiksel olarak yeniden tasarlanmasına ilişkin olarak geliştirilmiştir. Şekil 1.2'de derin öğrenme uygulamalarının temeli olan yapay sinir ağlarına ait bir yapay nöron modeli görülmektedir. x1..xn girdi değerlerini w1..wn ise her girdiye ait ağırlık değerlerini ifade etmektedir. Her girdi değeri kendisine ait olan ağırlık değeri ile çarpılır. Elde edilen sonuçlar transfer fonksiyonunda toplanarak aktivasyon fonksiyonundan geçirilir. Aktivasyon fonksiyonu transfer fonksiyonu ile elde edilen sonucu belirli bir aralıkta sınırlı tutar. Örneğin sigmoid aktivasyon fonksiyonu kullanıldığında Y çıkış değeri 0-1 arasında bir değer almaktadır (Bakshi, 2019).



Şekil 1.2. Yapay sinir ağı şeması (Bakshi, 2019)

Derin öğrenme, hiyerarşik katman mimarisi ile veri üzerinden öznitelik çıkarımlarını hesaplamalı olarak yapan makine öğrenme teknikleri sınıfına dahil bir alandır (Deng ve Yu, 2014). Burada yer alan derin kavramı, büyük bir yapay sinir ağını ve çok katmanlı yapay sinir ağlarında gizli katmanların sayısının birden fazla olması durumunu ifade etmektedir. Andrew Ng'ye göre derin öğrenmenin özünde büyük sinir ağlarını eğitebilmek için yeterli hızda bilgisayar ve veriye sahip olmamız yatmaktadır (Ng, 2015). Şekil 1.3'te derin öğrenme yöntemi ile klasik makine öğrenmesi algoritmalarının performansları veri miktarı üzerinden karşılaştırılmıştır. Burada veri miktarının artmasıyla derin öğrenme yönteminin performansınında artış gösterdiği görülmektedir. Büyük veri ile yapılabilecek uygulamalarda da derin öğrenme yöntemlerinin neden tercih edilebileceğini Andrew Ng açıklamış oluyor.



Şekil 1.3. Veri miktarının veri bilimi tekniklerine etkisi (Ng, 2015)

Derin öğrenme girdilerin analog olduğu alanlarda oldukça başarılıdır, bunlar piksel verileri, metin verileri veya ses verilerini kapsar (Brownlee, 2019a). Bu bağlamda derin öğrenme için görüntü tanıma, doğal dil işleme ve konuşma tanıma gibi görevlerde yaygın olarak kullanılmaktadır. Çizelge 1.2'de en çok kullanılan derin öğrenme mimarilerine örnek verilerek açıklaması yapılmıştır.

Çizelge 1.2. Derin öğrenme mimarileri

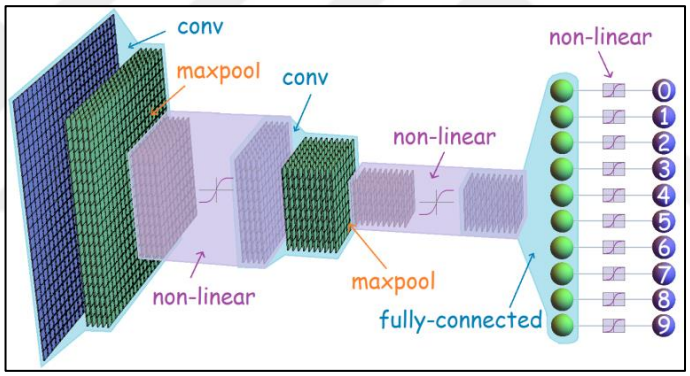
Derin Öğrenme Mimarileri	Açıklama
Konvolüsyonel Sinir Ağları (CNN)	Temel derin öğrenme aracı, konvolüsyonel sinir ağları (Lecun vd., 1998) karmaşık işlemlerin modellenmesi ve görüntü örneklerinin tanınması gibi büyük miktarda veri içeren uygulamalarda kullanılmaktadır (Ferentinos, 2018). Doğal dil işleme ve ses işleme gibi farklı alanlarda da uygulanmaktadır (Şeker vd., 2017). Özellikle görüntü işleme alanında CNN'ler oldukça başarılı sonuçlar vermiştir (Lu vd., 2017; Ferentinos, 2018).
Tekraralayan Sinir Ağları (RNN)	RNN, model içerisinde bir birime ait çıktı bir sonraki birimin girdisi olarak kullanılan ve birimler arasında döngüsel bağlantı kurulan yapay sinir ağı sınıfıdır (Şeker vd., 2017). RNN konuşma ve metin işleme gibi alanlarda yaygın olarak kullanılmaktadır (Sherstinsky, 2018).

Tez çalışmasında görüntüler üzerinden zeytin bitkisine ait hastalık ve zararlıların tanı ve teşhisi için derin öğrenme mimarilerinden konvolüsyonel sinir ağı (Convolutional Neural Network) kullanılmıştır.

1.2.1. Konvolüsyonel sinir ağları

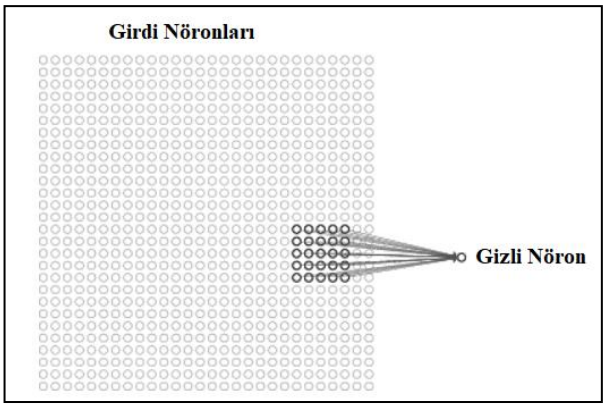
Konvolüsyonel sinir ağı, bir veriseti (görüntü) üzerinden öznitelik çıkarımı yapabilmek için tasarlanmış katmanlı bir sinir ağı modelidir (Anonim, 2019a). Her katmanda görüntüye ait öznitelikler otomatik olarak çıkartılmakta ve bir sonraki katmana iletilmektedir. Le Cun vd. (1999), el yazısı ile yazılmış olan rakamların

tanınmasında nesneye ait özniteliklerin CNN sayesinde başarılı bir şekilde ortaya çıkarıldığını, çalışmalarında kanıtlamışlardır. Örnek bir konvolüsyonel sinir ağı model mimarisini Şekil 1.4'te görülmektedir. Şekil 1.3'te yer alan konvolüsyonel sinir ağı mimarisi 0-9 arasında rakamları sınıflandırabilmesi için oluşturulmuş bir modeli temsil etmektedir. Öncelikle ilk olarak konvolüsyon katmanı ve devamında maksimum havuzlama katmanı kullanılmıştır. Havuzlama katmanından sonra ise doğrusal olmayan durumları arttırabilmek için aktivasyon fonksiyonundan yararlanılmıştır ve tekrar konvolüsyon beraberinde havuzlama katmanı ve aktivasyon fonksiyonu ile devam edilmiştir. Son olarak tam bağlı katman elde edilmiştir. Bu katmandan sonra sınıflandırma işlemini uygulayabilmek için yapay sinir ağı tarafından elde edilen skor değerleri kullanarak olasılık üzerinden sonuca ulaşılmıştır (Çarkacı, 2017). İki sınıf için uygulanacaksa sigmoid fonksiyonu, daha fazla sınıf söz konusu ise softmax fonksiyonu tercih edilir (Çarkacı, 2017).



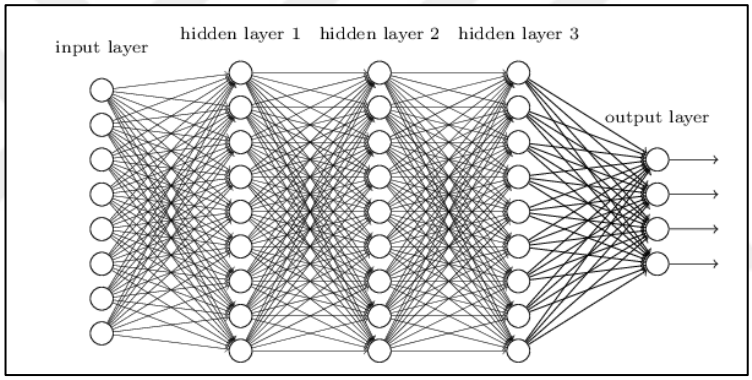
Şekil 1.4. Örnek konvolüsyonel sinir ağı (CNN) yapısı (Anonim, 2016a)

Günümüzde çoğunlukla konvolüsyon ağları görüntü tanıma amaçlı kullanılmaktadır (Nielsen, 2018). Konvolüsyonel sinir ağları 3 ana tema üzerinde tasarlanır: yerel alıcı alanları, paylaşılan ağırlıklar ve havuzlama (Nielsen, 2018). Yerel alıcı alanları kısaca ifade edecek olursak giriş görüntüsü üzerinde ayrılmış olan ve görüntünün özniteliklerini çıkartmada bize yardımcı olan (3x3, 5x5, 7x7 gibi) bölgelerdir. Şekil 1.5'te örnek yerel alıcı alan olarak 5x5'lik bir bölgenin bilgisi gizli nöronda nasıl saklandığı belirtilmiştir. Burada giriş görüntüsüne ait öznitelik bilgisi de bu alanlarda saklanmaktadır.



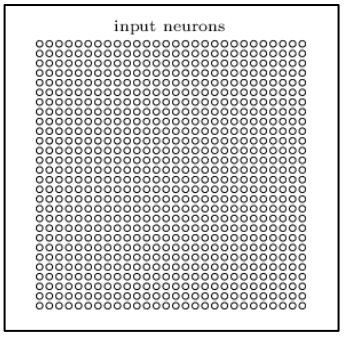
Şekil 1.5. Yerel alıcı alan örneği (Nielsen, 2018)

Şekil 1.6'da verilen yapay sinir ağı modeli incelediğinde giriş verilerinin dikey nöronlardan oluştuğunu görebiliriz. Bu modelde her bir nöron bir sonraki katmanlarda yer alan her bir nörona bağlanır.



Şekil 1.6. Yapay sinir ağı modeli (Nielsen, 2018)

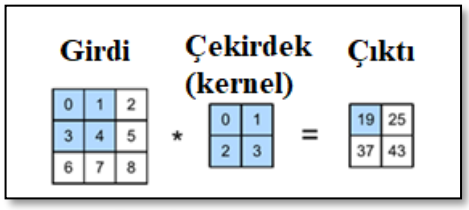
Ancak veri olarak 2 boyutlu bir resim ele alınacak olursa Şekil 1.6'da görüldüğü gibi resimde yer alan her bir pikselin bir nörona karşılık geldiğini ifade edebiliriz.



Şekil 1.7. 28x28'lik bir resmin piksel görünümü (Nielsen, 2018)

1.2.1.1. Konvolüsyon katmanı

Konvolüsyon işlemi ile giriş resmine ait matris üzerinde korelasyon çekirdeği (kernel) uygulanarak çıkış dizisi üretilir (Zhang vd., 2019). Şekil 1.8'de konvolüsyon işlemine ait bir örnek görülmektedir. Örnekte 3x3 boyutunda bir giriş resmine 2x2'lik bir korelasyon çekirdeği bir diğer ifade ile filtre uygulanmıştır.



Şekil 1.8. Konvolüsyon işlemi örneği (Zhang vd., 2019)

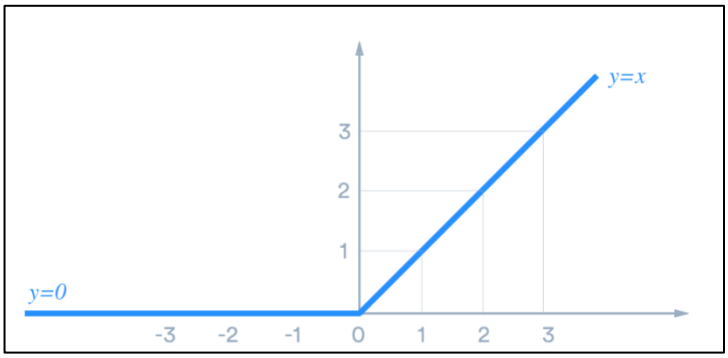
Filtre üzerinde bulunan her bir piksel giriş resminde yer alan her bir piksel ile çarpılıp toplanarak (0×0 + 1×1 + 3×2 + 4×3 = 19) sonuçta yeni bir matris elde edilmektedir (Zhang vd., 2019). Yapılan bu işleme konvolüsyon veya evrişim adı verilmektedir. Konvolüsyon işleminin bir amacı giriş resmine ait alt, orta ve üst düzey öznitelikleri otomatik olarak giriş görüntüsünden çıkarabilmektir. Genellikle ilk konvolüsyon katmanı resme ait kenar, renk gibi alt düzey öznitelikleri çıkarılmasından sorumludur (Sumit, 2018). Ek katmanlar uygulayarak giriş görüntüsüne ait çıkarılan öznitelikler ile ağın daha iyi anlamasını sağlayabiliriz.

1.2.1.2. ReLu (Rectified Linear Unit)

Görüntü üzerinde yapılan matris çarpım işleminin sonucunun doğrusal olmayan bir hale dönüştürülmesi gerekmektedir, bu işlem aktivasyon fonksiyonları ile gerçekleştirilir (Çarkacı, 2018). ReLu fonksiyonu doğrusal olmayan dönüşümleri sağlamak için kullanılmaktadır.

ReLu, CNN ile en sık kullanılan aktivasyon fonksiyonudur. Genellikle derin öğrenme modellerinde konvolüsyon katmanının ardından kullanılır. Herhangi bir negatif giriş değeri alırsa, fonksiyon 0 değerini döndürür, pozitif değerler için ise fonksiyon o değerin kendisini döndürür (Liu, 2017). Böylece matematiksel olarak

f(x)=max(0, x) şeklinde ifade edilebilir. Fonksiyona ait grafik Şekil 1.9'da görülmektedir.

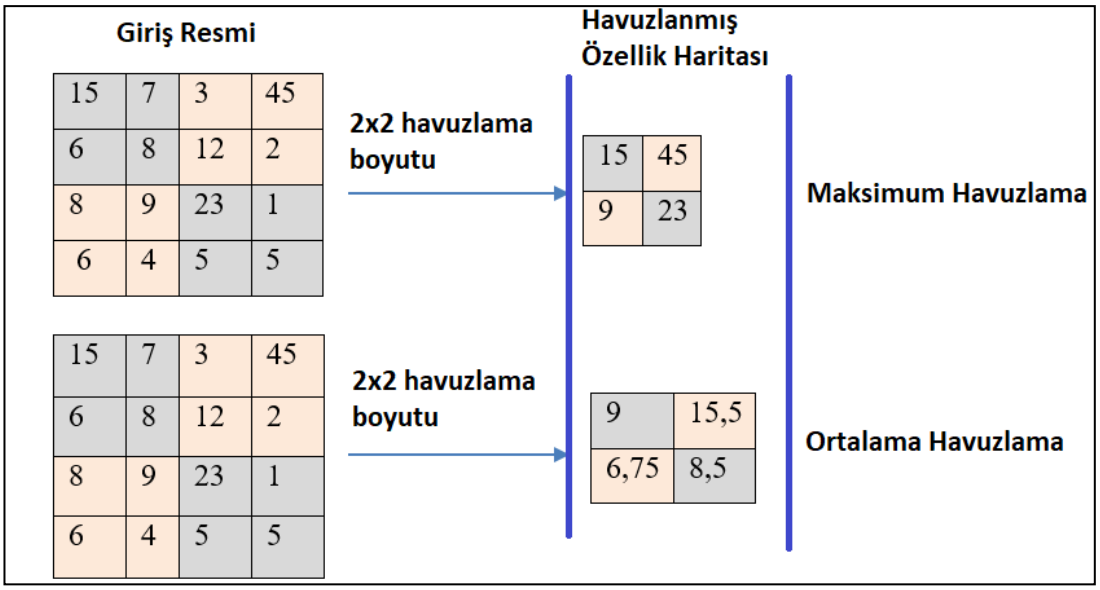


Şekil 1.9. ReLu fonksiyon grafiği (Liu, 2017)

1.2.1.3. Havuzlama (Pooling) katmanı

Aktivasyon katmanından geçtikten sonra elde edilen öznitelik haritası üzerinde görüntü boyutunu ve ağdaki işlem gücünü azaltmak için havuzlama katmanı uygulanır (Rao, 2019). Burada boyut azalır ancak derinlik artar. Yani giriş görüntüsünde aranılan nesneye ait asıl öznitelik bu katmandan sonra oluşan matris içerisinde tutulur.

İki tür havuzlama yöntemi vardır. Bunlar, maksimum havuzlama ve ortalama havuzlama yöntemleridir (Rao, 2019). Maksimum havuzlama yöntemi giriş resmi üzerine uygulanan çekirdeğe ait bölgedeki piksel değerleri içerisinden en yüksek olanını seçer. Ortalama havuzlama yöntemi ise çekirdeğe ait bölgedeki piksel değerlerinin ortalamasını hesaplar. Örneğin, Şekil 1.10'da görüldüğü gibi bu işlem tüm giriş matrisi üzerinde2adım kaydırılarak devam edecektir.

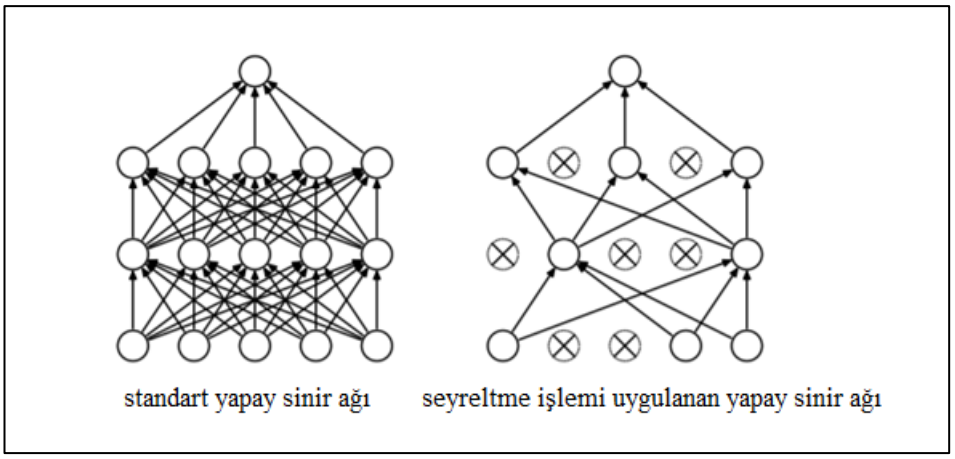


Şekil 1.10. Maksimum havuzlama ve ortalama havuzlama yöntemleri

Tez çalışmasında önerilen CNN mimarisi içerisinde maksimum havuzlama yöntemi tercih edilmiştir.

1.2.1.4. Seyreltme katmanı

Küçük verisetleri ile eğitilen derin sinir ağlarıaşırı öğrenme (overfitting) ile karşılaşılabilir. Aşırı öğrenme modelin eğitim verileri üzerinde çok iyi bir doğruluk (başarı) oranını yakalarken test verileri üzerinde düşük performans göstermesi anlamına gelir (Brownlee, 2018). Seyreltme (dropout) bu problemi çözmek için kullanılan düzenlileştirme (regularization) tekniklerinden birisidir (Srivastava vd., 2014). Burada yapılan işlem yapay sinir ağının eğitimi esnasında gizli katmanda bulunan bazı yapay nöronları silmektir. Şekil 1.11'de sol tarafta 2 adet gizli katmana sahip yapay sinir ağı, sağ tarafta ise bu ağa seyreltme (dropout) işlemi uygulandıktan sonraki yapay sinir ağı görülmektedir.



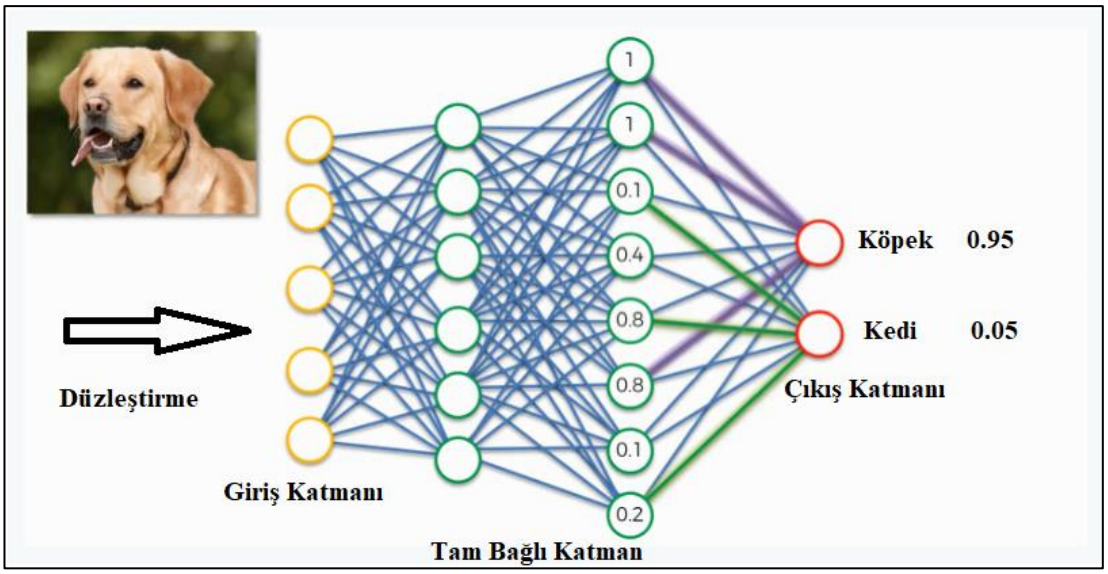
Şekil 1.11. Seyreltme işlemi (Srivastava vd., 2014)

Yapılan literatür taramalarında derin öğrenme modelleri üzerinde seyreltmekatmanın uygulanmasınınmodel performansını arttırdığı görülmüştür (Wallelign vd., 2018; Geetharamani ve Arun Pandian, 2019; Francis ve Deisy, 2019).

İncelenen modeller üzerinde, uygulanan seyreltme oranı 0.2 de daha iyi sonuç verdiği tespit edilmiştir (Francis ve Deisy, 2019). Bu nedenle tez çalışmasında tasarlanan model üzerinde ve transfer öğrenimi gerçekleştirilen modeller üzerinde seyreltme oranı 0.2 olarak sabit bırakılmıştır.

1.2.1.5. Tam bağlı katman

Konvolüsyonel sinir ağı, genellikle konvolüsyon ve havuzlama katmanlarından sonra eklenir. Yapay sinir ağında olduğu gibi birbirine bağlı olan düğümlerden (nöronlardan) oluşmaktadır (Anonim, 2017b). Konvolüsyon katmanlarından sonra giriş resimlerinden öznitelikler belirlenirve sonuç olarak çıktıyı elde etmek için tam bağlı katman kullanılır. Tam bağlı katmanda yer alan nöronlar belirli bir özniteliği taşımaktadır. Örneğin; giriş resminde uygulanan görüntüdeki bir öznitelik değer olarak tutuluyor ve bu değerin hangi sınıfa ait olduğu kontrol edilip karar veriliyor (Anonim, 2018a). Şekil 1.12'de verilen örnekte, görüntünün konvolüsyon katmanlarından geçtikten sonra düzleştirme işlemi uygulanmış olup bir sonraki işlem adımında tam bağlı katman mimarisi oluşturulmuştur. Bu işlem adımında bir yapay sinir ağı modeli görülmektedir. Yapay sinir ağı önceki katmanlardan elde edilen bilgilere bağlı olarak her iki sınıfında olasılık değerlerini hesaplayıp buna göre tahminde bulunmaktadır (Anonim, 2018a).



Şekil 1.12. Tam bağlı katman yapısı için bir örnek (Anonim, 2018a)

1.2.1.6. Yığın normalizasyonu

Derin sinir ağları eğitimi esnasında katmanlar arası geçişlerdeki girdi değerleri değiştiği için bu durum eğitimi karmaşıklaştırıyor. Bu nedenle eğitimin düşük öğrenme oranları ve dikkatli parametre değerleri ile başlatılması gerekmektedir. Sorunun çözümü için girdi katmanlarında yapılan normalleştirme işlemini gizli katman girişlerinde de uygulanması önerilmektedir (Ioffe ve Szegedy, 2015). Uygulanan bu işleme yığın normalizasyonu (Batch Normalization) denilmektedir. Yığın normalizasyonunun ağa uygulanması aynı zamanda ağın eğitimini de hızlandıracaktır. Ayrıca ağın aşırı öğrenme problemine de çözüm sunarak seyreltme işlemine olan ihtiyacı azaltacaktır (Anonim, 2017c; Ng, 2017).

1.2.1.7. Optimizasyon algoritmaları

Sinir ağı modelini eğitebilmek için modelin tahmin ettiği değer ile gerçek etiket değeri arasındaki farkı hesaplayan bir kayıp fonksiyonuna ihtiyaç vardır. Sinir ağının doğru bir tahmin yapabilmesi için kayıp fonksiyonunun otomatik olarak en düşük değere inmesini sağlayan ağırlık değerlerinin tespit edilmesi ve güncellenmesi gerekmektedir (Oppermann, 2019). Bunun için optimizasyon algoritmaları kullanılmaktadır (Ser ve Bati, 2019). Derin öğrenme uygulamalarında kullanılan birçok optimizasyon algoritması vardır. Bunlar arasında da gerek hız gerek başarı

yönünden farklılıklar bulunmaktadır (Çarkacı, 2018). Çizelge 1.3'te bu tez çalışmasında kullanılan optimizasyon algortimaları ve açıklamaları verilmiştir.

Çizelge 1.3. Optimizasyon algoritmaları ve açıklamaları

Optimizasyon Algoritmaları	Açıklama
Adam	Adam (Adaptive Moment Estimation Algorithm) optimizasyon algoritması rastgele eğim iniş algoritmasından (Stochastic Gradient Descent) farklı olarak öğrenme oranını eğitim sırasında değiştirebilmektedir (Brownlee, 2017a). Adam, momentumlu SGD algoritması ve Rmsprop algoritmasının birleştirilmiş halidir.
SGD	Rastgele eğim iniş algoritmasında (SGD), her bir iterasyon için ayarlanan verilerin tamamı yerine rastgele birkaç örnek seçilir. Her bir iterasyonda rastgele seçilen verilerin hesaplaması yapılacaktır (Srinivasan, 2019). Eğim iniş algoritmasındaki (Gradient Descent) hesaplama maliyetini azaltmak için kullanılan bir algoritmadır (Srinivasan, 2019).
Rmsprop	Rmspropoptimizasyon algoritması, Adagrad algoritmasının bir özelliği olan öğrenme oranlarının aşırı küçülmesi sorununa bir çözüm olarak önerilmiştir (Ser ve Bati, 2019).
AdaGrad	AdaGrad optimizasyon algoritmasında kullanılan parametrelerin kendi öğrenme hızları bulunur ve buna bağlı olarak öğrenme oranı güncellenir (Çarkacı, 2018). Daha çok doğal dil işleme ve görüntü işleme uygulamaları için uygundur (Çarkacı, 2018). Örneğin doğal dil işleme uygulamasında seyrek karşılaşılan bir kelime için öğrenme oranının dinamik olarak azaltılması gerekecektir. Bu algoritma özellikle seyrek veriler için iyi bir performans göstermektedir (Gylberth, 2018).

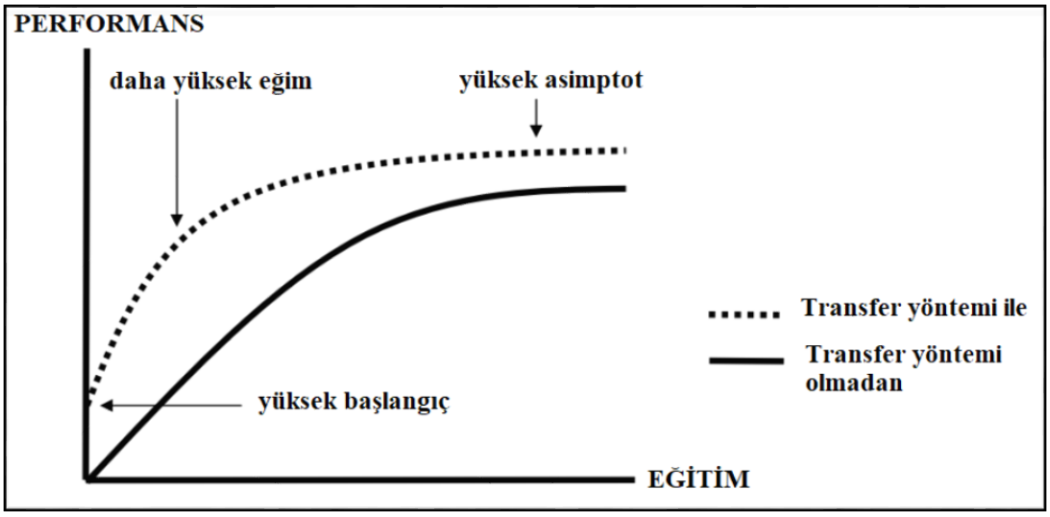
1.2.1.8. Kayıp fonksiyonu

Bir algoritmanın, verileri ne kadar iyi modellediğini değerlendirmenin bir yöntemi kayıp (loss) fonksiyonunu kullanmaktır (Parmar, 2018). Derin öğrenme uygulamalarında kullanılan optimizasyon algoritmalarının yardımıyla, kayıp fonksiyonu tahmindeki hatayı azaltmayı öğrenir (Parmar, 2018). Hata oranı ne kadar yüksek olursa gerçek değer ile tahmin edilen değer arasındaki fark da o kadar yüksek olacaktır. Tasarlanan modeller için ise bu durum model performansının düşüklüğü anlamına gelir.

Kayıp fonksiyonu ve maliyet fonksiyonu (cost function) birbirinin yerine kullanılsa da, aralarında fark vardır. Kayıp fonksiyonu, tek iterasyondaki eğitim örneğini kapsarken; maliyet fonksiyonu, tüm verisetindeki ortalama kayıp oranıdır (Mahendru, 2019). Yapay sinir ağında tasarlanan uygulamaya göre kayıp fonksiyonunun belirlenmesi gerekmektedir. İkili sınıflandırma görevi için "Binary Cross Entropy", çoklu sınıflandırma görevlerinde "Categorical Cross Entropy" kullanılır (Brownlee, 2019b). Tez çalışmamızda, hedeflenen yaprak görüntülerinden hastalık ve zararlı türü ile beraber sağlıklı yaprakların sınıflandırılması gibi çoklu sınıflandırma görevi olduğu için "Categorical Cross Entropy" kayıp fonksiyonu kullanılmıştır.

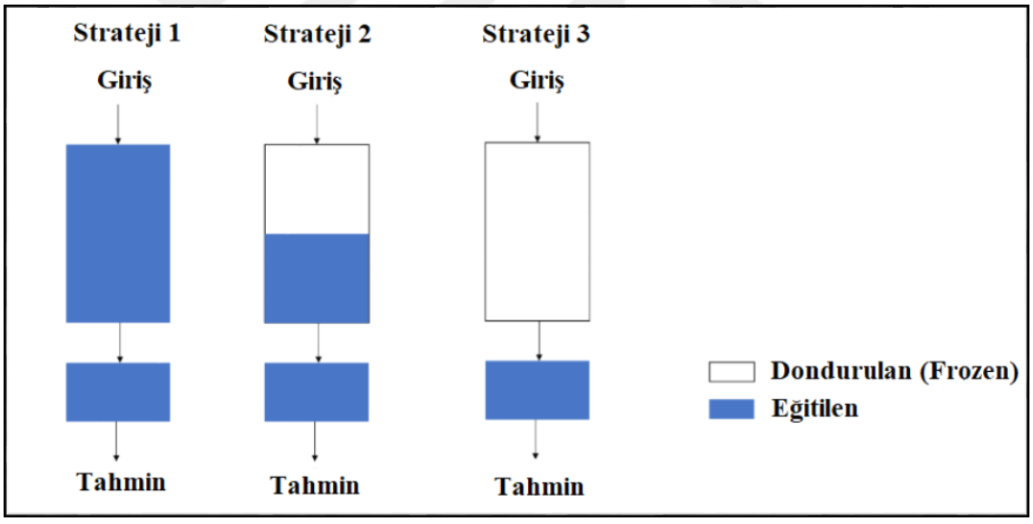
1.3. Transfer Öğrenme

Transfer öğrenme yöntemi bilgiyi önceden eğitilmiş olan model üzerinden yeni oluşturulan modele aktarma işlemi olarak tanımlanabilir (Geetharamani ve Arun Pandian, 2019). Transfer öğrenmenin amacı, çok çaba gerektiren modelleri yeniden oluşturmak yerine önceden edinilen bilgiler kullanılıp ve model geliştirme sürecini önemli ölçüde azaltarak model performansını iyileştirmektir (Kaya vd., 2019). Araştırmacılar tarafından makine öğrenmesi ile çözülmek istenilen karmaşık problemlerin sayısı arttıkça transfer öğrenme yöntemi de daha talep edilebilir hale gelmektedir (Torrey ve Shavlik, 2009). Transfer öğrenme, ilk görevden edinilen bilgiler sayesinde ikinci görevde hızlı ilerleme veya gelişmiş bir performans sağlamaktadır (Brownlee, 2017b). Şekil 1.13'te transfer öğrenme yönteminin model performansına olan 3 farklı etkisi gösterilmiştir (Torrey ve Shavlik, 2009).



Şekil 1.13. Transfer öğrenme yönteminin modelin öğrenmesi üzerindeki etkisi (Torrey ve Shavlik, 2009)

Transfer öğrenme yönteminde Şekil 1.14'de görüldüğü gibi üç farklı ince ayar stratejisi kullanılmaktadır.



Şekil 1.14. Transfer öğrenme yönteminde kullanılan ince ayar stratejileri (Marcelino, 2018)

Çizelge 1.4'te ise transfer öğrenme yöntemine ait stratejilerin birbirinden farkı üzerinde durulmuştur.

Çizelge 1.4. Transfer öğrenme stratejileri ve açıklamaları

Strateji 1	Modeli sıfırdan eğitmek	Önceden eğitilmiş olan model mimarisini kullanarak oluşturulan veri seti bu mimariye uygulanmış olur (Marcelino, 2018). Bu nedenle çok fazla sayıda veriye ihtiyaç olacaktır.
Strateji 2	Bazı katmanları eğitip diğer katmanları dondurmak	Bilindiği üzere ilk katmanlar görüntüye ait düşük seviye öznitelikleri, ileri katmanlar ise görüntüye ait yüksek seviye öznitelikleri ortaya koymaktadır. Buna göre ağda bulunan bazı katmanlar dondurma işlemine tabii tutulabilir. Burada dondurulmuş olan katmanlar eğitilmeyecektir (Marcelino, 2018). Bu yaklaşım oldukça hassas bir ayarlama gerektirir.
Strateji 3	Konvolüsyon tabanını dondurmak	Burada önceden eğitilmiş modeli bir öznitelik çıkarıcı olarak kullanılıyor. Konvolüsyon katmanları dondurularak orijinal biçimde tutuluyor. Sınıflandırıcı katmanları kullanılmış oluyor. Küçük verisetine sahip uygulamalarda tercih edilebilir (Marcelino, 2018).

Bu tez çalışmasında, transfer öğrenme yöntemi için ImageNet veriseti ile önceden eğitilmiş 2 farklı model, tercih edilmiştir. Bunlar VGG16 ve VGG19 modelleridir. Bu modeller öznitelik çıkarıcısı olarak kullanılmıştır. Seçilen bu modellerin hepsine de düzleştirme işleminden sonra 1000 yapay nörondan oluşan tam bağlı katman, 0.2 oranında seyreltme katmanı ve 3 yapay nörondan oluşan son tam bağlı katman eklenmiştir. Verisetinde yer alan örneklerin 3 farklı sınıfta (halkalı leke, zeytin yaprak pasakarı ve sağlıklı) tanımlanması hedeflendiği için son katmanda sınıf sayısını belirtilmektedir.

1.3.1. VGGNet

VGGNet modeli, Oxford Üniversitesinden, Karen Simonyan ve Andrew Zisserman tarafından önerilen konvolüsyonel sinir ağı modelidir. 2014 yılında yapılan ImageNet yarışmasında, testlerde %8.8 hata oranı ile en iyi performansı elde eder (Simonyan ve Zisserman, 2015). VGGNet konvolüsyonel sinir ağı modeli, 1.3 milyon resim üzerinde 1000 farklı sınıfı tanımlamak için eğitilir.

VGGNet modelinde girişte 224x224 piksel boyutunda RGB görüntüler uygulanmıştır. Genel olarak, Karen Simonyan ve Andrew Zisserman'ın makalesinde VGGNet modelinin farklı konvolüsyon katman sayıları deneysel olarak uygulanmış ve sonuçları gözlemlenmiştir. Çizelge 1.5'de uygulanan katman ve filtre sayıları görülmektedir. Çizelge 1.5'de yer alan model yapılandırmaları sırasıyla; VGG-11, VGG-11 (LRN), VGG-13, VGG-16 (Conv1), VGG-16 ve VGG-19' dur.

Çizelge 1.5. VGGNet modeline ait konvolüsyon mimarisi (Simonyan ve Zisserman, 2015)

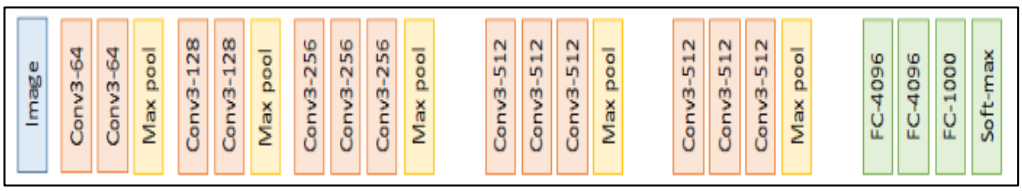
ConvNet Yapılandırması					
A	**A-LRN**	**B**	**C**	**D**	**E**
11 ağırlık katmanı	**11 ağırlık katmanı**	**13 ağırlık katmanı**	**16 ağırlık katmanı**	**16 ağırlık katmanı**	**19 ağırlık katmanı**
Giriş (224 x 224 RGB görüntü)					
Conv3-64	Conv3-64 **LRN**	Conv3-64 **Conv3-64**	Conv3-64 Conv3-64	Conv3-64 Conv3-64	Conv3-64 Conv3-64
Maksimum Havuzlama					
Conv3-128	Conv3-128	Conv3-128 **Conv3-128**	Conv3-128 Conv3-128	Conv3-128 Conv3-128	Conv3-128 Conv3-128
Maksimum Havuzlama					
Conv3-256 Conv3-256	Conv3-256 Conv3-256	Conv3-256 Conv3-256	Conv3-256 Conv3-256 **Conv3-256**	Conv3-256 Conv3-256 **Conv3-256**	Conv3-256 Conv3-256 Conv3-256 **Conv3-256**
Maksimum Havuzlama					
Conv3-512 Conv3-512	Conv3-512 Conv3-512	Conv3-512 Conv3-512	Conv3-512 Conv3-512 **Conv3-512**	Conv3-512 Conv3-512 **Conv3-512**	Conv3-512 Conv3-512 Conv3-512 **Conv3-512**
Maksimum Havuzlama					
Conv3-512 Conv3-512	Conv3-512 Conv3-512	Conv3-512 Conv3-512	Conv3-512 Conv3-512 **Conv3-512**	Conv3-512 Conv3-512 **Conv3-512**	Conv3-512 Conv3-512 Conv3-512 **Conv3-512**
Maksimum Havuzlama					
Tam Bağlı Katman - 4096					
Tam Bağlı Katman - 4096					
Tam Bağlı Katman - 1000					
Softmax					

Bu tez çalışmasında kullanılan ve VGGNet mimarileri olan VGG16 ve VGG19 araştırmacılar tarafından yaygın olarak kullanılmaktadır.

1.3.1.1. VGG16

Bu tez çalışmasında, transfer öğrenme yöntemi uygulanırken VGG16 ağında bulunan konvolüsyon katmanları öznitelik çıkarıcı olarak kullanılmış olup eğitim sırasında ağırlık değerleri değişmemesi için Çizelge 1.4'te ki transfer öğrenme stratejilerinden

biri uygulanmıştır. Buna göre konvolüsyon tabanı dondurulmuştur. VGG16 modelinin sınıflandırıcı katmanı eğitilebilir halde bırakılmıştır. Bu durum Şekil 1.16'da belirtilmiştir. Burada amaç, tez çalışmasında kullanılan verisetinin küçük olması nedeniyle kullanılan modelin en iyi sınıflandırma başarısına ulaşması istenilmektedir. VGG16 modeline ait genel mimari Şekil 1.15'te görülmektedir.



Şekil 1.15. VGG16 genel ağ mimarisi (Tsang, 2018)

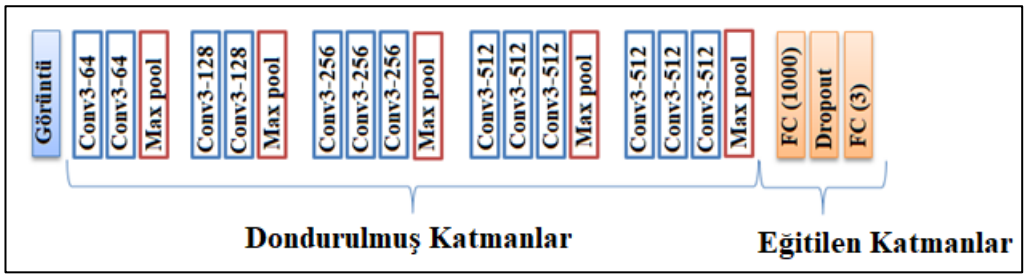
Çizelge 1.6'da VGG16 modeline ait yapının detayları verilmiştir. Burada oluşturulan katmanlar: InputLayer olarak ifade edilen kısım girişte kullanılan görüntünün boyutudur. 224 x 224 piksel boyutunda RGB görüntülerin modele verildiği görülmektedir. İki konvolüsyon katmanı ile beraber bir adet havuzlama katmanı blok1 olarak nitelendirilmiştir. Bu sayede mimaride yer alan ve ard arda uygulanan katmanların daha net ayırt edilebilmesi sağlanmış olacaktır. Çizelge 1.6'da görüldüğü üzere her blokta farklı sayıda konvolüsyon katmanları uygulanmaktadır. Ayrıca Çizelge 1.6'da konvolüsyon katmanlarına uygulanan VGGNet mimarisinde 3x3 boyutlarında kullanılan filtre sayıları da görülmektedir. Örneğin, 224, 224, 64 ifadesi ile birinci konvolüsyon katmanında 64 adet filtre kullanılmıştır. Bu filtre sayıları VGGNet mimarisinde giderek arttırılmış ve en son katmanda 512 adet filtre uygulanmıştır. Çizelge 1.6'da görüldüğü üzere, maksimum havuzlama katmanlarında iki adım kaydırma işlemi uygulanmıştır. Bu sayede görüntü boyutu azalacağından işlem yükü de azaltılmış olacaktır. Örneğin, girişte uygulanan 224x224 boyutunda ki görüntü, havuzlama katmanından sonra 112x112 boyutunda yarıya düşürülmüş olacaktır. Bu işlem her havuzlama katmanından sonra gerçekleşeceği içinVGG16 mimarisinde en son havuzlama katmanından sonra görüntü 7x7 boyutuna indirgenmiş olacaktır. Son havuzlama katmanında elde edilen görüntü üzerinden yapay sinir ağı modelini oluşturabilmek için Çizelge 1.6'da da görüleceği üzere 25088 adet piksel olacaktır. Bu pikselller tek boyutlu vektörel bir hale dönüştürülecektir. Bu adıma da düzleştirme denilmektedir. Düzleştirme işleminden sonra iki adet tam bağlı katman eklenmiştir. Bu katmanlarda yer alan yapay nöron sayıları yine Çizelge 1.6'da görülmektedir. Son katmanda 1000 adet nöron

bulunmasının nedeni VGGNet mimarisi 2014 yılında katılmış olduğu ImageNet yarışmasında 1000 farklı nesneyi sınıflandırmaya çalışmıştır.

Çizelge 1.6. VGG16 modeli genel katman mimarisi (Simonyan ve Zisserman, 2015)

Katman	**Çıktı Şekli**
input_1 (InputLayer)	(None, 224, 224, 3)
block1_conv1 (Conv2D)	(None, 224, 224, 64)
block1_conv2 (Conv2D)	(None, 224, 224, 64)
block1_pool (MaxPooling2D)	(None, 112, 112, 64)
block2_conv1 (Conv2D)	(None, 112, 112, 128)
block2_conv2 (Conv2D)	(None, 112, 112, 128)
block2_pool (MaxPooling2D)	(None, 56, 56, 128)
block3_conv1 (Conv2D)	(None, 56, 56, 256)
block3_conv2 (Conv2D)	(None, 56, 56, 256)
block3_conv3 (Conv2D)	(None, 56, 56, 256)
block3_pool (MaxPooling2D)	(None, 28, 28, 256)
block4_conv1 (Conv2D)	(None, 28, 28, 512)
block4_conv2 (Conv2D)	(None, 28, 28, 512)
block4_conv3 (Conv2D)	(None, 28, 28, 512)
block4_pool (MaxPooling2D)	(None, 14, 14, 512)
block5_conv1 (Conv2D)	(None, 14, 14, 512)
block5_conv2 (Conv2D)	(None, 14, 14, 512)
block5_conv3 (Conv2D)	(None, 14, 14, 512)
block5_pool (MaxPooling2D)	(None, 7, 7, 512)
flatten (Flatten)	(None, 25088)
fc1 (Dense)	(None, 4096)
fc2 (Dense)	(None, 4096)
predictions (Dense)	(None, 1000)

Bu tez çalışmasında düzleştirme katmanından sonra eğitim işlemi gerçekleştirilecek olan katmanlar eklenmiştir. Uygulanan transfer öğrenme yöntemi ile mimarideki son katmanlarda, sınıflandırmaya ağırlık verilmiştir. Oluşturulan mimari Şekil 1.16'da belirtilmiştir.



Şekil 1.16. Tez çalışmasında kullanılan VGG16 model mimarisi

Çizelge 1.7'de VGG16 modeli üzerinde eğitilen katmanların detayları görülmektedir. VGG16 modelinin son havuzlama katmanında 7x7x512 boyutunda görüntü elde edilmiştir. Bu işlem adımından sonra düzleştirme uygulanmıştır. Tez çalışmasında düzleştirme uygulamasının ardından bir adet tam bağlı katman eklenmiştir. Çizelge 1.7'de dense_1 olarak adlandırıldığı ve 1000 adet yapay nörondan oluştuğu görülmektedir. Dropout_1 olarak tanımlanan bir sonraki katmanda ise yapay sinir ağımızın aşırı öğrenme durumunu önlemek amacıyla 0.2 oranında seyreltme katmanı uygulanmıştır. Yapılan literatür taramalarında seyreltme katmanı için uygulanan bu oranının oldukça iyi performans sonucu verdiği görülmüştür (Francis ve Deisy, 2019). Seyreltme katmanından sonra ise tekrar tam bağlı katman uygulanmıştır ve verisetinde yer alan sınıf sayısı (üç) kadar bu tam bağlı katmanda yapay nöron eklenmiştir.

Çizelge 1.7. Tez çalışmasında kullanılan VGG16 modelinde eğitilen katmanlar

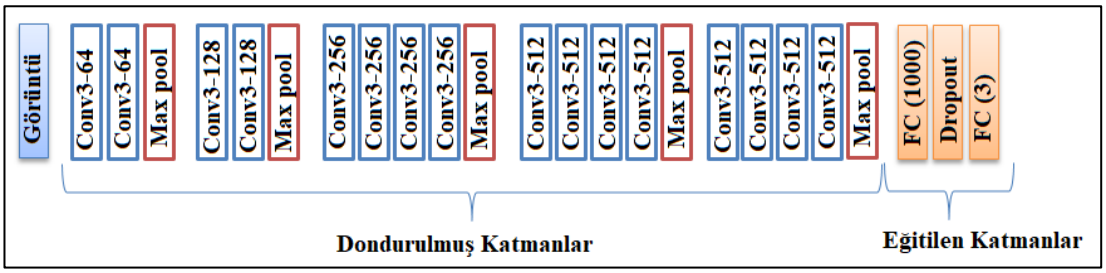
Katman	**Çıktı Şekli**
vgg16 (Model)	(None, 7, 7, 512)
flatten_1 (Flatten)	(None, 25088)
dense_1 (Dense)	(None, 1000)
dropout_1 (Dropout)	(None, 1000)
dense_2 (Dense)	(None, 3)

1.3.1.2. VGG19

VGG19 modeli, 2014 yılı ImageNet yarışmasında %9.0'lık hata oranı elde etmiştir (Simonyan ve Zisserman, 2015). Bu durum çalışmada ki katman sayısının arttırılmasını da durdurmuştur (Tsang, 2018). Şekil 1.17'de VGG19 modeline ait genel sinir ağı yapısı görülmektedir. Önceden eğitilmiş olan konvolüsyonel sinir ağı modelleri VGG16 ve VGG19 arasındaki fark ise VGG19'a daha fazla konvolüsyon katmanı eklenmiştir.

Şekil 1.17. VGG19 genel ağ mimarisi (Tsang, 2018)

Bu tez çalışmasında düzleştirme katmanından sonra eğitim işlemi gerçekleştirilecek olan katmanlar eklenmiştir. VGG16 mimarisinde olduğu gibi mimarideki son katmanlarda, sınıflandırmaya ağırlık verilmiştir. Oluşturulan mimari Şekil 1.17'de belirtilmiştir.



Şekil 1.18. Tez çalışmamızda kullanılan VGG19 model mimarisi

Çizelge 1.8'de VGG19 modeli üzerinde eğitilen katmanların detayları görülmektedir. Transfer öğrenme yöntemi uygulanacak olan VGG16 ve VGG19 model mimarilerinde karşılaştırmalı deneyler yapılacağı için seyreltme oranı, eğitilecek olan katmanlar gibi parametreler aynı şekilde bırakılmıştır.

Çizelge 1.8. Tez çalışmamızda kullanılan VGG19 modelinde eğitilen katmanlar

Katman	**Çıktı Şekli**
vgg19 (Model)	(None, 7, 7, 512)
flatten_1 (Flatten)	(None, 25088)
dense_1 (Dense)	(None, 1000)
dropout_1 (Dropout)	(None, 1000)
dense_2 (Dense)	(None, 3)

2. KAYNAK ÖZETLERİ

Ioffe ve Szegedy (2015), çalışmalarında derin sinir ağları mimarilerinde katman girişlerinin normalleştirilmesini önermişlerdir. Derin sinir ağları eğitimi esnasında katmanlar arası geçişlerdeki girdi değerleri değiştiği için bu durum eğitimi karmaşıklaştırıyor. Bu nedenle eğitimin düşük öğrenme oranları ve dikkatli parametre değerleri ile başlatılmasını zorunlu kılıyor. Bu sorunun çözümü için önerilen normalleştirme işlemi aynı zamanda seyreltme (dropout) işlemine olan ihtiyacı da azaltıyor. Ayrıca derin sinir ağında düzenleyici olarak işlev sunmuş oluyor. Önerilen bu yöntem, gücünü normalleştirme aktivasyonlarından ve normalleşmeyi de ağ mimarisine dâhil etmekten almıştır. Ayrıca, ağı eğitmek için kullanılan herhangi bir optimizasyon algoritması ile uygun bir şekilde normalizasyonun yapılması sağlanmıştır. Engellenmiş görüntü sınıflandırma modeline uygulanan yığın (batch) normalizasyon işlemi sayesinde, derin sinir ağının 14 kat daha az eğitim adımıyla aynı başarıya ulaştığı görülmüştür. Ağa uygulanan yığın (batch) normalizasyon işlemi ile sınıflandırma başarısında artış gözlemlenmişitr.

Mohanty vd. (2016), çalışmalarında gelişen teknolojiye bağlı olarak bitki hastalıklarının mobil telefonlar ile tespit edilebilmesi için derin öğrenme yöntemi kullanmışlardır. Derin öğrenme modeli olarak AlexNet ve GoogleNet gibi önceden eğitilmiş olan iki model ele alınmıştır. Herkesin erişimine açık olan PlantVillage veri seti üzerinden 54306 görüntü ile 14 bitki türüne ait 26 hastalık çeşidi ve sağlıklı olduğu bilgisi tespiti için derin bir konvolüsyonel sinir ağı eğitilmiştir. Bu görüntüler kontrollü koşullar altında toplanılmış ve 256 x 256 piksel boyutlarında ölçeklendirilmiştir. Model internet ortamından elde edilmiş olan bitki görüntüleri ile de test edilmiş olup %31.4'lük bir başarı oranı görülmüştür. Burada başarı oranını iyileştirebilmek için daha fazla sayıda ve çeşitte eğitim verisine ihtiyaç duyulduğu vurgulanmıştır. Makalede yapılan deneysel çalışmalarda Caffe açık kaynak derin öğrenme kütüphanesi kullanılmıştır. Makalede uygulanan deneysel araştırmalardan bir tanesi, girişte uygulanan görüntü türünün 3'e ayrılmasıdır. Grayscale (gri tonda görüntüler), RGB (renkli görüntüler) ve Segmented (bölütlenmiş yaprak görüntüleri). Genel olarak özetlemek gerekirse aşağıda yer alan parametre değişimlerine göre 60 farklı deneysel çalışma gerçekleştirilmiştir:

1. Derin öğrenme modelinin seçilmesi
    - AlexNet
    - GoogleNet
2. Eğitim mekanizmasının seçilmesi
    - Transfer öğrenimi
    - Modeli sıfırdan eğitmek
3. Veri setinde kullanılacak görüntü türünün seçilmesi
    - Renkli (RGB)
    - Gri tonda (Gray scale)
    - Yaprak bölütlemeli (Leaf segmented)
4. Eğitim ve test verilerinin dağıtım oranının seçilmesi
    - Eğitim : %80, Test : %20
    - Eğitim : %60, Test : %40
    - Eğitim : %50, Test : %50
    - Eğitim : %40, Test : %60
    - Eğitim : %20, Test : %80

Yapılan deneysel çalışmalarda 30 iterasyon ile modeller eğitilmiştir (Mohanty vd., 2016).

Lee vd. (2017), çalışmalarında yaprak görüntülerinden ayırt edici özellikleri öğrenme amaçlı derin öğrenme yöntemini araştırmış ve deneysel çalışmalar yapmışlardır. Çalışmalarında, CNN mimarisinin sınıflandırıcı uygulamaları incelenmiştir. Çalışma sonucunda elde edilen bulgular CNN kullanılarak yaprak özelliklerinin öğrenilmesi el ile belirlenen özelliklere kıyasla daha başarılı olduğu gözlemlenmiştir. Yapraklarda şekil özelliğinin yanı sıra damar yapısının da sınıflandırmada çok önemli olduğu bulgusuna ulaşılmıştır.

Lu vd. (2017), çalışmalarında pirinç bitkisine ait hastalıkları teşhis ve tanımlamada kullanmak için yeni bir yöntem önermişlerdir. Önerdikleri CNN modeli 10 çeşit pirinç hastalığı türünü tanımlamak için eğitilmiştir. Modelde elde edilen başarı oranı %95.48 olmuştur. Hastalığın teşhisi için örüntü tanımlama görevinde konvolüsyonel sinir ağlarının (CNNs) en iyi sınıflandırıcı olduğu kabul edilmiştir. Eğitim esnasında 500 adet pirinç yaprağı ve pirinç sap görüntüleri kullanılmıştır. Modelde konvolüsyon katmanından sonra stokastik havuzlama (stochastic pooling) denilen havuzlama katmanı uygulanmıştır. Bu yöntem özellik haritası üzerindeki değerlerin olasılık değerleri hesaplanarak büyük olasılıklı değerlerin seçilmesi ile oluşuyor. LeNet-5 ve AlexNet gibi CNN mimarilerinden esinlenerek kendi CNN modelini

ortaya koymuşlardır. Bu yapıda konvolüsyonel katman, stokastik havuzlama katmanı ve softmax katmanı yer almaktadır. Deneysel çalışmalar sonucunda giriş görüntü boyutu 224 x 224 piksel olarak seçilmiştir. Önerilen yapıda 3 adet konvolüsyon katmanı yer almıştır. İlk katman görüntünün kenar çizgisi ve köşeleri gibi düşük seviye özelliklerin çıkartılmasını sağlıyor. Diğer 2 konvolüsyon katmanı ise yüksek seviye özelliklerin çıkartılmasına yardımcı oluyor. Modelde kullanılan stokastikhavuzlama katmanı varyansı azaltmak için kullanılmıştır. Maksimum havuzlama ve ortalama havuzlama yöntemlerinin avantajlarını birleştirerek aşırı öğrenme durumunu önlediği görülmüştür. Veri setinde pirinç bitkisine ait 10 çeşit hastalık türünün bulunduğu 500 adet resim yer almaktadır. Dijital kamera ile 5760 x 3840 piksel boyutlarında bu resimler elde edilmiştir. Tüm görüntüler Matlab R2012 yazılımı ile resim ön işleme işleminden geçirilmiştir. Burada resimler 512 x 512 piksel boyutuna getirilip modele girişte RGB olarak uygulanmıştır. Yapılan deneysel çalışmayı özetleyecek olursak, ortalama havuzlama, maksimum havuzlama ve stochastic havuzlama tekniklerinin oluşturulan model üzerindeki etkisi gözlemlenmiştir. Buna göre stokastik (stochastic) havuzlama diğerlerine oranla daha iyi performans sunduğu görülmüştür. Ayrıca konvolüsyon işlemlerinde uygulanan filtre boyutları değiştirilerek de model başarı sonuçları gözlemlenmiştir. Ancak burada 5x5, 9x9, 16x16, 32x32 gibi oluşturulan farklı filtre boyutlarının model üzerindeki tanımlama başarısına etkisi çok az olduğu görülmüştür.

Shijie vd. (2017), çalışmalarında domates bitkisine ait yaprak üzerinde 10 çeşit hastalık ve haşerenin tanı ve tespitini gerçekleştirmiştir. VGG16 modeli üzerinden transfer öğrenimi yöntemi kullanmışlardır. Keras, tensorflow gibi derin öğrenme kütüphaneleri kullanılarak oluşturulan modelin ortalama sınıflandırma başarısı %89 olarak gözlemlenmiştir. Sırasıyla 2 algoritma kullanılmıştır. Birincisi (VGG16+SVM), görüntü özelliği çıkarımı için VGG16, SVM (suppurt vector machine) sınıflandırma algoritmasıyla beraber kullanılmıştır. İkinci olarak, orijinal VGG16 modelini temel alan uçtan uca bir sınıflandırma modeli oluşturmak için ince ayar yöntemi uygulanmıştır. Tüm deneylerin uygulandığı donanımsal özellikleri sıralayacak olursak, 2.4 Ghz CPU 16 GB hafıza (RAM) ve NVIDIA Geforce GT 1080 ekran kartı kullanıldığı görülmüştür. Linux Ubuntu 14.04 işletim sistemi üzerinde tensorflow 1.0 ve keras 1.0.3 derin öğenme kütüphaneleri kullanılmıştır. Veri seti üç bölüme ayrılmıştır. Bunlar, eğitim, onaylama (validation) ve test. Test

verisi 880, onaylama verisi 1600 ve eğitim verisi 4000 adet domates yaprak görüntüsünden oluşmaktadır. Modelde kullanılan bazı hiper parametre değerleri ise şu şekildedir: Learning rate: 0.01, Batch size: 40, Epoch: 180, Optimizer: SGD, Momentum: 0.9. VGG16+SVM'in birlikte kullanıldığı modelin sınıflandırma başarısı %88 iken ince ayar (fine tuning) işlemi uygulanan VGG16 modelinin sınıflandırma başarısı ise %89 olarak ölçülmüştür. Burada transfer öğrenimi modeli, VGG16+SVM modelinden biraz daha iyi performans gösterdiği görülmüştür.

Ferentinos (2018), çalışmasında bitkilerin yaprak görüntüleri üzerinden hastalıkların tanı ve teşhisi için konvolüsyonel sinir ağı modelleri kullanılmıştır. Erişime açık veri setinde bulunan 87848 adet yaprak görüntüsü ile eğitim gerçekleştirilmiştir. 25 farklı bitkiye ait toplamda hastalıklı ve sağlıklı yapraklarla beraber 58 farklı sınıfı tanımlayabilmiştir. Burada birkaç model eğitimi gerçekleştirilmiştir. Bu derin öğrenme modelleri arasında en iyi başarı oranı 99.53 olarak ölçülmüştür. 5 temel CNN mimarisi, Torch kütüphanesi ile yaprak görüntülerinden hastalıkların teşhisi için test edilmiştir. Bunlar, AlexNet, AlexNetOWTBn, GoogleNet, Overfeat ve VGG. Eğitim için donanım olarak NVIDIA GTX1080 GPU sistemi kullanılmıştır. %80 eğitim, %20 test için veri seti ayrılmıştır. Burada en başarılı model %99.53 ile VGG CNN modeli olmuştur. Bu yüksek performans seviyesine dayanarak, konvolüsyonlu sinir ağlarının basit yaprak görüntülerini analiz edip bitki hastalıklarını otomatik tanı ve teşhisinde oldukça başarılı olduğu görülmüştür. Önerilen derin öğrenme yaklaşımı başarısını göstermiştir. Önerilen sistemi geliştirmek ve daha sağlam hale getirebilmek mevcut verilerin nicelik ve niteliğine bağlı olduğu sonucuna ulaşılmıştır.

Maa vd. (2018), makalede 4 farklı salatalık hastalık türü tanımlanmaya çalışılmıştır. Derin konvolüsyonel sinir ağı (DCNN) modeli bu hastalıkların tanımlanması için önerilmiştir. Modelin ezberlemesinin (aşırı öğrenme) önlenebilmesi için bölümlenmiş belirti (semptom) görüntülerinden oluşan veri seti üzerinde veri artırımı yapılmıştır. Bu işlem ile beraber veri setinde toplam 14208 belirti (semptom) görüntü elde edilmiştir. Oluşturulan DCNN ile %93.4'lük doğrulukla hastalık tanıma sonuçlarına ulaşılmış olduğu görülmüştür. DCNN ile elde edilen başarı sonuçlarını karşılaştırmak için geleneksel makine öğrenme yöntemleri(Rastgele orman ve destek vektör makineleri) ve önceden eğitilmiş AlexNet mimarisi kullanılmıştır. Salatalık

bitkisine ait hastalıklı yaprak görüntüleri veri seti PlantVillage ve Forestry Images üzerinden ve kalan diğer görüntülerde dijital kamera ile elde edilmiştir. Hesaplama maliyetini azaltmak ve görüntü işleme verimliliğini arttırmak için tüm görüntüler 800 x 600 piksel boyutuna getirilmiştir. DCNN modeli CUDA 9.0 ile NVIDIA Quadro P4000 (8 GB Memory) üzerinden eğitilm gerçekleştirilmiştir. Ağ ağırlıklarını optimize etmek için momentum ile stochastic gradient descent kullanılmıştır. Mini batch size değeri 128 olarak belirlenmiştir. Eğitimin en yüksek iterasyon sayısı 800 olmuştur. Eğitim verileri değiştirilerek deneysel çalışmalar yapılmıştır. Salatalık hastalıklarının belirti (semptom) olarak tanınmasını sağlamak için derin bir evrimşimsel sinir ağı önerilmiştir. Nicel deneyler DCNN'nin mükemmel tanıma sonuçları elde ettiğini göstermiştir. DCNN'nin dengesiz veri seti ve dengeli veri seti üzerindeki doğruluğu sırasıyla %93.4 ve %92.2 idi. Karşılaştırmalı testler AlexNet'in önceden eğitilmiş bir model olarak zengin özellik çıkarımı sayesinde DCNN'den daha iyi performans gösterdiğini sonucuna ulaşılmıştır.

Rangarajan vd. (2018), çalışmada domates bitkisine ait yaprak görüntülerinin 6 hastalık ve 1 sağlıklı olmak üzere AlexNet ve VGG16 gibi modeller ile transfer öğrenimi metodu kullanılarak sınıflandırma işlemi gerçekleştirmiştir. Modeli eğitmek için kullanılan görüntüler internet üzerinden Plant Village veri setinden alınmıştır. Toplam 13262 adet bölümlendirilmiş görüntü kullanılmıştır. Burada kontrollü koşullar altında elde edilen yaprak görüntüleri arka plandan ayrıştırılıyor ve bölümlendirme dediğimiz işlem gerçekleştirilmiş oluyor. Hastalıkların sınıflandırılmasına ilişkin modeller üzerinde seçilen bazı hiper parametre değerleri değiştirilerek ağın başarısına olan etkisi gözlemlenmiştir. Modeller donanım olarak 4GB 1050 GPU ve 8 GB RAM ile Matlab2017b yazılımı kullanılarak eğitilmiştir. Veri setinde toplam 13262 adet bölümlendirilmiş domates bitkisine ait görüntüler yer almaktadır. Burada yer alan görüntülerin orijinal boyutları 256 x 256 piksel boyutundadır. Ancak VGG16 önceden eğitilmiş olan model için bu boyut 224 x 224 piksele, AlexNet önceden eğitilmiş modeli için ise 227 x 227 piksel boyutlarına yeniden ölçeklendirilmiştir. Transfer öğrenimi yaklaşımı kullanılarak sınıflandırma yapılmıştır. Eğitimler 10 iterasyon ile gerçekleştirilmiştir. Sonuç olarak AlexNet ve VGG16 gibi ImageNet veri seti üzerinde önceden eğitilmiş modeller, PlantVillage veri setinden alınan 13262 görüntü ile domates mahsulü hastalık sınıflandırma

başarıları gözlemlenmiştir. Yapılan deneysel araştırmalar sonucunda VGG16 modeli için %97.29 ve AlexNet modelinde %97.49 gibi başarı oranları elde edilmiştir.

Suh vd. (2018), makalede, alan koşullarında elde edilen şeker pancarı ve gönüllü(kendisi yetişen) patates bitkilerine ait görüntülerin transfer öğrenimi yöntemi ile sınıflandırılması üzerine çalışma yapmışlardır. Buna göre AlexNet ile önceden eğitilmiş konvolüsyonel sinir ağı modeli üzerinden 3 farklı senaryo uygulanılmıştır. Daha sonra AlexNet, VGG-19, GoogleNet, ResNet50, ResNet101 ve Inception-v3 gibi ImageNet veriseti üzerinde önceden eğitilmiş modellerin deneysel çalışmalar yapılarak başarı oranları karşılaştırılmış. Veri seti, 3 farklı yılda ve her yıl içerisinde 3 farklı dönemde 2 farklı toprak tipinden elde edilmiş olan 550 şeker pancarı ile 550 gönüllü patates bitkisine ait toplam 1100 adet görüntüden oluşmaktadır. Matlab ile bu görüntüler 227 x 227 piksele (RGB) yeniden ölçeklendirilmiştir. Yazılım tarafında, Matlab programı kullanılmıştır. Donanımsal platform olarak ise 2 farklı bulut servisinden yararlanılmış. Bunlar Amazon Elastic Compute Cloud (EC2) ve Paperspace GPU Cloud. Transfer öğrenimi yönteminin farklı ortam ve ışık koşulları altında şeker pancarı ve gönüllü patates görüntülerinin sınıflandırılması için başarılı bir performans sağladığı görülmüştür.

Traorea vd. (2018), çalışmalarında salgın hastalıkların erken tanı ve teşhisi amacıyla CNN kullanılarak tıbbi görüntü analizi yapmışlardır. Bu yöntemde salgın patojenin varlığını tespit etmeye yönelik araştırmalar gerçekleştirilmiştir. CNN modelinin eğitimi için 200 adet vibrio cholera ve 200 adet plasmodium falciparum bakterilerine ait görüntü kullanılmıştır. Test için ise 80 adet görüntü ile %94 sınıflandırma başarısına ulaşılmıştır. Salgın kriz yönetimi kapasitesini güçlendirmek amacıyla salgın patojenlerin görüntülerini mikroskopla otomatik sınıflandırmaya çalışılmıştır. Bunun için kullanılan derin öğrenme yönteminin görüntüleri tanıma hususunda ki başarısı incelenmiştir. Çalışmada 6 katmanlı bir konvolüsyon işlemi uygulanmıştır. Veri ön işleme sayesinde sınıflandırma başarısının arttığı gözlemlenmiştir. Yapılan veri ön işleme, iyi kalitede görüntü veri setinin oluşturulmasını sağlamıştır.

Wallelign vd. (2018), çalışmalarında soya fasulyesine ait yaprak görüntülerinden 3 farklı hastalık türü ile beraber 1 sağlıklı yaprak türünün tanı ve teşhisi için derin öğrenme yöntemini kullanmışlardır. Önerdikleri model, soya fasulyesi bitki hastalığı

sınıflandırmasını gerçekleştirmek için LeNet mimarisine dayanarak tasarlanmıştır. PlantVillage veri tabanından, sağlıklı yaprak görüntüleri de dahil olmak üzere dört sınıfın yaprak görüntülerini içeren 12,673 örnek imge alınmıştır. Uygulanan model, CNN'nin önemli özellikleri çıkarabildiğini ve bitki hastalıklarını doğal ortamda çekilen görüntülerden sınıflandırabileceğini açıkça gösteren %99.32 sınıflandırma doğruluğunu elde etmiştir. Modelde en yüksek başarı oranına ulaşabilmek için öğrenme hızı, filtre boyutu ve filtre sayısı gibi bir takım parametreler değiştirilip deneysel çalışma yapılmıştır. Ayrıca girdi olarak kullanılan yaprak görüntüleri; renkli, gri ve bölütlenmiş olarak üçe ayrılmıştır. Sonuçlar incelendiğinde ise renkli imgeler ile yapılan çalışmaların daha başarılı olduğu görülmüştür.

Zhang vd. (2018), GoogLeNet ve CIFAR10 modelleri kullanılarak mısır yaprağı görüntülerinden 8 farklı hastalık türü ile beraber sağlıklı yaprakların sınıflandırılması yapılmıştır. Çalışmalarında kullanılan modeller üzerinde veri arttırımı işlemi uygulanmıştır. 500 adet mısır yaprağı görüntüsünden veri arttırımı sayesinde 3060 adet görüntü elde edilmiştir. Veri setinde yer alan görüntülerin %80'i eğitim, %20'si test için ayrılmıştır. Bu çalışmada 9 farklı mısır yaprağı sınıflandırılırken GoogLeNet ve Cifar10 gibi gelişmiş konvolüsyonel sinir ağı modelleri sırasıyla %98.9 ve %98.8 gibi doğruluk oranlarını yakalamşlardır.

Geetharamani ve Arun Pandian (2019), çalışmalarında derin konvolüsyonel sinir ağı modeli ile bitkinin yaprak görüntülerinden hastalığın teşhisi için yeni bir CNN modeli önermişlerdir. Burada 39 farklı sınıf tanımlanmaya çalışılmış ve beraberinde modelin en iyi performansı sağlayabilmesi için bazı hiper parametreler değiştirilmiştir. Araştırma esnasında veri seti içerisindeki görüntüler üzerinde 6 farklı teknik (görüntü çevirme, PCA-renk artırımı, gama düzenlemesi, gürültü ekleme, döndürme ve ölçekleme) uygulanarak veri artırımı yöntemi uygulanmıştır. Veri artırımı sayesinde modelin performansının iyi sonuçlar verdiği gözlemlenmiştir. Önerilen model ile ResNet, AlexNet, VGG16 ve inceptionv3 gibi transfer öğrenimi yaklaşımları arasında karşılaştırmalı deneyler yapılmıştır. Elde edilen sonuçlara göre test verileri üzerinde önerilen modelin %97.87'lik doğruluk oranına ulaştığı görülmüştür. Ayrıca farklı iterasyon değerleri, batch-size ve seyreltme (dropout) değerleri modele uygulanmıştır. Önerilen modelin başarısının artması için çok sayıda eğitim verisine ihtiyaç olduğu saptanmıştır. Bu sebeple veri artırım tekniğinin gerekli

olacağı ifade edilmiştir. 55448 adet yaprak görüntüsü ile ve daha sonra veri setinde yer alan görüntüler arttırılarak 61486 adet yaprak görüntüsü ile önerilen model eğitilmiştir. Önerilen derin CNN mimarisi, geleneksel makine öğrenmesi modelleri arasında (SVM, Karar ağaçları, lojistik regresyon, KNN) yapılan karşılaştırmalı deneyler sonucunda en başarılı performansı göstermiştir.

Genshenga vd. (2019), makalede renk ve doku özellikleri çıkarılarak, çay yaprağının hastalık görüntülerinde hastalık lekeleri, destek vektör makinesi (SVM) yöntemi ile bölümlenmiştir. Ayrıca girdi olarak bölümlenmiş hastalık spot görüntüleri ile veri artışına yönelik geliştirilmiş koşullu derin konvolüsyonlu üretici ağlar (C-DCGAN) tarafından yeni eğitim örnekleri elde edilmiştir. Elde edilen bu görüntüler VGG16 derin öğrenme modeli ile eğitilmiştir. Bu deneyde, her çay yaprağı hastalık tipinden 40 görüntü seçilmiş ve hastalık spot görüntülerini elde etmek için 120 görüntünün tamamı bölümlenmiştir. Her tip hastalık spot görüntülerinden 20 örnek rastgele eğitim numunesi olarak seçilir ve geri kalan her 20 tip örnek test numunesi olarak kullanılmıştır. Eğitim örneklerinin renk özellikleri ve doku özellikleri çıkarılmış ve test örneklerini belirlemek için sırasıyla destek vektör makinesi, karar ağacı ve rastgele orman sınıfları eğitilmiştir. Her biri 4980 görüntü içeren 3 tür hastalık spot örnek görüntü üretmek için C-DCGAN ağından faydalanılmıştır. VGG16 veri arttırma yöntemiyle elde dilmiş olan örnekler üzerinden eğitilmiştir. Derin öğrenme yönteminin, çay yaprağına ait hastalıkları tanımlamak için etkili bir yöntem olduğu görülmüştür. Ancak derin öğrenme model eğitimi çok sayıda eğitim örneğine ihtiyaç duymaktadır. Deneysel sonuçlar, geleneksel makine öğrenme yöntemleri ile derin öğrenme yönteminin birlikte kullanılmasıyla, karmaşık bir arka plana sahip çay yaprağı görüntülerinde bile hastalığı etkili bir şekilde tanımlayabildiğini göstermiştir.

Francis ve Deisy (2019), çalışmalarında derin öğrenme modellerinden konvolüsyonel sinir ağını (CNN) kullanarak bitki hastalık tanı ve teşhisini gerçekleştirmişlerdir. Elma ve domates bitkilerine ait 3663 yaprak görüntüsünden oluşan veri seti ile önermiş oldukları CNN modelini eğitmişlerdir. Model 4 konvolüsyon + havuzlama katmanından oluşmaktadır. 8000 iterasyon ile eğitilen ağın doğruluk oranı %74 olarak ölçülürken iterasyon sayısı ve veri sayısı arttırılarak bu oran %87'ye ulaşmıştır. Çalışmalarında aşırı öğrenme durumunu önlemek amacıyla seyreltme

işlemi gerçekleştirilmiştir. Seyreltme hiper parametre değeri ise 0.2 olarak modele uygulanmıştır. Bunun sonucunda doğruluk oranı %88.7 olmuştur.

Kaya vd. (2019), çalışmalarında dört farklı transfer öğrenme modelinin bitki türlerinin sınıflandırılmasında başarı oranlarını karşılaştırmalı olarak deneysel çalışmalar yapmışlardır. Transfer öğrenimi metodunun farklı senaryolar ile farklı veri setleri üzerinden sınıflandırma başarıları ölçülmeye çalışılmıştır. Birçok deneysel işlem adımları uygulanmıştır. Transfer öğrenmenin amacı çok çaba gerektiren modelleri yeniden oluşturmak yerine önceden edinilen bilgiler kullanılıp ve model geliştirme sürecini önemli ölçüde azaltarak performansı iyileştirmektir. Bu çalışmada VGG16 ve AlexNet modelleri transfer öğrenimi yönteminde kullanılmıştır. Tüm derin öğrenme modelleri 100 iterasyon üzerinden eğitilmiştir. Kullanılan veri setleri %70 eğitim, %30 test için ayrılmıştır. Her veri seti türü içerdiği veri örnekleri sayıları bakımından farklılıklar göstermektedir. Bu çalışma transfer öğrenme yöntemi ile derin öğrenme modellerinin performansını önemli ölçüde geliştirdiğini göstermiştir. Özellikle transfer öğrenme yönteminde kullanılan derin özellik uygulaması ve ince ayar stratejilerinin diğer transfer öğrenimi stratejilerine göre daha iyi performans sunduğu da görülmüştür.

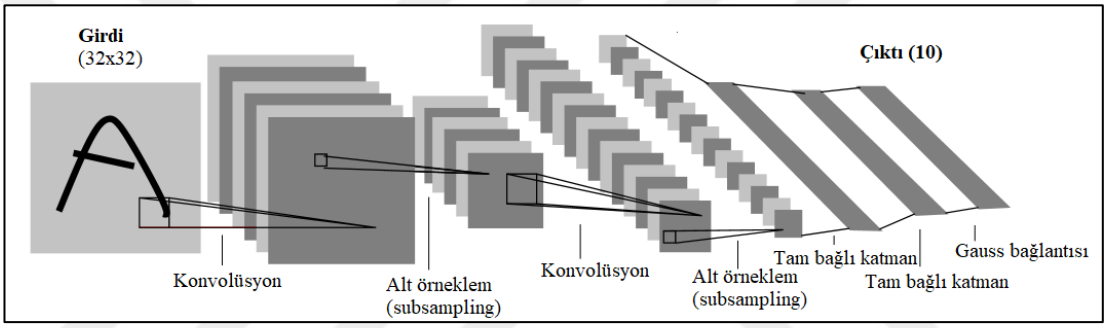
Picona vd. (2019), çalışmalarında daha önceden Johannes ve arkadaşları tarafından yapılmış olan bir çalışmayı genişletmişlerdir. Bunun için Deep Residual Neural Network (ResNet) tabanlı bir mimari kullanılmıştır. Gerçek koşullar altında bitki hastalıklarının tanı ve teşhisi için model tasarlanmıştır. Avrupada görülen endemik buğday türüne ait hastalığın erken teşhisine dayalı performansı analiz edilmiştir. Buna göre buğday bitkisine ait 3 hastalık türü ele alınmıştır. Çalışmada elde edilen sonuçlar johannes ve arkadaşlarının ulaştığı %78 gibi doğruluk oranından %87 doğruluk oranına yükseldiği görülmüştür. Derin evrişimsel sinir ağları (CNNs) görüntü sınıflandırma için bir atılım olarak görülüyor. Geleneksel bilgisayarlı görüntü işleme yaklaşımlarının olumsuz yönlerine alternatif CNN'ler hem tanımlayıcı hem de görüntüler üzerinden özellik çıkarıcı olarak kullanılmasına olanak sağlayan esnek bir çerçeve sunmuştur.

Toda ve Okura (2019), çalışmalarında kapalı kutu olarak tanımlanan CNN in bitki üzerindeki hastalıkları teşhisi noktasında katmanların etkilerini gözlemlemişlerdir.

Plantvillage veri setinde yer alan bitkiler kullanılmıştır. Hesaplama işlemlerini görselleştirme gibi insan tarafından yorumlanabilen bir biçimde ortaya koyarak CNN'leri anlamaya çalışmışlardır. Deneysel çalışmalarında önceden eğitilmiş olan inception v3 modelini kullanmışlardır.

3. MATERYAL VE YÖNTEM

Bu tez çalışmasındazeytin yaprağındaki hastalık ve zararlıların sınıflandırılması için derin öğrenme algoritmalarından CNN kullanılarak bir model tasarlanmıştır. Önerilen modelde VGGNet ve LeNet (LeCun vd., 1998) gibi mimarilerden esinlenilmiş olup deneysel çalışmalar sonucunda hiper parametreler ve katmanlar farklılaştırılmıştır. Şekil 3.1'de LeNet mimarisi verilmiştir. LeNet konvolüsyonel sinir ağı mimarisi LeCun ve arkadaşları tarafından MNIST verisetinde yer alan 0'dan 9'a kadar rakamların yer aldığı 60000 görüntü ile eğitilmiş bir modeldir (LeCun vd., 1998). Yapılan çalışmada 10000 test görüntüsü üzerinden model test edilmiş ve 0.7 hata oranı ile en iyi performansı yakalamıştır (LeCun vd., 1998).



Şekil 3.1. LeNet mimarisi (LeCun vd., 1998)

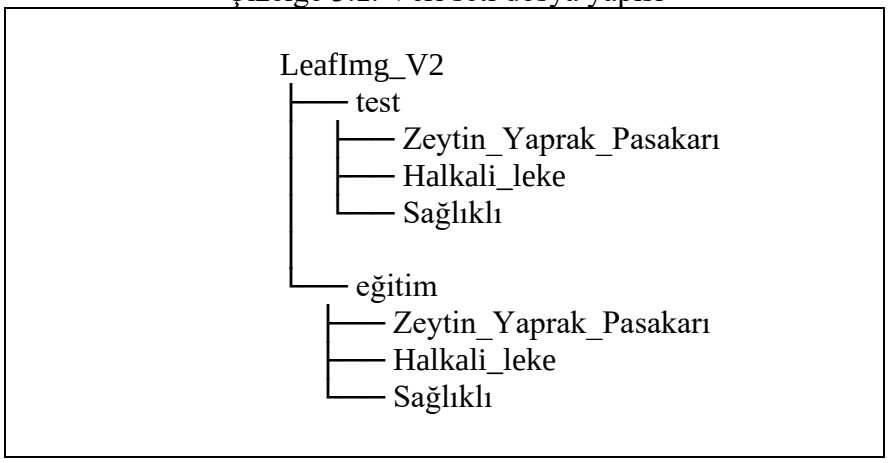
Tez çalışmasında önerilen CNN modeli ile farklı optimizasyon algoritmaları kullanılarak modelin test verileri üzerindeki başarı sonucu karşılaştırılmıştır. Görüntüler üzerinde hastalık ve zararlıların tanı ve teşhisinde en iyi başarı oranını yakalayabilmek için önerilen modelde bazı hiper parametre (katman sayısı, veri arttırımı, iterasyon sayısı, öğrenme hızı vb) değerleri değiştirilmiş ve buna bağlı olarak model tasarlanmıştır.

Yapılan deneysel çalışmalarda transfer öğrenme yöntemi de uygulanmıştır. Burada VGG16 ve VGG19 gibi mimariler kullanılmış olup karşılaştırmalı deneysel sonuçları tez çalışmasının sonuç bölümünde ele alınmıştır.

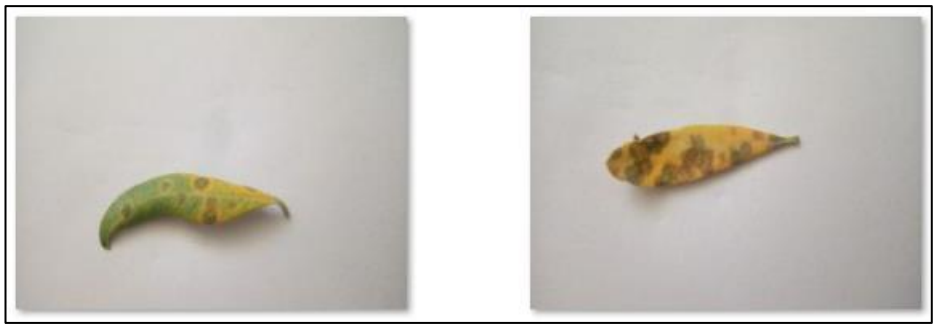
3.1. Veri Seti

Tez çalışmasında veriseti, kontrollü koşullar altında hazırlanan zeytin bitkisine ait yaprak görüntülerinden oluşmaktadır. Yapılan literatür araştırmasında kontrollü koşullar altında elde edilmiş zeytin yaprağı verisetine bilgimiz dahilindeulaşılamamıştır. Verisetinde yer alan zeytin yaprakları Denizli ilinden ilkbahar ve yaz mevsimlerinde toplanılmıştır. Mobil telefon (Sony Xperia C4 / Xiaomi Redmi 5 Plus**)** kullanılarak farklı ve yüksek çözünürlüklerde (3920 x 2208 pikselve 4000 x 3000 piksel) elde edilen görüntüler, verisetinin oluşturulduğu klasörde 800 x 600 piksel çözünürlüğünde yeniden ölçeklendilmiştir. Veriseti içerisinde yer alan yaprak görüntülerinin yeniden ölçeklendirilmesindeki amaç hesaplama maliyetini azaltarak görüntü işleme verimliliğini arttırmaktır (Maa vd., 2018). Verisetine ait hiyerarşik dosya yapısı Çizelge 3.1'de görülmektedir. Bilgisayarda LeafImg_V2 olarak adlandırılan klasör altında veriseti oluşturulmuştur. Burada toplam 3400 adet zeytin yaprağı görüntüsü yer almaktadır. Çizelge 3.1'de görüldüğü üzere test ve eğitim adında oluşturulan klasörlerin içerisinde Zeytin yaprak pasakarı, Halkalı leke ve Sağlıklı olmak üzere üç farklı sınıf yer almaktadır. Oluşturulan CNN modelimiz ile önceden eğitilmiş olan CNN modellerinin (VGG16, VGG19), verisetinde bulunan bu 3 sınıfı ayırt etmesi amaçlanmıştır.

Çizelge 3.1. Veri seti dosya yapısı



Şekil 3.2'de verisetinde yer alan halkalı leke hastalığına ait yaprak görüntü örnekleri verilmiştir.



Şekil 3.2. Halkalı leke hastalığı örnek görüntüleri

Şekil 3.3'te veri setinde yer alan zeytin akarı ve pasakarı zararlısına ait yaprak görüntü örnekleri verilmiştir. Burada belirtilen zararlı, yapraktan beslendiği için sınıflandırmada zeytin yaprak pasakarı olarak belirtilmiştir.

Şekil 3.3. Zeytin yaprak pasakarı örnek görüntüleri

Şekil 3.4'te veri setinde yer alan sağlıklı yaprak görüntü örnekleri verilmiştir.

Şekil 3.4. Sağlıklı zeytin yaprağı örnek görüntüleri

Çizelge 3.2'de veri setinde yer alan zeytin bitkisine ait yaprak görüntülerinin sınıflandırma bazında genel dağılımı verilmiştir. Bu görüntülerin %20'si test, %80'i eğitim olarak ayrılmıştır.

Çizelge 3.2. Veri setinde bulunan yaprak görüntülerine ait sayının genel dağılımı

EĞİTİM	Toplam eğitilen sağlıklı yaprak görüntü sayısı	830	2720	3400
	Toplam eğitilen halkalı leke hastalığı olan yaprak görüntü sayısı	1200		
	Toplam eğitilen zeytin yaprak pasakarı olan yaprak görüntü sayısı	690		
TEST	Toplam test için ayrılan sağlıklı yaprak görüntü sayısı	220	680	
	Toplam test için ayrılan halkalı leke yaprak görüntü sayısı	260		
	Toplam test için ayrılan zeytin yaprak pasakarı olan yaprak görüntü sayısı	200		

3.2. Kullanılan Yazılım ve Donanımlar

Tez çalışmasında tüm uygulamalar python programlama dili ile yazılmıştır. Pycharm ve Anaconda (Spyder) IDE platformları kullanılmıştır. Derin öğrenme kütüphanelerinden Keras 1.0.7 ve Tensorflow 1.13.1 kullanılmıştır. Web uygulama hazırlamak için Flask 0.12.2 kütüphanesi kullanılmıştır.

Model eğitimleri esnasındadonanımsal olarak Google Colab bulut servisinden yararlanılmıştır. Burada Tesla K80 GPU'lar üzerinden modeller farklı iterasyon sayılarında eğitilmiştir.

3.3. CNN Modelin Yapısı

Yapılan deneysel çalışmalar sonucunda önerilen CNN modeline ait katman mimarisi Çizelge 3.3'te belirtilmiştir. Buna göre 16 olan filtre sayısı artırım işlemi ile en son konvolüsyon katmanında 256 olarak uygulanmıştır. Literatürde filtre sayısının giderek artan bir sayıda belirlenmesi modelin başarısını da olumlu yönde etkilediği görülmüştür (Simonyan ve Zisserman, 2015). Önerilen mimaride filtre boyutu VGG16 modelinde olduğu gibi 3 x 3 olarak belirlenmiştir. Ayrıca ikili şekilde tekrar eden konvolüsyon katmanlarından sonra havuzlama katmanı mimariye dahil edilmiştir. Sadece ilk iki konvolüsyon katmanından hemen sonra yığın

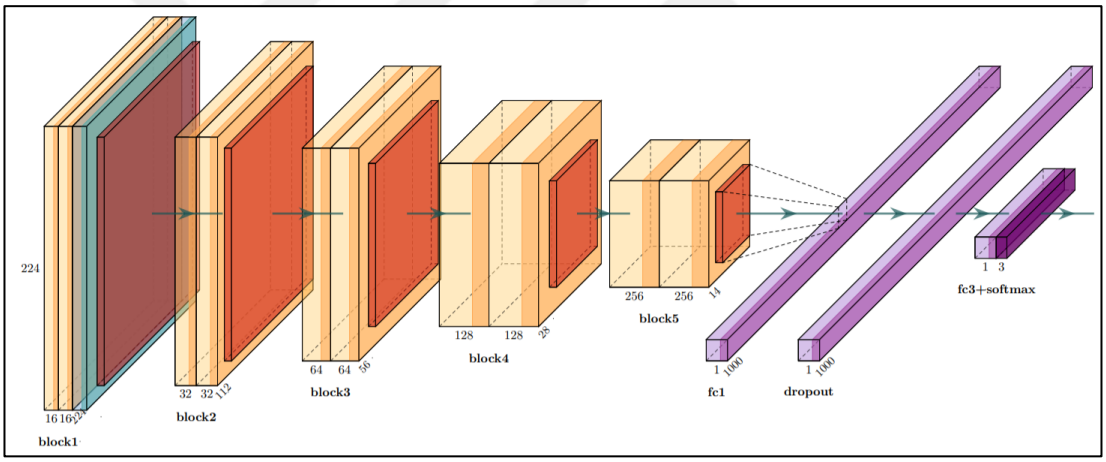
normalizasyonu işlemi bir defa uygulanmıştır. Son katmanlarda tam bağlı katman ile beraber seyreltme katmanı eklenmiştir.

Çizelge 3.3. Önerilen CNN modeline ait katman mimarisi

Katman	**Çıktı Şekli**
input_1 (InputLayer)	(None, 224, 224, 3)
block1_conv1 (Conv2D)	(None, 224, 224, 16)
block1_conv2 (Conv2D)	(None, 224, 224, 16)
block1_batch1 (BatchNormaliz)	(None, 224, 224, 16)
block1_maxpool1 (MaxPooling2)	(None, 112, 112, 16)
block2_conv1 (Conv2D)	(None, 112, 112, 32)
block2_conv2 (Conv2D)	(None, 112, 112, 32)
block2_maxpool1 (MaxPooling2)	(None, 56, 56, 32)
block3_conv1 (Conv2D)	(None, 56, 56, 64)
block3_conv2 (Conv2D)	(None, 56, 56, 64)
block3_maxpool1 (MaxPooling2)	(None, 28, 28, 64)
block4_conv1 (Conv2D)	(None, 28, 28, 128)
block4_conv2 (Conv2D)	(None, 28, 28, 128)
block4_maxpool1 (MaxPooling2)	(None, 14, 14, 128)
block5_conv1 (Conv2D)	(None, 14, 14, 256)
block5_conv2 (Conv2D)	(None, 14, 14, 256)
block5_maxpool1 (MaxPooling2)	(None, 7, 7, 256)
FC1 (Flatten)	(None, 12544)
dense1000 (Dense)	(None, 1000)
Drop1 (Dropout)	(None, 1000)
dense3 (Dense)	(None, 3)

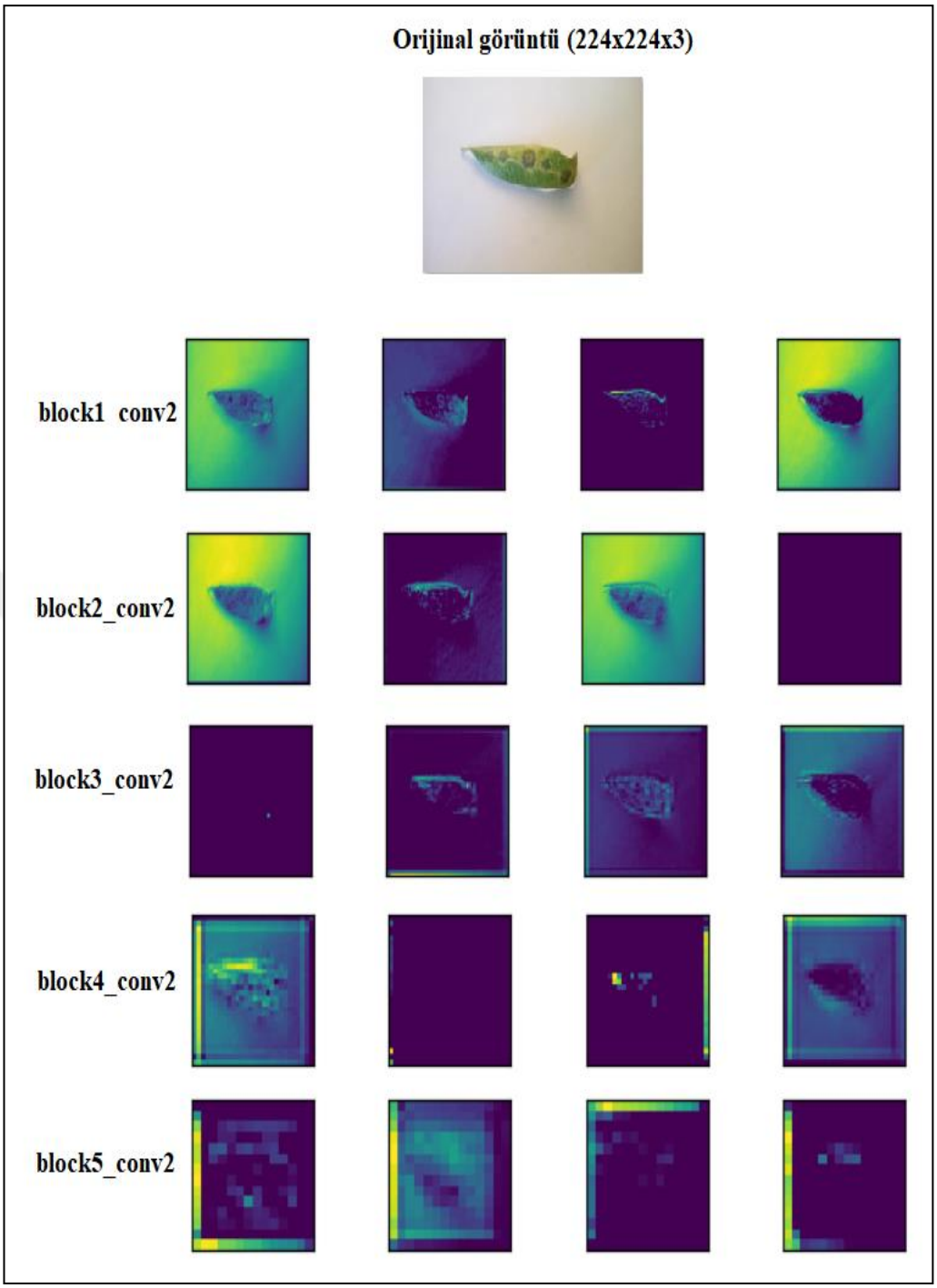
Şekil 3.5'te tez çalışmasında önerilen CNN modelinin genel yapısı görülmektedir. Model mimarisinde konvolüsyon katmanları için 3x3 lük filtre boyutu kullanılmıştır. Filtre sayıları ise Şekil 3.5'te görüldüğü üzere 16'dan 256'ya kadar arttırılarak uygulanmıştır. Birinci blokta ilk iki konvolüsyon işleminin hemen ardından yığın normalizasyonu işlemi eklenmiştir. Diğer bloklarda 2 konvolüsyon ve 1 maksimum havuzlama katmanı mevcuttur. Önerilen CNN mimarisini genel olarak özetleyecek olursak bu şekilde toplam 5 bloktan oluşmaktadır.

Konvolüsyon işlem adımlarının ardından görüntülerdeki doğrusal olmayan değerleri arttırabilmek için ReLu aktivasyon fonksiyonu kullanılmıştır. Son katmanda yer alan tam bağlı katman yapısında 3 yapay nöron ile beraber softmax aktivasyon fonksiyonu eklenmiştir. Zeytin bitkisine ait yaprak görüntülerinden 3 farklı sınıflandırma yapılacağı için softmax aktivasyon fonksiyonu kullanılmıştır.



Şekil 3.5. Önerilen CNN modelinin genel yapısı

Konvolüsyonel sinir ağına uygulanan giriş görüntülerinin katman geçişlerinde elde edilen sonuçları görselleştirilmiştir. Bu sayede konvolüsyonel sinir ağı mimarisinin nasıl çalıştığını gözlemleme imkanı elde edilmiştir. Şekil 3.6'da bu durum örnek üzerinde gösterilmiştir. Çizelge 3.3'te verilen CNN mimarisine göre her konvolüsyon katmanı sonrası elde edilen görüntü Şekil 3.6'da verilmiştir.



Şekil 3.6. Önerilen CNN modelinde farklı bloklardaki konvolüsyon katmanları sonrası elde edilen görüntüler

Keras derin öğrenme kütüphanesi kullanılarak oluşturulan mimaride 2 farklı şekilde konvolüsyonel sinir ağı modeli tasarlamak mümkündür. Bunlardan ilki Sıralı (Sequential) model, ikincisi ise Fonksiyonel (Functional API) modeldir. Sıralı modeller, ardışık katman mimarisine sahiptir. Ancak katmanların paylaşılmasına ve çoklu giriş çıkışlar tanımlanmasına izin vermez (Lin, 2017). Sıralı modele alternatif olarak fonksiyonel model geliştirilmiştir. Fonksiyonel model daha esnek CNN

modelleri tasarlanmasına imkân sağlar ve sıralı modellerin kısıtlı özelliklerine çözüm üretir (Brownlee, 2017c). Tez çalışmasında keras derin öğrenme kütüphanesi ile fonksiyonel model kullanılarak CNN mimarisi tasarlanmıştır.

Tez çalışmasında kullanılan CNN modelleri üzerinde sabit bırakılan bazı hiper parametre değerleri ise Çizelge 3.4'te verilmiştir.

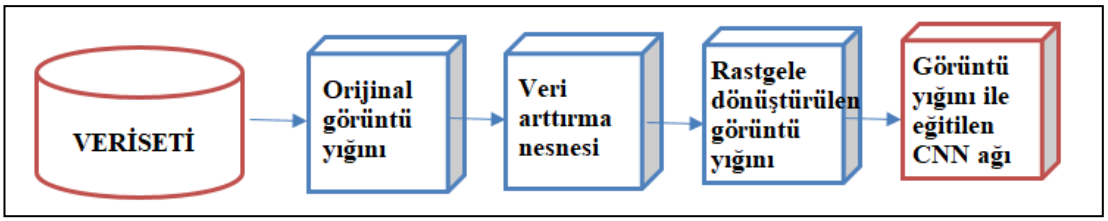
Çizelge 3.4. Tez çalışmasında belirlenen hiper parametre değerleri

Learning rate: 0.0001
Batch size: 2
Loss function: 'categorical_crossentropy'

3.4. Veri Arttırım Yönteminin Modele Uygulanması

Veri arttırma yöntemi, verisetinde yer alan görüntülere farklı teknikler uygulayarak (yakınlaştırma, döndürme, kırpma vb.) yeni eğitim örnekleri elde etme işlemidir (Rosebrock, 2019). Veri artırım yöntemi, tez çalışmasında sadece eğitim verileri üzerinde uygulanmıştır. Bu yöntemde dikkat edilmesi gereken bir husus, orijinal görüntü üzerinden elde edilmiş olan yeni görüntülerin verisetinde yer alan sınıf etiketlerinin aynı kalması olacaktır (Rosebrock, 2019). Yapılan literatür taramalarında, veri arttırma yönteminin konvolüsyonel sinir ağında performans artışı sağladığı görülmüştür (Wallelign vd., 2018; Zhang vd., 2018; Geetharamani ve Arun Pandian, 2019).

Veri arttırma işlemi için tez çalışmasında, Keras derin öğrenme kütüphanesi içerisinde yer alan Image Data Genarator sınıfı kullanılmıştır. Image Data Generator, eğitim esnasında görüntüler üzerinde istenilen dönüşümleri sağlamaktadır (Rosebrock, 2019). Bu sayede gerçek zamanlı olarak yeni veriler üretilmektedir. Image Data Genarator sınıfı, eğitim esnasında orijinal verileri alıp rastgele dönüştürerek yapay sinir ağına geri döndürüyor (Rosebrock, 2019). Bu süreç Şekil 3.7'de görülmektedir. Her iterasyonda bu durum gerçekleşmektedir.



Şekil 3.7. Keras ile uygulanan görüntü arttırma işlemi (Rosebrock, 2019)

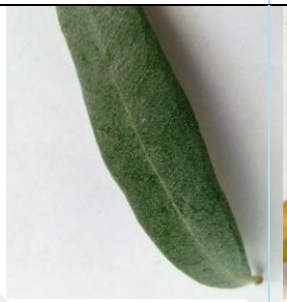
Tez çalışmasında kullanılan Image Data Genarator sınıfı ile veri arttırımı sağlama amaçlı uygulanan bazı görüntü işleme teknikleri Çizelge 3.5'te detaylı olarak verilmiştir.

Çizelge 3.5. Tez çalışmasında ImageDataGenerator ile uygulanan görüntü işleme teknikleri

Görüntü işleme tekniği	Açıklama
zoom_range=[0.5,1.0]	Görüntünün rastgele yakınlaştırılması işlemidir. Yakınlaştırma miktarı genişlik ve yükseklik belirtilerek gerçekleştirilir. Görüntüde yer alan nesnenin en boy oranına dikkat edilerek rastgele yakınlaştırma işlemi sağlanacaktır (Brownlee, 2019c).
brightness_range=[0.2,1.0]	Görüntüdeki aydınlatma seviyelerini değiştirmeyi sağlar. Uygulanan kod örneğinde %20 oranında görüntü üzerinde karartma işlemi yapılmıştır (Brownlee, 2019c).
horizontal_flip=True	Görüntünün yatayda ters çevrilmesi işlemidir (Brownlee, 2019c).
rotation_range=90	Görüntüyü 0-90 derece arasında saat yönünde rastgele döndürmesi işlemidir (Brownlee, 2019c).

Tez çalışmasında Image Data Generator sınıfı ile veri arttırımı sonucu oluşan görüntü örnekleri Çizelge 3.6'da verilmiştir.

Çizelge 3.6. Veri arttırımı sonucu oluşan görüntü örnekleri

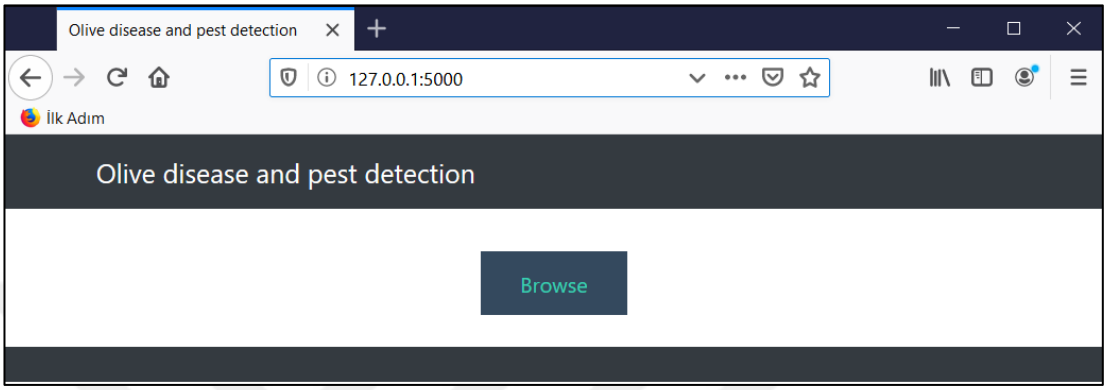
Orijinal görüntü			
Veri arttırım sonucu oluşan görüntüler			
			
			

3.5. Modelin Web Arayüzüne Aktarılması

Tez çalışmasında oluşturulan CNN modelinin uygulama olarak web sitesine eklenmesi aşamasında Flask Kütüphanesi kullanılmıştır. Flask, python ile yazılmış açık kaynak kodlu bir kütüphanedir. Web uygulamaları oluşturmak için oldukça kolay ve kullanışlıdır (Yu ve Malan, 2018).

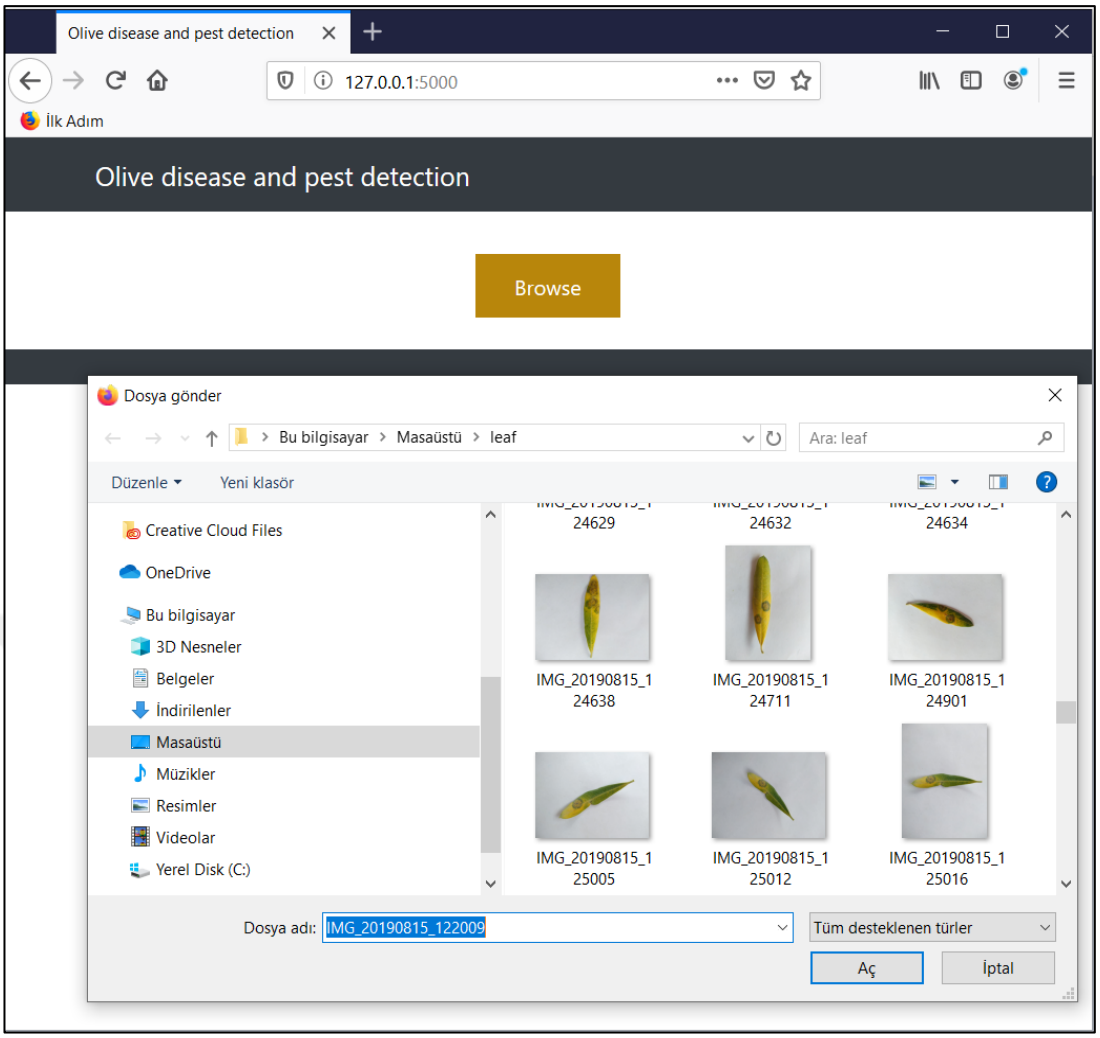
Anasayfanın tasarlandığı index.html dosyası python uygulamasına Flask sayesinde eklenmiştir (Anonim, 2019b). Tez çalışmasında kullanıcı tarafında görüntülerin

uygulamaya yüklenebilmesi ve buna görre uygulamanın kullanıcıya yanıt verebilmesi için GET ve POST metodları kullanılmıştır. Form oluşturmaya yarayan bu metodlar Flask Kütüphanesi ile python uygulamasına eklenmiştir. Web uygulaması yerelde (local host) çalıştırıldığında Şekil 3.8'de olduğu gibi bir arayüz kullanıcıyı karşılamaktadır.



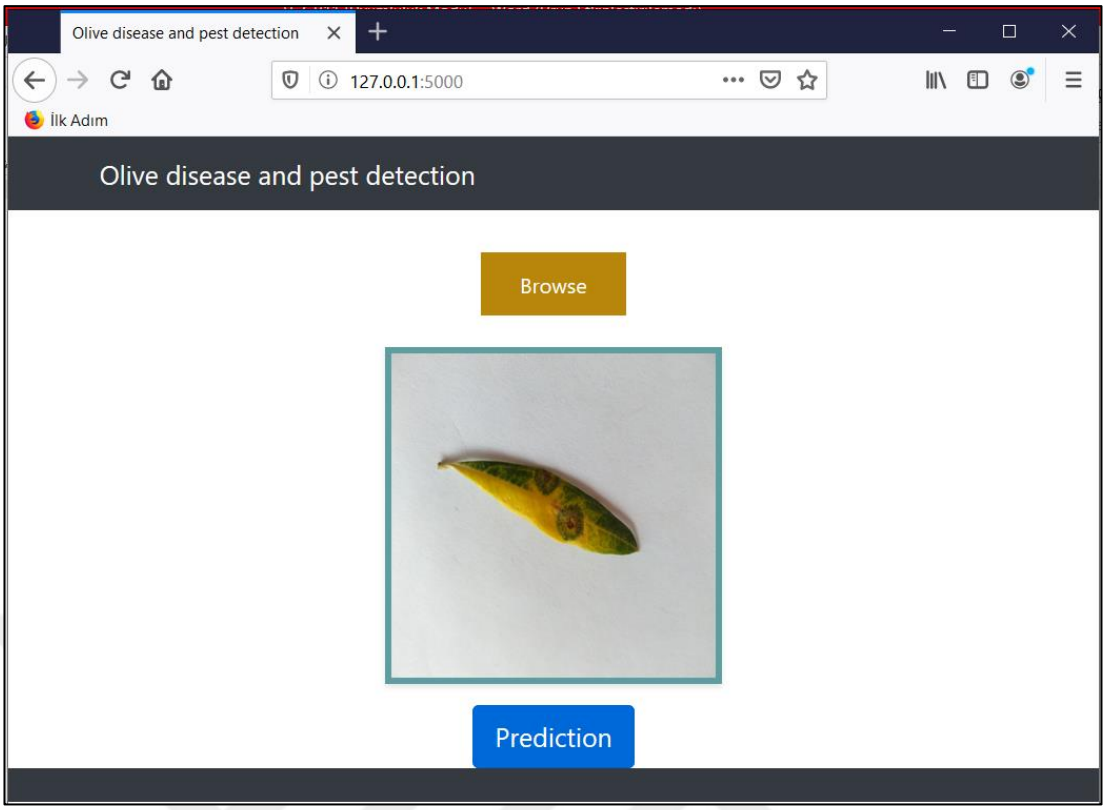
Şekil 3.8. Web site genel arayüzü (Anonim, 2019b)

Kullanıcı hazırlanan arayüzde hastalık ve zararlı yönünden sınıflandırmaya çalıştığı örnek görüntüyü 'Seç' butonuna tıklayarak uygulamaya yüklemesi gerekmektedir. Bu işlem adımının devamı Şekil 3.9'da verilmiştir.



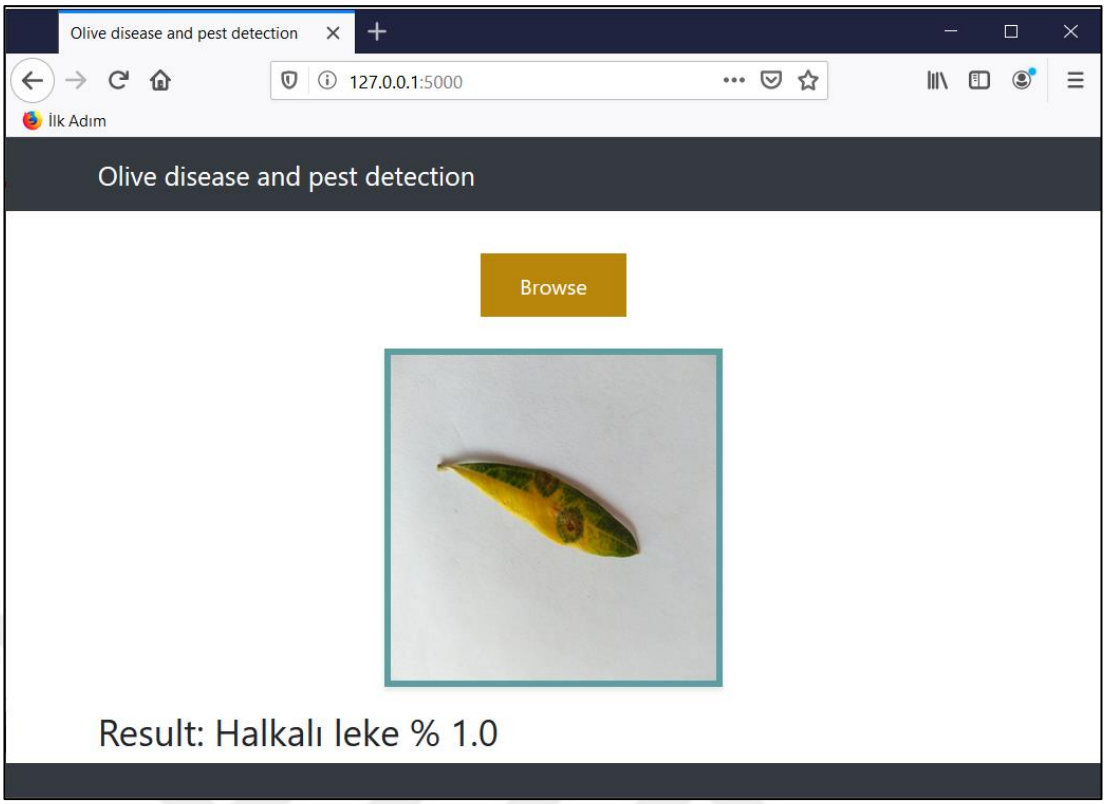
Şekil 3.9. Uygulamaya yüklenecek dosyanın seçilmesi

Şekil 3.10'da yüklenen görüntüyü derin öğrenme modelimizin zeytin bitkisi hastalık ve zararlıları tanı ve teşhisi için tahmin etmesi istenilmektedir.



Şekil 3.10. Yüklenen görüntünün tahmin edilmesi

Şekil 3.11'de Derin öğrenme modelinin tahmin sonucu olasılık değeri ile beraber döndürülmektedir.



Şekil 3 11. Derin öğrenme modelinin tahmin sonucu

4. BULGULAR VE TARTIŞMA

Oluşturulan verisetinde, önerilen CNN modeli ve transfer öğrenme yöntemi uygulanan modellerin sınıflandırma başarıları karşılaştırılmıştır. Yapılan deneyler esnasında önerilen CNN modeli, VGG16 ve VGG19 olmak üzere 3 farklı model tasarlanmıştır. Tez çalışmasında önerilen CNN modeli sıfırdan eğitilmiş olup VGG16 ve VGG19 gibi önceden eğitilmiş modeller öznitelik çıkarıcı olarak kullanılmış ve transfer öğrenme yöntemi uygulanmıştır. Herbir model üzerinde farklı optimizasyon algoritmaları (SGD, Adam, AdaGrad, RMSprop) sırasıyla uygulanmış ve bu algoritmaların model performansı üzerindeki sonuçları gözlemlenmiştir. Ayrıca eğitimler esnasında iterasyon sayıları 30 ve 100 olarak belirlenmiş olup konvolüsyonel sinir ağı modellerinin belirlenen iterasyon sayılarındaki başarı oranları karşılaştırılmış ve çizelgelere dökülmüştür. Karşılaştırmalı deneylerde doğruluk, hata matrisi, duyarlılık, kesinlik ve F1 skorlama sonuçları değerlendirilmiştir.

Doğruluk: Verisetinde doğru tahmin edilen değerlerin sayısı, verisetinde ki tüm örneklerin sayısına bölünmesidir.

$$\textbf{Doğruluk} = \frac{\textbf{Doğru Tahmin Sayısı}}{\textbf{Tüm Örnek Sayısı}} \tag{4.1}$$

Başarı ölçütlerinden hata matrisini açıklayabilmek için bazı kavramların örnekler üzerinden tanımlanması gerekmektedir. Bu kavramlar: TP (True Positive – Doğru Pozitif); gerçekte pozitif olan ve sınıflandırmadapozitif olarak tahmin edilen örnekleri belirtmektedir (Anonim, 2018b). FP (False Positive – Yanlış Pozitif); gerçekte negatif olan ve sınıflandırmada pozitif olarak tahmin edilen örnekleri belirtmektedir (Anonim, 2018b). TN (True Negative – Doğru Negatif); gerçekte negatif olan ve sınıflandırmada negatif olarak tahmin edilen örnekleri belirtmektedir (Anonim, 2018b). FN (False Negative – Yanlış Negatif); gerçekte pozitif olan ve sınıflandırmada negatif olarak tahmin edilen örnekleri belirtmektedir (Anonim, 2018b). Açıklanan bu kavramlar Çizelge 4.1'de hata matrisi tablosu olarak verilmiştir. Diğer tüm başarı ölçütleri (duyarlılık, kesinlik ve F1 skorlama) bu kavramlar üzerinden tanımlanacaktır.

Çizelge 4.1. Hata matrisi

		GERÇEK DEĞER	
		Pozitif	Negatif
TAHMİN	Pozitif	TP	FP
	Negatif	FN	TN

Duyarlılık; hasta olan yaprakları doğru tespit etme oranı olarak açıklanabilir (Şimşek, 2018).

$$\textbf{Duyarlılık} = \frac{\textbf{TP}}{\textbf{TP+FN}} \tag{4.2}$$

Kesinlik; hasta olarak sınıflandırılan yaprakların gerçekte kaç tanesinin hastalıklı olduğunun bulunmasıdır (Şimşek, 2018).

$$\textbf{Kesinlik} = \frac{\textbf{TP}}{\textbf{TP+FP}} \tag{4.3}$$

F1-skorlama; dengesiz bir şekilde dağılan verisetindeki durumları harmonik ortalama ile tespit edilmesidir (Şimşek, 2018).

$$\textbf{F1 - skorlama} = 2 * \frac{\textbf{duyarlılık*kesinlik}}{\textbf{duyarlılık+kesinlik}} \tag{4.4}$$

Çizelge 4.2'de 30 iterasyon sonucunda ve arttırım işlemi uygulanmadan kullanılan CNN modellerine ait farklı optimizasyon algoritmalarının uygulanmasıyla elde edilen başarı ölçütleri karşılaştırılmıştır. Veri örneklem sayısı dengesiz bir dağılıma sahip olan verisetleri için en uygun değerlendirme kriteri F1-skorlama değeridir. Çizelge 4.2'de görüleceği üzere sağlıklı zeytin yapaklarının sınıflandırılmasında en iyi F1-skorlama değeri %89 ile AdaGrad optimizasyon algoritması kullanılan VGG16 modeli olmuştur. En düşük F1-skorlama değeri %39 ile RMSProp optimizasyon algoritması kullanılan tez çalışmasında önerilen CNN modeli olmuştur. Benzer şekilde halkalı leke zeytin yapaklarının sınıflandırılmasında en iyi F1-skorlama değeri %89 ile SGD optimizasyon algoritması kullanılan VGG19 modeli olmuştur. Zeytin yaprak pasakarı zeytin yaprakları için ise en iyi F1-skorlama değeri

%90 ile Adam optimizasyon algoritması kullanılan önerilen CNN modeli olmuştur. Bu bağlamda sonuçlara genel olarak baktığımızda değerlendirme kriterlerinden F1-skorlama değeri için en iyi performansın gelişmiş CNN modellerinde görüldüğü söylenebilir.

Çizelge 4.2. 30 iterasyon sonucunda CNN modellerinin performans karşılaştırması

SINIFLANDIRMA	DERİN ÖĞRENME MODELLERİ	OPTİMİZASYON ALGORİTMALARI	Doğruluk (Accuracy)	Duyarlılık (Recall)	Kesinlik (Precision)	F1 Skoru
SAĞLIKLI	**ÖNERİLEN CNN MODELİ**	Adam	0.82	0.59	0.90	0.71
		AdaGrad	0.70	0.64	0.61	0.83
		SGD	0.65	0.74	0.45	0.56
		RMSprop	0.69	0.24	0.98	0.39
	VGG16	Adam	0.87	0.89	0.82	0.86
		AdaGrad	0.86	0.89	0.90	0.89
		SGD	0.87	0.84	0.91	0.88
		RMSprop	0.86	0.85	0.89	0.87
	VGG19	Adam	0.87	0.84	0.92	0.88
		AdaGrad	0.87	0.90	0.86	0.88
		SGD	0.89	0.82	0.93	0.87
		RMSprop	0.70	0.94	0.54	0.69
HALKALI LEKE	**ÖNERİLEN CNN MODELİ**	Adam	0.82	0.97	0.71	0.82
		AdaGrad	0.70	0.68	0.62	0.65
		SGD	0.65	0.35	0.66	0.46
		RMSprop	0.86	0.99	0.52	0.69
	VGG16	Adam	0.87	0.91	0.84	0.88
		AdaGrad	0.86	0.79	0.90	0.84
		SGD	0.87	0.82	0.90	0.86
		RMSprop	0.86	0.80	0.87	0.83
	VGG19	Adam	0.87	0.82	0.91	0.86
		AdaGrad	0.87	0.83	0.91	0.87
		SGD	0.89	0.90	0.88	0.89
		RMSprop	0.70	0.55	0.75	0.63
ZEYTİN YAPRAK PASAKARI	**ÖNERİLEN CNN MODELİ**	Adam	0.82	0.88	0.92	0.90
		AdaGrad	0.70	0.78	0.88	0.83
		SGD	0.65	0.83	0.92	0.87
		RMSprop	0.69	0.78	0.95	0.86
	VGG16	Adam	0.87	0.81	0.93	0.87
		AdaGrad	0.86	0.91	0.82	0.86
		SGD	0.87	0.93	0.82	0.87
		RMSprop	0.86	0.90	0.82	0.86
	VGG19	Adam	0.87	0.93	0.81	0.86
		AdaGrad	0.87	0.89	0.85	0.87
		SGD	0.89	0.92	0.86	0.89
		RMSprop	0.70	0.64	0.97	0.77

Çizelge 4.3'te 100 iterasyon sonucunda, kullanılan CNN modellerine ait farklı optimizasyon algoritmalarının uygulanmasıyla elde edilen başarı ölçütleri karşılaştırılmıştır. Modeller üzerinde iterasyon sayısının artmasıyla F1-skor değerinin önerilen CNN modeli için başarılı bir grafik çizdiği görülmektedir.

Çizelge 4.3. 100 iterasyon sonucunda CNN modellerinin performans karşılaştırması

SINIFLANDIRMA	DERİN ÖĞRENME MODELLERİ	OPTİMİZASYON ALGORİTMALARI	Doğruluk (Accuracy)	Duyarlılık (Recall)	Kesinlik (Precision)	F1 Skoru
SAĞLIKLI	**ÖNERİLEN CNN MODELİ**	Adam	0.85	0.88	0.80	0.84
		AdaGrad	0.73	0.77	0.64	0.70
		SGD	0.78	0.69	0.73	0.71
		RMSprop	0.71	0.36	0.87	0.51
	VGG16	Adam	0.88	0.83	0.91	0.87
		AdaGrad	0.86	0.87	0.91	0.89
		SGD	0.88	0.84	0.92	0.88
		RMSprop	0.86	0.77	0.94	0.85
	VGG19	Adam	0.81	0.78	0.85	0.81
		AdaGrad	0.87	0.86	0.88	0.87
		SGD	0.88	0.86	0.88	0.87
		RMSprop	0.57	0.12	0.92	0.20
HALKALI LEKE	**ÖNERİLEN CNN MODELİ**	Adam	0.85	0.81	0.84	0.83
		AdaGrad	0.73	0.58	0.73	0.65
		SGD	0.78	0.80	0.72	0.75
		RMSprop	0.71	0.98	0.56	0.71
	VGG16	Adam	0.88	0.91	0.87	0.89
		AdaGrad	0.86	0.80	0.90	0.84
		SGD	0.88	0.85	0.90	0.88
		RMSprop	0.86	0.95	0.80	0.87
	VGG19	Adam	0.81	0.77	0.85	0.81
		AdaGrad	0.87	0.81	0.91	0.86
		SGD	0.88	0.85	0.91	0.88
		RMSprop	0.57	0.58	0.73	0.64
ZEYTİN YAPRAK PASAKARI	**ÖNERİLEN CNN MODELİ**	Adam	0.85	0.85	0.89	0.87
		AdaGrad	0.73	0.82	0.80	0.81
		SGD	0.78	0.83	0.87	0.85
		RMSprop	0.71	0.77	0.93	0.84
	VGG16	Adam	0.88	0.89	0.86	0.88
		AdaGrad	0.86	0.92	0.82	0.86
		SGD	0.88	0.93	0.83	0.88
		RMSprop	0.86	0.85	0.86	0.86
	VGG19	Adam	0.81	0.87	0.76	0.81
		AdaGrad	0.87	0.92	0.83	0.87
		SGD	0.88	0.90	0.85	0.88
		RMSprop	0.57	0.93	0.50	0.65

Hata matrisi ile de oluşturulan modellerin sınıflandırma performansı incelenmiştir. Buna göre Çizelge 4.4'te önerilen CNN modeline, uygulanan farklı optimizasyon algoritmalarının 30 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.4. Önerilen CNN modeline ait 30 iterasyon sonucunda oluşan hata matrisi

Adam

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	229	21	10
	Halkalı Leke	4	213	3
	Sağlıklı	15	67	118

RMSprop

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	204	56	0
	Halkalı Leke	1	218	1
	Sağlıklı	9	142	49

AdaGrad

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	202	36	22
	Halkalı Leke	12	149	59
	Sağlıklı	15	57	128

SGD

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	216	5	39
	Halkalı Leke	3	77	140
	Sağlıklı	16	35	149

Çizelge 4.5'te önerilen CNN modeline ait, uygulanan farklı optimizasyon algoritmalarının 100 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.5. Önerilen CNN modeline ait 100 iterasyon sonucunda oluşan hata matrisi

Adam

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	220	16	24
	Halkalı Leke	20	179	21
	Sağlıklı	7	17	176

RMSprop

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	199	54	7
	Halkalı Leke	1	215	4
	Sağlıklı	13	115	72

AdaGrad

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	212	22	26
	Halkalı Leke	32	128	60
	Sağlıklı	21	25	154

SGD

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	216	24	20
	Halkalı Leke	15	175	30
	Sağlıklı	17	45	138

Çizelge 4.6'da VGG16 modeline ait, uygulanan farklı optimizasyon algoritmalarının 30 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.6. VGG16 modeline ait 30 iterasyon sonucunda oluşan hata matrisi

Adam					**RMSprop**				
		TAHMİN					TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı			Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	211	21	28	GERÇEK DEĞER	Zeytin Yaprak Pasakarı	234	14	12
	Halkalı Leke	9	201	10		Halkalı Leke	34	177	9
	Sağlıklı	6	16	178		Sağlıklı	16	13	171

AdaGrad					**SGD**				
		TAHMİN					TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı			Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	237	13	10	GERÇEK DEĞER	Zeytin Yaprak Pasakarı	241	11	8
	Halkalı Leke	37	174	9		Halkalı Leke	31	181	8
	Sağlıklı	16	7	177		Sağlıklı	22	9	169

Çizelge 4.7'de VGG16 modeline ait, uygulanan farklı optimizasyon algoritmalarının 100 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.7. VGG16 modeline ait 100 iterasyon sonucunda oluşan hata matrisi

Adam					**RMSprop**				
		TAHMİN					TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı			Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	231	18	11	GERÇEK DEĞER	Zeytin Yaprak Pasakarı	222	31	7
	Halkalı Leke	15	200	5		Halkalı Leke	10	208	2
	Sağlıklı	22	12	166		Sağlıklı	26	20	154

AdaGrad					**SGD**				
		TAHMİN					TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı			Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	239	12	9	GERÇEK DEĞER	Zeytin Yaprak Pasakarı	241	11	8
	Halkalı Leke	36	175	9		Halkalı Leke	26	187	7
	Sağlıklı	18	8	174		Sağlıklı	22	9	169

Çizelge 4.8'de VGG19 modeline ait, uygulanan farklı optimizasyon algoritmalarının 30 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.8. VGG19 moddeline ait 30 iterasyon sonucunda oluşan hata matrisi

Adam

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	241	8	11
	Halkalı Leke	36	180	4
	Sağlıklı	21	10	169

RMSprop

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	167	31	62
	Halkalı Leke	3	120	97
	Sağlıklı	3	9	188

AdaGrad

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	232	11	17
	Halkalı Leke	26	182	12
	Sağlıklı	14	6	180

SGD

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	239	12	9
	Halkalı Leke	18	198	4
	Sağlıklı	21	14	165

Çizelge 4.9'da VGG19 modeline ait, uygulanan farklı optimizasyon algoritmalarının 100 iterasyon sonucunda sınıflandırma performansları verilmiştir.

Çizelge 4.9. VGG19 modeline ait 100 iterasyon sonucunda oluşan hata matrisi

Adam

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	226	12	22
	Halkalı Leke	44	170	6
	Sağlıklı	27	18	155

RMSprop

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	241	19	0
	Halkalı Leke	91	127	2
	Sağlıklı	178	29	23

AdaGrad

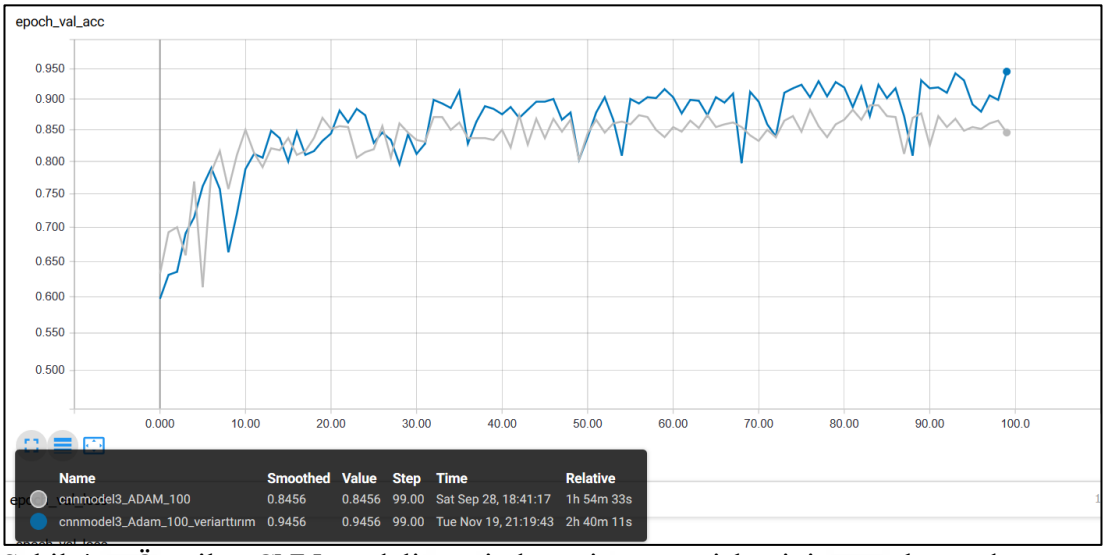
		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	238	11	11
	Halkalı Leke	30	178	12
	Sağlıklı	20	7	173

SGD

		TAHMİN		
		Zeytin Yaprak Pasakarı	Halkalı Leke	Sağlıklı
GERÇEK DEĞER	Zeytin Yaprak Pasakarı	235	12	13
	Halkalı Leke	22	188	10
	Sağlıklı	20	7	173

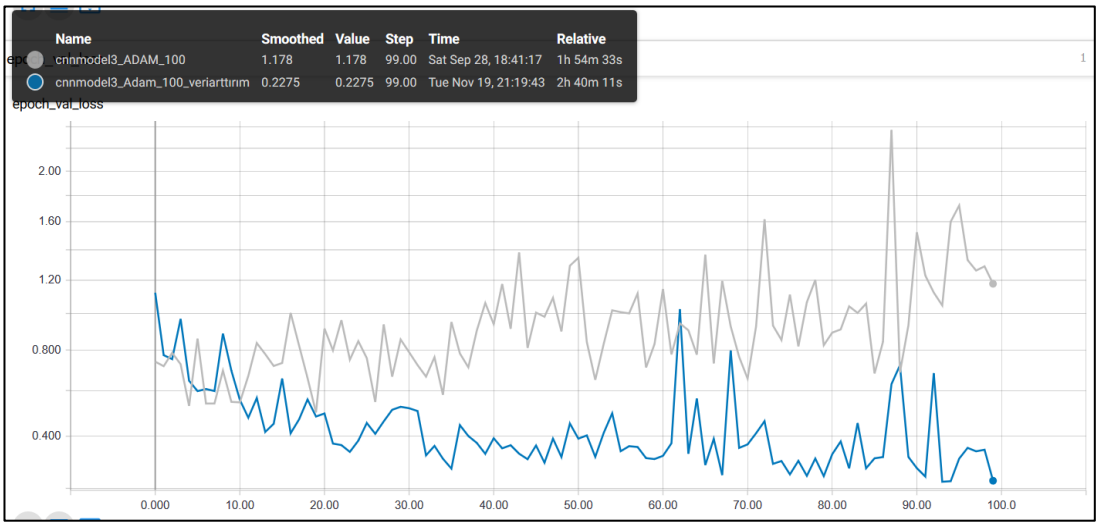
Önerilen CNN modeli üzerinde Keras kütüphanesinde yer alan ImageDataGenerator sınıfı ile veri arttırım işlemi uygulanmıştır. Modelde Adam optimizasyon algoritması kullanılmıştır. Bu işlemin ardından CNN modeli 100 iterasyon ile eğitimi gerçekleştirilmiştir. Eğitim sonucunda elde edilen bulgular Tensor Board kullanılarak Şekil 4.1'de verilmiştir. Grafikte verilen iki modeli de karşılaştırdığımızda veri arttırım işlemi uygulanmış olan Konvolüsyonel Sinir Ağı modelimizin daha iyi bir performans gösterdiği görülmektedir. Şekilde verilen grafikte veri arttırım işlemi uygulanan CNN modelinde doğruluk oranı %94 iken, aynı modelin veri arttırım işlemi uygulanmadan doğruluk oranı ise %84 olarak ölçülmüştür. TensorBoard'da verilen grafiğin parametreleri aşağıdaki gibi açıklanabilir.

1. Value: Modelin doğruluk oranı,
2. Step: Modelin iterasyon sayısı,
3. Time: Model eğitiminin gerçekleştirildiği tarih ve saat,
4. Realative: Modelin toplam eğitim süresidir.



Şekil 4.1. Önerilen CNN modeli üzerinde veri arttırım işleminin uygulanmadan önce ve sonra doğruluk oranı grafiğinin karşılaştırılması

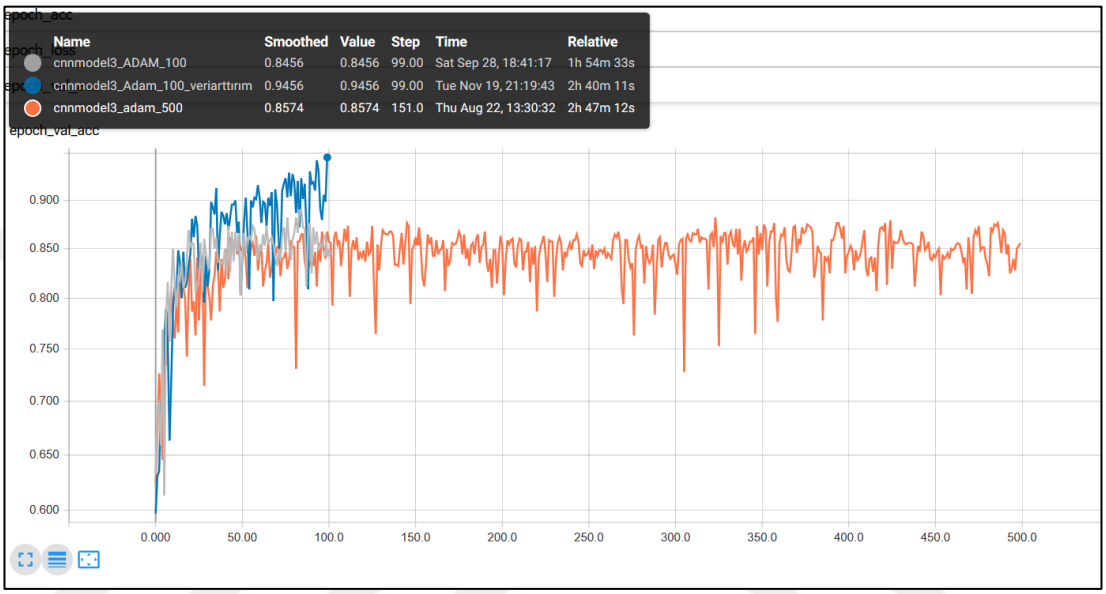
Şekil 4.2'de, Şekil 4.1'de gösterilen modellerin hata oranı verilmiştir. Grafik incelendiğinde veri arttırım işlemi uygulanmamış olan CNN modeline ait hata oranı 1.178 iken veri arttırım işlemi uygulanmış olan CNN modeline ait hata oranı 0.227 olarak ölçülmüştür.



Şekil 4.2. Önerilen CNN modeli üzerinde veri arttırım işleminin uygulanmadan önce ve sonra hata oranı grafiğinin karşılaştırılması

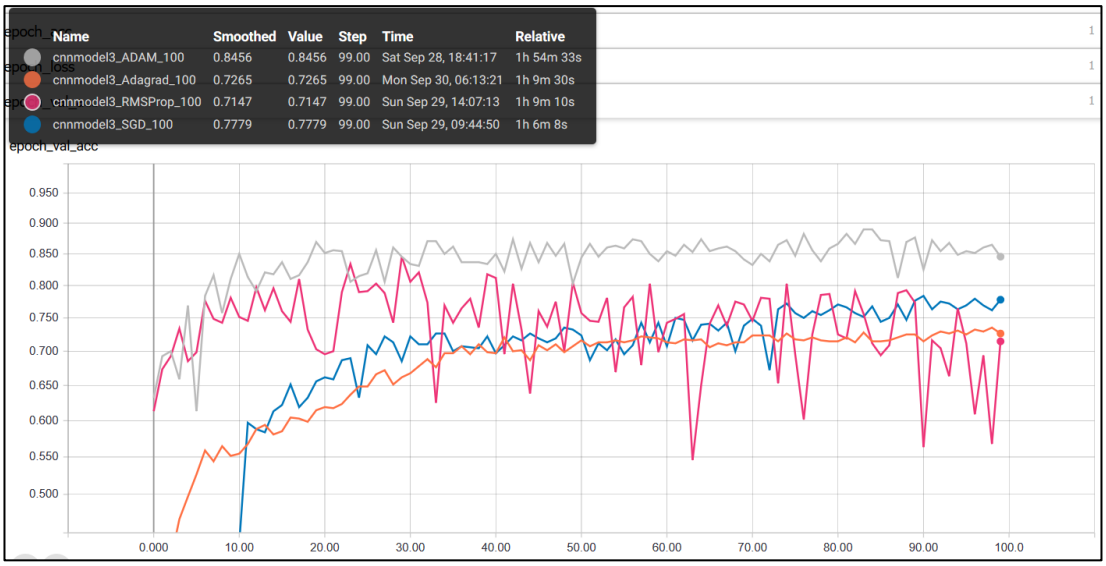
Şekil 4.3'te önerilen CNN modeline ait 3 durum verilmiştir. Verilen grafikte bu modele uygulanan veri arttırım tekniği ile iterasyon sayısının arttırılması yöntemi ayrı ayrı modele uygulanmıştır. Adam optimizasyon algoritması 3 durum için de

kullanılmıştır. İlk durumda 100 iterasyon ile model eğitilmiştir ve %84 doğruluk oranı elde edilmiştir. İkinci durumda veri arttırım tekniği 100 iterasyon ile uygulanmıştır. Şekil 4.3'te görüldüğü üzere modelin doğruluk oranı %94 olarak ölçülmüştür. Aynı modele üçüncü durumda ise veri arttırım tekniği uygulanmadan sadece iterasyon sayısı 500'e çıkartılarak sonuç gözlemlenmiştir. Buna göre %85 doğruluk oranına ulaşılmıştır.



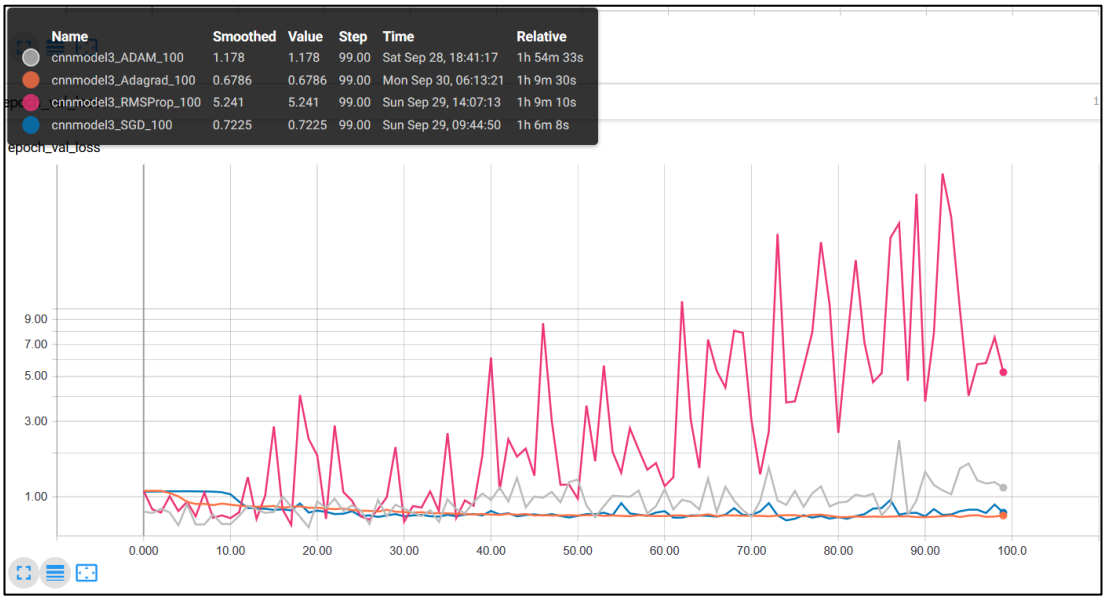
Şekil 4.3. Önerilen CNN modeline veri arttırım yöntemi uygulanması ve iterasyon sayısının arttırılması durumlarının doğruluk oranına etkisi

Şekil 4.4'te önerilen CNN modeli üzerinde 100 iterasyon ile optimizasyon algoritmalarının doğruluk oranı grafiksel olarak karşılaştırılmıştır. Elde edilen bulgular Tensor Board kullanılarak Şekil 4.4'te gösterilmiştir. Şekil 4.4'te verilen grafik incelendiğinde AdaGrad optimizasyon algoritması ile oluşturulan Konvolüsyonel Sinir ağı modelinin diğerlerine göre daha kararlı bir eğitim grafiği çizdiği görülmektedir. Ancak Adam optimizasyon algoritması ile oluşturulan modelde, 100 iterasyon sonunda %84 doğruluk oranı ile daha iyi bir performans elde edilmiştir.



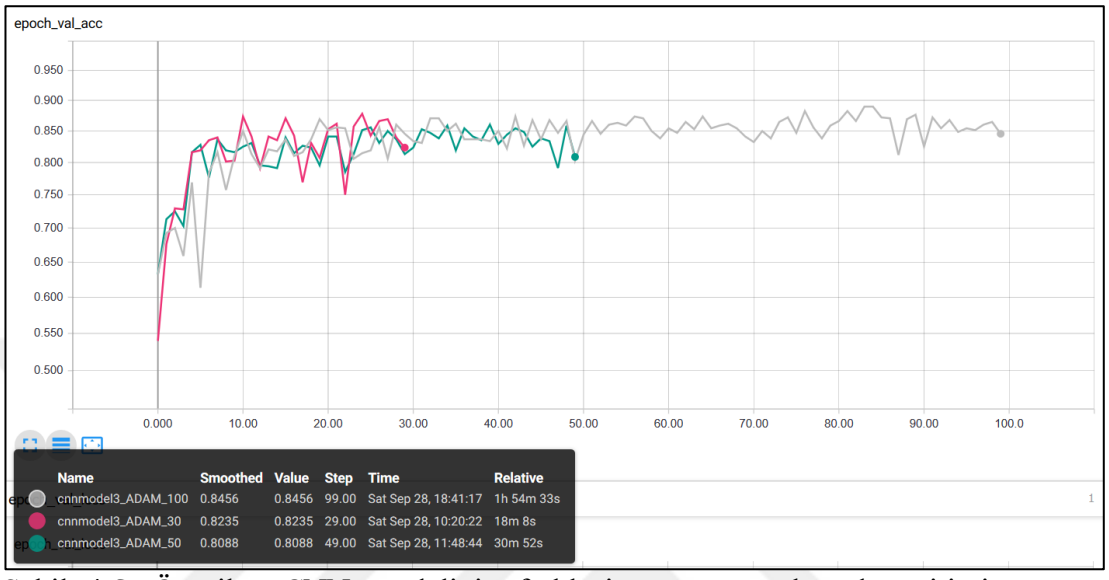
Şekil 4.4. Önerilen CNN modelinin farklı optimizasyon algoritmaları kullanılarak doğruluk oranlarının karşılaştırılması

Şekil 4.5'te önerilen CNN modeli üzerinde 100 iterasyon ile optimizasyon algoritmalarının hata oranı grafiksel olarak karşılaştırılmıştır. Elde edilen bulgular TensorBoard kullanılarak Şekil 4.5'te gösterilmiştir. Şekil 4.5'te Rmsprop optimizasyon algoritması ile oluşturulan Konvolüsyonel sinir ağı modelinin eğitim esnasında daha kararsız bir grafik çizdiği görülmektedir. 5.241 ile en yüksek hata oranı Rmsprop optimizasyon algoritması ile oluşturulan modelde tespit edilmiştir.



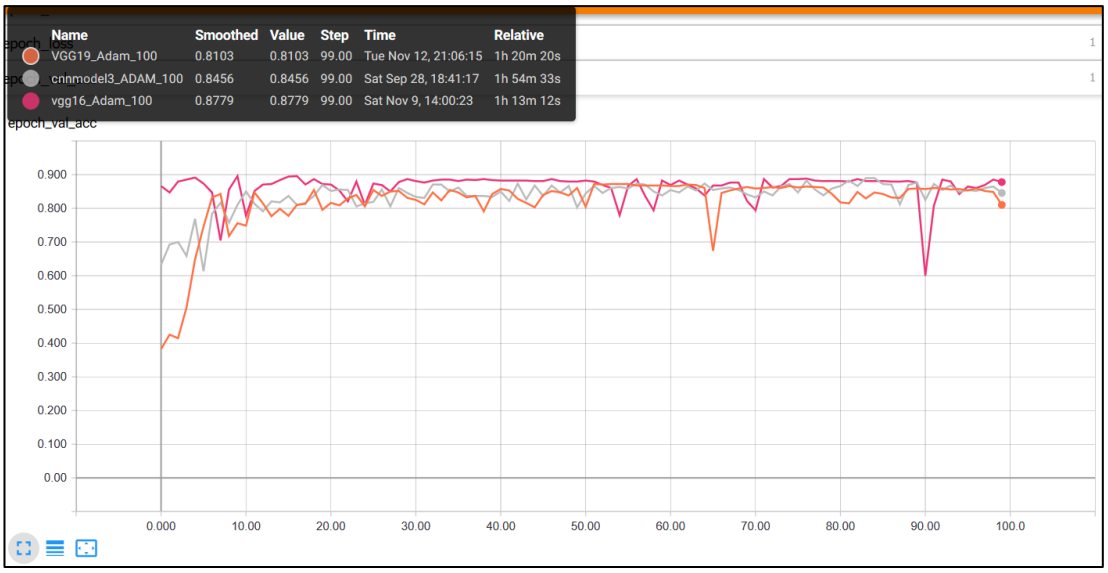
Şekil 4.5. Önerilen CNN modelinin farklı optimizasyon algoritmaları kullanılarak hata oranlarının karşılaştırılması

Şekil 4.6'da Önerilen CNN modeline ait eğitim sonucunda elde edilen doğruluk oranı grafik üzerinde verilmiştir. Verilen grafikte bu modele ait farklı iterasyon sayıları ile eğitilen modelin doğruluk oranları kıyaslanmaktadır. Burada 30 iterasyon ile 100 iterasyon arasında dengeli bir değer artışı olduğu Şekil 4.6'da görülmektedir.



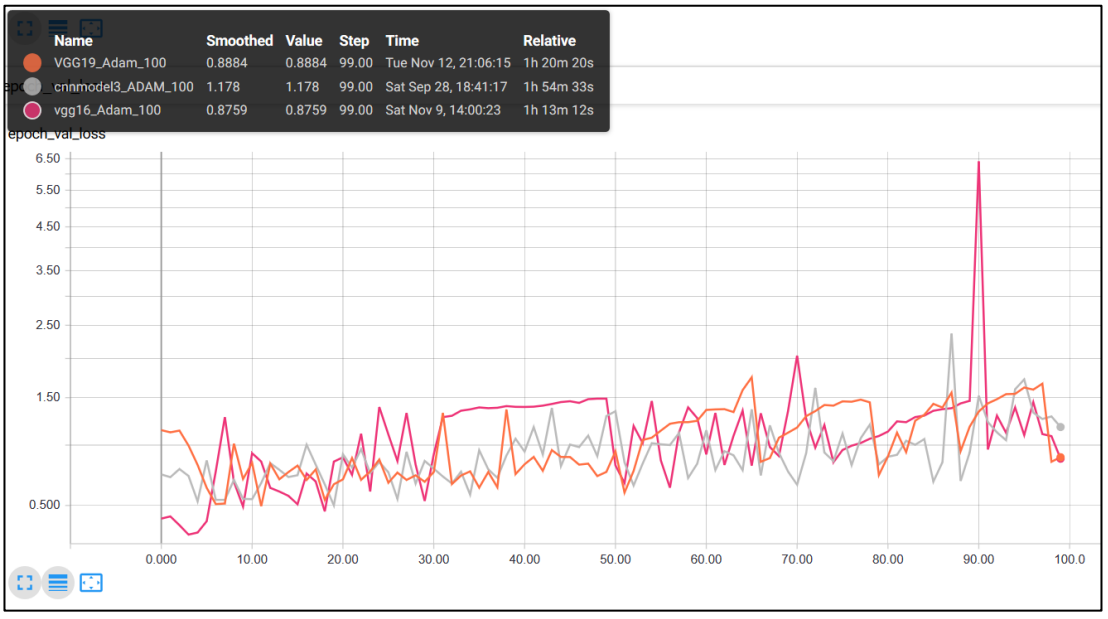
Şekil 4.6. Önerilen CNN modelinin farklı iterasyon sayılarında eğitimi sonucu doğruluk oranlarının karşılaştırılması

Şekil 4.7'de önceden eğitilmiş olan VGG16 ve VGG19 modelleri ile önerilen CNN modelinin doğruluk oranı karşılaştırılmıştır. Verilen grafikte modelleri 100 iterasyon üzerinden eğitilmiş olup tüm modellerde Adam optimizasyon algoritması kullanılmıştır. Elde edilen bulgular TensorBoard kullanılarak Şekil 4.7'de gösterilmiştir. Şekil 4.7'de transfer öğrenme tekniği ile kullanılan VGG16 gelişmiş Konvolüsyonel Sinir Ağı modelinin %87 ile VGG19 ve önerilen CNN modeline kıyasla daha yüksek doğruluk oranı elde ettiği görülmektedir.



Şekil 4.7. Tez çalışmasında kullanılan farklı modellerin doğruluk oranlarının karşılaştırılması

Şekil 4.8'de önceden eğitilmiş olan VGG16 ve VGG19 modelleri ile önerilen CNN modelinin hata oranı grafiksel olarak karşılaştırılmıştır. Şekil 4.8'de yer alan grafikte 0.87 hata oranı ile en düşük değer VGG16 gelişmiş Konvolüsyonel Sinir Ağı modelinde görülmektedir. VGGNet'e ait önceden eğitilmiş Konvolüsyonel sinir ağı modelinin diğer sinir ağı modeli VGG19 ile yakın değerlere sahip olduğu tespit edilmiştir.



Şekil 4.8. Tez çalışmasında kullanılan farklı modellerin hata oranlarının karşılaştırılması

5. SONUÇ VE ÖNERİLER

Bu tez çalışmasında zeytin bitkisine ait 3 farklı yaprak türü konvolüsyonel sinir ağı modeli ile sınıflandırıldı. Veri seti kontrollü koşullar altında elde edilen 3400 adet zeytin yaprak görüntüsünden oluşturuldu. Bu görüntülerin %80'i eğitim, %20'si test için ayrıştırıldı. Konvolüsyonel sinir ağı mimarisinin en iyi performans sonucunu verebilmesi için karşılaştırmalı deneysel çalışmalar yürütüldü. Bu deneysel çalışmalarda özellikle optimizasyon algoritmalarının ve iterasyon sayılarının konvolüsyonel sinir ağı modellerine, performans bakımından etkisi ölçüldü. Bu ölçüm hata matrisi, doğruluk, kesinlik, duyarlılık ve F1 skoru gibi performans metrikleri ile gerçekleştirildi. Sonuçlar çizelge ve grafikler üzerinde karşılaştırıldı. Buna göre; veri arttırım işlemi yapılmamış olan ve tez çalışmasında önerilen CNN modeli üzerinde 100 iterasyon sonucunda %85 en iyi doğruluk oranı Adam optimizasyon algoritması ile elde edildi. Ayrıca transfer öğrenme yönteminin sıfırdan eğitilen CNN modellerine göre başarısı gözlemlendi. Transfer öğrenimi yöntemi uygulanan modellerin 100 iterasyon sonucunda önerilen CNN modeline göre daha iyi performans yakaladığı görüldü. Burada en iyi performans sonucu %88 ile Adam optimizasyon algoritması kullanılan VGG16 gelişmiş konvolüsyonel sinir ağı modeli oldu.

Tez çalışmasının kısıtlamalarından bir tanesi veri seti boyutunun küçük olmasıdır. Bu durumu çözebilmek amacıyla Keras kütüphanesinde yer alan ImageDataGenarator sınıfı kullanılarak veri arttırım tekniği tez çalışmasında önerilen CNN modeline uygulandı. Model Adam optimizasyon algoritması ile 100 iterasyon üzerinden eğitimi gerçekleştirildi. Bu işlem sonucunda önerilen CNN modeli ile sınıflandırma başarısı olarak %94 doğruluk oranı elde edildi.

Çalışmanın son aşamasında kullanıcıya modelin sunulabilmesi için Flask Kütüphanesi yardımı ile web uygulaması tasarlandı. Uygulama esnasında kullanıcı zeytin bitkisine ait yaprak görüntüsünü fotoğraflayıp bu görüntüyü uygulamaya yükledikten sonra sonucu modele tahmin ettirebilmektedir.

Tez çalışmasında yer alan bir diğer kısıtlama ise veri setinde bulunan zeytin yaprak görüntüsü sınıf sayısının az olmasıdır. Veri seti sadece Denizli ilinde yetişen zeytin

türüne ait hastalık ve zararlıların bulunduğu yaprak görüntülerinden oluşturulmuştur. Ancak burada amaç; deneysel çalışmalar sonucunda elde edilen CNN model mimarisi ile zeytin yaprak görüntüleri üzerinde en iyi sınıflandırma başarısını yakalamaktır. Bu aşamadan sonra veri seti üzerinde daha detaylı bir çalışma yürütülerek farklı bölgelerde yetişen zeytin ağacı türlerine ait hastalık ve zararlılarının tanı ve teşhisini sağlamak mümkün olabilir. Ayrıca bu hastalık ve zararlıların tanı ve teşhisin sadece yaprak görüntüleri ile sınırlı kalmayıp meyve ve gövde görüntülerinden de gerçekleştirilebilir. Bu sayede ülkemiz tarım ekonomisi açısından önem arzeden zeytin ağacının diğer hastalık ve zararlılarının tanı ve teşhisi konusunda sahada çalışan çiftçilere ya da ziraat mühendislerine kolaylık sağlanabilir.

KAYNAKLAR

Anonim (2015). https://www.internationaloliveoil.org/wp-content/uploads/2019/11/INTERNATIONAL-OLIVE-OIL-PRODUCTION-COSTS-STUDY-.pdf (Son erişim tarihi: 04.01.2020)

Anonim (2016a). https://cp4space.wordpress.com/2016/02/06/deep-learning-with-the-analytical-engine/ (Son erişim tarihi: 03.11.2019)

Anonim (2016b). Zeytin Hastalık ve Zararlıları ile Mücadele. https://www.tarimorman.gov.tr/GKGM/Belgeler/Bitki%20Sa%C4%9Fl%C4%B1%C4%9F%C4%B1%20Hizmetleri/hastalik_zararlilari_ile_m%C3%BCcadele_dokumanlari/zeytin.pdf (Son erişim tarihi: 16.11.2019)

Anonim (2017a). https://www.tarimorman.gov.tr/TAGEM/Belgeler/Entegre/zeytin%20entegre.pdf (Son erişim tarihi: 12.10.2019)

Anonim (2017b). http://cs231n.github.io/convolutional-networks/#fc (Son erişim tarihi: 10.11.2019)

Anonim (2017c). https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c (Son erişim tarihi: 10.11.2019)

Anonim (2018a). https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection

Anonim (2018b). https://github.com/datafloyd/datafloyd_examples/blob/master/Conf_Matrix/Conf_Matrix.ipynb

Anonim (2019a). https://www.guru99.com/deep-learning-tutorial.html (Son erişim tarihi: 25.10.2019)

Anonim (2019b). https://github.com/krishnaik06/Deployment-Deep-Learning-Model (Son erişim tarihi: 19.11.2019)

Bakshi, A., (2019). Deep Learnin Tutorial: Artifical Intellegence Using Deep Learning. https://www.edureka.co/blog/deep-learning-tutorial#disqus_thread (Son erişim tarihi: 27.10.2019)

Brownlee, J., (2017a). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/(Son erişim tarihi: 10.11.2019)

Brownlee, J., (2017b). A Gentle Introduction to Transfer Learning for Deep Learning. https://machinelearningmastery.com/transfer-learning-for-deep-learning/ (Son erişim tarihi: 11.11.2019)

Brownlee, J., (2017c). How to Use the Keras Functional API for Deep Learning. https://machinelearningmastery.com/keras-functional-api-deep-learning/ (Son erişim tarihi: 11.11.2019)

Brownlee, J., (2018). A Gentle Introduction to Dropout for Regularizing Deep Neural Networks. https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/ (Son erişim tarihi: 09.11.2019)

Brownlee, J., (2019a). What is Deep Learning? https://machinelearningmastery.com/what-is-deep-learning/(Son erişim tarihi: 26.10.2019)

Brownlee, J., (2019b). A Gentle Introduction to Cross-Entropy for Machine Learning. https://machinelearningmastery.com/cross-entropy-for-machine-learning/ (Son erişim tarihi: 17.11.2019)

Brownlee, J., (2019c). How to Configure Image Data Augmentation in Keras. https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/ (Son erişim tarihi: 19.11.2019)

Çarkacı, N., (2017). Derin öğrenme uygulamalarında temel kavramlar: perceptron, skor fonksiyonu ve hata hesaplaması (loss function). https://www.linkedin.com/pulse/derin-%C3%B6%C4%9Frenme-uygulamalar%C4%B1nda-temel-kavramlar-skor-ve-%C3%A7arkac%C4%B1

Çarkacı, N., (2018). Derin öğrenme uygulamalarında en sık kullanılan hiper parametreler. https://medium.com/deep-learning-turkiye/derin-ogrenme-uygulamalarinda-en-sik-kullanilan-hiper-parametreler-ece8e9125c4 (Son erişim tarihi: 03.11.2019)

Deng, L. & Yu, D. (2014). Deep Learning Methods and Applications. *Three Classes of Deep Learning Networks.* (pp.214-226)

Ferentinos, K.P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*,145, 311-318. https://doi.org/10.1016/j.compag.2018.01.009

Francis, M. & Deisy, C. (2019). *Disease Detection and Classification in Agricultural Plants Using Convolutional Neural Networks –A Visual Understanding.* 6. International Conference on Signal Processing and Integrated Networks (SPIN).

Gylberth, R., (2018). An Introduction to AdaGrad. https://medium.com/konvergen/an-introduction-to-adagrad-f130ae871827 (Son erişim tarihi: 17.11.2019)

Geetharamani, G. & Arun Pandian, J. (2019). Identification of plant leaf disease using a nine layer deep convolutional neural network. *Computers and*

Electrical Engineering, 76, 323-338. https://doi.org/10.1016/j.compeleceng.2019.04.011

Genshenga, H., Haoyua, W., Yana, Z. & Mingzhub, W. (2019). A low shot learning method for tea leaf's disease identification. *Computers and Electronics in Agriculture*,163, 104852.

Gümrük ve Ticaret Bakanlığı, Kooperatifçilik Genel Müdürlüğü, (2018). 2017 Yılı Zeytin ve Zeytin Yağı Raporu. (Son erişim tarihi: 06.10.2019). http://koop.gtb.gov.tr/data/5ad06f17ddee7dd8b423eb2e/2017%20Zeytinya%C4%9F%C4%B1%20Raporu.pdf

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. & Adam, H. (2017). Mobile Nets: Efficient convolutional neural networks for mobile visions applications. arXiv:1704.04861v1 [cs.CV]

Ioffe, S. & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167v3[cs.LG]

Kaya, A., Keceli, A.S., Çatal, Ç., Yalica, H.Y., Temucin, T. & Tekinerdogan, B. (2019). Analysis of transfer learning for deep neural network based plant classification models. *Computers and Electronics in Agriculture*, 158, 20-29. https://doi.org/10.1016/j.compag.2019.01.041

LeCun, Y., Haffner, P., Bottou, L. & Bengio, Y. (1999). Object recognition with gradient-based learning. *Shape, Contour and Grouping in Computer Vision.* (pp. 319-345)

LeCun, Y., Haffner, P., Bottou, L. & Bengio, Y. (1998). Gradient Based Learning Applied to Document Recognition.

Lin, J., (2017). Keras Models: Sequential vs. Functional. https://jovianlin.io/keras-models-sequential-vs-functional/ (Son erişim tarihi: 18.11.2019)

Liu, D., (2017). A Practical Guide to ReLU. https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7 (Son erişim tarihi: 03.11.2019)

Lu, Y., Yi, S., Zeng, N., Liu, Y. & Zhang, Y. (2017). Identification of rice diseases using deep Convolutional neural network. *Neurocomputing, 267,* 378-384. http://dx.doi.org/10.1016/j.neucom.2017.06.023

Maa, J., Dua, K., Zhenga, F., Zhangb, L., Gongc, Z. & Suna, Z. (2018). A recognition method for cucumber diseases using leaf symptom images based on deep convolutional neural network. *Computers and Electronics in Agriculture*, 154, 18-24. https://doi.org/10.1016/j.compag.2018.08.048

Mahendru, K., (2019). A Detailed Guide to 7 Loss Functions for Machine Learning Algorithms with Python Code. https://www.analyticsvidhya.com/blog/2019/08/detailed-guide-7-loss-functions-machine-learning-python-code/ (Son erişim tarihi: 10.11.2019)

Marcelino, P., (2018). Transfer learning from pre-trained models. https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751 (Son erişim tarihi: 11.11.2019).

Mohanty, P.S., Hughes, D. & Salathe, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. doi: 10.3389/fpls.2016.01419

Mureşan, H. & Oltean, M. (2018). Fruit recognition from images using deep learning. doi: 10.2478/ausi-2018-0002

Ng, A., (2015). https://www.youtube.com/watch?v=O0VN0pGgBZM (Son erişim tarihi: 26.10.2019)

Ng, A., (2017). https://www.youtube.com/watch?v=nUUqwaxLnWs (Son erişim tarihi: 10.11.2019)

Nielsen, M. (2018). Neural Networks and Deep Learning. *Deep Learning.* (pp. 167-209)

Oppermann, A., (2019). Optimization Algorithms in Deep Learning. https://towardsdatascience.com/optimization-algorithms-in-deep-learning-191bfc2737a4 (Son erişim tarihi: 10.11.2019)

Parmar, R., (2018). Common Loss functions in machine learning. https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23 (Son erişim tarihi: 10.11.2019)

Picona, A., Alvarez-Gilaa, A., Seitzc, M., Ortiz-Barredob, A., Echazarraa, J. & Johannes, A. (2019). Deep convolutional neural networks for mobile capture device-based crop disease classification in the wild. *Computers and Electronics in Agriculture*, 161, 280-290. https://doi.org/10.1016/j.compag.2018.04.002

Rangarajani A.K., Purushothaman, R. & Ramesh, A. (2018). Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia Computer Science*, 133, 1040-1047.

Rao, A., (2019). Convolutional Neural Network Tutorial (CNN) – Developing An Image Classifier In Python Using TensorFlow. https://www.edureka.co/blog/convolutional-neural-network/#z7 (Son erişim tarihi: 03.11.2019)

Rosebrock, A., (2019). Keras ImageDataGenerator and Data Augmentation. https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/ (Son erişim tarihi: 19.11.2019)

Ser, G. & Bati, C.T. (2019). Derin Sinir Ağları ile En İyi Modelin Belirlenmesi: Mantar Verileri Üzerine Keras Uygulaması. *Yüzüncü Yıl Üniversitesi Tarım Bilimleri Dergisi, Cilt 29, Sayı 3* doi: 10.29133/yyutbd.505086.

Sherstinsky, A. (2018). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network. arXiv:1808.03314v4 [cs.LG]

Shijie, J., Peiyi, J., Siping, H. & Haibo L. (2017). *Automatic Detection of Tomato Diseases and Pests Based on Leaf Images*. 2017 Chinese Automation Congress (CAC).20-22 October, doi: 10.1109/CAC.2017.8243388.

Simonyan, K. & Zisserman, A. (2015). Very deep Convolutional networks for large-scale recognation. https://arxiv.org/abs/1409.1556

Suh, H.K., Ijsselmuiden, J., Hofstee, J.W. & Henten, E.J.V. (2018). Transfer learning fort he classification of sugar beet and volunteer potato under field conditions. *BIOSYSTEMS ENGINEERING*, 174. 50-65. www.elsevier.com/locate/issn/15375110

Sumit, S., (2018). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (Son erişim tarihi: 03.11.2019)

Srinivasan, A.V. (2019). Stochastic Gradient Descent Clearly Explained. https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31 (Son erişim tarihi: 17.11.2019)

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research.

Şeker, A., Diri, B. & Balık, H.H. (2017). Derin Öğrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme. *Gazi Mühendislik Bilimleri Dergisi, 3(3),* 47-64.

Şimşek, H.K., 2018. Sınıflandırma Modellerinde Başarı Kriterleri. https://medium.com/data-science-tr/s%C4%B1n%C4%B1fland%C4%B1rma-modellerinde-ba%C5%9Far%C4%B1-kriterleri-2d86488799c6 (Son erişim tarihi: 20.11.2019)

Toda, Y. & Okura, F. (2019). How Convolutional Neural Networks Diagnose Plant Disease. *Plant Phenomics,* https://doi.org/10.34133/2019/9237136.

Torrey, L. & Shavlik, J. (2009). Transfer Learning. *Handbook of Research on Machine Learning Applications.*

Traorea, B.B., Foguema, B.K. & Tangarab, F. (2018). Deep Convolutional neural network for image recognation. https://doi.org/10.1016/j.ecoinf.2018.10.002

Tsang, S., (2018). Review: VGGNet — 1st Runner-Up (Image Classification), Winner (Localization) in ILSVRC 2014. https://medium.com/coinmonks/paper-review-of-vggnet-1st-runner-up-of-

ilsvlc-2014-image-classification-d02355543a11 (Son erişim tarihi: 11.11.2019)

Tunç, C. & Onoğur, E., (2013). Güncel Verilerle Zeytin Halkalı Leke Hastalığı. ANADOLU, J. of AARI 23 (2), 44 - 59

Wallelign, S., Polceanu, M. & Buche C. (2018). *Soybean Plant Disease Identification Using Convolutional Neural Network.* The Thirty-First International Flairs Conference Artificial Intelligence Research Society Conference (FLAIRS-31)

Yu, B. & Malan, J., D. 2018. CS50's Web Programming with Python and JavaScript. https://www.youtube.com/watch?v=j5wysXqaIV8&list=WL&index=26&t=3160s (Son erişim tarihi: 19.11.2019)

Zhang, X., Qiao, Y., Meng, F., Fan, C. & Zhang, M. (2018). Identification of Maize Leaf Diseases Using Improved Deep Convolutional Neural Networks. doi: 10.1109/ACCESS.2018.2844405

Zhang, A., Lipton, Z.C., Li, M. & Smola A.J. (2019). Dive into Deep Learning. *Convolutional Neural Network.* (pp. 215-244)

ÖZGEÇMİŞ

Adı Soyadı : Neşe UYSAL

Doğum Yeri ve Yılı : Denizli, 1987

Medeni Hali : Bekar

Yabancı Dili : İngilizce

E-posta : neseuysal87@gmail.com

Eğitim Durumu

Lise : Denizli Şehit Öğretmen Yusuf Batur EML, 2004

Lisans : PAÜ, Teknik Eğitim Fakültesi, 2010

Mesleki Deneyim

Muş, Rekabet Kurumu Mesleki ve Teknik AL	2011-2015
Burdur, Karamanlı Mesleki ve Teknik AL	2015-2018
Burdur, Hakan Sevim Fen Lisesi	2018.....(halen)