ANKARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

KÖTÜ AMAÇLI YAZILIMLARIN DERİN ÖĞRENME YÖNTEMİ İLE TESPİT EDİLMESİ

ÜMİT KÖSE

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ANKARA
2022

ÖZET

Yüksek Lisans Tezi

KÖTÜ AMAÇLI YAZILIMLARIN DERİN ÖĞRENME YÖNTEMİ İLE TESPİT EDİLMESİ

Ümit KÖSE

Ankara Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Refik SAMET

Teknolojik cihazlardaki verilerin ele geçirilmesi, sistemlere zarar verilmesi amacıyla siber saldırılar yapılmakta olup, kurumlara ve bireylere yüksek maliyette zararlar verilmektedir. Siber saldırı türlerinde en çok karşılan tür olan zararlı yazılımlar, veri güvenliği ve mahremiyeti ile ilgili endişeler oluşturmaktadır. Kötü amaçlı yazılımlar her gün farklı türleri ile piyasaya sunulmakta olup, zararlı yazılımların tespiti için yüksek maliyette harcamalar yapılmaktadır. Farklı türlerde olan zararlı yazılımların tespitleri için imza, dinamik ve spesifikasyon tabanlı yöntemler kullanılmaktadır. İmza ve dinamik tabanlı tespit yöntemlerinin zararlı yazılım tespitindeki eksikliklerinden dolayı gelişen teknolojik imkanlardan yararlanılarak derin öğrenme yöntemini içeren mimari model önerilmiştir. Geliştirilen modelde derin öğrenme algoritmaları kullanılarak, makine öğrenmesi algoritmaları ve diğer çalışmalarla kıyaslama yapılarak güçlü ve zayıf yönleri yapılan tez çalışmasında anlatılmıştır. Önerilen derin öğrenme modelinde %99.02 başarı oranında kötü amaçlı yazılım tespiti yapılmıştır. İlgili çalışma statik ve dinamik analiz yöntemlerinin yeni zararlı yazılımlara karşı eksikliklerini ortaya koyarak, yüksek veri seti üzerinde Windows işletim sistemlerinde mevcut çalışmalara kıyasla yüksek oranda zararlı yazılımları tespit etmiştir.

Mart 2022, 58 sayfa

Anahtar Kelimeler: Zararlı Yazılım, Siber Güvenlik, Siber Saldırılar, Saldırı Tespit Sistemleri, Makine Öğrenmesi, Derin Öğrenme, Zararlı Yazılım Analizi, Zararlı Yazılım Tespiti

ABSTRACT

Master Thesis

DETECTION MALICIOUS SOFTWARE WITH DEEP LEARNING METHOD

Ümit KÖSE

Ankara University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisors: Prof. Dr. Refik SAMET

Cyber attacks are carried out in order to seize data on technological devices and damage systems, causing high-cost damages to institutions and individuals. Malware, which is the most common type of cyber attack, raises concerns about data security and privacy. Malware is released to the market with different types every day, spend high costs for the detection of malicious software. Signature, dynamic and specification based methods are used for the detection of different types of malicious software. A architectural model including the deep learning method has been proposed to take advantage of the developing technological possibilities due to the deficiencies in malware detection of signature and dynamic based detection methods. In the developed model, deep learning algorithms are used and the strengths and weaknesses are explained in the thesis study by making comparisons with machine learning algorithms and other studies. In the proposed deep learning model, malware detection was achieved with a success rate of 99.02%. The related study revealed the shortcomings of static and dynamic analysis methods against new malware, and detected a high rate of malware compared to existing studies on Windows operating systems on a high data set.

March 2022, 58 pages

Key Words: Malware, Cyber Security , Cyber Attacks, Intrusion Detection Systems, Machine Learning, Deep Learning, Malware Analysis, Malware Detection

TEŞEKKÜR

Kötü Amaçlı Yazılımların Derin Öğrenme Yöntemi ile tespit edilmesi isimli bu çalışma, Ankara Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda yüksek lisans tezi olarak hazırlanmıştır.

Bu tez çalışmasının başından sonuna kadar her aşamasında, bilgisi, yardımları ve destekleriyle çalışmamın tamamlanmasını sağlayan, kendisinden sürekli yararlandığım danışman hocam Sayın Prof. Dr. Refik SAMET 'e teşekkürlerimi sunarım.

Eğitim ve öğrenim hayatımda bugüne kadar en büyük destekçim olan başta babam Orhan KÖSE ve annem Ayşe KÖSE 'ye, ablam Ülkü UYAR abim Musa UYAR ve sevgili yeğenlerim Ömer Barış ve Ayşe Nisa UYAR 'a teşekkürlerimi sunarım. Akademik olarak bu yola çıkmamda bana olan inançlarını her zaman gösteren arkadaşlarıma, birlikte çalıştığım meslektaşlarıma teşekkürlerimi sunmayı borç bilirim. Yapılan bu çalışmanın farklı araştırmalara öncü olması ve katkıda bulunabilmesini dilerim.

Ümit KÖSE
Ankara, Mart 2022

İÇİNDEKİLER

KISALTMALAR DİZİNİ

ANN	Yapay Sinir Ağı (Artificial Neural Network)
API	Uygulama Programlama Arayüzü (Application Programming Interface)
APK	Android Uygulama Paketi (Android Application Package)
CNN	Konvülüsyon Sinir Ağı (Convolutional Neural Network)
CPU	Merkezi İşlem Birimi (Central Process Unit)
DOS	Disk İşletim Sistemi (Disk Operationg System)
DDOS	Dağıtık Hizmet Engelleme (Denial of Service Attack)
DLL	Dinamik Link Kütüphanesi (Dynamic Link Library)
DNN	Derin Sinir Ağı (Deep Neural Network)
DNS	Alan Adı Sistemi (Domain Name System)
DT	Karar Ağacı (Decision Tree)
ELF	Yürütülebilir ve Bağlanabilir Biçim (Executable and Linkable Format)
ESA	Evrişimsel Sinir Ağı
GPU	Grafik İşlem Birimi (Graphics Processing Unit)
HTML	Hiper Metin İşaretleme Dili (Hypertext Markup Language)
IDS	Saldırı Tespit Sistemi (Instrusion Detection System)
JPEG	Birleşik Fotoğraf Uzmanları Grubu (Joint Photographic Experts Group)
KNN	K-En Yakın Komşu (K-Neares Neighbors)
LSTM	Uzun Kısa Süreli Bellek (Long Short Term Memory)
MP3	Film Uzmanlar Grubu Ses Katmanı 3 (MPEG-1 Audio Layer III)
NB	Naive Bayes
NIPS	Ağ Tabanlı Saldırı Tespit Sistemi
PDF	Taşınabilir Belge Biçimi (Portable Document Format)
PE	Taşınabilir Yürütülebilir Dosya (Portable Executable)
PEF	Tercih Edilen Yürütülebilir Format (Preferred Executable Format)
PNG	Taşınabilir Ağ Grafiği (Portable Network Graphics)
ROC	Alıcı İşletim Karakteristiği (Receiver Operating Characteristic)
RNN	Tekrarlayan Sinir Ağı (Recurrent Neural Network)
SVM	Destek Vektör Makineleri (Support Vector Machine)
SQL	Yapılandırılmış Sorgu Dili (Structured Query Language)
WIPS	Kablosuz Saldırı Önleme Sistemleri

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

İnternet, akıllı cihaz ve bilgisayar kullanımı gündelik yaşamdan kamu kurumlarına, sağlık alanından eğitime, bilimden tarıma kadar geniş bir alana yayılmıştır. Teknolojinin yaygın kullanımı ile birlikte zararlı yazılımlar ve sistem açıklıklarından faydalanarak kullanıcıların kişisel bilgilerinin çalınması, dijital varlıkların manipüle edilmesi, değiştirilmesi ya da yok edilmesi ile karşılaşılabilmektedir (Hatipoğlu ve Tunacan 2021). Yapılan bu eylemler siber saldırı olarak adlandırılmaktadır.

Siber saldırılar, kullanıcılara zarar vermek için farklı türde saldırı yöntemleri kullanmaktadır. Siber saldırılarda en çok kullanılan yöntem olan zararlı yazılım (Aslan 2021), sahibinin rızası olmadan bir bilgisayar sistemine sızmak veya zarar vermek için tasarlanmış herhangi bir kod parçası olarak tanımlanmıştır (Ray ve Asoke 2016). Kötü amaçlı yazılım, bilgisayarlara ve bilgisayar kullanıcılarına zarar vererek, sistemdeki dosyaları bozarak veya kullanıcıları rahatsız edici faaliyetlerde bulunmaya yönelik geliştirilmiş yazılımlardır (Sanjaa ve Chuluun 2013). Kötü amaçlı yazılım kelimesi 1990 yılında Yisrail Radei tarafından Malicious Software (Malware) olarak isimlendirilmiştir (Sukhbaatar 2007).

Teknolojinin günden güne gelişmesi ile birlikte kullanıcıların ve kurumların sahip olduğu veri sayısında büyük bir artış olmuştur. Artan veri sayısı ile birlikte kullanıcıların ve kurumların verisini ele geçirmek için çıkarılan zararlı yazılımların sayısı artmıştır (Anonymous 2022). AvAtlas tarafından sunulan kötü amaçlı yazılım verilerine göre 2022 yılında dünyada 702953430 adet zararlı yazılım bulunmaktadır (Anonymous 2022). Son 12 ayda 147451035 tane kötü amaçlı yazılım kullanıcıların verilerini ele geçirmek, zarar vermek amacıyla geliştirilmiştir (Anonymous 2022). Bu tez çalışmasında kötü amaçlı yazılımların tercih edilmesindeki en önemli sebep zararlı yazılımların hızlı bir miktarda artışı olmuştur.

Siber saldırılar zararlı yazılımların dışında kimlik avı saldırıları, ağ üzerinde yük oluşturabilecek ve sistem açıklarından yararlanılabilecek kablosuz ağ saldırıları (Man in the Middle), dağıtık hizmet engelleme (DDOS) ve disk işletim sistemi (DOS) saldırıları,

veri tabanları için kullanılan yapılandırılmış sorgu dilinin güvenlik açıklarından faydalanma (SQL Enjeksiyon), kullanıcıların sistemlerinde kripto para kazanmak için sistem üzerindeki kaynakları tüketen saldırılar (Cryptojacking) veya kullanıcıların şifrelerine elde etmeye yönelik saldırılar olarak karşımıza çıkabilmektedir (Eskandari ve Shayan 2018).

Teknolojinin hızlı ilerlemesi ve kullanımının yaygınlaşmasıyla birlikte artan siber saldırı ve zararlı yazılımlardan korunmak için belirli yöntem ve uygulamalar çıkmıştır. Bu yöntem ve uygulamalar siber güvenlik adı altında toplanmıştır. Teknolojik cihazı korumak için güvenlik duvarı, anti virüs uygulamaları, saldırı tespit ve koruma sistemleri (IDPS) gibi siber güvenlik yöntemleri kullanılmaktadır (Ozkan vd 2021). Siber güvenlikte zararlı yazılım tespiti yapılırken kullanılan yöntemlerden biri de sistemin zararlı yazılıma maruz kalıp kalmayacağıyla karar verilerek sistem üzerindeki davranışının izlenmesidir.

Siber güvenlik uygulamaları temelinde imza, dinamik ve spesifikasyon tabanlı yöntemler kullanmaktadır. Çoğunlukla anti virüs uygulamalarında imza tabanlı yöntem kullanılmakta olup, imza tabanlı yöntemin yeni zararlı yazılımları tespit etmesindeki yetersizliklerinden dolayı zararlı yazılımların tespitinde çok başarılı bir yöntem olmamaktadır (Aslan ve Samet 2020). Yeni çıkan zararlı yazılımların tespitinde daha çok kullanılan dinamik tabanlı yöntem sistem davranışlarını inceleyerek zararlı yazılım tespiti yapmaktadır. Bu yöntem kullanılarak yeni çıkan zararlı yazılımların tespitinde zaman alabileceği için, sistem üzerinde zararlı yazılımın bırakmış olduğu hasar artabilecektir (Yalçinkaya 2020).

Bu tez çalışmasında zararlı yazılımların tespit yöntemleri incelenerek, derin öğrenme algoritmalarını içeren model önerilmiştir. Yapılan araştırmalar sonucunda zararlı yazılım tespitindeki yetersizlikler tespit edilmiş, detaylı bir literatür araştırması yapılarak diğer çalışmalarda kullanılan makine öğrenmesi ve derin öğrenme algoritmaları incelenmiştir. Yapılan literatür çalışmasında, çalışmaların detaylarıyla birlikte çalışmalardaki eksik yöntemler ele alınarak tez çalışmasındaki farklar anlatılmıştır. Karşılaştırılmalı olarak

literatür çalışması sunulmuş ve literatür çalışmasındaki algoritmalardan farklı olarak zararlı yazılım tespiti için yeni bir mimari model ortaya atılmıştır.

Derin öğrenme yöntemi için modelde Evrişimli Sinir Ağı (CNN) ve yapay sinir ağından (ANN) faydalanılmıştır. Hazır olarak kullanılan veri setinde modelin hiper parametreleri değiştirilecek şekilde çok fazla eğitimden geçirilmiştir. Modelin zararlı yazılım tespit oranı literatürdeki çalışmalarda kullanılan makine öğrenmesi algoritmalarından ve yüksek sayıda kötü amaçlı yazılım içeren veri setlerindeki derin öğrenme modellerinden daha başarılı bir oranda tespiti sağlanmıştır. Kullanılan veri seti makine öğrenmesi algoritmalarıyla da eğitilerek başarı oranı tespit edilmiştir. Geliştirilen model seçilen en uygun hiper parametreler ile birlikte %99.02 oranında tahminde bulunmaktadır. Modelin başarı oranı ile birlikte karmaşıklık matrisi ve alıcı işletim karakteristiği (ROC) eğrisi başarı durumlar paylaşılmıştır. Geliştirilen modelde kullanılan yöntem ve parametre bilgileri ilerleyen bölümlerde anlatılmıştır.

Yapılan tez çalışmasının, ikinci bölümünde kötü amaçlı yazılım ve derin öğrenme hakkında temel kavramlar ve bilgiler paylaşılmıştır. Bu bölümde kötü amaçlı yazılımın tespit ve analiz yöntemleriyle birlikte derin öğrenme algoritmaları ve kullanılan matematiksel fonksiyonlardan bahsedilmiştir. Üçüncü bölümde zararlı yazılımlar ile ilgili yapılan diğer çalışmalar detaylı bir şekilde incelenerek ve tez çalışmasının diğer çalışmalarla karşılaştırılması yapılarak literatür araştırması başlığında paylaşılmıştır. Dördüncü bölümde statik ve dinamik olarak zararlı yazılım incelemesi yapılmıştır. Yapılan inceleme sonucunda neden derin öğrenme kullanılarak zararlı yazılım tespit edilmesine ihtiyaç duyulduğu ve bunun sonucunda önerilen modelin akış şeması ve mimarisi paylaşılmıştır. Önerilen modelde kullanılan veri seti, uygulama platformu ve uygulama yöntemleri beşinci bölümde Uygulama başlığı altında anlatılmıştır. Son bölümde uygulanan yöntemle ilgili sonuçlar ve sonuçlar üzerine değerlendirmeler, yapılan çalışmanın katkıları ve gelecekte yapılacak olan çalışmalar hakkında bilgiler verilmiştir.

2. KÖTÜ AMAÇLI YAZILIM VE DERİN ÖĞRENME TEMEL KAVRAMLARI

Yapılan tez çalışmasının bu bölümünde kötü amaçlı yazılımların ne olduğu, türleri, günümüz teknolojisinde zararlı yazılımların analiz ve tespit yöntemlerinde hangi çalışmaların kullanıldığı, analiz ve tespit yöntemlerindeki alt başlıkların açıklamaları ve karşılaştırılmaları ile birlikte kötü amaçlı yazılımların tespiti için kullanılan derin öğrenme teknolojisinin kullanım amacı ve mimari model olarak kullanılan ANN ve CNN anlatılmıştır.

2.1 Kötü Amaçlı Yazılım

Kötü amaçlı yazılım kullanıcıların verilerini ele geçirmek, sistemi manipüle etmek için kullanılmakta olan yazılımlardır. Bu yazılımlar kullanıcılarının karşısına farklı bir dosya ya da farklı bir zararlı yazılım formatında çıkabilmektedir. Zararlı yazılımlar kullanıcıların kişisel bilgilerini çalabilir, dinleme, izleme, loglama faaliyetlerinde bulunabilir, kullanılan cihazdaki işletim sistemlerine zarar verebilir ya da dağıtık hizmet engelleme (DDOS) saldırılarında, spam e-posta gönderiminde kullanılabilir. Kullanıcılara internet üzerinden, flash belleklerden, işletim sistemlerinin sisteme takılan cihazın durumuna göre ne yapacağını belirlendiği autorun özelliğinden, network üzerindeki güvenlik zafiyetinden, sosyal mühendislikle ya da mobil sistemler üzerinden bulaşabilmektedir.

Kötü amaçlı yazılımların işletim sistemleri üzerinde çalışması için işletim sisteminin çalıştırabileceği formatta uygulamaların olması gerekmektedir. Android işletim sisteminde Android uygulama paketi (APK), Linux işletim sistemlerinde yürütülebilir ve bağlanabilir biçim (ELF), MacOS işletim sisteminde tercih edilen yürütülebilir format (PEF) ve Windows işletim sistemlerinde ise taşınabilir yürütülebilir dosya (PE) formatında olduğu taktirde çalışmaktadır.

PE formatında bulunan uygulamalar Windows işletim sisteminde uygulamanın çalışması için içerisinde bilgiler içeren veri yapısıdır. 32 ve 64 bit Windows sistemlerinde çalışan bu dosya formatı Linux işletim sistemlerinde kullanılan ELF formatı gibi belirli

bölümlerden oluşmaktadır. Windows PE dosya formatındaki bölümler **çizelge 2.1**'de paylaşılmıştır (Tokmak ve Küçüksille 2019).

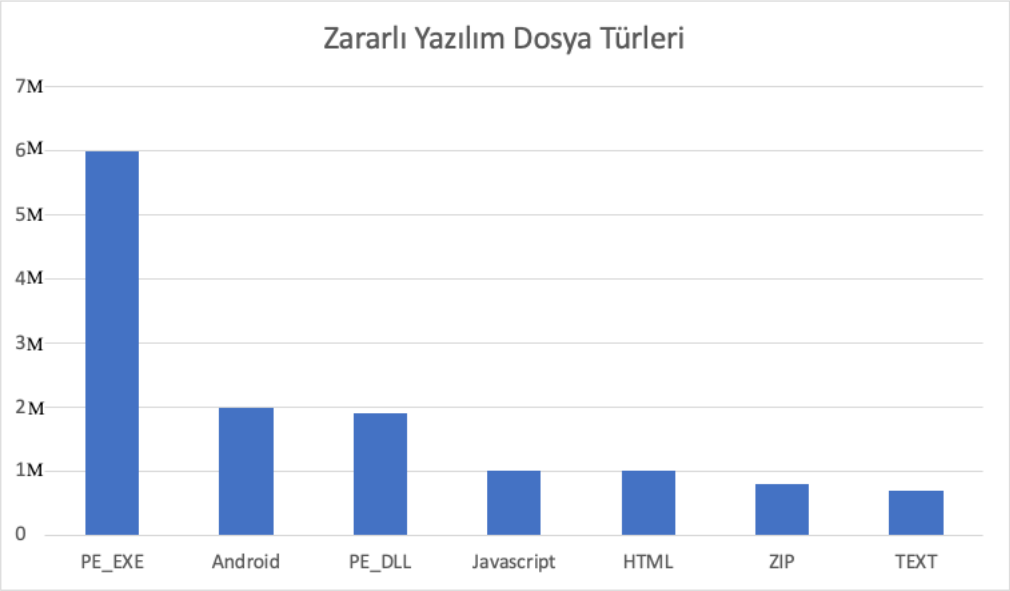
Çizelge 2.1 Windows PE Dosya İçeriği

MS-DOS Başlığı
MS-DOS Mesaj Programı
PE Başlığı
Bölüm Başlığı
Bölüm 1
Bölüm 2
…….
Bölüm n

Disk Operation System (DOS) Başlığı PE formatında uyumluluğu sağlamak için kullanılır. PE başlığı DOS başlığından sonra gelmektedir. PE başlığı uygulamanın kimlik imzasını doğrulamak için kullanılmaktadır. Uygulamanın bilgilerini içermektedir. Opsiyonel başlık ismi opsiyonel olmasına rağmen zorunludur, bütün taşınabilir dosyalar için aynı yapıda olmalıdır. Veri dizinleri uygulamanın önemli bilgilerini depolamaktadır. Bölüm başlıkları uygulamanın erişim hakları, bir sonraki bölümde bağlanmış olduğu yer bilgisi veya bellek boyutu gibi bilgileri içermektedir. Taşınabilir yürütülebilir dosya bölümleri ise uygulamanın çalıştırılabilir kodunu, başlangıç verilerini, Windows uygulama programlama arayüzünün (API) Dinamik link kütüphanesi (DLL) bilgilerini ve hata ayıklama bilgilerini içerir.

2.1.1 Kötü amaçlı yazılım türleri

İşletim sistemlerinin çok fazla sayıda desteklediği formatta dosya türü bulunmaktadır. Resim için birleşik fotoğraf uzmanları grubu (JPEG), taşınabilir ağ grafiği (PNG) vb. Müzik için film uzmanlar grubu ses katmanı 3 (MP3) ve işletim sistemine göre farklılık gösterebilmekle birlikte ZIP, taşınabilir belge biçimi (PDF), EXE, SH vb. çok fazla sayıda dosya bulunmaktadır (Köse ve Samet 2021).



Şekil 2.1 Kötü amaçlı yazılım dosya türleri

Bu dosyalar her ne kadar kendi amaçları doğrultusunda kullanıldığı düşünülse de içerisinde birçok farklı kötü amaçlı yazılım içerebilmektedirler. Zararlı yazılımların dosya türlerinden en çok karşılaşılanları **şekil 2.1**'de paylaşılmıştır (Anonymous 2022). Son 7 günlük karşılaştırılan zararlı yazılım dosya türleri içerisinde en çok Windows işletim sistemlerinde uygulama yüklemek için kullanılan EXE uzantılı dosyalar olduğu görülmüştür. Sırasıyla EXE dosya türünü Android, DLL, Javascript ve hiper metin işaretleme dili (HTML) dosya türleri takip etmektedir.

2.1.1.1 Virüs

Virüs, bilgisayarın çalışma şeklini kullanıcının bilgisi ve izni dahilinde olmadan değiştiren ve kendini gizleyen yazılımlardır (Egemen vd 2015). Virüsün aktif olabilmesi için kullanıcı tarafından sistemde çalıştırılması gerekmektedir.

2.1.1.2 Solucan (Worm)

Solucan, bağımsız olarak bulunan kötü amaçlı bir kod parçasıdır (Aslan vd 2020). Solucanlar, kendisini çoğaltma özelliğine sahiptir. Depolama aygıtları ve e-postalar yoluyla yayılırlar. Bilgisayar ve ağ üzerinde performans sorunlarına ve kaynak tüketimine neden olmaktadır.

2.1.1.3 Truva atı (Trojan)

Truva atı yararlı bir program gibi davranır ama zararlı bir amacı vardır. Kendilerini çoğaltmazlar, ancak bir bilgisayara indirme gibi internet etkileşimi ile aktarılırlar. Hassas bilgileri çalar, kullanıcıların etkinliklerini gözlemler ve bulunduğu sistemdeki dosyaları silebilmekte, değiştirebilmekte veya bozabilmektedir (Salmani vd 2011).

2.1.1.4 Casus yazılım (Spyware)

Casus yazılım, kişisel bilgileri çalmak veya kullanıcının etkinliklerini izlemek için kullanılır (Aslan vd 2020). Sistem sahibinin bilgisi olmadan kurulur ve gizlice bilgiyi toplar ve yaratıcısına geri gönderir. Google gibi büyük isimleri olan bir şirket bile, kullanıcılarının gerekli bilgilerini toplamak için casus yazılım kullanmaktadır.

2.1.1.5 Reklam oynatıcı (Adware)

Kullanıcıların kullandıkları akıllı cihaz, bilgisayar sistemlerinde izinsiz olarak reklam oynatır ve kullanıcının yapmış olduğu çalışmanın kesintiye uğramasını sağlar. Reklamın temel amacı kullanıcının zamanını çalmakta ve reklam sağlayıcı kişiye finansal fayda sağlamaktadır.

2.1.1.6 Bot

Saldırganın virüs içeren bir bilgisayarı kontrol etmesine izin veren bir yazılımlardır. Bilgisayar korsanları ya da saldırganlar tarafından kontrol edilen virüslü bilgisayarlardan oluşurlar. Sahibinin bilgisi olmadan kötü niyetli faaliyetler gerçekleştirebilmektedirler. Örneğin, Hizmet reddi saldırıları yapabilmekte, spam mesajları gönderebilmekte, bilgi çalabilmektedir.

2.1.1.7 Spam

Önemsiz e-postalar, Aynı e-postalar oluşturularak, aynı anda birden fazla alıcıya gönderilmesidir (Thomas ve David 2010). Çok fazla bant genişliği tüketmekte ve aynı zamanda sistemi yavaşlatmaktadır.

2.1.1.8 Fidye yazılımı (Ransomware)

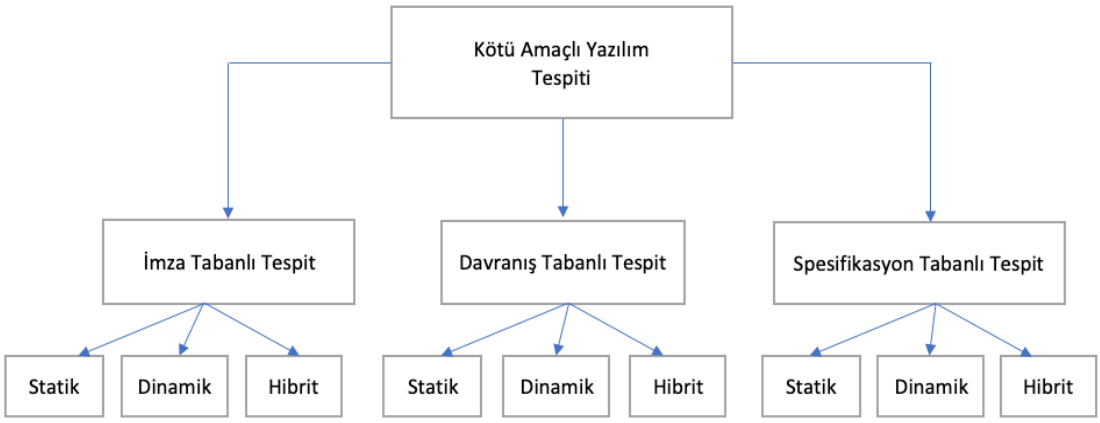
Fidye yazılımı, verilerimizi şifrelemek için kullanılan, kullandığımız uygulamaları durduran ya da yazılımı yapan geliştiricinin taleplerini yerine getirene kadar işletim sistemimizi kullanmamıza izin vermeyen, bilgisayarımızın kontrolünü ele geçiren yazılımlardır (Aslan vd 2020). Yazılımı geliştiren kişi taleplerini çoğunlukla para cinsinden istemektedir. Para ödendikten sonra sistemin kontrolünü geri alacağımızın garantisi bulunmamaktadır.

2.1.1.9 Keylogger

Şifreleri, kredi kartı detaylarını ve diğer önemli ve hassas verileri çalmak için tuş vuruşlarını kaydeden bir tür casus yazılımdır. Başka bir kötü amaçlı yazılım yüklendiğinde veya virüslü herhangi bir site kullanıcı tarafından ziyaret edildiğinde bilgisayara aktarılmaktadır.

2.1.2 Kötü amaçlı yazılım analiz ve tespit yöntemleri

Kötü amaçlı yazılımların tespit ve analiz yöntemleri 3 ayrı alt başlık altında incelenmektedir. Tespit yöntemleri için imza tabanlı, davranış ve spesifikasyon tabanlıdır. Analiz yöntemleri ise statik, dinamik ve hibrit olarak kullanılmaktadır (Idika vd 2007, Barsiya vd 2016). Yöntemlerle ilgili bilgiler ve tez süresince kullanılarak kötü amaçlı yazılım tespit ve analiz yöntemleri aşağıda paylaşılmıştır. Kötü amaçlı yazılımların analiz ve tespit yöntemleri **şekil 2.2**'de gösterilmiştir.



Şekil 2.2 Kötü amaçlı yazılım analiz ve tespit türleri

2.1.2.1 İmza tabanlı tespit yöntemi

Kötü amaçlı yazılım yazıldığında, zararlı yazılımın hangi aile ait olduğunun bilinmesi amacıyla genellikle imza olarak adlandırılan bit dizisi koda eklenir. Antivirüs programları bit dizisini kendi sistemlerinde depolarlar (Idika vd 2007). Zararlı yazılım ile ilgili imza sistemlerinde arandığında arama yaparak, sistemlerinde bulunan zararlı yazılımla ilgili bilgiyi kullanıcısına gösterir. Statik analiz uygulaması olan md5sum kullanılarak zararlı yazılımın özet bilgisi olarak şifrelenmiş hali oluşturulmaktadır. Oluşturulan özet bilgisi virustotal vb. anti virüs sitelerinde taratılarak zararlı yazılım ile ilgili bilgiler alınmaktadır (Anonymous 2022). Tez çalışmasında statik ve dinamik analiz yönteminde kullanılan trickster.exe isimli zararlı yazılımın özet bilgisi **şekil 2.3**'de paylaşılmıştır. Yeni çıkan zararlı yazılımlarda bu yöntem başarısız olarak cevap vermektedir (Aydoğan ve Şen 2014).

```
(base) umitkose@umitkose-N550JK:~/Desktop$ md5sum trickster.exe
fc592cc04d54e2b1af2a2bc63319709a  trickster.exe
```

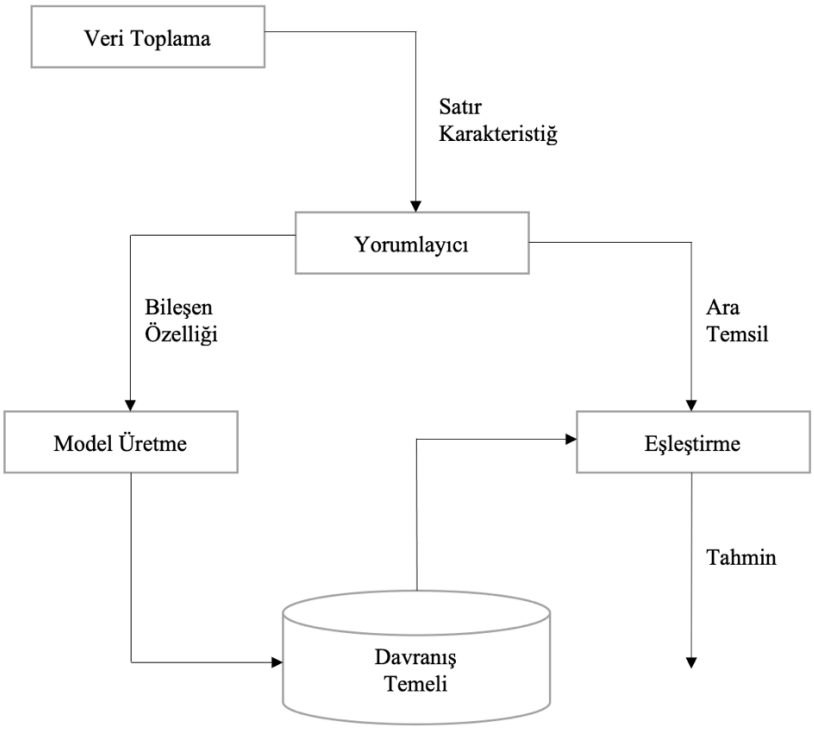
Şekil 2.3 İmza tabanlı yaklaşım örneği

2.1.2.2 Davranış tabanlı yöntem

Davranış tabanlı yöntem de sistemin normal ve anormal davranışlarını gözlemleyerek, aralarındaki farkları tespit ve ayırt edilmesiyle birlikte bilinen veya bilinmeyen kötü

amaçlı yazılımların tespit edilmesi yöntemidir (Aslan ve Samet 2020). Bu yöntemde davranış saldırısı gözlemlenir ve saldırı durumunda önemli bilgilerin kaydı tutulur.

Davranış tabanlı yaklaşımın yöntemi **şekil 2.4**'te gösterilmiştir.



Şekil 2.4 Davranış tabanlı yaklaşım yöntemi (Sewak vd 2018)

Davranış tabanlı yöntem 3 temel bileşenden oluşmaktadır.

Veri toplama : Adından da anlaşılacağı gibi, bu bileşen statik veya dinamik veri toplanmasıdır.

Yorumlama : Veri toplama bileşininden toplanan veriler bir takım normalizasyon yöntemlerinden geçirilerek yorumlanacak formatlara dönüştürülür.

Algoritma eşleştirme : Bu bileşen, davranış imzasını yorumlama ve bileşeninde dönüştürülen bilgilerle eşleştirilmek için kullanılmaktadır.

2.1.2.3 Spesifikasyon tabanlı yöntem

Bu yöntem davranış tabanlı yöntemin bir alt türü olarak geçmektedir. Bu yöntemde uygulamalar spesifikasyonlara göre izlenerek, normal ve anormal davranışlar tespit

edilmektedir. Bu yöntemde makine öğrenmesi veya yapay zeka ile tespit işlemlerine gerek kalmadan sistem davranışları izlenmektedir.

2.1.2.4 İmza ve davranış tabanlı yaklaşımların karşılaştırılması

İmza ve davranış tabanlı tespit yöntemlerin birbirlerine göre avantaj ve dezavantajları bulunmaktadır. İmza tabanlı tespit yöntemi kullanılarak bilinen zararlı yazılım tespiti oldukça kolay olmakta ve diğer yönteme göre daha az kaynak kullanmaktadır. Fakat yeni bir kötü amaçlı yazılım tespitinde yetersiz kalmaktadır. Davranış tabanlı yaklaşım ise yeni bir zararlı yazılım tespiti için kullanılması avantajlı olmaktadır. Yeni bir zararlı yazılımın tespitin de ise daha çok zamana ihtiyaç duymaktadır. İmza ve davranış tabanlı yöntemlerin karşılaştırılmaları **çizelge 2.2**'de gösterilmiştir.

Çizelge 2.2 İmza tabanlı ve davranış tabanlı yöntemlerin karşılaştırılması

	Avantaj	**Dezavantaj**
İmza Tabanlı Yöntem	Bilinen KAY tespitinde kolaydır.	Bilinmeyen KAY tespitinde iyi değildir.
	Diğer yöntemlere göre daha az kaynak kullanmaktadır.	
Davranış Tabanlı Yöntem	Bilinmeyen ve bilinen KAY tespiti yapabilir	Zararlı yazılımın çalıştırılmasına rağmen bütün davranışlarını gösteremeyebilir.
		KAY tespiti için daha çok zamana ihtiyaç duyar.

İmza tabanlı yöntemin avantajı bilinen zararlı yazılımların tespitinde hızlı, kolay ve davranış tabanlı yönteme göre programın davranışına bakılmasına gerek olmamasıdır. Fakat bilinmeyen zararlı yazılımların tespiti için imza tabanlı yöntem başarılı olamamaktadır (Aydoğan ve Şen 2014). Davranış tabanlı yöntem ise imza tabanlı yöntemin aksine bilinen ya da bilinmeyen zararlı yazılım tespiti için kullanılmaktadır. Kötü amaçlı yazılımın davranışına göre incelemeler yapılarak zararlı yazılım tespitinde bulunulmaktadır. Davranış tabanlı yöntemin dezavantajı ise sanal makine ve sanbox ortamlarında zararlı yazılımın çalıştırılarak tespit edilmesi gerektiğinden, tespit edilme

süreci yavaş veya zararlı yazılım davranışına denk gelinmemesi durumunda yetersiz kalabilmektedir (Alosefer 2012).

2.1.2.5 Statik analiz yöntemi

Zararlı yazılımların analizi yapılırken eğer bir parça kod veya yazılım çalıştırılmadan üzerinde incelemeler yapılıyorsa bu analiz yöntemi statik analizdir. Statik analizde daha çok kod içerisinde karakter dizileri, kodun yapısı ya da zararlı yazılım ile ilgili özellikler tespit edilmeye çalışılmaktadır. Bu analiz yönteminde tersine mühendislik teknikleri kullanılmaktadır.

2.1.2.6 Dinamik analiz yöntemi

Zararlı yazılımların analizi yapılırken sistemle aynı benzerliğe sahip bir ortamda zararlı yazılımın çalıştırılarak, sistem üzerindeki etkilerinin gözlemlenmesi yöntemi ise dinamik analiz olarak geçmektedir (Aslan ve Samet 2017). Dinamik analizde daha çok zararlı yazılımın etkileri gözlemlenmektedir. Statik analiz'e göre daha tehlikelidir. Sebebi ise kötü amaçlı yazılımın sistem üzerindeki tüm etkilerine maruz kalınmasıdır.

2.1.2.7 Hibrit analiz yöntemi

Hibrit analiz yöntemi zararlı yazılım tespitinde hem dinamik hem de statik analiz tekniklerinin birleştirilerek, her iki yaklaşımdan faydalanılan bir yöntemdir. Bu yöntemde kötü amaçlı yazılımın imzası kontrol edilerek kod analizi yazılımlar ile gözlemlenir ve sonrasında davranışın gözlemlenmesi amacıyla zararlı yazılım sistemde çalıştırılır.

2.1.2.8 Statik ve dinamik analiz yöntemlerinin karşılaştırılması

Statik ve dinamik analizin birbirlerine göre avantaj ve dezavantajları bulunmaktadır. Dinamik analiz zararlı yazılımın etkilerine maruz kaldığı için sistemi olumsuz etkileyebilmektedir. Statik analiz ise özellikle yeni kötü amaçlı yazılımların tespitinde yetersiz kalmaktadır (Damodaran ve Anusha 2017). Statik analiz dinamik analize göre hızlı bir tespit yöntemidir. Statik analizde sanal bir ortam kurulmasına gerek bulunmamaktadır. Statik ve dinamik analiz yöntemlerinin karşılaştırılması **çizelge 2.3**'de gösterilmiştir.

Çizelge 2.3 İmza tabanlı ve davranış tabanlı yöntemlerin karşılaştırılması

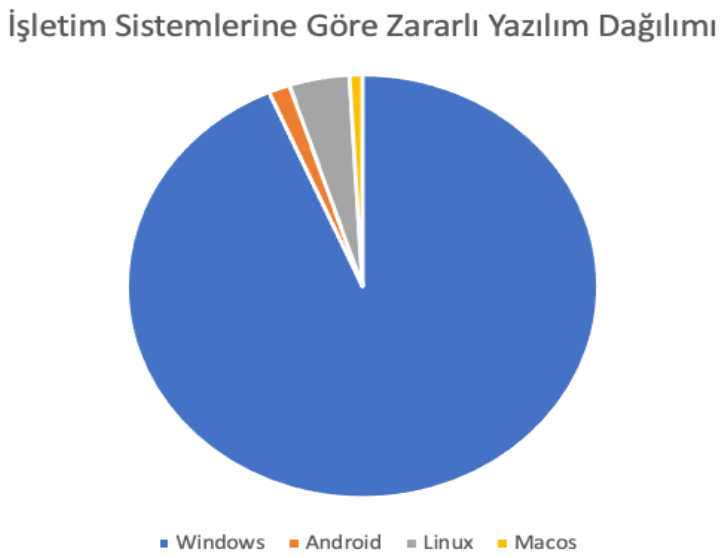
Statik analiz	**Dinamik analiz**
Zararlı yazılım çalıştırılmadan analiz yapar.	Zararlı yazılım çalıştırılarak analiz yapar.
Tersine mühendislik, binary kod, hafıza etkileri, paketleme ve debugging içerir.	Sistem ve Network çağrıları, api çağrıları, registry, process değişiklikleri ve komutların izlenmesini içerir.
Hızlı	Yavaş
Güvenli	Güvensiz
Sanal ortam üzerinde kontroller yapılması iyi olmakla birlikte ihtiyaç yoktur.	Sanal ortam üzerinde kontroller yapılmasına ihtiyaç vardır.
Yeni kötü amaçlı yazılım tespitinde yetersiz bulunmaktadır.	Yeni kötü amaçlı yazılım tespitinde tercih edilen yöntemdir.

Statik analiz yönteminde zararlı yazılım çalıştırılmadan analiz yapılmaktadır. Dinamik analiz yönteminde ise zararlı yazılımın çalıştırılması gerekmektedir. Dinamik analiz yönteminde zararlı yazılım çalıştırılmadan önce sanal ortamın kurulması ve oluşturulan sanal ortamda zararlı yazılımın çalıştırılmadan önce yedeğinin alınması önerilir (Kara ve Aydos 2018). Statik analiz için sanal ortama ihtiyaç bulunmamasına rağmen güvenlik amaçlı çalışmalar bu ortamda yapılabilir. Statik analiz yönteminde kötü amaçlı yazılımın içerisinde kod ve yazı içerikleri tersine mühendislik teknikleri kullanılarak incelemeler yapılmaktadır. Dinamik analiz yönteminde ise zararlı yazılım çalıştırılarak sistem üzerindeki etkileri incelenir. Statik analiz yönteminde zararlı yazılım çalıştırılmadığı için dinamik analize göre daha hızlı ve güvenli bir yöntemdir. Dinamik analiz yönteminde ise kötü amaçlı yazılımın sistem üzerindeki davranışları incelenirken yavaş ve çalıştırılmasından dolayı güvensiz olmaktadır. Statik analiz yöntemi yeni çıkan kötü amaçlı yöntemlerde yetersiz bulunuyorken (Aydoğan ve Şen 2014), dinamik analiz yöntemi yeni çıkan zararlı yazılımların tespitinde daha çok kullanılan yöntemdir.

2.2 Zararlı Yazılımlardan En Çok Etkilenen İşletim Sistemleri

İşletim sistemi görsel bir arayüz üzerinden donanım ve yazılım arasındaki iletişimi sağlayan teknoloji olmuştur. Bilgisayar, cep telefonu gibi cihazların hepsinde işletim sistemi bulunmaktadır. Kullanıcıların en çok tercih ettiği işletim sistemleri ocak ayı 2022 yılı itibariyle %39 seviyelerinde Android ve %32 seviyelerinde Windows işletim sistemi olmuştur (Anonymous 2022). Masaüstü ve dizüstü işletim sistemleri arasında en çok

kullanılanı windows işletim sistemleridir. Zararlı yazılım geliştiricilerinin en çok kötü amaçlı yazılım geliştirdiği işletim sistemi bilgisi **şekil 2.5**'te paylaşılmıştır.



Şekil 2.5 KAY'ların işletim sistemi özelinde kullanım istatistiği (Anonymous 2022)

Paylaşılan grafikte 2022 yılı için %95 oranında Windows işletim sistemi için kötü amaçlı yazılım geliştirilmiştir (Anonymous 2022). Windows zararlı yazılımlara en çok maruz kalan işletim sistemidir. Yukarıda anlatılan zararlı yazılım tespit ve analiz yöntemleri incelendiğinde Windows işletim sistemleri için uygulanması daha başarılı ve doğru olacağı düşünülmektedir.

2.3 Derin Öğrenme

Yapay zeka ilk olarak 1943 yılında insan sinir sisteminin çalışmasını mantıksal olarak modelleyen çalışmayla başlamıştır (McCulloch ve Pitts 1943). Yapay sinir ağlarının temellerini oluşturan bu çalışmadan sonra Perceptron (Rosenblatt 1958, Rosenblatt 1962) ve Adaptive Linear Element (Widrow ve Hoff 1960) çalışmaları ortaya çıkmıştır. Oluşturulan bu modeller XOR problemine çözüm sağlayamaması yapay zeka çalışmalarını yavaşlatsa da, paralel işlemlerle (Rumelhart vd 1986, McClelland, vd 1995) ve geri yayılım algoritmalarıyla (Rumelhart vd 1986, LeCun 1987) ilgili yapılan çalışmalar derin öğrenmenin temellerini oluşturmuştur.

Derin öğrenme yöntemi 2012 yılına kadar teknolojik yetersizlik ve veri sıkıntılarından ötürü çok hızlı bir şekilde ilerleyememiştir. 2012 yılında İmageNet yarışmasıyla derin öğrenme teknolojisinin popülerliği ve teknolojik olarak kullanımı artmaktadır (You 2018). Yarışmada birçok farklı kategoride 100 'lerce resim üzerinden katılımcıların oluşturmuş olduğu derin öğrenme modelleriyle birlikte nesne sınıflandırma yapılmaktadır. ImageNet yarışmasının sonucunda insanlar derin öğrenme uygulamaları için çok fazla sayıda kullanılan evrişimsel sinir ağı (ESA) (CNN) ile tanışmıştır (Krizhevsky vd 2012). Yapılan yarışmanın daha sonraki yıllarında ortaya birçok derin öğrenme modeli çıkmış ve derin öğrenmenin gelişimine katkı sağlanmıştır. Bu modeller Alex Net, GoogLeNet, Microsoft RestNet, R-CNN, Fast R-CNN'lerdir (Girshick 2015, Iandola 2016).

Derin öğrenme içerisinde en çok bilinen mimariler CNN, DNN ve RNN'dir (Larochelle 2009). Yapmış olduğumuz çalışma da CNN sinir ağları kullanarak kötü amaçlı yazılımlar tespit edilmiştir. CNN içerisinde 3 bölüme ayrılmıştır. Konvolüsyon katmanı bir resimdeki özellikleri ayrılan kısımdır. Havuzlama katmanı resmin en önemli özelliklerinin ön plana alındığı bölümdür. Fully connected katmanı ise üstteki 2 kısımdan önemli özelliklerini elde ettiğimiz resmin ANN ile veriyi öğrendiğimiz ve doğruluğunu elde etmeye çalıştığımız alandır. Burada kötü amaçlı yazılım özelliklerinden ve elde edilen resimden bilgiler öğrenilmektedir.

Derin öğrenme günümüz teknolojisinde ses tanımlama, görüntü renklendirme, görüntü tamamlama, nesne tespiti gibi çok fazla alanda kullanılmaktadır. Siyah beyaz bir resmin renklendirilmesi (Hwang ve Zhou 2015, Larsson ve Maire 2016, Zhang ve Isola 2016), Google ve Facebook gibi büyük firmaların sosyal medyalarında yüklenen resimler için nesne etiketlenmesi (Kiros 2014), insan gibi oyun oynanması (Mnih 2013) gibi alanlarda derin öğrenme modelleri kullanılmıştır.

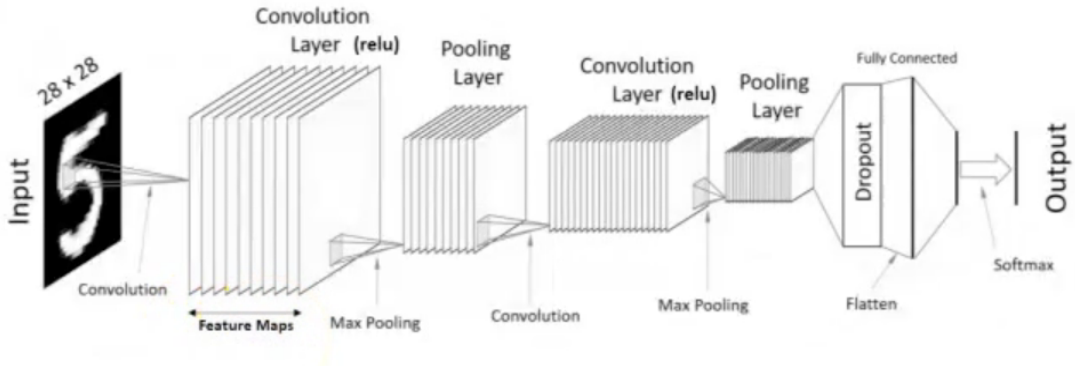
2.3.1 Yapay sinir ağı (ANN)

Yapay Sinir ağı insan beynine benzeyen yapıda içerisinde nöronlardan oluşan, bilgiyi işleyerek ve analiz ederek kendi kendine öğrenebilen bilgisayar sistemidir. Etiketli bir öğrenme yöntemi olan ANN, etiketli veriler'den oluşan bir veri setini öğrenerek etiketsiz

veri setlerindeki çıktıları tahmin eden bir yöntemdir. Girdi katmanı, gizli katmanlar ve sonuç katmanı olmak üzere 3 katmanı vardır. Katmanlar öğrenmenin olduğu kısımlardır. Kayıp (Loss) fonksiyonları ile geliştirilen modelin performansı bilinmektedir. Optimizasyon algoritmaları ile öğrenen modelin bilgisi sürekli güncellenmektedir. Aktivasyon fonksiyonları ile bir modelin çıktısı tanımlanmaktadır. Öğrenme işlemi en yüksek oranda başarı oranı bulununcaya kadar devam etmektedir.

2.3.2 Evrişimsel sinir ağı (CNN)

İlk kez karşımıza 2012 yılındaki ImageNet yarışmasından sonra çıkan CNN arkasında çok sayıda matematiksel formül barındıran modeldir (You 2018). CNN ile birlikte bilgisayarla görme, ses işleme veya görüntülerle ilgili renklendirme, çoğaltma, tamamlama gibi işlemler uygulanmaktadır. Evrişimsel sinir ağının ortaya çıkmasıyla birlikte derin öğrenme üzerine çalışmalar hızlanmıştır.



Şekil 2.6 CNN mimarisi

CNN mimarisi 5 kısımda incelenmektedir. CNN Mimarisi **şekil 2.7** 'de gösterilmiştir. İlk olarak CNN mimarisi konvolüsyon katmanı ile başlamaktadır. Konvolüsyon katmanı resim üzerindeki özellik haritasının çıkarıldığı, resmin özelliklerini ayırdığımız kısımdır. Köşe, göz, kulak gibi kısımları, sayıdaki kenarları resim üzerindeki özellikleri tespit eden bölümdür. Konvolüsyon katmanında bir filtre belirlenmekte ve verilen giriş üzerinde filtre gezerek resmi taramaktadır. Filtre işlemi yapılırken veri azaltma işlemlerinde aktivasyon fonksiyonlarından faydalanılmaktadır. Konvolüsyon katmanından sonra resim üzerinde vermiş olduğumuz girişin boyutu azalmaktadır. Daha sonraki katmanlarda

resim boyut olarak eski haline 0'lar eklenecek şekilde gelmiştir. Same padding adı verilen bu yöntem 2. kısmı oluşturmaktadır. Etkisiz eleman olan 0 sayesinde boyut olarak azaltmamış gibi görünse de bu yöntem sayesinde matematiksel işlemlerde hız ve zamandan tasarrufu sağlanmaktadır.

3. bölümde konvolüsyon katmanından sonra elimizde özellikleri daha ortaya çıkmış bir resim bulunmaktadır. Seçtiğimiz modele göre en büyük, en küçük ya da ortalama bulma işlemlerinden birini havuzlama adı verilen yöntemle birlikte uygulanarak resmin boyutu daha fazla oranda küçültmüş ve veriler tek düze hale gelmiştir. Havuzlama işlemleri aşırı öğrenmeyi ortadan kaldıran bir yöntemdir. 3. bölümde matris içerisinde bulunan resim bilgileri yapay sinir ağına verilebilmesi için flattening yöntemi denilen X,1'lik matris haline gelmiştir.

CNN mimarisinin 5. Maddesi olan Fully Connected kısmında flatten ile X,1'lik matris halinde bulunan giriş verisini yapay sinir ağında işlem yapılabilmesine olanak sağlamaktayız. Yapay sinir ağı ile ilgili işlemler **2.3.1 Yapay Sinir Ağı** bölümünde anlatılmıştır.

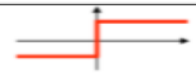
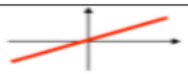
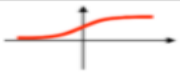
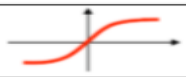
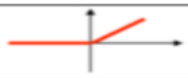
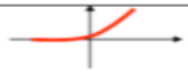
2.3.3 Derin öğrenmede kullanılan matematiksel fonksiyonlar

Derin öğrenme arkasında çok sayıda matematiksel işlem ve formül barındırmaktadır. Geliştirilen derin öğrenme modelindeki hataları tespit etmek, veri seti üzerindeki başarı ölçütünü hesaplayabilmek ve hata oranını en aza indirebilmek için aktivasyon, kayıp ve optimizasyon fonksiyonları kullanılmıştır. Bu fonksiyonlardan önerilen model içerisinde tercih edilen algoritmaların neden tercih edildiği alt başlıklarda anlatılmıştır.

2.3.3.1 Aktivasyon fonksiyonu

Aktivasyon fonksiyonları matematiksel formüller içeren ve bu formüller sayesinde oluşturulan mimarinin aşamalarında ya da sonuçlarında toplam değerin karakterize edilmesini sağlayan işlemlerdir. Veri setinin değerlerine göre aktivasyon fonksiyonları kullanılır. Yapılan tez çalışmasında iyi ve kötü huylu yazılımların çıktılarını elde etmek için Sigmoid fonksiyonu ve ölü nöronların çıkarılması için RELU fonksiyonu ile CNN

mimarisinde kullanılmıştır. Derin öğrenme modellerinde en çok kullanılan aktivasyon fonksiyonlarının listesi **şekil 2.8**'de paylaşılmıştır.

Sign (Signum)	$f(x) = \begin{cases} -1 & z < 0 \\ 0 & z = 0 \\ 1 & z > 0 \end{cases}$	Perceptron	
Linear	$f(x) = x$	Linear regression	
Logistic (sigmoid)	$f(x) = \frac{1}{1+e^{-x}}$	Logistic regression, Multi layer NN	
Hiperbolik Tanjant	$f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	Multi layer NN	
ReLU (Rectified Linear Unit)	$f(x) = max(0, x)$	Multi layer NN	
Softplus	$f(x) = ln(1 + e^{x})$	Multi layer NN	

Şekil 2.7 Aktivasyon fonksiyonlarının listesi (Anonymous 2022)

Derin öğrenme mimarilerinin aşama ve sonuçlarında toplam değerin karakterize edilmesi için çok fazla sayıda kullanılan aktivasyon fonksiyonları aşama ve çıktı durumlarına göre çeşitlilik göstermektedir (Bircanoğlu ve Arıca 2018). Sigmod aktivasyon fonksiyonu -1'den küçük ve 0 değerleri için 0, 0'dan büyük değerler için 1 döndürmektedir. Sigmoid çıktı fonksiyonlarında ikili sonuç durumlarında tercih edilebilmektedir. Bu tez çalışmasında çıktı fonksiyonunda zararlı yazılımın iyi veya kötü huylu olmasının belirlenmesi durumu için sigmoid kullanılmıştır. Softmax aktivasyon fonksiyonu çıktı birden fazla olan çok sınıflandırma problemleri için kullanılmaktadır. Giriş değerlerinin hangi sınıfa ait olduğunu [0,1] arasındaki değerlere göre gruplamaktadır. Tanh aktivasyon fonksiyonu doğrusal olmayan ve [-1,1] arasında çıktı üreten fonksiyondur. ReLU (düzeltilmiş doğrusal birim) fonksiyonu negatif girdiler için 0 ve pozitif girdiler için mevcut değerini koruyan aktivasyon fonksiyonudur.

2.3.3.2 Kayıp (Loss) fonksiyonu

Loss fonksiyonları derin öğrenme modellerinin veri seti üzerindeki başarısını ölçen matematiksel fonksiyonlardır. Eğer model veri seti üzerinde yapmış olduğu tahminlerde yüksek başarı gösteriyorsa loss fonksiyonu düşük, eğer düşük başarı gösteriyorsa loss fonksiyonu yüksek oranda çıkmaktadır. Ortalama kare, Ortalama mutlak, Binary Cross gibi birden fazla loss fonksiyonu bulunmaktadır. Yapılan tez çalışmasında çıktı ikili

sınıflandırma içerdiği için sigmoid aktivasyon fonksiyonunun arkasında geliştirilen mimari modelin başarısını ölçmek için binary cross entropy kullanılmıştır.

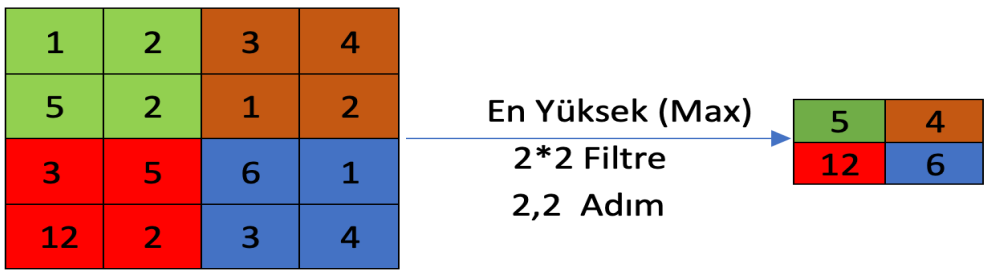
$$Binary\ Cross\ Entropy = -y1\ log\ log\ (p1)\ -(1-y1)log\ (1-p1)$$

2.3.3.3 Optimizasyon fonksiyonu

Derin öğrenme yöntemlerinde doğrusal olmayan problem çözümü için en doğru optimum değeri bulabilmek ve hata oranını en aza indirebilmek için optimizasyon fonksiyonları kullanılmaktadır. Bu fonksiyonlar Adam, Gradient Descent, Adagrad, Adamax, Rmsprob gibi fonksiyonlardır. Tez çalışması kapsamında optimizasyon fonksiyonu olarak Adam algoritması seçilmiştir. Adam algoritması 2014 yılında Rmsprob ve momentum gibi diğer optimizasyon algoritmaların avantajlı yönlerini kullanan ve güncel olan gradient descent algoritmasıdır (Seyyarer 2020).

2.3.3.4 Havuzlama (Pooling) katmanında kullanılan fonksiyonlar

Havuzlama katmanları CNN mimarisinde öznitelik çıktılarını özetleme ve boyut düşürme için kullanılmaktadır. Girdinin özet bilgisi ile veri üzerindeki özellikler daha belirgin olarak ortaya çıkmaktadır. Havuzlama katmanında en yüksek, ortalama ve en düşük bulma yöntemleri kullanılmaktadır. En yüksek bulma verilen bir filtre kullanılarak belirli bir adımda ilerlendiğinde oluşan kare içerisinde en yüksek sayıyı alarak ilerlemektedir. Ortalama bulma da ise oluşan alt kare içerisinde ortalama sayı hesaplanarak oluşturulmaktadır. En küçükte ise oluşan alt kare içerisindeki en küçük sayı alınmaktadır. **şekil 2.9**'da tez kapsamında kullanılan en yüksek havuzlama yöntemi anlatılmıştır.



Şekil 2.8 Pooling katmanına en yüksek fonksiyon uygulanması (Max Pooling)

3. LİTERATÜR ÇALIŞMASI

Makine öğrenmesi ve derin öğrenme yöntemi ile kötü amaçlı yazılım tespiti için çok sayıda çalışma ve araştırma yapılmaktadır. Tez çalışması yapılmadan önce, tez konusuyla daha önceden ilgilenmiş kişilerin yapmış olduğu araştırmalar ve çalışmalar incelenmiştir. Yapılan çalışmaların sonucunda elde edilen sonuçlar incelenmiştir. Mevcut çalışmalardan tez konusuyla örtüşen çalışmalar aşağıda paylaşılmıştır.

Kötü amaçlı yazılım tespiti için Hansen vd (2016) tarafından yapılan çalışma da 270000 fazla zararlı yazılım ve 837 zararsız yazılımdan oluşan veri kümesinde Rassal Orman algoritması kullanılarak %98,1 oranında kötü amaçlı yazılım tespiti yapıldığı paylaşılmıştır. Yapılan bu çalışma da verinin %99 oranında zararlı yazılım içermesi yanlılık problemine neden olacağı ve yapılan çalışmanın yeni veri setlerinde ya da yeni çıkan zararlı yazılımlarda sorunlar oluşturacağı düşünülmüştür.

Hardy (2016) yapmış olduğu çalışmada ise kendi oluşturmuş olduğu 50000 iyi ve kötü huylu yazılımı ANN, Destek Vektör Makinesi (SVM), Naive Bayes (NB), Karar Ağaçları (DT) ve kendi yöntemleri olan Deep Learning Freamework for Intelligent Malware Detection (DL4MD) yöntemleriyle karşılaştırma yapmaktadır. Yapmış olduğu çalışmada en iyi başarı oranı 3 gizli katman ile denemiş olduğu kendi yöntemleri olan DL4MD %96 oranında vermiştir.

Aslan ve Samet (2017) tarafından yapılan çalışma da spesifikasyon tabanlı herhangi bir tespit yöntemi kullanılmadan dinamik analiz yöntemleri ve uygulamaları kullanılarak zararlı yazılım tespiti yapılmaya çalışılmıştır. Yapılan çalışma da dinamik analiz yöntemi için Wireshark, Cuckoo Sandbox, Process Monitor, PEView vb ve statik analiz yöntemi için Dependency Walker, IDAPro, BinText, PEid, PeView vb uygulamalar kullanılarak %87,5 oranında kötü amaçlı yazılım tespiti yapılmıştır. Yapılan çalışmanın başarı oranı derin öğrenme ve makine öğrenmesi algoritmalarına göre daha az olduğu görülmüştür.

Mohit vd. (2018), makine öğrenmesi algoritmalarından Rassal Orman (Random Forest) ve Derin Sinir Ağı (DNN) 2, 4 ve 7 katmalı yapısı oluşturularak yaklaşık 14000 test

verisinde bu teknikler karşılaştırılmıştır. Test verisi için 11308 zararlı ve 2819 iyi huylu yazılım ile karşılaştırma yapılmıştır. Rassal Orman algoritması için Python'ın Skilearn ve derin öğrenme için Keras ile Tensarflow kütüphaneleri kullanılmıştır. Loss fonksiyonu olarak Adam ve aktivasyon fonksiyonu olarak RELU tercih edilmiştir. En iyi başarı olarak 7 Katmanlı derin öğrenme yöntemi ile %99,21 başarı oranında kötü amaçlı yazılım tespiti yapılmıştır.

Irshad vd. (2019) çalışmalarında, dinamik analiz yöntemiyle genetik algoritma kullanarak Windows işletim sistemleri için en önemli parametreleri içeren 41 özellik elde etmişlerdir. Yapılan çalışma da 236 adet veri bulunmaktadır. Yaklaşık olarak yarı yarıya zararlı ve iyi huylu yazılım içeren veri setinde Rassal Orman, SVM ve NB algoritmaları kullanılarak zararlı yazılım tespit çalışmaları yapılmıştır. Yapılan çalışma da sırasıyla %86,8, %81,3 ve %64,7 oranında zararlı yazılım tespiti yapılmıştır. Günümüz ve yapılan tez çalışması ile bu çalışma karşılaştırıldığında zararlı yazılım tespit oranının düşük olduğu görülmektedir.

Cho (2019), Microsoftun kötü amaçlı yazılımların tespiti için oluşturmuş olduğu veri setinin 10868'ini kullanan çalışmada, rastgele uzunlukta veri boyutları ve konvolüsyon ile uzun kısa süreli bellek (LSTM)'i birleştirerek adını koydukları ConvLSTM, CNN ve Attention mekanizması kullanılmıştır. Çalışmada rastgele uzunlukta veri setine sahip modelin performansının, özellik çıkarma ve boyut azaltma olmadan doğrulanabileceğini gösterilmiştir. Yapılan çalışmada %96 oranında başarı oranı elde edilmiştir.

Poonguzhali vd (2019) tarafından yapılan kötü amaçlı yazılım için derin öğrenme yöntemlerinden CNN'i kullanarak yapmış olduğu çalışma da %99,01 oranında başarım elde edilmiştir. Yapılan çalışmada dosyalar pandas kütüphanesi ile özellik çıkarımı yapılmıştır. Numpy kütüphanesi kullanılarak özellik çıkarımından elde edilen bilgiler binary kodlara dönüştürülerek, matplotlib kütüphanesi aracılığıyla gri resimlere dönüştürülmüştür. Buradaki normal çalışmadan farkı gri resimleri CNN algoritması ile eğitimi yapılmıştır: Bat algoritması adını verdikleri resimdeki gürültü ve özellik haritasını bulmadaki davranışları derin öğrenmede kullanılmak için çıkarılmıştır. Son aşamada ise karar destek sınıflandırma ile çalışmayı tamamlayarak doğruluk oranını hesaplamıştır.

Kartal vd (2019) yapmış olduğu Özellik Çıkarma Yöntemlerinin zararlı yazılım sınıflandırmasına etkileri çalışmasında 25 farklı aileye ait 9339 zararlı yazılım Ma1Img veri setinden toplanarak gri tonlamalı resimlere dönüştürülmüştür. Genişlik her bir çalışma için aynı boyuta sahiptir. K en yakın komşu (KNN), Densenet, VGG19 ve VGG16 algoritmalarıyla birlikte yapılan karşılaştırmada da en başarılı çözüm VGG16 algoritmasıyla %98,94 ile sağlanmıştır.

Liu vd. (2019) tarafından yapılan çalışma, 13518 kötü huylu ve 7860 iyi huylu olmak üzere toplam 21378 adet veriden yararlanmıştır. Dinamik analiz yöntemi için sandbox üzerinde zararlı yazılımların davranışları incelenmiş ve veri seti oluşturulmuştur. Veri setinde kayıt defterindeki bilgiler kullanılmaması veri seti hakkında yanlış sonuç doğurabilmektedir. Yapılan çalışmada veri seti üzerinde BLSTM, LSTM, BGRU, GRU ve basit RNN yöntemlerini kullanarak kötü amaçlı yazılım tespiti üzerine çalışmışlardır. Yapılan çalışmada BLSTM yöntemi kullanılarak %97,85 oranında zararlı yazılım tespiti yapmışlardır.

Tokmak ve Küçüksille (2019) tarafından yapılan çalışma, Windows 10 işletim sisteminin API çağrıları ve çalıştırılabilir dosyalarının opsiyonel başlıklarını analiz ederek oluşturdukları veri setinde zararlı yazılım tespiti üzerine çalışmışlardır. VirusShare internet sitesinden indirmiş oldukları 592 tane zararlı yazılım ve sistem dosyalarından elde edilen 283 adet zararsız yazılımdan veri seti oluşturmuşlardır. DNN ile yapmış oldukları çalışmada, %80 eğitim, %20 test verisi olarak veri seti ayrılmıştır. 15 farklı aileye ait kötü amaçlı yazılımdan oluşturmuş oldukları veri setinde %100 oranında zararlı yazılım tespiti yapmışlardır.

Kötü amaçlı yazılım tespitinde üç farklı derin öğrenme yöntemi kullanılarak Microsoftun kötü amaçlı yazılım test verilerinde %99,23 oranında tespiti sağlanmıştır. Microsoftun kötü amaçlı yazılım veri setindeki zararlı yazılımın tespiti için CNN, RNN ve Dense Network modellerini birlikte kullanılmıştır. Kullanılan üç model ile verilerin farklı özelliklerinin verimli bir şekilde tespitini yapmak ve eğitim sırasında önemli özelliklerini yakalamak için kullanılmıştır. Veri setinde 9 farklı kötü amaçlı yazılım türü içermektedir.

Veri setindeki resimler CNN ve RNN ile eğitilirken diğer dosya türleri Dense Network ile eğitilmiştir (Snow 2020).

Yapılan bu çalışmada kullanıcıların kişisel bilgilerini değiştirmek, çalmak ya da bilgisayar veya cep telefonlarına sızmak için kullanılan farklı türlerdeki kötü amaçlı yazılımların incelenerek derin öğrenmeye dayalı analiz ve karşılaştırma yapılmıştır.

Çizelge 3.1 Kötü amaçlı yazılım tespitinde Sreekumari'nin çalışması (Sreekumari 2020)

Derin öğrenme Modeli	**Analiz Metodu**	**Toplam Veri seti Boyutu**	**Doğruluk**	**İşletim Sistemi**
Evrişimli Sinir Ağı (CNN)	Komut Gruplama	70000	%95	Windows
Bidirectional LSTM	Statik	21378	%97,85	Windows, Linux ve Android
Evrişimli ve Tekrarlayan Sinir Ağı (CNN ve RNN)	Statik ve Dinamik	21760	%96,3	Windows ve Linux

Sreekumari tarafından yapılan çalışmada farklı türde işletim sistemlerine ait iyi ve kötü huylu yazılım veri setlerinden zararlı yazılım tespiti yapan birçok çalışma paylaşılmıştır (Sreekumari 2020). Windows işletim sistemini içeren zararlı yazılımın derin öğrenme yöntemleriyle tespit çalışmaları **çizelge 3.1**'de paylaşılmıştır. CNN ve RNN ile birlikte 3 farklı zararlı yazılım tespitinde statik analiz yöntemiyle toplanan veriler Bidirectional LSTM yöntemi kullanılarak %97,85 doğrulukta zararlı yazılım tespiti yapılmıştır.

2021 yılında yapılan kötü amaçlı yazılımların derin öğrenme yöntemiyle tespitinde CNN mimarisi kullanılarak Malware Analysis Datasets: Top 1000 PE Imports (Angelo 2019) veri seti statik analiz yöntemiyle oluşturulmuştur. İçerisinde 47500 veri ve 999 parametre bulunan veri setinde 32*32 'lik resimlere dönüştürülerek oluşturulan CNN ve ANN mimarisiyle %98,1 oranında zararlı yazılımların tespiti sağlanmıştır (Köse ve Samet 2021). Yapılan çalışma sonrasında veri setinin Rassal Orman ve Lojistik Regresyon algoritmaları ile eğitildiğinde sırasıyla %97,36 ve %96,12 başarı oranında zararlı yazılım tespiti yapıldığı görülmektedir. Yapılan tez çalışması veri setinin zararlı yazılım tespiti

için kullanılabilir olduğunu ve parametreleri değiştirilerek oluşturulan derin öğrenme modelinin makine öğrenmesi algoritmalarıyla kıyaslandığında daha başarılı bir şekilde zararlı yazılım tespiti yapabildiği **çizelge 3.2**'de gösterilmiştir.

Çizelge 3.2 Literatür çalışmasında araştırılan çalışmalar

Çalışmalar	**Veri Seti**	**Algoritma**	**Başarı Oranı**
Köse ve Samet 2021	Malware Analysis Datasets: Top 1000 PE Imports – 47500	CNN, ANN	%98,1 -> %99,02'e çıkmıştır.
Hansen vd 2016	-	Rassal Orman	%98,1
Aslan ve Samet 2017	Kendilerinin 200	Manuel	%87,5
Mohit vd 2018	Kendilerinin ~14000	Rassal Orman DNN	%99,21
Cho 2019	Microsoft – 10868	CNN, Attention Mechanism ve ConvLSTM	%96
Irshad vd 2019	Kendilerinin – 236	Rassal Orman, NB, SVM	%86,8
Hardy 2016	Kendilerinin ~50000	ANN, SVM, NB, DT ve DL4MD	%96
Poonguzhali vd 2019	-	CNN	%99,01
Tokmak ve Küçüksille 2019	Kendilerinin - 875	DNN	%100
Kartal vd 2019	MalImg – 9339	KNN, DenseNet, VGG19 ve VGG16	%98,94
Snow 2020	Microsoft – 10868	CNN, RNN ve Dense Network	%99,23
Sreekumari 2020	21378	Bidirectional LSTM	%97,85
Liu 2019	21378	BLSTM	%97,85

Literatürde yer alan çalışmalar yukarıda ayrıntılarıyla birlikte anlatılmıştır. Yapılan çalışmalarda çoğunlukla makine öğrenmesi ve derin öğrenme yöntem ve algoritmalarını kullananlar araştırılmıştır. Yukarıdaki sonuçlar ile birlikte yapılan tez çalışmasının önemi artmaktadır. Tez çalışmasında kullanılan yöntem, uygulama ve sonuçlar sonraki bölümlerde anlatılmaktadır.

4. MATERYAL VE YÖNTEM

Bu bölümde yukarıda tanımlamaları yapılan kavramlar ve tez çalışması sırasında incelenen kötü amaçlı yazılımların analiz ve tespit yöntemleri kullanılarak önerilen yöntemin hazırlığı sağlanmıştır. Öğrenilen bilgiler neticesinde kötü amaçlı yazılımların statik ve dinamik analiz yöntemlerinde yeni çıkan ve mevcut zararlı yazılımların tespitindeki yavaşlık, tespit edememe durumları sonucunda zararlı yazılımların tespitinde derin öğrenme yöntemleri kullanılarak yeni bir model geliştirilmiştir.

Yapılan tez çalışmasında zararlı yazılım içeren ve içermeyen veri setinin derin öğrenme ve makine öğrenmesi algoritmaları ile birlikte tespit etme çalışması yapılmıştır. Önerilen modelde resim olarak kullanılan veri seti için evrişimsel sinir ağının öznitelik tespitinden faydalanılarak zararlı yazılım resminin özelliklerinin daha belirginliği sağlanmıştır. Resim üzerindeki nitelikleri daha belirgin olan zararlı yazılımlar yapay sinir ağında geliştirilen modele verilerek zararlı yazılım tespiti yapılmıştır. Bu geliştirme için derin öğrenme modeli üzerindeki parametreler ve yapay sinir ağındaki katmanlar sürekli değiştirilerek zararlı yazılım tespiti için en doğru parametreler bulunmuş ve zararlı yazılım tespitindeki başarı oranı sürekli incelenerek arttırılmıştır. Zararlı yazılım tespitini geliştirilen model %99,02 oranında başarılı olarak tespit etmektedir. Bu bölümde önerilen modelin hazırlığı, önerilen modelle ilgili akış ve mimari bilgisi, önerilen modelin detayları ve kullanılan veri seti ile ilgili çalışmalar anlatılmıştır.

4.1 Önerilen Modelin Hazırlığı

Bölüm 2.2'de anlatılan istatistiksel bilgiler ışığı ile yapılan tez çalışmasında masaüstü ve dizüstü bilgisayarların en çok kullanıldığı Windows işletim sistemlerinde geliştirilen kötü amaçlı yazılımların tespiti üzerine olmasına karar verilmiştir. Çalışmanın ilk zamanlarında kötü amaçlı yazılımların analiz ve tespit yöntemleri araştırılmıştır. Zararlı bir yazılımın Windows işletim sisteminde analizini incelemek için statik ve dinamik analiz yöntemlerinden yararlanılmıştır.

Statik analiz yöntemi için kötü amaçlı bir yazılımın uygulama çalıştırılmadan özet bilgisi elde edilerek anti virüs uygulamalarındaki tespit edilip edilmediği sağlanmıştır. Çok

bilinen zararlı yazılım için hızlı yapılan bu yöntem, yeni çıkarılan zararlı yazılımlar için tespit edilemediği yetersiz kaldığı görülmüştür. Tez çalışması kapsamında incelenen statik analiz yöntemi uygulamaları **çizelge 4.1'de** paylaşılmıştır.

Çizelge 4.1 Yapılan tez çalışmasında incelenen statik analiz yöntemleri

Hashcalculator = İncelenmek istenen uygulamanın hash değerlerini hesaplar.
md5sum, sha256sum = Hash değerleri almak için kullanılır.
BinText = Kötü Amaçlı Yazılım içerisinde geçen karakter dizileri ve karakter dizilerinin konumunu bulmamıza yarayan uygulamadır.
Floss, Strings = Çalıştırılabilir dosya içerisinde geçen karakterleri ekranda gösteren programdır.
Dependecy Walker = Windows İşletim sistemindeki çalıştırılabilir dosya formatı olan PE dosya formatı için sistemde import edilmiş .dll ve fonksiyon dosyalarını gösterir.
Detect It Easy = Kötü amaçlı yazılımın hangi programlama dili ile yazıldığını, hangi derleyici ile derlendiğini, programın boyutunu, sisteme dahil edilmiş kütüphane bilgilerini gösteren uygulamadır.
Resource Hacker = Dosyaya ait kaynak dosyaları görüntüler.
PEStudio = Kötü amaçlı yazılıma ait PE dosyalarını analiz eder ve tehlikeli gördüğü yerlerde kullanıcıyı uyarır.
IDA = Zararlı yazılıma ait kodun hata ayıklanması veya assembly için kullanılır.
Radare2 = IDA alternatifi, terminal üzerinden çalışan assembly aracıdır.
Ollydbg = Çalıştırılan uygulamaların makine dili seviyesinde hata ayıklanmasını sağlar.

Zararlı yazılım tespitinde statik analiz yöntemini kullanan uygulamalar test edilerek zararlı yazılım tespiti üzerine çalışmalar yapılmıştır. Yapılan bu çalışmada Hashcalculator, Md5sum, sha256sum uygulamaları kullanılarak zararlı yazılım uygulamasının özet bilgisi alınmış ve anti virüs sitelerinde zararlı yazılımın bulunup bulunmadığı incelenmiştir. Floss, Strings, Resource Hacker, IDA, Radare2, Ollydbg

uygulamaları kullanılarak örnek bir zararlı yazılımın kod bilgileri incelenmiştir. Detect It Easy uygulaması kullanılarak geliştirilen zararlı yazılım ile ilgili bilgiler elde edilmiştir.

Zararlı yazılımın dinamik yöntem ile analizinin yapılması için sistem üzerinde çalıştırılması gerekmektedir. Zararlı yazılımın sistem üzerinde çalıştırılması için analiz programlarının olduğu bir işletim sistemine ve bu ortamın fiziksel ortamdan izole olacak şekilde bir sanallaştırma uygulamasına ihtiyaç bulunmaktadır (Aslan ve Samet 2017). Sanal makine olarak VMware, ESXİ, KVM veya VirtualBox uygulamaları kullanılabilir. Sistem ile ilgili bütün analiz programları kurulduktan sonra zararlı yazılımın yeniden çalıştırıldığın aynı işlemleri yapmamak ve sistemin kötü amaçlı yazılımdan etkilenmediği halini görebilmek için yedeğini almak gerekmektedir.

Windows işletim sisteminde çalışan kötü amaçlı yazılımın dinamik analiz yöntemiyle tespiti için FlareVM uygulaması kurulmuştur. FlareVM Windows işletim sistemlerinde kötü amaçlı yazılımların tespitinde kullanılan uygulamaları içerisinde barındıran analiz ortamıdır. Tez çalışması kapsamında kötü amaçlı yazılım tespiti için incelenen ve kullanılan dinamik analiz yöntemi uygulamaları **çizelge 4.2**'de paylaşılmıştır (Aslan ve Samet 2017). Dinamik analiz yönteminde sürekli olarak zararlı yazılım tespiti çalıştırıldıktan sonra sistemin temiz haline geri dönüşü için geri dönüşü sağlanmıştır. Bu yöntem, yeni yada eski zararlı yazılım tespitindeki yavaşlığı ile dikkat çekmekte ve zararlı yazılımın çalıştırıldıktan sonra sistem üzerindeki etkileri tespit edilene kadar devam ettiği görülmüştür.

Çizelge 4.2 Yapılan tez çalışmasında incelenen dinamik analiz yöntemleri

apateDNS = Zararlı yazılımın yapmış olduğu DNS isteklerini 53 numaralı UDP portunu dinleyerek bilgi veren uygulamadır.
Netstat = Ağ bağlantılarını, yönlendirme tablosunu ve benzer ağ bilgilerini ayrıntılı olarak konsolda gösteren araçtır.
TcpView = Netstat programına benzer işlevde Microsoft tarafından yapılmış uygulamadır.
Netcat = Port dinlemek ya da gelen giden bağlantıyı göstermek için kullanılan araçtır.

Çizelge 4.2 Yapılan tez çalışmasında incelenen dinamik analiz yöntemleri (devam)

WireShark = Ağ üzerindeki paketleri canlı olarak yakalamaya yarayan ve ağ üzerinde derinlemesine analiz yapan açık kaynaklı paket analiz aracıdır.
RegShot = Zararlı yazılım çalıştırılmadan ve çalıştırılarak sistemde meydana gelecek farklılıkları göstermektedir.
Procmon = Registry, dosya, ağ, process ve thread analizi yaparak sistemde meydana gelen farklılıkları analiz yapmaktadır.
Process Explorer = Çalışan process'leri ve hangi klasörde çalıştığı bilgisini sunmaktadır**.**
Autoruns = İşletim sistemi başladığında sistem tarafında otomatik başlayan programlar, sisteme import edilmiş .dll dosyaları veya otomatik çalışan servisleri listelemektedir.
CaptureBat = Dosya, kayıt defteri ve process üzerinde meydana gelen değişiklikleri göstermek için kullanılmaktadır.

Zararlı yazılım tespitinde dinamik analiz yöntemini kullanan uygulamalar test edilerek zararlı yazılım tespiti üzerine çalışmalar yapılmıştır. Yapılan bu çalışmada Netstat, WireShark, TcpView, Netcat ve apateDNS uygulamaları kullanılarak zararlı yazılımın ağ üzerinde yapmış olduğu işlemler incelenmiştir. RegShot, Process Explorer, Autoruns ve CaptureBat uygulamaları kullanılarak zararlı yazılım çalıştırıldıktan sonra sistem üzerinde oluşturmuş olduğu etkiler gözlemlenmiştir. Yukarıdaki uygulamalar ile yapılan çalışmalar neticesinde zararlı yazılımın sistem üzerinde göstermiş olduğu davranışlar incelenmiş ve sistem üzerindeki etkileri gözlemlenmiştir.

Literatür çalışmalarına Windows işletim sistemi üzerine yapılan zararlı yazılım tespit çalışmaları paylaşılmıştır. Yapılan araştırmalar, uygulamalar ve literatür çalışmaları neticesinde kötü amaçlı yazılımların tespitinde statik ve dinamik yöntemlerin yetersiz kaldığı gözlemlenmiştir. Tez çalışmasında zararlı yazılımların tespiti için derin öğrenme yöntemlerinin kullanılmasına karar verilmiştir. Derin öğrenme yöntemlerinden CNN ve ANN literatürdeki diğer çalışmalarla kıyaslamak ve günümüzde etiketli veri için en çok tercih edilen derin öğrenme yöntemleri olduğu için seçilmiştir. Konuyla ilgili derin öğrenme ve makine öğrenmesi algoritmaları incelenmiştir. Derin öğrenme ve makine

öğrenmesi algoritmaları incelenerek önerilecek modelin mimarisi ve akış şeması **Bölüm 4.2**'de paylaşılmıştır.

4.2 Önerilen Modelin Mimarisi ve Akış Şeması

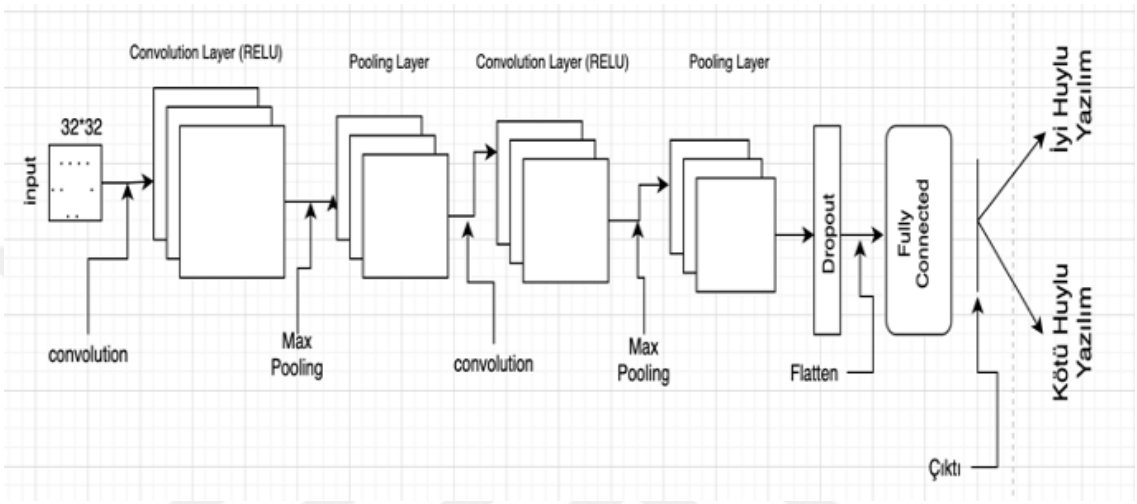
Derin öğrenme ile kötü amaçlı yazılımın tespitinde modele giriş verisi olarak resim verilmesi gerekmektedir. Veri setinde önerilen modelin mimarisinde veri setleri 32*32 'lik resim formatına dönüştürülmüştür. **Bölüm 5.1**'de veri seti ile ilgili bilgi verilmektedir.

Evrişimli Sinir Ağı içerisinde konvülüsyon, havuzlama katmanlarıyla resmin özniteliklerinin çıkarıldığı ve sonrasında yapay sinir ağları ile eğitildiği yöntemdir (You 2018). İmagenet yarışmasında ortaya çıkan bu modelde resmin boyutu, konvülüsyon ve havuzlama katmanında kullanılacak matematiksel fonksiyonlar, yapay sinir ağında kullanılacak katman sayısı ve parametreler net bir şekilde ortaya konmamıştır. Araştırılan çalışmalarda fazla parametreyle zararlı yazılım tespitine rastlanmamıştır. Önerilen yöntem için kullanılan sistemde resimlerin 32*32 boyutunda olması daha az özellikli resimlere oranla daha fazla bilgi içerdiğinden önerilmektedir. Daha fazla boyuttaki resimlerin oluşması zararlı yazılım tespiti için doğruluk oranını düşürebilir ve hangi parametrenin modelin eğitimde sıkıntılar yarattığının tespiti kolay olmayacaktır.

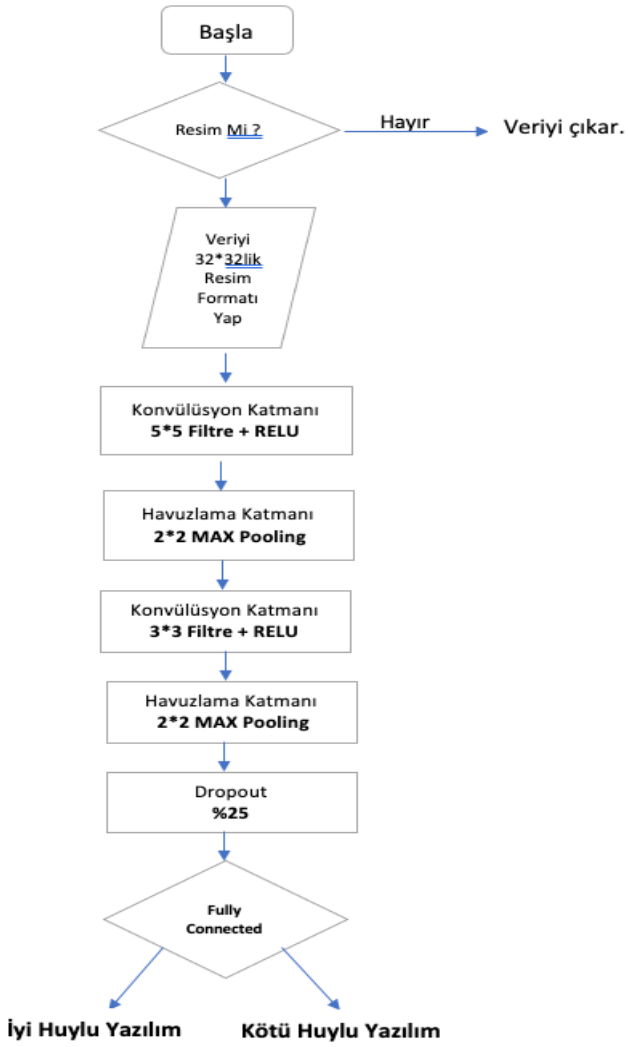
Model oluşturulurken çok farklı teknikler ile çok sayıda eğitim denenmiştir. Kullanılan tekniklerde evrişimli sinir ağında kullanılan konvolüsyon katmanı ve havuzlama katmanının 2 kez art arda resim üzerinde işlem yapması bu sayının artması ya da azalması durumunda zararlı yazılım tespitini düşürmektedir. Önerilen modeldeki konvolüsyon katmanı en başarılı için filtreler 1. Katmanda 5*5'lik 8 filtre ve RELU aktivasyon fonksiyonu ve 2. Katmanda 3*3'lük 16 filtre ve RELU fonksiyonuyla olduğu görülmüştür. Havuzlama katmanı için en başarılı yöntemler 2*2'lik filtreler ile sağlanmıştır. Bu katmanlarda dropout %25 kullanılarak her bir işlemde azalma sağlanmış ve modelin zararlı yazılım tespiti hızlanmıştır.

Önerilen mimari modelde Yapay sinir ağı olarak 256 sinir hücreli giriş katmanı ve katman sayısına göre değişiklik göstermek olan 128, 64 ve 32 nöronluk gizli katmanlı bir yapı

kurulmuştur. Zararlı yazılım tespitinin bu parametrelerin artış ya da azalışı durumunda düştüğü görülmektedir. Önerilen modelde 150 adımlık işlem yapılmış ve artışı durumunda modelin aynı sonucu vererek aşırı öğrendiği, azaltıldığında ise yüksek orandaki başarı durumuna çıkamadığı gözlemlenmiştir. Kötü amaçlı yazılım tespiti için önerilen modele ait mimari ve akış şeması **şekil 4.1 ve şekil 4.2**'de paylaşılmıştır.



Şekil 4.1 KAY tespitinde önerilen model mimarisi



Şekil 4.2 KAY tespitinde önerilen modelin akış şeması

Önerilen modelde giriş verisi olarak 32*32 boyutunda resim bilgileri giriş verisi olarak eklenmiştir. Giriş verisinden sonra sırasıyla konvülüsyon ve havuzlama katmanlarına 2 kez işlenmiştir. Flatten yöntemi kullanılarak sinir ağına giriş verisinin dönüşümü sağlanmıştır. Sinir ağı içerisinde katman sayıları ve yöntemler üzerinde işlemler yapılarak eğitim sağlanmıştır. Çıktı olarak zararlı yazılımın iyi huylu ya da kötü huylu olduğu tespit edilmiştir.

5. ÖNERİLEN MODELİN UYGULANMASI

Bu bölümde uygulanan modelin veri seti tanıtımı, veri seti ile ilgili bilgiler, önerilen modelin uygulama platformu ve kullanılan kütüphaneler, önerilen modelin başarısının ölçümlenmesi için kullanılan değerlendirme metrikleri, önerilen model için kullanılan matematiksel fonksiyonlarının tercih nedenleri, önerilen modelin kıyaslanması açısından Rassal Orman ve Lojistik Regresyon algoritmalarıyla ilgili bilgi ve yöntem anlatılmaktadır.

5.1 Önerilen Model İçin Seçilen Veri Setinin Tanıtılması

Zararlı yazılım tespitinde veri seti derin öğrenme modeli için 32*32'lik bir resim üzerine işlem yapılması gerektiğinden 1024 özelliğe ihtiyaç bulunmaktadır. Yapılan araştırmalar neticesinde Windows işletim sisteminde PE formatında veri seti sunan Malware Analysis Datasets: TOP-1000 PE IMPORTS veri setinde yaklaşık olarak 1000'e yakın özellik ile virusshare.com üzerinden indirilmiş zararlı ve iyi huylu yazılımların özellikleri çıkarılmıştır. Bu veriler statik analiz yöntemi kullanılarak çıkarılmıştır.

Veri setinde toplamda 47500 veri ve 999 parametresi bulunmaktadır. 32*32'lik bir resme dönüştürülmesi için 1024 özelliğe ihtiyaç duyulacağı için, veri setinde her bir resim için başına 25 tane 0 eklenmiştir.

Veri setinde Windows işletim sisteminde normal zamanda ya da zararlı yazılım bulunduğu zamandaki kayıt defterindeki durumlar, sistemin durumları gibi bilgileri içermektedir. Veri setinde bulunan bilgilerin bir kısmı **çizelge 5.1**'de paylaşılmıştır.

Çizelge 5.1 Veri setinin örnek alanları

Hash	getProcAdress	ExitProcess	WriteFile	Close Handle	Free Library	Sleep
396AC2A33748BB784077F411	0	1	0	1	0	1
619DA0D78EBD8E0F56B7D0	0	0	0	1	1	0
4347A75A24D60D8D83A3F40	1	0	1	1	0	1
4506D44E5E7D91211A68EBF	1	1	1	0	1	0
4297F44B139552355B2497399	1	0	0	0	1	0
D435E9E84CDE79834785EEF	0	1	0	0	0	1
335EF23ASD8E16C1FDBCC7	1	0	0	1	1	0

Yapılan tez çalışmasında kullanılan veri setinde zararlı yazılımın etiket bilgisi, zararlı yazılım tespit edildiği durumda işletim sistemi üzerindeki etkileri statik analiz yöntemi kullanılarak oluşturulmuştur. Zararlı yazılımın tespit edildiği andaki sistem metrikleri parametre olarak veri setinin giriş verisini oluşturmaktadır.

5.2 Uygulama Platformu

Uygulama platformu oluşturulurken seçilen bilgisayar sistemi ve yazılım platformu aşağıdaki gibidir.

Geliştirilen model boyunca kullanılan bilgisayarın özelliği aşağıdaki gibidir.

- Intel Core i7 4700HQ İşlemci
- 2.4 Ghz İşlemci Hızı
- 16 GB Ram
- NVIDIA GTX850M 4 GB Ekran Kartı

Geliştirilen model boyunca kullanılan uygulamalar aşağıdaki gibidir.

- Python Programlama dili
- Numpy Kütüphanesi

- Pandas Kütüphanesi
- Matplotlib Kütüphanesi
- Kaggle
- Spider IDE
- Google Colab
- Windows İşletim Sistemi
- Keras
- Tensorflow

Python, nesne tabanlı, yüksek seviyeli programlama dillerinden biridir (McKinney 2012). Özellikle diğer python kütüphaneleri de eklendiğinde veri işlemedeki yeteneği, nesne tabanlı olması, basit bir söz dizimine sahip olması özelliklerinden yapılan tez çalışmasında programlama dili olarak seçilmiştir.

Python programlama dili kütüphanelerle birlikte veriler üzerinde işlem yapma özelliği çok büyük hız ve önem kazanmıştır. Veri üzerinde işlem yapan ve tez çalışmasında da kullanılan 2 kütüphane bulunmaktadır. Bu kütüphaneler Pandas ve Numpy kütüphaneleridir. Numpy, Python programlama diline bilimsel hesaplama özelliği kazandırmış, N-Diziler üzerinde güçlü matematiksel işlemler yapabilen kütüphanedir (Anonymous 2022). Numpy kütüphanesi kullanılarak uzun ve karmaşık seviyede yazılacak kodlar kısa ve basit bir şekilde yazılabilmektedir. Pandas kütüphanesi ise veri yapıları ve veri analizi üzerine geliştirilmiş hızlı ve yüksek performansta çalışan kütüphanedir (Anonymous 2022). Pandas kütüphanesi özellike CSV, resim ya da dosyalar üzerinde çok hızlı bir şekilde veri işleme, temizleme işlemleri yapabilmektedir. Matplotlib kütüphanesi ise 2D boyutunda kullanıcılara verilerle yapılan işlemleri görsel formatta sunan bir kütüphanedir (Anonymous 2022). İçerisinde bulunan hata, çubuk ve çizgi grafikleri, histogramlar ile görsel çizimler oluşturulabilmektedir. Tensorflow, Google tarafından geliştirilen makine öğrenimi ve derin öğrenme işlemlerinde açık kaynaklı olarak sunulan ve arkasında güçlü bir ekip ve destekleyici bulunan kütüphanedir (Anonymous 2022). Tensorflow tek bir API ile platform bağımsız CPU ve GPU üzerine uygulamaları çalıştırmaya yarar. Keras kütüphanesi de Python dilinde yazılmış, Derin

öğrenme yöntemlerinin hızlı ve pratik olarak uygulanmasını sağlayan, CPU ve GPU üzerinde sorunsuz çalışan kütüphanedir (Anonymous 2022).

5.3 Önerilen Modelin Değerlendirme Metrikleri

Sınıflandırma algoritmalarında sınıflandırma performans ölçümünün yapılması ve diğer algoritmalarla performans karşılaştırılabilmesi için başarım ölçüm metrikleri kullanılır. Önerilen modelin başarılı olup olmadığına karar vermek için bu metrikler hesaplanarak yorumlanır. Bu metriklerin hesaplanmasında karmaşıklık matrisinden yararlanılır. Karmaşıklık matrisi (Confusion Matrix), yapılan tahminleri gerçek değerleri ile karşılaştırarak tahminlerde bulunan tablodur (Türker 2019). Karmaşıklık matrisinin örnek tablosu **çizelge 5.2**'de paylaşılmıştır.

Çizelge 5.2 Karmaşıklık matris tablosu

	Gerçek Değer		
Tahmin Edilen Değer	Sonuç	Pozitif (1)	Negatif (0)
	Pozitif (1)	TP	FP
	Negatif (0)	FN	TN

Matriste 4 durum bulunmaktadır. Bu durumlar doğru pozitif (TP), yanlış pozitif (FP), yanlış negatif (FN) ve doğru negatif (TN)'tir (BrownLee 2020). Doğru pozitifler zararlı yazılım olan örneklerin zararlı yazılım olarak tahmin edildiğini, doğru negatifler zararlı yazılım olmayan örneklerin zararlı yazılım olmadığını göstermekte, yanlış pozitifler zararlı yazılım olmayan etiketine sahip verilerin zararlı yazılım olarak tahmin edildiğini ve yanlış negatifler zararlı yazılım olan örneklerin zararlı yazılım olmadığının tahmin edildiği durumları göstermektedir. Bu değerler sonucunda aşağıdaki metrik hesaplamaları yapılmaktadır.

- **Doğruluk (Accuracy):** Doğru tahmin edilen örnek sayısının, toplam örnek sayısına oranıdır.

$$Doğruluk\ (Accuracy) = \frac{(TP + TN)}{(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})}$$

- **Duyarlılık (Recall):** Zararlı yazılım olarak tahmin edilen ve zararlı yazılım olan örnek sayısının, gerçekten zararlı yazılım olan tüm örneklere oranıdır.

$$Duyarlılık\ (Recall) = \frac{TP}{(\mathrm{TP} + \mathrm{FN})}$$

- **Hassasiyet (Precision):** Zararlı yazılım olarak tahmin edilen ve zararlı yazılım olan örnek sayısının, sistemin zararlı yazılım olarak tahmin ettiği örneklere oranıdır.

$$Hassasiyet\ (Precision) = \frac{TP}{(\mathrm{TP} + \mathrm{FP})}$$

- **F1 Skoru:** Hassasiyet ve duyarlılık değerlerinin harmonik ortalaması alınarak hesaplanmaktadır. Yanlış pozitif ve yanlış negatif değerlerini hesaba kattığı için, doğruluk değerine göre daha faydalı bir metrik olduğu söylenebilir (Türker 2019).

$$F1\ Skoru = \frac{2 * Hassasiyet * Duyarlılık}{\mathrm{Hassasiyet} + \mathrm{Duyarlılık}}$$

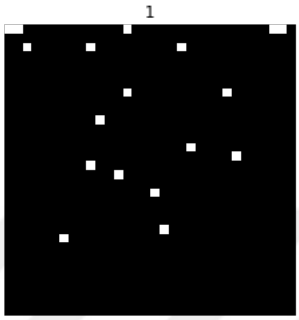
Alıcı İşletim Karakteristiği (ROC) Eğrisi: Alıcı İşletim Karakteristiği (ROC) eğrisi, geliştirilen modelin tüm olası değerlerini göstererek performansını özetleyen grafiktir. ROC eğrisi tahmin edilen gerçek değerlerin, yanlış tahmin edilen pozitif değerlere oranı ile çizilmektedir. Bir ROC eğrisi, farklı eşik değerleri için dikey eksen üzerinde doğru pozitiflerin tüm pozitiflere oranlarının, yatay eksen üzerinde ise yanlış pozitiflerin tüm negatiflere oranlarının yer aldığı bir eğridir. Doğru pozitif oranının yüksek, yanlış pozitif oranının düşük olması modelin başarılı olduğunu gösterir. Böyle bir durumda eğri sol üst köşeye doğru yaklaşmaktadır.

5.4 Önerilen Modelin Uygulanması

Yapılan tez çalışmasında kötü amaçlı yazılımların tespiti için geliştirilen mimari modelin nasıl uygulandığı burada paylaşılmıştır. Geliştirilecek model için 47580 veri'den 33306

tanesi eğitim, 14274 tanesi ise eğitilen verinin doğruluğunu tespit etmek için kullanılmıştır.

Kötü Amaçlı Yazılım tespitinde Derin öğrenme yönteminin CNN mimarisi kullanılmıştır. Veri seti ilk başta 32*32'lik bir resim haline getirilmiştir. Zararlı Yazılım'ın resim olarak dönüştürülmüş hali **şekil 5.1**'de paylaşılmıştır.



Şekil 5.1 KAY resim olarak gösterilmesi

Resim haline dönüştürülen ZY'lar için veri setinde veri ön işleme kuralları uygulanmıştır. Veri ön işleme verilerin temizlenmesi, anlamsız verilerin çıkarılması ya da verilerle ilgili eksikliklerin tamamlanması için kullanılan tekniktir. Yapılan tez çalışmasında kullanılan veri setinde Windows işletim sistemleriyle ilgili bilgiler ve özellikler 999 parametreden oluşmuştur. Her bir parametre özelinde anlamsız ya da tutarsız veriler temizlenmiştir. 32*32'lik bir resim haline getirilmesi için 25 adet önemsiz değer içeren parametrelerle veri desteklenmiştir. Veri tutarlılığında boş, hatalı ya da gürültülü veriler tespit edilerek giderilmiştir.

Veriler üzerinde yapılan veri ön işleme işlemleri sonrasında Evrişimsel sinir ağının giriş verileri elde edilmiştir. CNN mimarisi giriş verisi olarak 32*32 resim haline getirilmiş kötü amaçlı yazılımlar verilmiştir. Elde edilen veriler ile birlikte CNN'de 2 katmanlı özellik çıkarımı ve fully connected katmanında 2, 5 ve 7 katmanlı bir sinir ağı kullanılmıştır. İlk aşamada konvolüsyon katmanında 5*5'lik filtre ile birlikte RELU aktivasyon fonksiyonu kullanılarak işlemler yapılmıştır. RELU Aktivasyon fonksiyonu yapılan hesaplamalarda negatif değerleri sıfıra, pozitif değerleri ise mevcut sayıyı koruyarak kendisiyle eşleştirmektedir. Havuzlama Katmanında 2*2 adımlar halinde Max pooling yöntemi kullanılmıştır.

2. katmanda 3*3 filtre ile birlikte RELU aktivasyon fonksiyonu kullanılmıştır. 2*2 adım ile En büyük veri bulunan havuzlama yöntemi kullanılmıştır. Her bir katmanda %25 oranında Dropout kullanılmıştır. Dropout aşırı öğrenmeyi (overfitting) önlemek amacıyla eğitim süresinde bazı nöronları devre dışı bırakmıştır.

Fully connected katmanda ise flatten ile 1024,1'lik resimlerle sinir ağı katmanına hazır hale gelmiştir. 256 ve 64'lük nöronlara indirgenerek son aşamada 2, 5 ve 7 katman ile kötü amaçlı yazılım tespit edilmiştir. Aktivasyon fonksiyonu RELU ve son aşamada iyi ve kötü huylu yazılımı tespit etmek amacıyla sigmoid aktivasyon fonksiyonu kullanılmıştır. Model içerisinde optimizasyon fonksiyonu olarak ADAM algoritması ve loss fonksiyonu için de binary cross entropy kullanılmıştır.

Bu çalışma için 16 GB RAM, 4 GB ekran kartı (Nvidia GTX850M), 2.4 ghz hızına sahip i7 4700HQ işlemci olan bir bilgisayar ve google colab ile kaggle'ın sağlamış olduğu bilgisayarlar kullanılmıştır. Test süresi 47500 veri için geliştirilen mimari de 10 adımlı mimari için 2 katmanlı 1 dakika, 5 katmanlı 3 dakika ve 7 katmanlı için 8 dakika, 30 adımlı mimari için 2 katmanlı için 3 dakika, 5 katmanlı için 15 dakika ve 7 katmanlı mimari için 45 dakika, 150 adımlı mimari için 2 katmanlı 10 dakika, 5 katmanlı için 1,5 saat ve 7 katmanlı mimari için 2,5 saat sürmüştür.

47500 veri setinde %70 oranında veri eğitim amaçlı kullanılmıştır. Eğitim süreci sonrasında ise %30 oranında test seti doğrulamadan geçirilmiştir. 10 adımlı 2 katmanlı mimari için %96,4, 5 katmanlı için %96,8 ve 7 katmanlı mimari için %97,30 adımlı ve 2 Katmanlı mimari için %98,1, 5 katmanlı mimari için %98,5, 7 katmanlı mimari için %98,4 ve 150 adımlı 2 katmanlı mimari için %98,5, 5 katmanlı %99,02 ve 7 katmanlı için %98,7 oranında zararlı yazılımların tespiti sağlanmıştır. Yapılan geliştirmede adım sayısı 150 adımdan sonra aşırı öğrenmeye girmekte ve modelin veri setinden başarılı tahmin etme oranında değişiklik olmamaktadır. Eğitim süresince önerilen mimari için elde edilen tüm sonuçlar **çizelge 5.3**'de görülmektedir.

Çizelge 5.3 Önerilen modelin katman ve adımlara göre doğruluk oranları

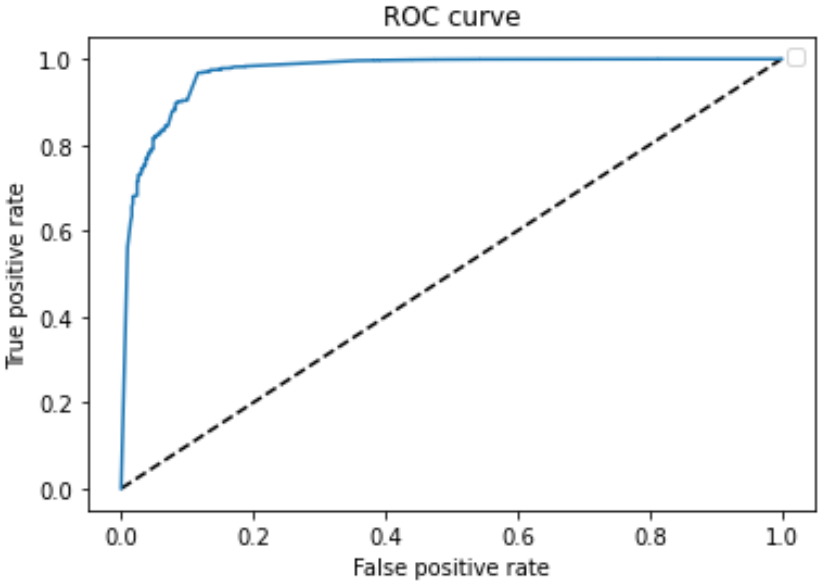
	10 adım	**30 adım**	**150 adım**
ANN 2 Katmanlı Model	%96,4	%98,1	%98,5
ANN 5 Katmanlı Model	%96,8	%98,5	**%99,02**
ANN 7 Katmanlı Model	%97,30	%98,4	%98,7

Modelde katman sayısı CNN kısmında 2 katmanlı ve ANN kısmında 2, 5 ve 7 katmanlı olacak şekilde eğitim ve test süreci olmuştur. 2 katmanlı mimaride eğitim ve test süresi toplamda 5 dakika olurken, 5 katmanlı mimari için 10 adımda 10 dakika, 30 adımda süreç 20 ile 35 dakika arasında ve 150 adımda 2 saate yakın bir zaman almaktadır. 7 katmanlı mimari için 2 katmanlı mimari 20 dakikaya yakın, 30 adımda 2 saate yakın ve 150 adımda 6 saate yakın eğitim ve test süreci olmuştur. 150 adımdan sonra modelde ezbere başladığı görüldüğü ve öğrenme oranında değişime rastlanmadığı için adım sayısında artırım yapılmamıştır. Önerilen mimari model içerisinde 150 adımlı 5 katmanlı mimari en yüksek oranda tahminde bulunan modeldir. Önerilen mimari modelin karmaşıklık matrisi **çizelge 5.4'**te paylaşılmıştır.

Çizelge 5.4 Önerilen modelin karmaşıklık matris tablosu

	Gerçek Değer		
Tahmin Edilen Değer	Sonuç	Negatif (0)	Pozitif (1)
	Negatif (0)	474	122
	Pozitif (0)	22	13660

Önerilen bu modele ait ROC eğrisi **şekil 5.2**'de gösterilmiştir.



Şekil 5.2 Önerilen modelin ROC eğrisi

Önerilen modelde veri setinin çoğunluğu zararlı yazılım içerdiği ve dağınık olmadığı için hassasiyet, duyarlılık ve f1 skorlarına bakılarak modelin başarısı değerlendirilmiştir. Duyarlılık oranı 0,9911, hassasiyet oranı 0,9983 ve F1 skoru ise 0,9946 olarak çıkmıştır. Önerilen model için kullanılan ROC eğrisinde, modelin kötü amaçlı olan yazılımları yüksek doğrulukta tespit ettiği görülmüştür. Önerilen model Rassal Orman ve Lojistik Regresyon yöntemleri ile kıyaslandığında, önerilen modelin f1 skoru ve zararlı yazılım tespit oranının daha başarılı olduğu gözlemlenmiştir.

5.5 Önerilen Modelde Kullanılan Fonksiyonların Seçilmesi

Yapılan tez çalışmasında önerilen model için aktivasyon, optimizasyon ve kayıp fonksiyonları ile çalışılmıştır. Aktivasyon fonksiyonu için RELU ve sigmoid, optimizasyon fonksiyonu için adam ve kayıp fonksiyonu için binary cross entropy fonksiyonları tercih edilmiştir.

Aktivasyon fonksiyonunun çıktısı sıfır ise geri yayılım algoritmasında ağırlıklar kaybolmaktadır. Bu problem aktivasyon fonksiyonlarından tanh ve sigmoid fonksiyonlarında görülmekte olup, kaybolan gradyan (Vanishing Gradients) problemi olarak isimlendirilmektedir (Hochreiter 1998). Sigmoid fonksiyonu çıktı katmanının iyi ya da kötü huylu zararlı yazılım sonucunu vereceği için kullanılmıştır. CNN katmanları

ve ANN katmanlarında güncel olan RELU aktivasyon fonksiyonları kullanılarak modelin eğitilmesi sağlanmıştır (Ramachandran vd 2017). Yapılan tez çalışmasında, RELU fonksiyonunun 0 değerleri dying RELU problemine sebep olabileceği için leakly RELU fonksiyonu kullanılarak RELU fonksiyonu ile karşılaştırmaları yapılmıştır (Lu 2019). Veri seti içeriğinin 0 ve 1 değerlerinden oluşmasından dolayı RELU, leakly RELU'ya göre daha başarılı olmuştur.

Optimizasyon fonksiyonları içerisinde Adam algoritması, RMSprob ve momentum yöntemlerinin avantajlı yönlerinin birleşimi olarak karşımıza çıkmaktadır (Seyyarar vd 2020). Diğer optimizasyon algoritmalarındaki eksikliklerinde giderilerek daha güncel olan adam algoritması tez çalışmasında tercih edilmiştir.

Kayıp fonksiyonları yapay sinir ağı için geliştirilen modellerin veri seti üzerindeki başarısının ölçümünü yapmaktadır. Yapılan tez çalışmasında iyi ve kötü huylu yazılım tespiti yapılacağı ve çıktının 0 veya 1 sonucu olacağı için binary cross entrpy fonksiyonu tez çalışmasında kullanılmıştır. **Çizelge 5.5**'te farklı aktivasyon, optimizasyon ve kayıp fonksiyonları kullanılarak zararlı yazılım tespit çalışmaları yapılmış ve tabloda sonuçlar paylaşılmıştır.

Çizelge 5.5 Aktivasyon fonksiyonlarının karşılaştırılması

ANN Katman Sayısı	Aktivasyon Fonksiyonu	Kayıp Fonksiyonu	Optimizasyon Fonksiyonu	Adım Sayısı	Başarı Oranı
5	RELU	Binary Cross Entropy	Adam	30	%98,5
5	TANH	Binary Cross Entropy	Adam	30	%97,4
5	Leakly RELU	Binary Cross Entropy	Adam	30	%98,1
5	RELU	Binary Cross Entropy	Adadelta	30	%96,07

Çizelge 5.5 Aktivasyon fonksiyonlarının karşılaştırılması (devam)

5	RELU	Binary Cross Entropy	Adam	150	%99,02
5	Leakly RELU	Binary Cross Entropy	Adam	150	%98,5

Bu tez çalışmasında kullanılan veri seti üzerinde farklı aktivasyon, optimizasyon ve kayıp fonksiyonları kullanılarak karşılaştırma yapılmıştır. Kullanılan algoritmalardan 30 adım sayısı ile karşılaştırma yapıldığında aktivasyon fonksiyonlarından RELU, kayıp fonksiyonlarından binary cross entropy ve optimizasyon fonksiyonundan Adam en yüksek doğrulukta zararlı yazılım tespiti yapmıştır. RELU ve leakly RELU aktivasyon fonksiyonlarını 150 adımlı öğrenme gerçekleştirildiğinde RELU'nun leakly RELU'ya göre daha başarılı olduğu görülmüştür. Bu durumun veri setinin 0 ve 1 değerlerini içermesi ve RELU aktivasyon fonksiyonu için daha uygun bir veri seti ile çalışıldığından leakly RELU aktivasyon fonksiyonuna göre daha yüksek çıktığı düşünülmektedir.

5.6 Önerilen Modelin Rassal Orman Modeli ve Logistik Regression Algoritmalarıyla karşılaştırılması

Derin öğrenme Makine öğrenmesinin bir alt dalıdır. Makine öğrenmesi insanlar gibi davranıp, insanların öğrenme yeteneklerini anlamaya ve uygulamaya çalışan bilgisayar bilimidir. Makine öğrenmesi veriler ile birlikte öğrenerek karar verme mekanizması geliştirmektedir. Bu karar verme mekanizmasını geliştirirken içerisinde çok fazla sayıda algoritma, yöntem ve matematiksel formül barındırmaktadır. Rassal Orman ve Logistik Regresyon makine öğrenmesi algoritmalarından en popüler olanlarındandır.

Rassal Orman Algoritması veri ile ilgili sınıflandırma yapan, regresyon problemlerine uygulanabilen bir makine öğrenmesi algoritmasıdır. Rassal Orman Algoritmasında birden fazla karar ağacı kullanılarak model üzerinde sınıflandırma işlemi yapılmaktadır. Karar ağaçlarının en büyük dezavantajı aşırı öğrenme(overfitting)'tir. Rassal Orman karar ağlaçlarından farklı olarak veri seti üzerinde farklı alt veri setleri oluştabilmekte ve bunları eğitebilmektedir. Bu özelliği aşırı öğrenmeyi oldukça aza düşürmektedir. Yapılan

tez aşamasında önerilen CNN modelinin yanında veri seti Rassal Orman algoritması ile de eğitilmiştir.

Rassal Orman Algoritması ile eğitilen veri setinin %96 oranında başarılı olmuştur. Önerdiğimiz model ile karşılaştırıldığında yaklaşık %3 başarı oranında yanlış tahminleme yapmıştır.

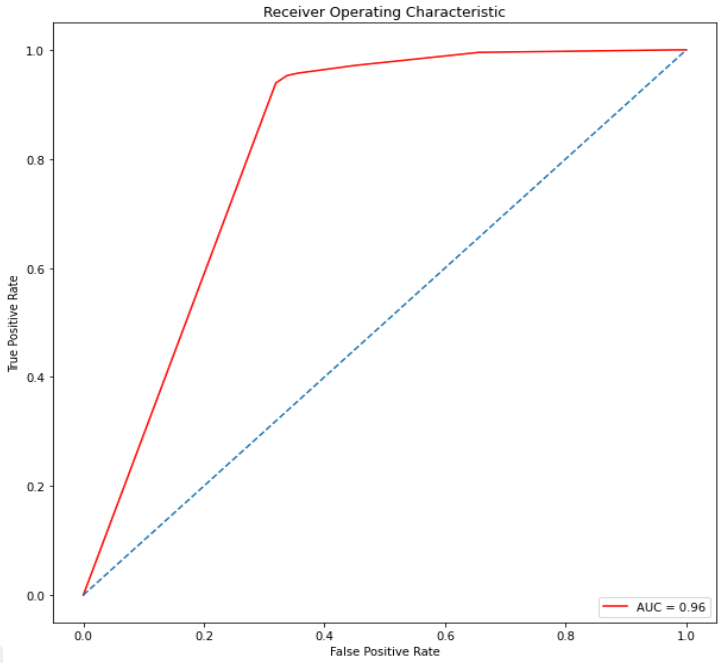
Rassal Orman Algoritması için karmaşıklık matrisi **çizelge 5.6**'da gösterilmiştir.

Çizelge 5.6 Rassal Orman Algoritması için karmaşıklık matris tablosu

	Gerçek Değer		
Tahmin Edilen Değer	Sonuç	Negatif (0)	Pozitif (1)
	Negatif (0)	67	553
	Pozitif (1)	1	13653

Eğitilen modelde 13654 kötü amaçlı yazılım olan test verisinde Rassal Orman Algoritması yalnızca 1 tanesi için hatalı tahmin etmiş diğerlerini ise doğru tahmin etmiştir. Zararlı yazılım olmayanlar tahmin edildiğinde ise 620 veriden 67 tanesi doğru tahmin edilmiş, 553 tanesini algoritma zararlı yazılım olmadığı halde zararlı yazılım olarak değerlendirmiştir.

Rassal Orman algoritması için ROC eğrisi **şekil 5.3**'te gösterilmiştir.



Şekil 5.3 Rassal Orman algoritması için ROC eğrisi

Önerilen modelde veri setinin çoğunluğu zararlı yazılım içerdiği ve dağınık olmadığı için hassasiyet, duyarlılık ve f1 skorlarına bakılarak modelin başarısı değerlendirilmiştir. Duyarlılık oranı 0,9951, hassasiyet oranı 0,9598 ve F1 skoru ise 0,9771 olarak çıkmıştır. Rassal Orman algoritması için kullanılan ROC eğrisinde, zararlı yazılım tespitinde yüksek doğrulukta başarılı olduğu görülmektedir. Derin öğrenme modeline kıyasla başarı oranı daha düşüktür. Rassal Orman algoritmasının önerilen yöntem ve Lojistik Regresyon ile kıyaslandığında f1 skorunun ve başarı oranının diğer çalışmalara kıyasla daha düşük olduğu gözlemlenmiştir.

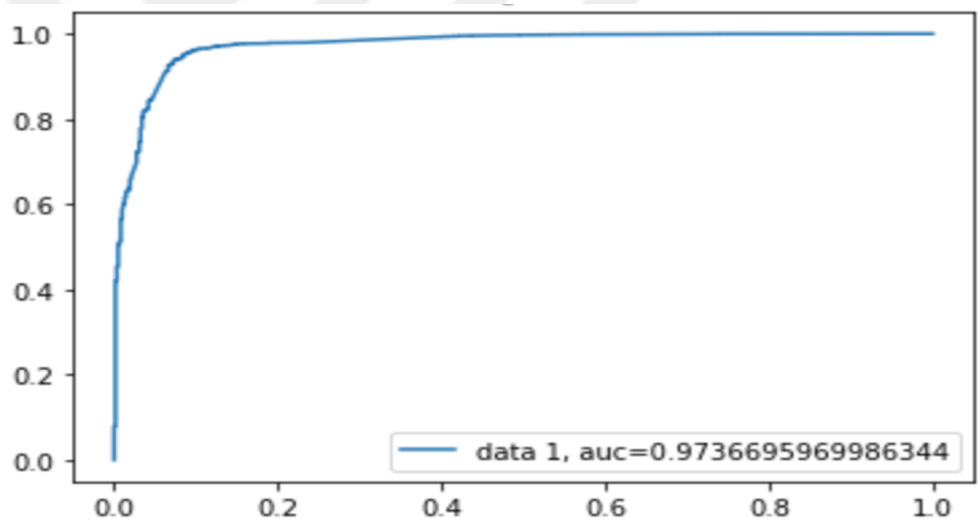
Lojistik Regresyon veri üzerinde bir sonucun birden fazla değişkene bağlı olarak veri kümesini analiz etmek için kullanılan sınıflandırma yöntemdir. Lojistik Regresyon 0 ile 1 arasındakı verileri sınıflandırmak için kullanılır. Veri setimiz bu duruma uygun olduğu için Lojistik Regresyon yöntemiyle de verimiz eğitilmiştir. Eğitim sonunda %97,36 oranında başarı oranı elde edilmiştir. Lojistik Regresyon için karmaşıklık matrisi **çizelge 5.7**'de gösterilmiştir.

Çizelge 5.7 Lojistik Regresyon için karmaşıklık matrisi

	Gerçek Değer		
Tahmin Edilen Değer	Sonuç	Negatif (0)	Pozitif (1)
	Negatif (0)	358	262
	Pozitif (1)	70	13584

Eğitilen modelde 13584 kötü amaçlı yazılım olan test verisinde 70 tanesi için hatalı tahmin etmiş diğerlerini ise doğru tahmin etmiştir. Zararlı yazılım olmayanlar tahmin edildiğinde ise 620 veriden 358 tanesi doğru tahmin edilmiş, 262 tanesini algoritma zararlı yazılım olmadığı halde zararlı yazılım olarak değerlendirmiştir.

Lojistik Regresyon yöntemi için ROC eğrisi **şekil 5.4**'te gösterilmiştir.



Şekil 5.4 Lojistik Regresyon için ROC eğrisi

Önerilen modelde veri setinin çoğunluğu zararlı yazılım içerdiği ve dağınık olmadığı için hassasiyet, duyarlılık ve f1 skorlarına bakılarak modelin başarısı değerlendirilmiştir. Duyarlılık oranı 0,9743, hassasiyet oranı 0,9810 ve F1 skoru ise 0,9776 olarak çıkmıştır. Lojistik Regresyon algoritması için kullanılan ROC eğrisinde, zararlı yazılım tespitinde yüksek doğrulukta başarılı olduğu görülmektedir. Rassal Orman algoritmasına oranla daha başarılı fakat önerilen derin öğrenme modeline kıyasla daha az başarılı olduğu görülmektedir. Lojistik Regresyon algoritmasının F1 skoru olarak karşılaştırıldığında

önerilen modele göre düşük, Rassal Orman algoritmasına oranla daha iyi oranda zararlı yazılım tespit edildiği görülmüştür.

Çizelge 5.8 Kötü amaçlı yazılım tespitinde geliştirilen modellerin karşılaştırılması

	10 adım	**30 adım**	**100 adım**	**150 adım**
CNN 2 Katmanlı Mimari	%96,4	%98,1	-	%98,5
CNN 5 Katmanlı Mimari	%96,8	%98,5	**-**	**%99,02**
CNN 7 Katmanlı Mimari	%97, 30	%98,4	**-**	%98,7
ANN 2 Katmanlı	%96,2	%96,5	-	%97.3
ANN 5 Katmanlı	%96,4	%96,8	-	%97.8
Lojistik Regresyon	-	-	%97,36	-
Rassal Orman	-	-	%96,12	

Yapılan çalışma neticesinde makine öğrenmesi algoritmaları ve derin öğrenme yönteminde özelleştirilen model ile birlikte elde edilen sonuçlar **çizelge 5.5**'te paylaşılmıştır. Makine öğrenmesi algoritmalarına oranla daha yüksek oranda Evrişimli Sinir Ağının 5 katmanlı mimaride zararlı yazılım tespiti yapıldığı görülmüştür. Sonuçlar ve tartışma bölümünde yapılan çalışma ile ilgili bilgiler anlatılmıştır.

6. SONUÇ VE TARTIŞMA

İnternet, akıllı cihaz ve bilgisayar kullanımının artışıyla birlikte teknoloji cihazları ve verinin siber güvenliği önemli bir alan olmuştur. Yapılan araştırmalar neticesinde siber saldırılarda kötü amaçlı yazılım kullanımı büyük çoğunluğu oluşturmaktadır. Geniş çapta ve küresel boyutta geliştirilen kötü amaçlı yazılımlar kullanıcıların sistemlerine zarar vermek, veri ve bilgilerine erişimlerini kısıtlamak amacıyla kullanılmakta olup sürekli olarak güncellenmektedir. Zararlı yazılımlar yüzünden kullanıcılar zaman ve maliyet açısından zararlar görmekte, ülke ekonomilerine ciddi anlamda zararları olmaktadır. Yukarıdaki sebeplerden dolayı kötü amaçlı yazılım tespiti önemli bilimsel konu olmakta ve bu konuda çalışmaların arttırılması gerekmektedir.

Yapılan tez çalışmasında var olan çalışmalarda bulunan eksiklikler belirlenerek kötü amaçlı yazılımların analiz ve tespit yöntemleri incelenmiştir. Derin öğrenme teknolojisinin evrişimsel sinir ağı üzerinde yeni yaklaşım ve farklı teknikler kullanılarak zararlı yazılım tespiti için model önerilmiştir. Yapılan çalışma sonucunda zararlı yazılımların yüksek oranda tespiti yapılmıştır. Veri setinin dengeli olmaması nedeniyle hassasiyet, duyarlılık ve f1 skorları incelenmiştir. Yapılan çalışma da kullanılan veri seti, yöntem ve uygulama adımları yukarıdaki bölümlerde anlatılmıştır. Yapılan çalışma da elde edilen sonuçlar **çizelge 5.3'**te paylaşılmıştır.

CNN ve ANN ile birlikte kullanılmasının amacı kötü amaçlı yazılım tespitinin başarı oranını arttırmaktır. Yalnızca ANN ile %97.8 oranında kötü amaçlı yazılım tespiti sağlanmıştır. CNN ile birleştirilerek geliştirilen modelde zararlı yazılımın tespiti %1.3 daha arttırılarak %99,02'e ulaşmıştır. Köse ve Samet (2021) yılında ilk yapılan çalışma da %98,1 oranında kötü amaçlı yazılım tespiti yapılabilmekteydi. Donanımsal kaynak yetersizliğinden dolayı tespit oranının düşük olduğu tespit edilmiş ve Adım sayısı 5 katmanlı mimari için yükseltilerek %1,3 oranında zararlı yazılım tespitinde artış sağlanmıştır.

Yapılan modelin çalışabilirliğini tespit etmek ve diğer algoritmaların veri seti üzerindeki doğruluğunu gösterebilmek için yapılan tez çalışmasında makine öğrenmesindeki Lojistik Regresyon ve Rassal Orman algoritmalarından yararlanılmıştır. Bu yöntemlerle yapılan zararlı yazılım tespitinde %2 ve %3 daha az doğrulukta zararlı yazılım tespiti yapıldığı görülmüştür. CNN mimarisindeki öznitelik çıkarmak için kullanılan konvolüsyon ve havuzlama katmanları çıkarılarak model öğretildiğinde CNN mimarisine göre yaklaşık %2 oranında daha az doğrulukta sonuç vermiştir. Önerilen model için daha fazla ANN katmanı ya da daha fazla adım da modelin eğitilmesinin model de aşırı öğrenmesiyle ezberleme yaptığı ve modelin kendini tekrarlayan doğrulukta çıktı vermesine neden olduğu görülmüştür.

Zararlı yazılımların tespiti ile ilgili anti virüs ve saldırı tespit sistemlerinde sıklıkla kullanılan statik ve dinamik tabanlı yöntemlerin eksik kaldığı yerler gözlemlenmiştir. Eksik kalan yerler paylaşıldığında:

- Statik tabanlı yöntem ile kullanılmakta olan uygulama imzaların zararlı yazılım tespiti için yetersiz kalması,
- Dinamik tabanlı yöntem ve yaklaşımların zararlı yazılım tespitinde uzun bir süreç olması,
- Dinamik tabanlı yöntemde kötü amaçlı yazılım davranışlarının net bir şekilde ortaya konulamaması,
- Dinamik tabanlı yöntem kullanılarak yapılan analizlerde derin öğrenme, makine öğrenmesi gibi yöntemlerin kullanımının fazla olmaması,
- Davranış ve statik tabanlı yöntemlerin gizlenme ve şaşırtma tekniklerine karşı zararlı yazılım tespitinde yetersiz kalmasıdır.
- Önerilen derin öğrenme ve makine öğrenmesi tabanlı yöntemlerde veri setlerindeki parametre sayısındaki eksiklik,
- Önerilen derin öğrenme ve makine öğrenmesi tabanlı yöntemlerde veri setlerinin az olması,
- Önerilen derin öğrenme ve makine öğrenmesi tabanlı yöntemlerde veri setleri artışında ortaya çıkabilecek sorunlarla ilgili bilgilerin bulunamaması,

- Önerilen derin öğrenme ve makine öğrenmesi tabanlı yöntemlerde sadece tek bir makine öğrenimi algoritması ya da derin öğrenme yöntemlerinden birinden yararlanılmasıdır.

Yukarıdaki bilgiler sonucunda statik ve dinamik tabanlı tespit yöntemlerinin zararlı yazılım tespitinde eksik kaldığı yerler görülmüştür. Yapılan tez çalışmasında bu yöntemler yerine makine öğrenmesi ve derin öğrenme yöntemlerini içeren model önerilmiştir. Zararlı yazılımların tespitiyle ilgili yapılan diğer çalışmaların kullanmış olduğu metot ve yöntemler incelendiğinde, kötü amaçlı yazılımların tespitinde veri setlerinin azlığı ve bu durumun sağladığı avantaj gibi görünen yüksek doğrulukta zararlı yazılım tespitinin, yeni veri setleri eklendiğinde modelin nasıl yanıt verebileceğinin bulunmaması, tek bir makine ve derin öğrenme yöntemi kullanılarak modelin tam çalıştığının gösterilememesi, bizlere bu tez çalışmasının yapılmasına yönlendirmiştir.

Mevcut çalışmalar ile tez çalışması kapsamında yapılan çalışma karşılaştırıldığında, bu çalışmanın önemi aşağıdaki şekilde paylaşılmıştır.

- Yapılan çalışmalardaki veri setleri incelendiğinde çoğunlukla dışarıdan hazır olarak kullanılan Microsoft ve MalImg 'in veri setleri kullanıldığı görülmüştür. Çalışma yapanların oluşturdukları veri seti ve hazır kullanılan veri setlerinin sayısı incelendiğinde, yapılan tez çalışmasına göre çok daha düşük oranda olduğu görülmektedir. Veri azlığının derin öğrenme ile makine öğrenimi algoritmalarında daha kolay bir şekilde normalizasyon işlemleri yapıldığı için daha yüksek oranda zararlı yazılım tespiti yapılabilmesine olanak sağlarken, yapılan tez çalışmasında yüksek oranda zararlı yazılım veri seti olmasına rağmen doğruluk oranı diğer çalışmalara oranla daha yüksek olduğu görülmektedir.
- İncelenen çalışmalarda dinamik analiz yöntemiyle oldukça az sayıda veri setlerinde parametre sayısı olduğu görülmektedir. Yapılan çalışmalar ve tez çalışmasında işletim sistemi Windows olduğu için bu sayının düşük olması kötü amaçlı yazılımın tespitinde başarısız sonuçlara yol açacaktır. Tez çalışmasında 32*32 'lik resim olacak şekilde 1024 parametre ile zararlı yazılım tespiti yapılmaktayken diğer çalışmalarda bu sayı oldukça az olduğu görülmektedir.

- Mevcut çalışmaların çoğunda makine ya da derin öğrenme yöntemlerinden teki kullanılmış ve çalışmaların bu iki yönteme göre doğruluk oranı ortaya konmamışken bu çalışmada iki yöntem ile de yüksek oranda zararlı yazılım tespiti yapabilmektedir.
- Yukarıda anlatılan çalışmalarda başarı oranı yüksekliği daha çok makine öğrenmesi algoritmaları kullanılarak yapılmışken, yapılan tez çalışmasında derin öğrenme yöntemi makine öğrenmesi algoritmalarına kıyasla daha başarılı oranda zararlı yazılım tespiti yapabilmiştir.

Farklı işletim sistemlerindeki kötü amaçlı yazılımların tespiti önerilen model ile yapılabilecektir. Kötü amaçlı yazılımların işletim sistemi ya da veri seti bağımlılığı farklı platformlarda yeni geliştirilecek olan kötü amaçlı yazılım veri seti ile birlikte kullanılabilecek ve karşılaştırılmaları yapılabilecektir. Bu işlem için önerilen modele giriş verisi olarak resim verilmesi yeterli bulunmaktadır.

Çalışmanın daha yüksek doğrulukta zararlı yazılım tespit edebilmesi ile ilgili çalışmalar devam etmektedir. Gelecekte yapılacak ve önerilen çalışmalar kötü amaçlı yazılımların etiketli veriler yerine etiketsiz verilerle gözetimsiz öğrenme yöntemleri kullanılarak tespiti yapılmasıdır. RNN modeli ve farklı gözetimsiz öğrenme yöntemleri geliştirilerek zararlı yazılım tespitinde kullanılacaktır. Veri seti daha dengeli hale getirilerek yeniden öğrenme işlemleri yapılacak ve değerlendirmeler paylaşılacaktır. Literatürde bulunan farklı mimari modeller veri setine uygulanarak sonuçları paylaşılacaktır.

Tez çalışmasının göstermiş olduğu bilgiler ışığında zararlı yazılım tespiti için statik tabanlı yöntemin blok zinciri tabanlı yöntemlerle desteklenmesi ve statik tabanlı yöntemlerin daha başarılı olması için dinamik tabanlı yöntemin derin öğrenme ve makine öğrenmesi yöntemleriyle birlikte kullanılması gerektiği açıktır. Derin öğrenme ve makine öğrenmesi teknolojilerinin zararlı yazılım tespitindeki sürelerin düşürülmesi için donanımsal kısıtların ortadan kaldırılarak yeni teknolojilerle desteklenmesi gerekmektedir. Zararlı yazılım tespit işlemleri ile ilgilenen teknoloji firmalarının online zararlı yazılım tespit sistemleri sunması, yukarıdaki yöntemlerin daha başarılı olması için veri setlerinin paylaşılması gerekmektedir. Paylaşılan veri setleri ile birlikte farklı işletim

sistemlerinde sunulan kötü amaçlı yazılım tespitinin de daha hızlı ve kolay yapılacağı düşünülmektedir

Tez çalışmasında kötü amaçlı yazılımların tespiti için mimari model önerilmiştir. Önerilen model farklı makine öğrenmesi algoritmaları ve fazla boyutta kötü amaçlı yazılım veri seti bulunan çalışmalar ile kıyaslandığında daha başarılı doğrulukta zararlı yazılımları tespit etmektedir. Bu çalışmanın daha fazla sayıda test verisi ile birlikte yeni çıkmış kötü amaçlı yazılımların tespitinin analiz yöntemlerindeki zorluklarını daha hızlı ve kolay bir şekilde aşarak tespitlerini kolaylaştıracağı kuşkusuzdur. Bu sayede zararlı yazılımın neden olacağı zararlar azalacağı ümit edilmektedir.

KAYNAKLAR

Alosefer, Y. 2012. Analysing Web-based Malware Behaviourthrough Client Honeypots. Doctoral dissertation, Cardiff University School of Computer Science & Informatics.

Alzammam, A. Binsalleeh, H. AsSadhan, B. Kyriakopoulos, K. G. Lambotharan, S. 2020. Comparative analysis on imbalanced multi-class classification for Malware samples using CNN. In 2019 International Conference on Advances in the Emerging Computing Technologies (AECT), 1-6, IEEE.

Angelo, O. 2019. Malware Analysis Datasets: Top-1000 PE Imports, IEEE Dataport

Anonymous. 2022. Aktivasyon Fonksiyonu, Web Sitesi: https://ennurcitir.wordpress.com/2020/02/19/yapay-sinir-aglari/, erişim tarihi: 20/02/2022

Anonymous. 2022. İşletim Sistemi İstatistikleri, Web Sitesi: https://gs.statcounter.com/os-market-share, Erişim Tarihi: 20.02.2022

Anonymous. 2022. Kötü Amaçlı Yazılım İstatistik Bilgisi, Web Sitesi: https://portal.av-atlas.org/ , erişim tarihi: 20/02/2022

Anonymous. 2022. Kötü Amaçlı Yazılım istatistik bilgisi, Web Sitesi: https://www.virustotal.com/tr/statistics/ Erişim Tarihi: 23.02.2022.

Anonymous. 2022. Numpy kütüphanesi, Web Sitesi: https://numpy.org, Erişim Tarihi: 20.02.2022

Anonymous. 2022. Pandas kütüphanesi, Python Data Analysis Library, Web Sitesi: https://pandas.pydata.org/, erişim tarihi: 20.02.2022

Anonymous. 2022. Tensorflow kütüphanesi, Web Sitesi: https://www.tensorflow.org/, Erişim Tarihi: 20.02.2022

Aslan, Ö. 2020. Zararlı yazılımların göstermiş oldukları davranışlara göre analiz ve tespit edilmesi. Doktora Tezi, Ankara Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı, 150, Ankara

Aslan, Ö. ve Samet, R. 2020. A comprehensive review on malware detection approaches. IEEE Access 8, 6249-6271.

Aslan, Ö, ve Samet, R. Investigation of possibilities to detect malware using existing tools. 2017. IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), IEEE.

Aslan, Ö, Samet, R., ve Tanrıöver, Ö. Ö. 2020. Using a subtractive center behavioral model to detect malware. Security and Communication Networks 2020.

Aydoğan, E ve Şen, S. 2014. Analysis of machine learning methods on malware detection. 22nd Signal Processing and Communications Applications Conference (SIU). IEEE, 2014.

Barsiya, T. K., Gyanchandani, M. ve Wadhwani, R. 2016. Android Malware Analysis: A Survey Paper. International Journal of Control, Automation, Communication and Systems (IJCACS), 35-42.

Bircanoğlu, C. ve Arıca, N. 2018. A comparison of activation functions in artificial neural networks. 26th Signal Processing and Communications Applications Conference (SIU). IEEE.

Bissell, K., Lasalle, R. M. ve Cin, P. D. 2019. THE COST OF CYBER CRIME, Technical Report No: Ninth Annual Cost of Cybercrime Study, Department of Security, Accenture Security PLC, USA.

BrownLee, J. 2020. How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification , Imbalanced Classification, Machine Learning Mastery PTY. LTD., USA.

Damodaran, A., 2017. A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques 13.1 (2017): 1-12.

Egemen, E. İnal, E. ve Levi, A. 2015. Mobile malware classification based on permission data. 23nd Signal Processing and Communications Applications Conference (SIU). IEEE.

Eskandari, S. 2018. A first look at browser-based cryptojacking. IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). IEEE.

Girshick, R. 2015. Fast r-cnn. Proceedings of the IEEE international conference on computer vision.

Hansen, S. S., Larsen, T.M.T., Stevanovic, M. ve Pedersen, J.M. 2016. An approach for detection and family classification of malware based on behavioral analysis. In 2016 International conference on computing, networking and communications (ICNC) (pp. 1-5). IEEE.

Hardy, W. 2016. Dl4md: A deep learning framework for intelligent malware detection. Proceedings of the International Conference on Data Science (ICDATA). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Hatipoğlu, C. ve Tunacan, T. 2021. Türkiye'de Siber Saldırı ve Tespit Yöntemleri: Bir Literatür Taraması. Bilecik Şeyh Edebali Üniversitesi Fen Bilimleri Dergisi.

Hochreiter, S. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6.02, 107-116.

Hwang, J. and Zhou, Y. 2015. Image Colorization with Deep Convolutional Neural Networks.

Iandola, F. N. 2016. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size.

Idika, N. ve Aditya, P. M. 2007. A survey of malware detection techniques. Purdue University 48.2.

Irshad, A., Maurya, R., Dutta, M.K., Burget, R. ve Uher, V. 2019. Feature Optimization for Run Time Analysis of Malware in Windows Operating System using Machine Learning Approach. In 2019 42nd International Conference on Telecommunications and Signal Processing (TSP) (pp. 255-260). IEEE.

Kara, I. ve Aydos M. 2018 Static and dynamic analysis of third generation cerber ransomware. 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT). IEEE.

Kartal, Y., Ozkan, K. ve Yılmaz, F. 2019. Effects of Feature Extraction Methods to Malware Classification. 10.13140/RG.2.2.20703.1040

Kiros, R., Salakhutdinov, R. ve Zemel, R. S. 2014. Unifying visual-semantic embeddings with multimodal neural language models. arXiv preprint arXiv:1411.2539.

Krizhevsky, A., Sutskever, I. ve Hinton, G. 2012. ImageNet classification with deep convolutional neural networks. In NIPS'2012 . 23, 24, 27, 100, 200, 371, 456, 460.

Köse, Ü. ve Samet, R. 2021 Detection of Malware with Deep Learning Method. 2021 6th International Conference on Computer Science and Engineering (UBMK). IEEE.

Larochelle, H. 2009. Exploring strategies for training deep neural networks. Journal of machine learning research 10.1 (2009).

Larsson, G., Maire, M. ve Shakhnarovich, G. 2016. Learning representations for automatic colorization. European Conference on Computer Vision, Springer

LeCun, Y. 1987. Modèles connexionistes de l'apprentissage, Université de Paris VI. 18, 504, 517.

Liu, Y. ve Wang, Y. 2019. A robust malware detection system using deep learning on api calls. In 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC) (pp. 1456-1460). IEEE.

Lu, L. 2019. Dying relu and initialization: Theory and numerical examples. arXiv preprint arXiv:1903.06733.

McClelland, J., Rumelhart, D. ve Hinton, G. 1995. The appeal of parallel distributed processing . In Computation & intelligence, American Association for Artificial Intelligence. 17.

McCulloch, W. S. ve Pitts, W. 1943. A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics 5(4): 115–133.

McKinney, W. 2012 Python for data analysis: Data wrangling with Pandas, NumPy, and IPython. " O'Reilly Media, Inc.".

Mnih, V. K., Kavukcuoglu, D., Silver, A., Graves, I., Antonoglou, D., Wierstra ve M. Riedmiller 2013. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Ozkan, O. M. Samet, R., Aslan, Ö., & Gupta, D. (2021). A Comprehensive Systematic Literature Review on Intrusion Detection Systems. IEEE Access.

Özkan, İ. N. İ. K. ve Ülker, Erkan. 2017. Derin öğrenme ve görüntü analizinde kullanılan derin öğrenme modelleri. Gaziosmanpaşa Bilimsel Araştırma Dergisi 6.3 (2017): 85-104.

Poonguzhali, N. P. 2019. Identification of malware using CNN and bio-inspired technique. 2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN). IEEE.

Ramachandran, P., Barret Z. ve Quoc V. Le. 2017. Searching for activation functions. arXiv preprint arXiv:1710.05941.

Ray, A. ve Asoke N. 2016. Introduction to Malware and Malware Analysis: A brief overview. International Journal 4.10.

Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. Psychological review 65(6): 386–408.

Rosenblatt, F. 1962. Principles of Neurodynamics. Spartan, New York. 15, 27

Rumelhart, D. E., McClelland, J. L. ve Group, T. P. R. 1986. Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MIT Press, Cambridge. 17.

Salmani, H., Tehranipoor M. ve Plusquellic J. 2011 A novel technique for improving hardware trojan detection and reducing trojan activation time. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20.1: 112-125.

Sanjaa, B, ve Chuluun, E. 2013. Malware detection using linear SVM. In Strategic Technology (IFOST), 2013 8th International Forum on (Vol. 2, pp. 136-138). IEEE.

Sewak, M., Sanjay, K., Sahay ve Hemant R. 2018. Comparison of deep learning and the classical machine learning algorithm for the malware detection. 2018 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD). IEEE.

Seyyarer, E., Ayata, F., Uçkan, T. ve Karci, A. 2020. Derin Öğrenmede Kullanilan Optimizasyon Algoritmalarinin Uygulanmasi Ve Kiyaslanmasi. Computer Science, 5(2), 90-98.

Snow, E. 2020 End-to-end Multimodel Deep Learning for Malware Classification. 2020 International Joint Conference on Neural Networks (IJCNN). IEEE.

Sreekumari, P. 2020. Malware detection techniques based on deep learning. 2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS). IEEE.

Sukhbaatar, B. 2007. Introduction of new information and Communication technologies in mongolia. In Strategic Technology, 2007. IFOST 2007. International Forum on (pp. 24-27). IEEE.

Thomas, K. ve David M. N. 2010. The Koobface botnet and the rise of social malware. 2010 5th International Conference on Malicious and Unwanted Software. IEEE.

Traore, B. B., Bernard K. F. ve Fana T. 2018. Deep convolution neural network for image recognition. Ecological Informatics 48 (2018): 257-268.

Tokmak, M. ve Ecir U. K. 2019. Kötü Amaçlı Windows Çalıştırılabilir Dosyalarının Derin Öğrenme İle Tespiti. Bilge International Journal of Science and Technology Research 3.1 (2019): 67-76.

Tur, A. O. ve Keles H. Y. 2021. Evaluation of hidden Markov models using deep CNN features in isolated sign recognition. Multimedia Tools and Applications 80.13 (2021): 19137-19155.

Türker, S. 2019. Zararlı android yazılımlarının makine öğrenmesi ile ailelerine göre sınıflandırılması. MS thesis. Fen Bilimleri Enstitüsü.

Widrow, B. ve Hoff, M. E. 1960. Adaptive switching circuits. Adaptive switching circuits. In 1960 IRE WESCON Convention Record, volume 4, pages 96–104. IRE, New York. 15, 21, 24, 27.

Yalçinkaya, M. A. 2020. "Yapay zeka ve dinamik analiz tabanlı web uygulama zafiyet tarayıcısı."

You, Y. 2018. Imagenet training in minutes. Proceedings of the 47th International Conference on Parallel Processing.

Zhang, R., Isola, P. ve Efros, A. A. 2016. Colorful image colorization. European Conference on Computer Vision, Springer.