
GAZİ ÜNİVERSİTESİ
Gazi
1926

AĞ TRAFİĞİNDE ANORMALLİK TESPİTİ İÇİN VERİ SETİ OLUŞTURMA VE TEST YÖNTEMLERİNİN KARŞILAŞTIRILMASI

Emrullah ERGİNAY

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2019

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

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

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

Emrullah ERGİNAY

21/06/2019

AĞ TRAFİĞİNDE ANORMALLİK TESPİTİ İÇİN VERİ SETİ OLUŞTURMA VE TEST YÖNTEMLERİNİN KARŞILAŞTIRILMASI

(Yüksek Lisans Tezi)

Emrullah ERGİNAY

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2019

ÖZET

Verilerin sayısal ortamlarda saklanmasıyla beraber siber saldırıların sayısı ve çeşitliliği çoğalmış, sayısal ortamlara yönelik saldırılar çok hızlı bir şekilde artmıştır. Saldırganlar tarafından bir silah olarak kullanılan zararlı yazılımlara karşı antivirüs, güvenlik duvarları, saldırı tespit/önleme sistemleri gibi çözümler geliştirilmiştir. İmza tabanlı çözümler, önceden bilinen zararlı yazılımları tespit edebilmesinden dolayı, yeni ortaya çıkan zararlı yazılımlara karşı etkisiz kalmaktadırlar. Bu nedenle zararlı yazılım tespitinde davranışsal analiz yöntemleri önemli hale gelmiştir. Bu tezde, sistemden veri sızıntısı yapan zararlı/casus yazılımlara ait İnternet trafiğinin tespit edilmesi amacıyla makine öğrenmesi yöntemleri kullanılarak bir uygulama geliştirilmiştir. Tez kapsamında, uygulama katmanı üzerinde zararlı ve normal İnternet trafiklerine ait veri toplanmış ve 100 tane öznitelik oluşturulmuştur. Bir kurumsal ağda gerçek zamanlı olarak 16221 tane akış verisi toplanarak; yapay sinir ağları, derin öğrenme, karar ağacı vb. çeşitli makine öğrenmesi algoritmaları uygulanmıştır. Farklı yöntemlerle başarı durumları kapsamlı bir şekilde incelenerek karşılaştırma yapılmıştır.

Bilim Kodu	:	92403
Anahtar Kelimeler	:	Derin öğrenme, yapay sinir ağları, zararlı yazılım, anormallik tespiti, makine öğrenmesi
Sayfa Adedi	:	89
Danışman	:	Prof. Dr. M. Ali AKCAYOL

CREATING A DATA SET FOR ANOMALY DETECTION IN NETWORK TRAFFIC AND COMPARISON OF TEST METHODS

(M. Sc. Thesis)

Emrullah ERGİNAY

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2019

ABSTRACT

As the data were stored in digital environments, the number and diversity of cyber attacks increased and the attacks on digital environments increased very rapidly. Antiviruses, firewalls, intrusion detection / prevention systems have been developed against the malware used as a weapon by the attackers. Signature-based solutions can detect previously known malware and are ineffective against new emerging malware. Therefore, behavioral analyses methods have become important in malware detection. In this thesis, an application has been developed by using machine learning methods in order to determine internet traffic of malware / spyware that leaked data from the system. Within the scope of the thesis, data on harmful and normal Internet traffic were collected on the application layer and 100 attributes were created. In a corporate network, 16221 stream data was collected in real time and artificial neural networks, deep learning, decision tree, etc. various machine learning algorithms have been applied. The success cases were analyzed by different methods comprehensively and compared.

Science Code	:	92403
Key Words	:	Deep learning, artificial neural networks, malware, anomaly detection, machine learning
Page Number	:	89
Supervisor	:	Prof. Dr. M. Ali AKCAYOL

TEŞEKKÜR

Bu tez çalışması boyunca yardım ve desteklerini esirgemeyen, kıymetli bilgilerinden yararlandığım danışmanım Sayın Prof. Dr. M. Ali AKCAYOL'a, maddi manevi destekleriyle yanımda olan değerli aile üyelerime ve kıymetli eşim Tuğba'ya teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGE LİSTESİ

Çizelge **Sayfa**

Çizelge **Sayfa**

Çizelge **Sayfa**

ŞEKİL LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

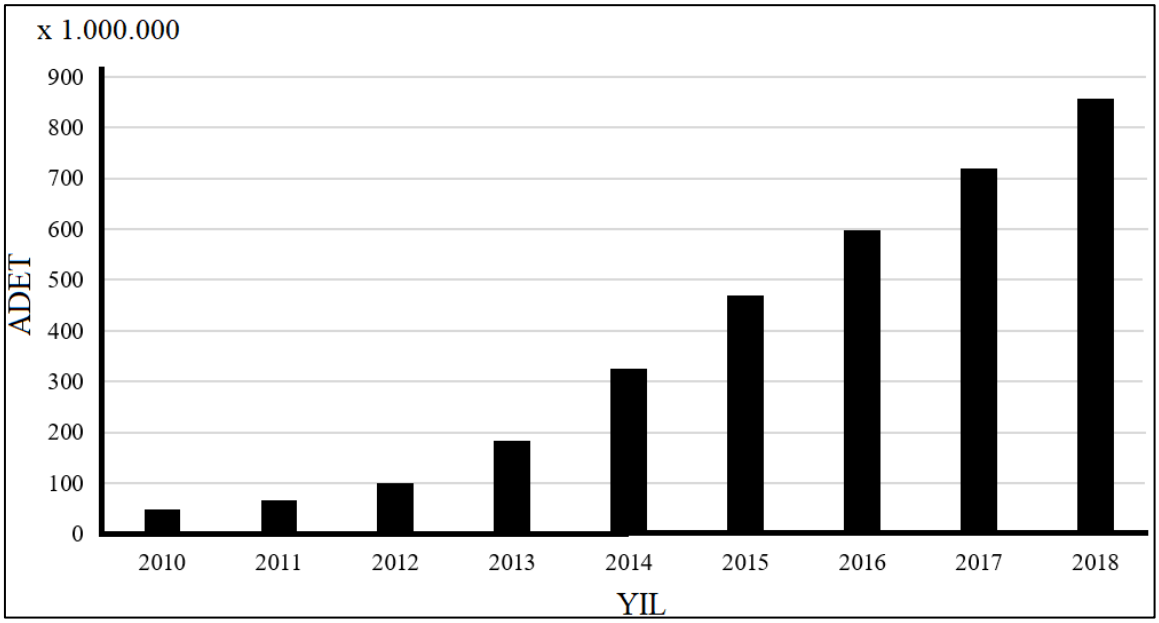
Kısaltmalar	Açıklamalar
AIM	AOL Instant Messenger
ARP	Address Resolution Protocol
API	Application Programming Interface
APT	Advanced Persistent Threat
AUC	Area Under Curve
CAIDA	Center for Applied Internet Data Analysis
CCC	Cyber Clean Center
CFS	Correlation-Based Feature Selection
CIDDS	Coburg Network Intrusion Detection Dataset
CNN	Convolutional Neural Network
CSV	Comma-Separated Values
CTU	Czech Technical University
DBN	Deep Belief Network
DDOS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DMOZ	Directory Mozilla
DNN	Deep Neural Network
DNS	Domain Name System
DPI	Deep Packet Inspection
DTW	Dynamic Time Warping
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
FTPS	File Transfer Protocol SSH
GAF	Global Abnormal Forest
GIST	Global Image Descriptor
GPU	Graphics Processing Unit

Kısaltmalar	Açıklamalar
HTTP	Hypter Text Transfer Protocol
ICQ	I seek you
ID	Identification
IDS	Intrusion Detection System
IGMP	Internet Group Management Protocol
IRC	Internet Relay Chat
JSON	JavaScript Object Notation
KNN	K-Nearest Neighbour
LBNL	Lawrence Berkeley National Laboratory
LBP	Local Binary Pattern
LOF	Local Outlier Factor
NBNS	NetBIOS Name Service
P2P	Peer to Peer
PART	Projective Adaptive Resonance Theory
PCA	Principle Component Analysis
REPTree	Reduced Error Pruning Tree
RF	Random Forest
RIDOR	Ripple Down Rule Rule Learner
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SDN	Software Defined Network
SFTP	SSH File Transfer Protocol
SMB	Server Message Block
SSDP	Simple Service Discovery Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer
SVM	Support Vector Machine
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UNB-CIC	University of New Brunswick- Canadian Institute for Cybersecurity
USB	Universal Serial Bus

Kısaltmalar	Açıklamalar
VPN	Virtual Private Network
WISNET	Wireless and Secure Networks Research Lab
YSA	Yapay Sinir Ağı

1. GİRİŞ

Dünya genelinde teknolojinin gelişmesi, İnternet kullanımının yaygınlaşması ve sunulan servislerin sayısal ortama aktarılmasıyla beraber veri miktarı ve veri çeşitliliğinde büyük artış olmuştur. Verilerin sayısal ortamlarda saklanmasıyla beraber de sayısal ortamlar, siber saldırganlar için önemli bir hedef haline gelmiştir. Tüm bunlarla birlikte siber saldırıların sayısında ve çeşitliliğinde de artış meydana gelmiştir. Her geçen gün siber saldırı sayısı artmakta ve saldırılar daha karmaşık hale gelmektedir. "AV-TEST Bilgi Teknolojileri Güvenlik Enstitüsü" tarafından yapılan çalışmada; 2010 yılından itibaren zararlı yazılımlara ilişkin istatistiksel veriler paylaşılmıştır. Şekil 1.1'de görüldüğü gibi 2018 yılı itibariyle yaklaşık 856 000 000 tane zararlı yazılım tespit edilmiştir [1].



Şekil 1.1. Son 10 yılda tespit edilen zararlı yazılım sayıları

Hedef sistemlere bulaşan zararlı yazılımlar, bulaştığı sistemler hakkındaki bilgileri (iç IP adresi, bilgisayar adı, MAC adresi vb.) ve/veya sistemde yer alan çeşitli dosyaları (ofis dokümanları, resim, video vb.) komuta kontrol sunucularına iletebilmektedir. Bununla beraber saldırganlar, hedef sisteme uzaktan yetkisiz erişim sağlamak amacıyla, hedef sistem üzerinde bir arka kapı (backdoor) bırakabilmekte ve söz konusu arka kapı aracılığıyla hedef sisteme komut yollayabilmekte veya veri çalabilmektedir. Kısacası zararlı yazılımlar, saldırganların amaçları doğrultusunda tasarlanmakta ve genellikle bulaştığı sistemin haricinde bir sistem veya sistemlerle iletişim kurmaktadır.

Saldırganlar herhangi bir bireyin bankacılık bilgilerini, e-posta veya sosyal medya hesaplarını hedef seçebileceği gibi bilgi çalmak amacıyla şirketleri de hedef haline getirebilmektedir. Saldırılarda bireyler, şirketler, çeşitli organizasyonların yanı sıra devletler de çeşitli amaçlar doğrultusunda hedef alınabilmektedir. Özellikle son yıllarda devletlere yönelik geliştirilen gelişmiş sürekli tehditlerin (Advanced Persistent Threat – APT) sayısında artış görülmektedir. APT tehditleri, bir saldırganın ağa erişim sağlayarak uzun süre tespit edilmeden orada kalabildiği sofistike ağ saldırıları olarak tanımlanmaktadır. APT saldırılarının amacı sisteme hasar vermekten ziyade veri çalmaktır. Bu nedenle APT saldırıları genellikle değerli veri barındıran sistemlere yönelik gerçekleştirilmektedir. Yeni nesil tehditler Web, e-posta, masaüstü/mobil uygulamalar vs. üzerinden yayılabildiği için çok vektörlü olmaktadır. Saldırılarla ağa sızılabilmekte, ağ üzerinde başka sistemlere yayılabilmekte ve değerli veriler çalınabilmektedir. Bu nedenle saldırılar aynı zamanda çok aşamalıdır [2]. Bununla beraber son yıllarda fidye yazılımların görülme sayısında artış olmuştur. Özellikle şirketlerin hedef alındığı fidye yazılımlar ile fidye yazılımın bulaştığı bilgisayardaki dosyalar kriptolanmakta ve kriptolu dosyaların açılması için kullanıcılardan para talep edilmektedir.

Söz konusu tehditlere karşılık antivirüs, güvenlik duvarları, saldırı tespit/önleme sistemleri gibi çözümler geliştirilmiştir. İmza tabanlı çözümler, yalnızca daha önceden bilinen zararlı yazılımları tespit edebilmesinden dolayı yeni ortaya çıkan zararlı yazılımlara karşı etkisiz kalmaktadırlar. Bu nedenle zararlı yazılım tespitine yönelik yapılan çalışmalarda, davranışsal analiz yöntemleri önemli hale gelmiştir. Literatüre bakıldığında, zararlı yazılımlara ait İnternet trafiğinin tespit edilmesine ilişkin yapılan çalışmalar kapsamında yapay zeka çözümlerinin sıklıkla kullanıldığı görülmektedir.

Bu tezde makine öğrenmesi algoritmaları kullanılarak gerek önceden bilinen gerekse henüz tespit edilmemiş zararlı yazılımlara ait trafiğin tespit edilmesi amaçlanmıştır. Bunun için öncelikle siber saldırıların ne şekilde yapıldığı, zararlı yazılım bulaşmış sistemdeki anormallikler, literatürde zararlı yazılımların trafiğinin tespit edilmesine yönelik daha önceden yapılmış çalışmalar ve kullanılan veri setlerinin özellikleri incelenmiştir. Bu kapsamda araştırmalar yapıldıktan sonra pcap dosya formatında veri seti toplanmış, söz konusu veri seti bir Python scripti vasıtasıyla filtrelenmiştir. Son olarak da kullanılabilir hale getirilen veri seti üzerinde makine öğrenmesi algoritmaları uygulanarak sonuçları karşılaştırılmıştır.

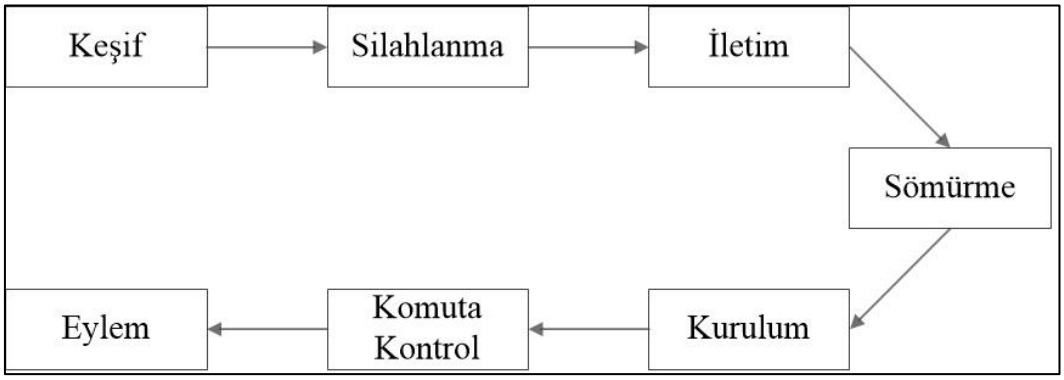
Bu tezin ikinci bölümünde siber saldırılar hakkında genel bir bilgi verilmiş ve ağ trafiğindeki anormallik tespitine yönelik literatür taramasına yer verilmiştir. Üçüncü bölümde ağ trafiğindeki anormallik tespitinde kullanılan makine öğrenmesi yöntemlerinden bahsedilmiştir. Dördüncü bölümde veri seti oluşturmaya yönelik literatürde yer alan yöntemler ve tez kapsamında oluşturulan veri seti oluşturma aşaması paylaşılmıştır. Beşinci bölümde çeşitli makine öğrenmesi yöntemlerinin, farklı parametrelere göre deneysel sonuçlarına yer verilmiştir. Son bölümde ise tez çalışmasının neticesinde varılan sonuç ve gelecekte yapılabilecek çalışmalar (öneriler) paylaşılmıştır.

2. KURUMSAL AĞLARDA SİBER SALDIRI VE ANORMALLİK TESPİTİ

2.1. Siber Saldırılar

Siber saldırılar bireyler, hacker grupları veya devletler tarafından gerçekleştirilmektedir ve bu tür saldırılarda izlenen belirli aşamalar bulunmaktadır.

2011 yılında Lockheed Martin firması tarafından, düşman saldırılarını daha iyi tespit etmek ve cevap vermek için karar verme sürecine yardımcı olmak amacıyla "Siber Ölüm Zinciri" modeli oluşturulmuştur. Askeri alanda kullanılmak amacıyla oluşturulan bu model, daha sonradan Bilgi Teknolojilerine (BT) uyarlanmıştır [3]. Söz konusu model, servis aksatma, veri hırsızlığı gibi zararlı aktiviteleri gerçekleştirmek ve hedef ağı istismar etmek amacıyla siber saldırılarda da kullanılmakta olup; Şekil 2.1'de görüldüğü gibi 7 aşamadan oluşmaktadır [4].



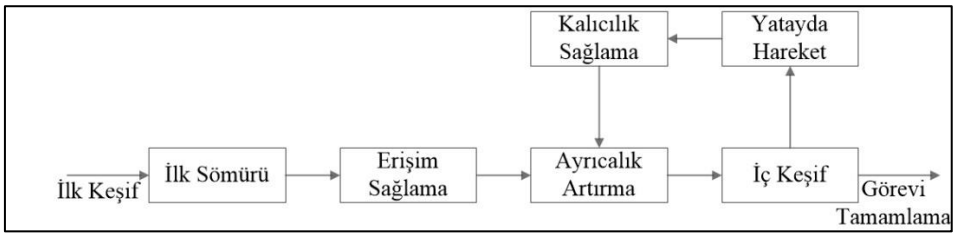
Şekil 2.1. "Siber ölüm zinciri" modeli

Yukarıdaki şekilde belirtilen ölüm zincirinin 7 adımı şu şekilde açıklanabilir [5]:

- Keşif: Hedef üzerinde saldırıya geçmeden önce port taramaları, açık kaynak taramaları vb. yapılır ve hedefin güvenlik açıklıkları tespit edilmeye çalışılır.
- Silahlanma: İkinci aşamada sisteme sızmak amacıyla siber silah oluşturma/edinme evresi yer almaktadır. Silahlandırma aşamasında saldırgan, kendi istismar kodunu (exploit) hazırlayabilir, hazır elde ettiği veya yine kendi hazırladığı zararlı yazılımı kullanabilmektedir. Bulunan güvenlik açıklığına göre saldırıyı gerçekleştiren kişi/kişiler, herhangi bir tür dosya içerisine zararlı yazılım yerleştirebilmektedir.

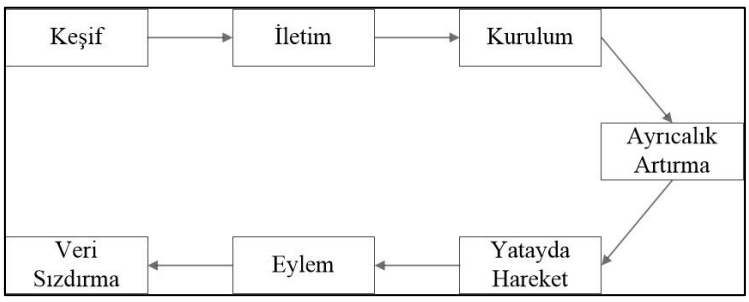
- İletim: Bu aşamada bir önceki aşamada edinilen siber silah, hedef üzerinde kullanılmaktadır. Burada e-posta ekleri, web siteleri veya USB bellekler aracılığıyla, oltalama (phishing) saldırısı sıklıkla kullanılmaktadır.
- Sömürme: Dördüncü aşamada zararlı yazılım veya istismar kodu hedef sistem üzerinde çalıştırılarak, sistemin istismar edilmesi amaçlanmaktadır.
- Kurulum: Sistem üzerinde arka kapı açılabilir, ayrıcalıklı kullanıcı yetkisi elde edilmeye çalışılabilir.
- Komuta ve Kontrol: Sistem üzerinde kalıcı olmak için çaba sarf edilir.
- Eylem: Son aşamada ise sistemden veri sızdırmak ve sistem hakkında detaylı bilgi edinilerek bağlantı ve erişimi kalıcı hale getirmek için ek yöntemler uygulanır. Ayrıca sistemdeki veriler kriptolanarak fidye talep edilebilir.

Lockedheed Martin şirketine ait ölüm zinciri modeline benzer olarak; 2012 yılında FireEye bünyesinde faaliyet gösteren danışmanlık şirketi olan Mandiant tarafından farklı bir "ölüm zinciri" modeli önerilmiştir. Şekil 2.3'te görüldüğü gibi bu modelde, iç ağ aktivitelerine (iç ağ keşfi, yanal hareketler) daha çok önem verilmiştir [4].



Şekil 2.2. Mandiant ölüm zinciri

Bryant ve Saiedian tarafından yapılan çalışmada ise Şekil 2.4'te görüldüğü gibi 7 aşamadan oluşan bir ölüm zinciri önerilmiştir [4].



Şekil 2.3. Bryant ölüm zinciri

Genel olarak yukarıdaki şekillerde gösterilen siber ölüm zincirleri incelendiğinde, hepsinde de saldırgan tarafından kullanılan istismar kodu veya zararlı yazılım vardır. Saldırgan hedef sisteme zarar vermeye çalışırken "kurulum" aşamasında, istismar kodunu / zararlı yazılımı kendisi de hazırlayabilmekte veya hali hazırda yayımlanmış olanları da kullanabilmektedir. Son aşamada genellikle saldırganın hedef sisteme bulaştırdığı istismar kodu veya zararlı yazılım aracılığı ile saldırganın kontrolündeki komuta kontrol sunucuları arasında iletişim kurulmaktadır.

Miloslavskaya tarafından uzaktan saldırı taksonomisi ve göstergelerine ilişkin yapılan çalışmada aşağıda verilen göstergelere değinilmiştir [5]:

- Tek bir yere giden veya tek bir yerden gelen aşırı İnternet trafiği
- Normalde İnternete kapalı olan iç ağdan İnternete çıkış
- Kara listede yer alan zararlı İnternet sitelerine yönelik aşırı trafik
- Gece saatlerinde veya hafta sonları oluşan İnternet trafiği
- Kısa süre içinde, farklı lokasyonlardan, aynı ID ile login olma durumu
- İç ağdan, kurumun herhangi bir şubesinin/ iş ortağının bulunmadığı bir ülkeye yönelik oluşan trafik
- Yanlış protokol – port eşleşmeleri
- Tek bir kullanıcı hesabı üzerinden, kısa süre içerisinde farklı bölgelerdeki sunuculara bağlanma isteği
- Antivirüs gibi izleme sistemlerinin gerçekleştirdiği aşırı port engelleme girişimleri
- Çok sayıda şüpheli içeriğe sahip e-posta gelmesi
- Kısa süre içinde, yönetici yetkilerine sahip hesaplarda değişiklikler olması

Yukarıda belirtilen göstergeler incelendiğinde, göstergelerin tamamının İnternet trafiğindeki anormalliklerden oluştuğu görülmektedir. Bu nedenle anormallik tespitinde, İnternet trafiği analiz edilirken davranışsal tabanlı çözümlerin kullanılması gereklidir.

2.2. Siber Saldırılara Karşı Yapılan Çalışmalar ve Kullanılan Anormallik Tespit Yöntemleri

Literatürdeki anormallik tespitine yönelik geliştirilen yapay zeka temelli çözümler incelenmiştir. İncelenen çalışmalarda, özellikle kullanılan veri setlerinin özellikleri, algoritmalar ve elde edilen başarı oranları irdelenmiştir.

Zhao, Traore, Sayed, Lu, Saad, Ghorbani ve Garant, makine öğrenmesi algoritmaları ile ağ trafiğinin davranışsal analizini yaparak, botnet tespitine yönelik bir çalışma yapmışlardır [19]. Akış karakteristiklerini sınıflandırmak için, her bir akıştaki gerekli davranış bilgileri seçilerek, öznitelikler elde edilmektedir. Veri setini eğitmek amacıyla, bilinen zararlı ve zararsız trafikler bir araya getirilmiş, veri setinde iki tane sınıf etiketi belirlenmiştir. Eğitim aşaması bittikten sonra, oluşturulan sistem, ağ trafiğini aktif olarak dinleyerek, tespit aşamasına geçmektedir.

Veri seti için öznitelikler belirlenirken, botnetlerin komuta kontrol sunucuları ile iletişime geçme süresi ve komuta kontrol sunucularına ilettiği paket boyutlarına ilişkin bilgilerin önemsendiği görülmektedir. Bununla birlikte paket başlık bilgilerinden elde edilebilecek IP adresi, port ve protokol bilgileri de veri setinde kullanılmıştır. Veri seti oluştururken kullanılan öznitelikler, aşağıdaki tabloda belirtilmiştir:

Çizelge 2.1. Botnet tespitinde kullanılan öznitelikler

Öznitelik	Açıklama
SrcIp	Akıştaki kaynak IP adresi
SrcPort	Akıştaki kaynak port
DstIp	Akıştaki hedef IP adresi
DstPort	Akıştaki hedef port
Protocol	Protokol
APL	Zaman aralığındaki ortalama paket boyutu
PV	Zaman aralığındaki paket boyutunun varyansı
PX	Zaman aralığında değiştirilen paket sayısı
PPS	Her 1 saniyede değiştirilen paket sayısı
FPS	Akışta gönderilen ilk paketin boyutu
TBP	Zaman aralığında gönderilen iki paket arasında geçen ortalama süre
NR	Bir akıştaki yeniden bağlanma sayısı
FPH	Tek bir adresten üretilen akış sayısının, belli bir zaman aralığında (1 saat) üretilen toplam akış sayısına oranı

Veri seti oluşturulduktan sonra karar ağacı, bayes ağları, yapay sinir ağları (YSA), destek vektör makineleri (Support Vector Machines - SVM), k en yakın komşu (k Nearest Neighbour - kNN) sınıflandırıcıları gibi çeşitli makine öğrenmesi algoritmaları denenmiş ve çalışmada Azaltılmış Hata Budama Ağacı (Reduced Error Pruning Tree – REPTree) algoritması kullanılarak karar ağacı seçilmiştir. REPTree algoritması ile sınıflandırmanın karmaşıklığını azaltmak ve gürültülü verileri (noisy data) yok etmek amaçlanmıştır.

Çalışmada sınıflandırma işlemleri için WEKA programı kullanılmış ve Java ortamında bir program geliştirilmiştir. Geliştirilen program ile bir pcap dosyasından akış bilgileri toplanarak, parse edilmiştir. Çalışmada test veri seti için 1 672 575 tane ağ akışı kullanılmıştır. Söz konusu veri setinin yaklaşık %5,8'i zararlı, geri kalan kısmı zararsız ağ akışı olarak belirtilmiştir. Eğitim aşamasında 10-fold cross-validation tekniği kullanılmıştır. Çalışma sonucunda aşağıdaki belirtilen sonuçlar elde edilmiştir:

Çizelge 2.2. Elde edilen sonuçlar

Çalışma Süresi	Doğruluk (%)	TP (%)	FP (%)	TN (%)	FN (%)
29,4 sn	99,1	98,3	<0,1	99,9	1,7

Ichino, Kawamoto, Iwano, Hatada ve Yoshiura, Linde-Buzo-Gray ve splitting algoritmalarını kullanarak, zararlı yazılımların oluşturduğu trafikleri inceleyerek, zararlı yazılımların oluşturduğu trafiği inceleme ve etkilerini tespit etme konusunda, bir ağ trafiğindeki özniteliklerden hangilerinin daha etkili olduğunu belirlemeye yönelik bir çalışma yapmışlardır [20].

Kullanılan yöntemin doğruluğunu ölçmek için aşağıdaki tabloda belirtilen 36 tane öznitelik kullanılmıştır:

Çizelge 2.3. Zararlı yazılım trafiği için kullanılan öznitelikler

	Öznitelik		Öznitelik
1	Paket sayısı	19	PSH paket sayısı/ACK paket sayısı
2	Toplam paket boyutu (byte)	20	RST paket sayısı/ACK paket sayısı
3	Ortalama paket boyutu (byte)	21	SYN paketlerinin TCP paketlerine oranı
4	Minimum paket boyutu (byte)	22	FIN paketlerinin TCP paketlerine oranı
5	Maksimum paket boyutu (byte)	23	PSH paketlerinin TCP paketlerine oranı
6	Paket boyutu standart sapması (byte)	24	ACK paketlerinin TCP paketlerine oranı
7	Ortalama iletim süresi (saniye)	25	RST paketlerinin TCP paketlerine oranı
8	Minimum iletim süresi (saniye)	26	URG paketlerinin TCP paketlerine oranı

Çizelge 2.3. (devam) Zararlı yazılım trafiği için kullanılan öznitelikler

9	Maksimum iletim süresi (saniye)	27	SYN-ACK oranının TCP paketlerine oranı
10	İletim süresi standart sapması (saniye)	28	FIN-ACK oranının TCP paketlerine oranı
11	SYN paket sayısı	29	PSH-ACK oranının TCP paketlerine oranı
12	FIN paket sayısı	30	RST-ACK oranının TCP paketlerine oranı
13	PSH paket sayısı	31	ICMP paket sayısı
14	ACK paket sayısı	32	UDP paket sayısı
15	RST paket sayısı	33	69 (UDP) portunu kullanan paket sayısı
16	URG paket sayısı	34	80 (TCP) portunu kullanan paket sayısı
17	SYN paket sayısı/ACK paket sayısı	35	110 (TCP) portunu kullanan paket sayısı
18	FIN paket sayısı/ACK paket sayısı	36	443 (TCP) portunu kullanan paket sayısı

1-10 arasındaki öznitelikler İnternet uygulaması ve zararlı yazılım etkisini tespit etmek amacıyla kullanılmaktadır. 11-32 arasındaki öznitelikler zararlı yazılım tespiti için kullanılmaktadır. 80 ve 443 portları normal kullanıcıların kullanabileceği portlar olduğu gibi aynı zamanda birçok zararlı yazılımın, komuta kontrol sunucuları ile iletişime geçmek amacıyla da kullandığı portlar olabilmektedir. Bu yüzden 34 ve 36 numaralı öznitelikler hem normal hem de zararlı yazılım etkilerini temsil etmektedir. 69 UDP portu da normal trafik olabileceği gibi zararlı yazılımların kullandığı port da olabilmektedir. Bu nedenle 33 numaralı öznitelik de hem normal hem de zararlı yazılımların etkisini temsil etmektedir. 110 portu e-posta trafiğini temsil ettiği için 35 numaralı öznitelik, normal işlemler olarak temsil edilmektedir.

Balküpü sistemlerinden toplanan Cyber Clean Center (CCC) veri setinden CCC2009, CCC2010 ve CCC2011 olmak üzere 3 farklı veri seti kullanılmıştır

Veri seti oluşturulduktan sonra vektör niceleme (vector quantization) ile normal ve anormal olmak üzere iki tane “codebook” oluşturulmuştur. Vektör niceleme için LBG ve splitting algoritmaları kullanılmıştır. Çalışmada kullanılan yöntem ile üç farklı veri seti için %90-100 aralığında doğruluk oranları elde edilmiştir.

Çalışmada ayrıca farklı özniteliklerden oluşan iki farklı veri seti üzerinde, karar ağacı ve naive bayes algoritmaları denenmiştir. İlk veri setinde yukarıdaki tabloda belirtilen tüm

öznitelikler kullanılmış olup; ikinci veri setinde SYN paket sayısı, SYN paket sayısının TCP paket sayısına oranı ve ACK paket sayısı kullanılmıştır. Her iki durum için de CCC2010 ve CCC2011 veri setleri denenmiştir ve doğruluk oranları aşağıdaki tabloda sunulmuştur:

Çizelge 2.4. CCC2010 ve CCC2011 veri setleri için elde edilen doğruluk oranları

Durum (Case)	Algoritma	CCC2010 veri seti (%)	CCC2011 veri seti (%)
1	Karar Ağacı	99,8	89,8
	Naive Bayes	73,4	60,4
2	Karar Ağacı	99,9	91,1
	Naive Bayes	89,0	86,4

Kumar, Viinikainen ve Hamalainen, ağ tabanlı saldırı tespit sistemleri için çeşitli denetimli makine öğrenmesi sınıflandırıcılarının kullanımına yönelik bir çalışma yapmışlardır [21]. Çalışmada, Android tabanlı zararlı yazılımların tespit edilmesine odaklanılmıştır.

Veri önişleme safhasında, UDP verileri silinerek, TCP verilerine odaklanılmıştır. Pcap dosyalarından aşağıdaki tabloda belirtilen 10 tane öznitelik ayıklanmıştır:

Çizelge 2.5. Ağ saldırı tespit sisteminde kullanılan öznitelikler

	Öznitelik	Açıklama
1	Duration	Bağlantı süresi
2	DP	Hedef port
3	PktSent	Gönderilen paket
4	PktRcv	Alınan paket
5	PLBytesSent	Gönderilen payload (byte)
6	PLBytesRcv	Alınan payload (byte)
7	IFlagF	İletilen ilk bayrak (flag)
8	IFlagR	Alınan ilk bayrak (flag)
9	UFlagsF	İletilen bayrakların tümü
10	UFlagR	Alınan bayrakların tümü

Çalışmada, WEKA platformu üzerinde J48, Random Forest, RIDOR, JRIP ve PART algoritmaları kullanılmıştır. Söz konusu algoritmalar doğruluk, eğitim süresi, çalışma prensibi, popülerlik ve verimlilik alanlarında analiz edilmiştir. Veri setinin eğitimi kısmında 10-fold cross validation metodunun kullanılmasının yanı sıra, eğitim ve test verisi %70-%30 oranında alınarak, eğitim aşaması gerçekleştirilmiş ve doğruluk oranları karşılaştırılmıştır.

10-fold cross validation yönteminin kullanıldığı deneyde aşağıdaki sonuçlar elde edilmiştir:

Çizelge 2.6. 10-fold cross validation ile elde edilen sonuçlar

Algoritma	TP - Recall (%)	FP (%)	TN (%)	FN (%)	Precision (%)	F1-Ölçütü (%)	Doğruluk (%)	Süre (ms)
J48	99,3	6,4	93,6	0,7	93,9	96,5	98,4	30
Random Forest	99,6	1,8	98,2	0,4	98,2	98,8	99,4	660
JRIP	99,3	5,0	95,0	0,7	95,2	97,2	98,6	310
RIDOR	99,3	6,4	93,6	0,7	93,9	96,2	98,4	190
PART	99,1	3,0	97,0	0,9	97,0	98,0	98,7	150

Eğitim ve test verisi %70-%30 oranında alındığında elde edilen karmaşıklık matrisi şu şekilde olmuştur:

Çizelge 2.7. Eğitim ve test verisi %70-%30 oranında alındığında elde edilen sonuçlar

Algoritma	TP – Recall (%)	FP (%)	TN (%)	FN (%)	Precision (%)	F1-Ölçütü (%)	Doğruluk (%)	Süre (ms)
J48	99,4	5,0	95,0	0,6	95,2	97,2	98,6	20
Random Forest	99,2	2,9	97,1	0,8	97,1	98,1	98,8	470
JRIP	99,0	2,9	97,1	1,0	97,1	98,0	98,7	300
RIDOR	99,2	6,5	93,5	0,8	93,8	96,4	98,2	51
PART	99,0	4,3	95,7	1,0	95,8	97,3	98,4	40

Azab, Alazab ve Aiash, makine öğrenmesi algoritmaları kullanılarak, zararlı yazılımlar ile komuta kontrol sunucuları arasındaki ağ trafiği akışında, botnet tespitine yönelik bir çalışma yapmışlardır [22]. Çalışmada, botnetin bulaştığı sisteme zarar vermeden önce tespit edilmesi amaçlanmış ve söz konusu çalışma, popüler bir botnet olan “Zeus” üzerinde uygulanmıştır. Çalışmada kullanılan veri seti şu sayılardan oluşmaktadır:

Çizelge 2.8. Zeus zararlı yazılım trafiğinde kullanılan veri setindeki eğtim ve test verisi sayısı

	Eğitim	Test
Normal HTTP Trafiği	2774	2396
Zeus v1.x	432	0
Zeus v2.x	0	144
Toplam	3206	2540

Sınıflandırma için kullanılan öznitelikler aşağıdaki çizelgede sunulmuştur:

Çizelge 2.9. Botnet tespitinde kullanılan öznitelikler

	Öznitelik	Açıklama
1	MPF	İletilen paket boyutlarının ortalaması
2	LPF	İletilen paketlerin en büyük boyutlu olanı
3	STDTF	İletilen paketlerin süresinin standart sapması
4	STB	Alınan paketlerin süresinin en küçüğü
5	MTB	Alınan paketlerin süresinin ortalaması
6	STDTB	Alınan paketlerin süresinin standart sapması
7	SActive	Minimum aktif akış sayısı
8	LIdle	Maksimum pasif akış sayısı
9	Bpush	Alınan “Push” bayrak sayısı

Çalışmada, gereksiz ve ilgisiz öznitelikleri filtrelemek için korelasyon tabanlı öznitelik seçme (Correlation-based Feature Selection - CFS) algoritması, sınıflandırma işlemi için ise C4.5 karar ağacı algoritması kullanılarak; Precision, Recall ve F-Measure metrikleri değerlendirilmiştir.

Oluşan karar ağacında, “LPF” ve “BPush” özniteliklerinin belirleyici olduğu tespit edilmiştir.

Bou-Harb, Lakhdari, Binsalleeh ve Debbabi, Kasım 2013’te Cisco Systems tarafından oluşturulan bir raporda, kaynak portu “0” olan inrernet trafiğinde anormal bir artış olduğunun belirtilmesi üzerine; kaynak portu “0” olan İnternet trafiklerine yönelik, denetimsiz makine öğrenmesi algoritmaları kullanarak, benzer davranışlar gösteren trafikleri kümelemişlerdir [23]. Ayrıca bir dizi istatistiksel tabanlı davranışsal analizler uygulanarak, oluşturulan kümelerin stratejileri, teknikleri ve doğal davranışları incelenmiş; incelenen trafiğin herhangi bir zararlı yazılım ile bağlantısının olup olmadığı hususuna yönelik araştırma yapılmıştır.

Çalışmada 30 GB boyutunda darknet trafiği, 1.4 milyar DNS kaydı ve 30 milyon zararlı yazılım analiz raporu kullanılmıştır. Çalışmada belirtilen çeşitli zararlı yazılım örneklerinin, kaynak port olarak 0’ı kullandığı belirtilmiştir. İncelenen trafikte kaynak portu 0 olup; hedef portu 445 (Microsoft directory) olan 9000, hedef portu 22 (secure shell) olan 7000 ve hedef portu 3389 (remote desktop) olan 5500 tane trafik tespit edilmiştir. Data link, network ve transport katmanlarında, aşağıdaki tabloda belirtilen öznitelikler kullanılmıştır:

Çizelge 2.10. Port “0” trafiği incelemesinde kullanılan öznitelikler

1	Paket iletim süresi
2	Paket boyutu
3	Frame boyutu
4	Capture boyutu
5	İşaretli bayrak
6	IP başlık boyutu
7	IP bayrakları
8	IP bayrakları: reversed bit
9	IP bayrakları: do not fragment bit
10	IP bayrakları: more fragments bit
11	IP fragment offset
12	IP time to live
13	IP protokolü
14	TCP segment boyutu
15	TCP sequence number
16	TCP next sequence number
17	TCP acknowledgment number
18	TCP başlık boyutu
19	TCP bayrakları
20	TCP bayrakları: congestion window
21	TCP bayrakları:ECN-echo
22	Urgent flag
23	ACK flag
24	PUSH flag
25	RESET flag
26	SYN flag
27	FIN flag
28	TCP window size
29	UDP boyutu

Mirza, Awan ve Younas tarafından yapılan bu çalışmada, zararlı ve zararsız dosyalardan oluşan geniş bir veri setinden çıkarılan öznitelikler üzerinde SVM, karar ağaçları ve karar ağaçları üzerinde boosting algoritması birlikte kullanılmıştır [24]. Ayrıca Amazon web servisleri üzerinde barındırılan bulut tabanlı ölçeklenebilir bir mimari sunulmuştur.

Çalışmada kullanılan veri seti incelendiğinde, zararlı yazılımlara ait dosyaların çoğunluğunun Trojan ve reklam yazılımlarından (adware) oluştuğu görülmektedir.

Öznitelikleri ayıklamak maksadıyla, Python tabanlı bir araç geliştirilmiştir. Geliştirilen araç ile zararlı dosyalar üzerinde statik analiz gerçekleştirilmesine yarayan açık kaynak kodlu "PEframe" aracı (github.com/guelfoweb/peframe) birbirine entegre edilmiştir. Oluşturulan sistem ile birlikte, bir dosyanın statik analizi yapılmasına müteakip, JSON tabanlı dosyalar üretilmektedir. Bununla beraber VirusTotal'in özel uygulama programlama arayüzü (Application Programming Interface – API) de kullanılarak, elde edilen sonuçlar birleştirilmektedir. Test aşamasında "10-fold cross validation" metodu kullanılmıştır.

Elde edilen sonuçlara göre en başarılı eğri altı alan (Area Under Curve – AUC) oranını veren algoritma, karar ağacı üzerinde uygulanan boosting algoritması olmuştur. Elde edilen AUC oranları aşağıda sunulmuştur:

Çizelge 2.11. Çalışmada elde edilen AUC oranları

Metot	AUC (%)
Karar Ağacı	97,7
SVM	98,9
Boosting	99,6

Buczak ve Guven tarafından yapılan bu çalışmada, saldırı tespitine yönelik siber analiz için makine öğrenmesi ve veri madenciliği konularında, bir literatür taraması yapılmıştır [25]. Söz konusu literatür taraması incelendiğinde, akademik yayınlarda geçen makine öğrenmesi algoritmaları, bahse konu algoritmaların kullanılmasıyla elde edilen sonuçların paylaşıldığı akademik makaleler ve ne tür veri setleri kullanıldığına dair hususların yer aldığı görülmektedir. İncelenen makalede yer alan veri setlerine bakıldığında; bazı akademik çalışmada kullanılan veri setleri, kdd cup 99 veri seti ve Cisco tarafından geliştirilen netflow ağ protokolünde kullanılan öğeler vb. özniteliklerin bulunduğu görülmektedir. Çalışmada yer verilen öznitelikler incelendiğinde; protokol, port, paket boyutları, bayraklar (SYN, ACK vb.), IP başlık bilgileri, iletim süreleri vb. gibi öznitelikler görülmektedir.

Makalenin geri kalan kısmı incelendiğinde; muhtelif makine öğrenmesi algoritmaları, hangi yazarların hangi metotları ve veri setlerini kullandığı, kullanılan algoritmaların çalışma süresi karmaşıklığı gibi hususlara yer verildiği görülmektedir.

McLaren, Russell ve Buchanan tarafından yapılan çalışmada; zararlı yazılımların komuta kontrol sunucuları ile oluşturdukları trafik üzerinden, sınıflandırma algoritmaları ile veri madenciliği yaklaşımı kullanılarak bir analiz gerçekleştirilmiştir [26].

Uygulama safhasında, çeşitli zararlı yazılımların trafikleri veri setine dahil edilerek; akış tabanlı ve paket tabanlı olmak üzere sınıflandırma işlemi yapılmıştır. Sınıflandırma işlemi yapılırken, her iki şekilde de karar ağacı kuralları çıkarılmıştır. Yalnızca TCP paketlerinin analiz edilmesiyle, paket tabanlı sınıflandırma işlemi sonucunda, aşağıdaki tabloda belirtilen kurallar çıkarılmıştır:

Çizelge 2.12. Paket tabanlı sınıflandırma işlemi sonucunda elde edilen kurallar

Zararlı Yazılım	Oluşan Ağaç Yapısı	FP (%)
Zeus	1. TCP window size> 64376 2. TCP window size <=64376 ve başlık uzunluğu > 30 3. TCP window size <= 64376 ve header length > 30 ve TCP window size scale factor > 1,5	0,0
Zeus Outbound	1. header length < 30	0,0
Cutwail	1. TCP window size değeri > 64887 2. TCP window size değeri <= 64887 ve hedef port <= 52	0,0
Zeus Gameover	1. frame uzunluğu <= 57 2. frame uzunluğu > 57 ve TCP window size değeri > 64245 3. frame uzunluğu > 57 ve TCP window size değeri <= 64245 ve başlık uzunluğu > 30	0,0
Citadel	1. frame uzunluğu <= 57 ve kaynak port > 1055 2. frame uzunluğu > 57 ve iletim süresi > 4967 ve başlık uzunluğu < 24	5,5
AlienSpy	1. iletim süresi < 0,065 ve frame uzunluğu < 59 2. iletim süresi > 0,065 ve frame uzunluğu < 72 ve FIN flag <= 0,5	5,1
IRCBot	Eksik TCP paketleri	
xTreme RAT	1. frame uzunluğu < 59 ve iletim süresi > 0	2,8

Yukarıdaki tablo incelendiğinde, “TCP windows size” ve “header length” özniteliklerinin değerlerine göre oluşturulan karar ağacında, Zeus zararlı yazılımının trafiğinin tespit edilmesinde, %0’lık false positive oranı dikkat çekici bulunmuştur. Aynı şekilde Zeus Gameover zararlı yazılımının trafiğinin tespit edilmesine yönelik oluşturulan karar ağacında sadece “frame length”, “TCP window size” ve “header length” özniteliklerinin kullanılarak %0’lık false positive oranının elde edilmesi dikkat çekicidir.

Aynı çalışmada yalnızca TCP paketlerinin analiz edilmesiyle, akış tabanlı sınıflandırma işlemi sonucunda, aşağıdaki tabloda belirtilen kurallar çıkarılmıştır:

Çizelge 2.13. Akış tabanlı sınıflandırma işlemi sonucunda elde edilen kurallar

Zararlı Yazılım	Oluşan Ağaç Yapısı	FP (%)
Zeus	Öznitelik bulunamadı	
Zeus Outbound	Öznitelik bulunamadı	
Cutwail	1. hedef port < 39 ve > 12	0,0
Zeus Gameover	1. Alınan paket boyutu > 56 ve 80 < gönderilen paket boyutu < 897 2. Alınan paket boyutu <= 56 ve gönderilen paket boyutu <= 2242 ve gönderilen paket sayısı <= 2	8,0
Citadel	1. Alınan paket sayısı = 1 2. Hedef port <= 107 3. Gönderilen paket sayısı >= 4	5,5
AlienSpy	1. 784 < hedef port < 1112 2. 44 < gönderilen paket boyutu <= 50 ve alınan paket sayısı <= 10	0,0
IRCBot	Eksik veri	
xTreme RAT	1. Gönderilen paket boyutu > 1043885	50,0

Yukarıdaki tabloya bakıldığında, Cutwail zararlı yazılımının yalnızca “destination port” niteliği üzerinden oluşturulan karar ağacının %0’lık false positive oranına sahip olduğu; ayrıca AlienSpy zararlı yazılımının “destination port”, “IP bytes sent” ve “IP packets received” özniteliklerinin yer aldığı karar ağacı üzerinden %0’lık false positive oranına sahip olduğu görülmektedir.

Singh, Guntuku, Thakur ve Hota tarafından hazırlanan makalede, Hadoop, Hive ve Mahout gibi açık kaynak araçları kullanılarak; makine öğrenmesi yaklaşımı ile yarı gerçek zamanlı peer-to-peer botnet saldırılarının tespit edilmesine yönelik bir çalışma gerçekleştirilmiştir [27]. Bu makale ile;

1) Dinamik ağ trafiğinin özniteliklerini çıkarmaya ve ağ trafiğini dinlemeye yarayan Hive aracı kullanılarak; dağıtık bir framework oluşturmak,
2) Yarı gerçek zamanlı olarak, peer-to-peer botnet tespitinde kullanılmak üzere, random forest tabanlı karar ağacı modelini oluşturmak için Mahout aracının paralel işlem gücünün kullanımı amaçlanmıştır.

Çalışma genel olarak 3 modülden oluşmaktadır:

1) Paket önişleme için trafik dinleme (trafik sniffer) modülü
2) Öznitelik seti oluşturmak için öznitelik çıkarım (feature extraction) modülü
3) Zararlı trafiği tespit etmek için makine öğrenmesi modülü

Çalışmanın öznitelik çıkarım modülü incelendiğinde, aşağıdaki tabloda yer alan özniteliklerin yer aldığı görülmektedir:

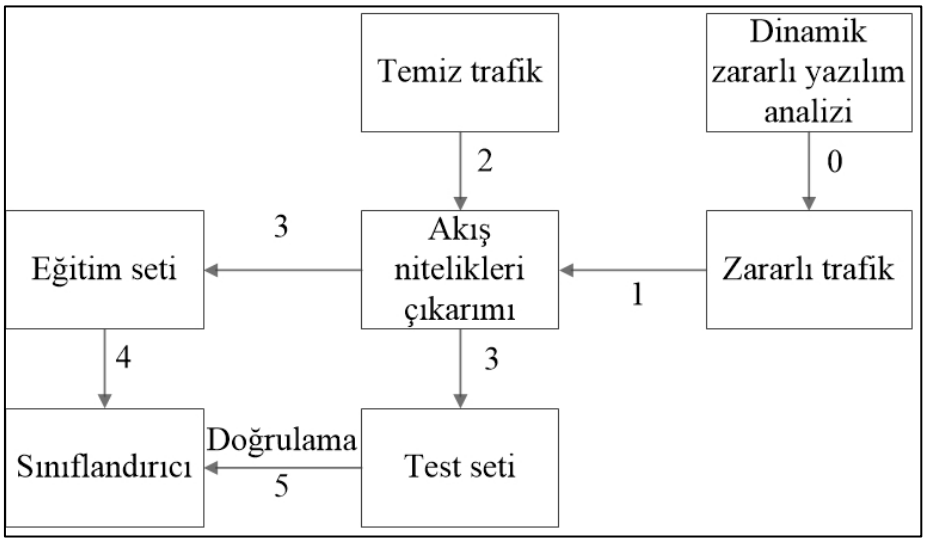
Çizelge 2.14. Botnet saldırı tespitinde kullanılan öznitelikler

Öznitelik	Açıklama
Srcip	Kaynak IP adresi
Srcport	Kaynak port
Dstip	Hedef IP adresi
Dstport	Hedef port
Proto	Protokol
Total_fpackets	İletilen toplam paket sayısı
Total_fvolume	İletilen toplam paket boyutu (byte)
Total_bpackets	Gelen toplam paket sayısı
Total_bvolume	Gelen toplam paket boyutu (byte)
Min_fpktl	İletilen en küçük paket boyutu (byte)
Mean_fpktl	İletilen ortalama paket boyutu (byte)
Max_fpktl	İletilen en büyük paket boyutu (byte)
Std_fpktl	İletilen paketlerin standart sapması (byte)
Min_bpktl	Gelen en küçük paket boyutu (byte)
Mean_bpktl	Gelen ortalama paket boyutu (byte)
Max_bpktl	Gelen en büyük paket boyutu (byte)
Std_bpktl	Gelen paketlerin standart sapması (byte)
Min_fiat	İletilen iki paket arasındaki minimum süre (milisaniye)
Mean_fiat	İletilen iki paket arasındaki ortalama süre (milisaniye)
Max_fiat	İletilen iki paket arasındaki maksimum süre (milisaniye)
Std_fiat	İletilen iki paket arasındaki sürenin standart sapması (milisaniye)
Min_biat	Gelen iki paket arasındaki minimum süre (milisaniye)
Mean_biat	Gelen iki paket arasındaki ortalama süre (milisaniye)
Max_biat	Gelen iki paket arasındaki maksimum süre (milisaniye)
Std_biat	Gelen iki paket arasındaki sürenin standart sapması (milisaniye)
Duration	Bağlantı süresi (mikrosaniye)
Fpsh_cnt	İletilen paketlerdeki PSH flag sayısı (UDP paketleri için “0”)
Bpsh_cnt	Gelen paketlerdeki PSH flag sayısı (UDP paketleri için “0”)
Furg_cnt	İletilen paketlerdeki URG flag sayısı (UDP paketleri için “0”)
Burg_cnt	Gelen paketlerdeki URG flag sayısı (UDP paketleri için “0”)
Total_bhlen	Gelen paketlerin toplam başlık boyutu (byte)
Total_fhlen	İletilen paketlerin toplam başlık boyutu (byte)

Veri setinin eğitimi aşamasında, botnet trafiği için Conficker, Kelihos-Hlux, Zeus, Storm ve Waledac gibi zararlı yazılımların, pcap formatındaki trafiği kullanılmıştır. Çalışmanın son bölümünde oluşturulan veri seti üzerinde random forest algoritması uygulanmış ve zararlı trafiği tespit etmede 99,8 TP oranı; 0,3 FP oranı elde edilmiştir.

Boukhtouta, Mokhov ve Lakhdari tarafından yapılan bu çalışmada; derinlemesine paket incelemesi (Deep Packet Inspection – DPI) ve flow paket başlıkları kullanılarak, zararlı yazılım sınıflandırma metotlarının karşılaştırması yapılmıştır [28].

Flow-based paket başlıkları üzerinden yapılan çalışmada J48 (karar ağacı), naive bayes, SVM ve boosting algoritmaları gibi çeşitli makine öğrenmesi teknikleri uygulanmıştır. Çalışmada kullanılan trafik örneği, çift yönlü (bidirectional) ve tek yönlü (unidirectional) flow paket başlıklarından toplanan özniteliklerden oluşmaktadır. Flow-based tespit yaklaşımı aşağıdaki şekilde gösterilmektedir:



Şekil 2.4. Akış tabanlı tespit yaklaşımı

Yukarıdaki şekilde de görüldüğü üzere, ilk aşamada dinamik zararlı yazılım analizinden (ThreatTrack Sandbox üzerinde çalıştırılarak) zararlı trafiği üretilerek ve normal trafik ("WISNET - Wireless and Secure Networks Research Lab" üzerinden) dahil edilerek, akış öznitelikleri çıkarılmaktadır. Elde edilen veri seti test ve eğitim olmak üzere parçalandıktan sonra sınıflandırma işlemi yapılmaktadır.

Veri setinde kullanılan çift yönlü ve tek yönlü flow öznitelikleri incelendiğinde; bir pcap dosyasından çeşitli katmanlardan direkt olarak alınabilecek öznitelikler olduğu gibi, hali hazırda pcap dosyasında yer alan özniteliklerin kullanılmasıyla yeni öznitelikler (maksimum/minimum/ortalama iletim süresi, iletim süresinin varyansı, toplam flag sayıları vb.) de üretilmektedir:

Çizelge 2.15. Çift yönlü akış öznitelikleri

1	Akış süresi
2	İletilen paket sayısı
3	Gelen paket sayısı
4	Protokol
5	İletilen paketlerin en küçük iletim süresi
6	İletilen paketlerin en büyük iletim süresi
7	İletilen paketlerin ortalama iletim süresi
8	İletilen paketlerin iletim sürelerinin standart sapması
9	İletilen toplam paket boyutu
10	İletilen minimum paket boyutu
11	İletilen maksimum paket boyutu
12	İletilen ortalama paket boyutu
13	İletilen paket boyutlarının standart sapması
14	Gelen paketlerin en küçük iletim süresi
15	Gelen paketlerin en büyük iletim süresi
16	Gelen paketlerin ortalama iletim süresi
17	Gelen paketlerin iletim sürelerinin standart sapması
18	Gelen toplam paket boyutu
19	Gelen minimum paket boyutu
20	Gelen maksimum paket boyutu
21	Gelen ortalama paket boyutu
22	Gelen paket boyutlarının standart sapması

Çizelge 2.16. Tek yönlü akış öznitelikleri

1	Toplam paket sayısı
2	Akış süresi
3	Minimum iletim süresi (saniye)
4	İlk çeyrekteki paketlerin iletim süresi
5	İletim süresinin medyanı
6	İletim süresinin ortalaması
7	Üçüncü çeyrekteki paketlerin iletim süresi
8	Maksimum iletim süresi (saniye)
9	İletim sürelerinin varyansı

Çizelge 2.16. (devam) Tek yönlü akış öznitelikleri

10		Minimum kontrol veri boyutu
11		İlk çeyrekteki kontrol veri boyutu
12		Kontrol veri boyutunun medyanı
13		Kontrol veri boyutu ortalaması
14		Üçüncü çeyrekteki kontrol veri boyutu
15		Maksimum kontrol veri boyutu
16		Kontrol veri boyutunun varyansı
17		Boş olmayan paket sayısı
18		Toplam paket boyutu (byte)
19		Minimum ethernet paketi boyutu
20		İlk çeyrekteki ethernet paketi boyutu
21		Ethernet paket boyutlarının medyanı
22		Ortalama ethernet paketi boyutu
23		Üçüncü çeyrekteki ethernet paketi boyutu
24		Maksimum ethernet paketi boyutu
25		Ethernet paket boyutlarının varyansı
26		Minimum IP paket boyutu
27		İlk çeyreketi IP paket boyutu
28		IP paket boyutlarının medyanı
29		Ortalama IP paket boyutu
30		Üçüncü çeyreketi IP paket boyutu
31		Maksimum IP paket boyutu
32		IP paket boyutlarının varyansı
33		Toplam ACK paket sayısı
34		Toplam PUSH paket sayısı
35		Toplam SYN paket sayısı
36		Toplam FINE paket sayısı
37		Toplam URGENT paket sayısı
38		Toplam URGENT paket boyutu
39		Minimum TCP segment boyutu
40		Maksimum TCP segment boyutu
41		TCP segment boyutu ortalaması
42		Minimum TCP window boyutu
43		Maksimum TCP window boyutu
44		Ortalama TCP window boyutu
45		Toplam boş TCP window paket sayısı

Çeşitli denetimli makine öğrenmesi tekniklerinin uygulanması neticesinde, J48 ve boosted J48 algoritmalarının en iyi sonucu verdiği (%99'un üzerinde doğruluk oranı) tespit edilmiştir.

Çizelge 2.17. Elde edilen sonuçlar

Algoritma	Doğruluk (%)	FP (%)	FN (%)	Precision (%)
Karar Ağacı	99,5	0,2	0,2	>99,0
Boosted Karar Ağacı	99,7	0,1	0,1	>99,0
Naive Bayes	64,8	-	-	-
Boosted Naive Bayes	64,8	-	-	-
SVM	93,5	4,5	1,9	-

Mimura, Otsubo, Hidehiko ve Hidema, proxy sunucu logları üzerinde http tabanlı uzaktan erişim trojeni (Remote Access Trojan - RAT) tespitine yönelik bir çalışma yapmışlardır [29]. Geliştirilen yöntem ile proxy sunucu loglarında yer alan;

- Tarih,
- Zaman,
- Metot (GET, POST),
- URL,
- User agent bilgileri,
- HTTP durum kodu (status code),
- İstemciye dönen nesne boyutu bilgileri

kullanılmaktadır.

Log satırlarından öznitelik vektörleri çıkarılırken, aşağıdaki adımlar izlenmektedir:

1) HTTP durum kodu başarılı olan satırların belirlenmesi
2) Satırlardan gerekli içeriklerin ayıklanması
3) Demet dizisine dönüştürülecek satır sayısının (N) belirlenmesi
4) Her tam öznitelikli alan adı (Fully Qualified Domain Name – FQDN) için, önceden belirlenmiş her bir sayı (N) için log kayıtlarının demetlenmesi
5) Demetlerden öznitelik vektörlerinin oluşturulması

Oluşturulan öznitelik vektörleri aşağıdaki tabloda sunulmuştur:

Çizelge 2.18. Öznitelik vektörleri

	Öznitelik Vektörleri
v1	İstemciye en çok gelen nesne boyutu (byte)
v2	İstemciye en çok gelen nesne boyutu sayısı (v1 sayısı)
v3	Loglanan süre bilgileri içerisinde en sık geçen aralık (saniye)
v4	Loglanan süre bilgileri içerisinde en sık geçen aralık sayısı (v3 sayısı)
v5	HTTP istekleri içinde en sık geçen path uzunluğu
v6	HTTP istekleri içinde en sık geçen path uzunluğu sayısı (v5 sayısı)
v7	POST metodu kullanan HTTP istek sayısı
v8	User agent bilgisi uzunluğu

Öznitelik vektörlerinin belirlenmesiyle beraber oluşturulan veri seti üzerinde, Random Forest (RF) ve SVM algoritmaları kullanılmıştır. Söz konusu algoritmaların kullanılmasıyla elde edilen sonuçlar, aşağıda gösterilmiştir:

Çizelge 2.19. Elde edilen sonuçlar

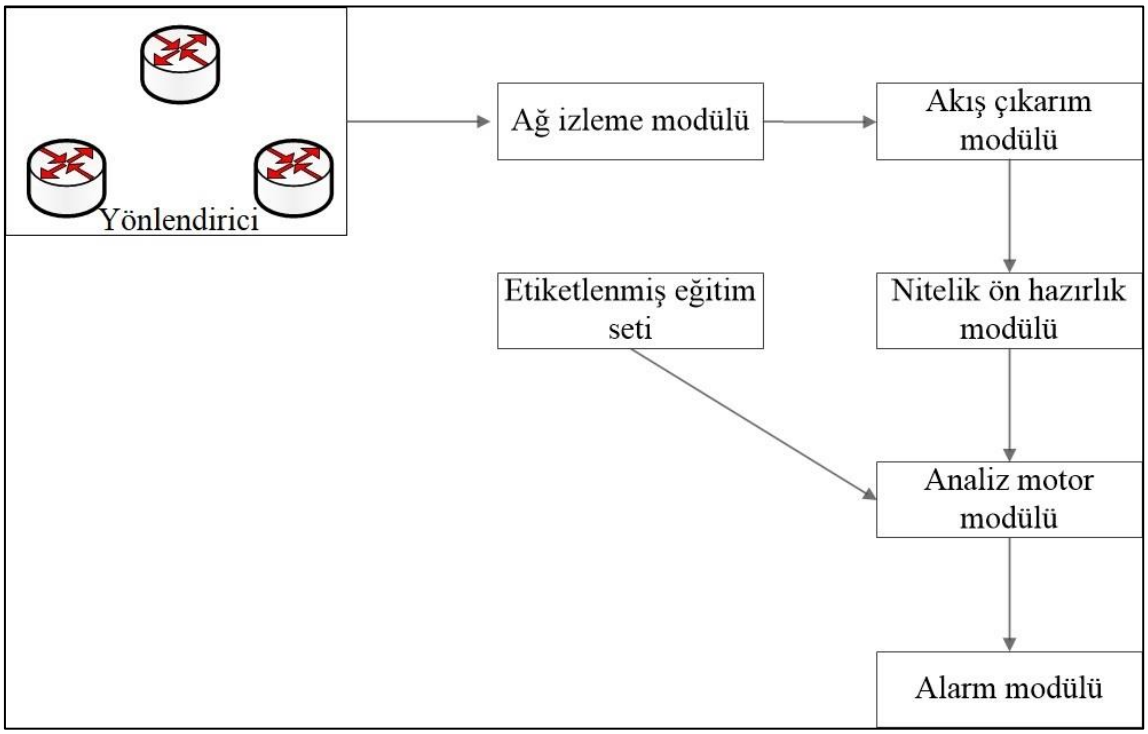
Algoritma	Çalışma Süresi (sn)	Doğruluk (%)	FP Oranı (%)
SVM	105	98,6	<0,1
Random Forest	35	99,2	<0,1

Yukarıdaki tablo incelendiğinde, önceden belirlenen satır sayılarına (N) göre sonuçların değişiklik gösterdiği görülmektedir. Sayı az tutuldukça tespit oranı (Detection Rate - DR) artmakla birlikte, false positive tespit sayısının da arttığı görülmektedir. Sayının fazla belirlenmesi durumunda ise false positive sayısının azalmasıyla beraber, tespit oranının da azaldığı görülmektedir. Makalede bu durum şu şekilde açıklanmaktadır:

1) Sayının az tutulması durumunda, kısa vadeli davranışları analiz etmek gereklidir. Kısa vadeli davranışlar da yeterince karakteristik davranışlar içermeyebilmektedir.
2) Sayının fazla tutulması durumunda, uzun vadeli davranışı analiz etmek gereklidir. Uzun vadeli davranışlarda da karakteristik olmayan davranışlar sergilenebilmektedir. Bu nedenle gerekli davranışsal özellikler tam manasıyla çıkarılamamaktadır. Uzun süreli RAT aktivitelerini tespit etmek zorlaşmaktadır.

Radoglou-Grammatikis ve Sarigiannidis'in yapmış olduğu bu çalışmada; Android mobil cihazlarda anormal davranışları tespit etmeye yarayan bir saldırı tespit sistemi (Intrusion Detection System - IDS) geliştirilmiştir [30]. Bahse konu IDS ile mobil cihazların ağ trafiği sürekli olarak izlenmekte ve Cisco tarafından geliştirilen, IP trafik bilgisini toplayan

"NetFlow" ağ protokolünün çeşitli öznitelikleri toplanmaktadır. YSA algoritması ile veri akışları toplanarak, bir saldırı olup olmadığı tespit edilmeye çalışılmaktadır. Önerilen IDS mimarisi şekli aşağıda gösterilmiştir:



Şekil 2.5. Önerilen IDS mimarisi

Yukarıdaki şekilde de görüldüğü üzere, sistem genel olarak 5 aşamadan oluşmaktadır. Öznitelik çıkarım modülünde, NetFlow bilgisi kapsamında;

- Ortalama NetFlow boyutu
- Ortalama paket sayısı
- Ortalama paket boyutu
- NetFlow süresi

öznitelikleri kullanılmıştır.

Analiz modülünde YSA ile sınıflandırma yapılmıştır. Çıktı katmanından çıkan değer, 0,5'ten büyük ise "anormal", küçük ise "normal" olarak sınıflandırılmıştır. Deneysel sonuçlar incelendiğinde, yaklaşık %85 doğruluk oranı elde edilmiştir.

Boero, Marchese ve Zappatore tarafından yayınlanmış olan bu makalede, yazılım tanımlı ağlar (Software Defined Networking - SDN) yaklaşımı ile elde edilen öznitelikler ve bu yaklaşım kullanılmadan (non-SDN) elde edilen öznitelikler üzerinden SVM algoritması kullanılarak, bir IDS geliştirilmiştir [31]. Çalışma sonucunda farklı oranlardaki eğitim ve test veri seti ile SDN ve non-SDN özniteliklerinin kullanılmasıyla elde edilen sonuçlar karşılaştırılmıştır. Non-SDN yaklaşımı ile birlikte kullanılan öznitelikler aşağıdaki tabloda sunulmuştur:

Çizelge 2.20. Saldırı tespit sisteminde kullanılan öznitelikler

Öznitelikler	Açıklama
Num_Pack	Paket sayısı
Tot_Byte_Flow	Paket boyutu (byte)
Flow_Duration	Akış süresi
Byte_Rate	Birim zamandaki byte oranı
Packet_Rate	Birim zamandaki paket oranı
Delta_Mean	Paketlerin ortalama iletim süresi
Delta_Std	İletim süresinin standart sapması
LE	Paket uzunluklarının entropisi
DPL	Akışın toplam paket sayısına bölünen, aynı uzunluğa sahip paketlerin alt kümelerinin toplam sayısı
First_Len	İlk paketin uzunluğu
Max_Len	En uzun paketin uzunluğu
Min_Len	En kısa paketin uzunluğu
Mean_Len	Ortalama paket uzunluğu
Std_Len	Paket uzunluğunun standart sapması

SDN yaklaşımı ile yukarıdaki tabloda belirtilen özniteliklerden yalnızca 7 adedi kullanılabilmektedir:

- Num_Pack
- Tot_Byte_Flow
- Flow_Duration
- Byte_Rate
- Packet_Rate
- First_Len
- Mean_Len

Veri setinde kullanılmak üzere normal trafik, yazarlar tarafından kendi laboratuvarlarındaki ağdan elde edilmiştir. Zararlı yazılım trafiği için ise çeşitli İnternet sitelerinden botnet, uzaktan erişim trojeni (Remote Access Trojan - RAT), trojan ve reklam yazılımlarının trafikleri toplanmıştır. Çalışma sonucunda elde edilen TP ve FP oranları aşağıdaki tabloda gösterilmektedir:

Çizelge 2.21. Çalışma sonucunda elde edilen TP ve FP oranları

Deney	TP (%)	FP (%)	Sınıf
%33 Eğitim, tüm öznitelikler	99,0	1,5	zararlı
	98,5	1,0	normal
%50 Eğitim, tüm öznitelikler	98,9	1,1	zararlı
	98,9	1,1	normal
%33 Eğitim, SDN öznitelikleri	98,4	13,8	zararlı
	86,2	1,6	normal
%50 Eğitim, SDN öznitelikleri	98,4	12,4	zararlı
	87,6	1,6	normal

Cuzzocrea, Martinelli, Mercaldo ve Vercelli'nin yapmış olduğu çalışmada, J48, J48Consolidated, BayesNet, jRip, OneR ve REPTree algoritmaları kullanılarak, bir sunucunun Tor trafiği kullanıp kullanmadığı tespit edilmeye çalışılmıştır [32]. Ayrıca pcap dosyalarından akış (flow) üretmek ve bu akışlardan öznitelik çıkarmak için geliştirilen açık kaynak kodlu "ISCXFlowMeter" aracı kullanılmıştır. Veri setinde 23 tane öznitelik kullanılmıştır:

- fiat: iki paket arasındaki ileri yönde (forward) iletim süresi (ortalama, minimum, maksimum ve standart sapma değerleri)
- biat: iki paket arasındaki geri yönde (backward) iletim süresi (ortalama, minimum, maksimum ve standart sapma değerleri)
- flowiar: ileri ve geri yönde gönderilen paketlerin akış iletim süresi (ortalama, minimum, maksimum ve standart sapma değerleri)
- active: bir akışın boşta (idle) olmadan önce aktif olduğu süre miktarı
- idle: bir akışın aktif olmadan önce boşta olduğu süre miktarı
- fb psec: bir akışta saniyedeki byte miktarı
- fp psec: bir akışta saniyedeki paket sayısı

- duration: akış süresi

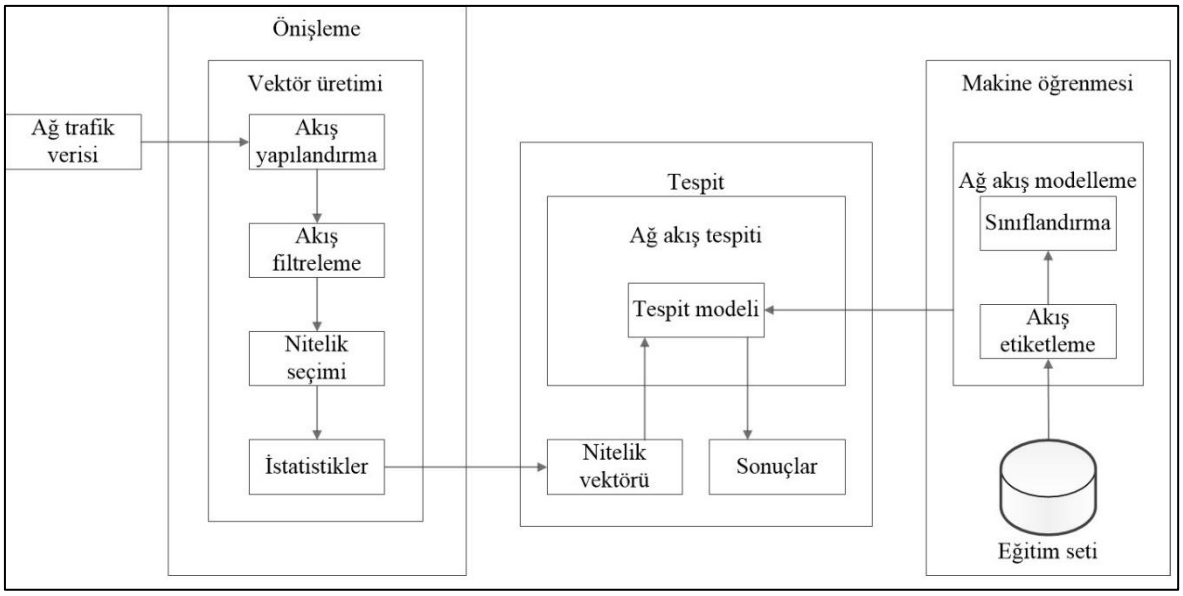
Yukarıda bahsedilen algoritmalar ve öznitelikler, WEKA programı üzerinde denenerek, aşağıdaki tabloda sunulan sonuçlar elde edilmiştir:

Çizelge 2.22. WEKA ile elde edilen sonuçlar

Algoritma	TP (%)	FP (%)	Precision (%)	Recall (%)	F-Ölçütü (%)	ROC (%)	Sınıf
J48 Karar Ağacı	99,9	0,7	99,8	99,9	99,9	99,9	nonTOR
	99,3	0,1	99,7	99,3	99,5	99,9	TOR
Bayes Ağları	98,2	2,0	99,6	98,2	98,9	99,9	nonTOR
	98,0	1,8	91,8	98,0	94,8	99,9	TOR
Jrip	100,0	0,0	100,0	100,0	100,0	100,0	nonTOR
	100,0	0,0	99,8	100,0	99,9	100,0	TOR
OneR	99,7	0,0	100,0	99,7	99,9	99,9	nonTOR
	100,0	0,3	98,7	100,0	99,4	99,9	TOR
RepTree	98,7	3,2	99,7	98,7	99,2	99,9	nonTOR
	98,2	1,9	92,3	98,3	95,1	99,8	TOR

Yukarıdaki tabloda da görüldüğü üzere kullanılan algoritmaların tamamından, yaklaşık %98 - %100 oranında True Positive sonucu elde edilmiştir.

Cheng, Chen, Zhang, Li ve Sang'ın yapmış olduğu bu çalışmada, Android platformundaki mobil ağ akışları üzerinde bilgi hırsızlığının tespit edilmesi amaçlanmıştır [33]. Ağ akışlarını sınıflandırmada, Random Forest algoritması kullanılmış olup; zaman serileri analizinde, iki zaman aralığı arasındaki benzerliği ölçmek maksadıyla dinamik zaman bükmesi (DTW - Dynamic Time Warping) algoritmasından faydalanılmıştır. Çalışma neticesinde %95'in üzerinde doğruluk oranı elde edilmiştir. Çalışmada kullanılan ağ akış tespiti şeması aşağıda sunulmuştur:



Şekil 2.6. Ağ akış tespiti şeması

Verma ve Ranga, k-means kümeleme ve kNN sınıflandırma algoritmalarını kullanarak; CIDDS-001 (Coburg Network Intrusion Detection Dataset) veri seti üzerinde ağ saldırı tespit sistemi geliştirmeye yönelik analiz çalışması yapmışlardır [34]. Söz konusu veri seti 14 tane niteliğe sahip olmakla birlikte, bu çalışmada 12 adedi:

Çizelge 2.23. Ağ saldırı tespit sisteminde kullanılan öznitelikler

Öznitelik	Açıklama
Src IP	Kaynak IP adres
Src Port	Kaynak port
Dest IP	Hedef IP adres
Dest Port	Hedef port
Proto	Ulaşım katmanı protokolü (ICMP, TCP, UDP)
Date first seen	Akışın başlama zamanı
Duration	Akış süresi
Bytes	İletilen byte miktarı
Packets	İletilen paket sayısı
Flags	TCP bayrakları
Class	Sınıf etiketi (normal, saldırı, kurban, şüpheli, bilinmeyen)
Attack Type	Saldırı türü (PortScan, DoS, Bruteforce, Ping scan)
AttackID	Tekil saldırı ID'si
Attack Description	Belirlenen saldırı parametreleri hakkında ek bilgi.

Sınıflandırma işlemi gerçekleştirilirken WEKA programı kullanılmış ve kNN algoritması için yukarıdaki tabloda belirtilen sınıf etiketleri dikkate alınmıştır. Sınıflandırma neticesinde, her iki algoritma için de %99'un üzerinde doğruluk oranı elde edilmiştir.

Çizelge 2.24. Elde edilen sonuçlar

Algoritma	Doğruluk (%)	TP (%)	FP (%)	F1 Ölçütü (%)	Precision (%)
kNN (k=1)	99,5	100,0	0,0	100,0	100,0

Shah ve Issac tarafından yapılan bu çalışmada, açık kaynak IDS olan Snort ve Suricata araçlarının (varsayılan konfigürasyonlar ile) performans (CPU, bellek kullanımı, hız) ve doğruluk oranlarına yönelik elde edilen sonuçlar karşılaştırılmıştır [35]. Ayrıca Snort aracının muhtelif makine öğrenmesi algoritmaları eklentileri ile de kullanılması sağlanarak, elde edilen doğruluk oranlarının analizi ve karşılaştırılması yapılmıştır. Zararlı ağ trafiği üretilirken Kali Linux platformu üzerinde, Metasploit Framework kullanılmıştır. Normal ağ trafiği üretmek için ise Ostinato, Nmap ve Nping araçları kullanılmıştır.

Herhangi bir eklenti kullanmadan, çalışma hızı bakımından, Suricata'nın Snort'a göre daha iyi bir performans gösterdiği; doğruluk oranı bakımından ise Snort'un daha iyi sonuç verdiği tespit edilmiştir. 10 Gbps bant genişliğine sahip trafik üzerinde, 12 saatlik bir çalışma ile Snort'un ortalama %55,2, Suricata'nın ise ortalama % 74,3 oranında false positive alarm ürettiği belirlenmiştir.

Çalışmanın devamında, yüksek çıkan false positive oranlarını azaltmak maksadıyla, Snort aracı muhtelif makine öğrenmesi algoritmaları eklentileri ile kullanılmıştır. Ayrıca aşağıda belirtilen 5 farklı makine öğrenmesi algoritmalarının her biri de tek başına WEKA ortamında test edilmiştir:

- Destek vektör makineleri
- Karar ağaçları
- Bulanık mantık
- BayesNet
- NaiveBayes

Yukarıda bahsedilen makine öğrenmesi algoritmaları test edilirken, 3 farklı veri seti kullanılmıştır:

- NSA Snort IDS Alert Logs

- DARPA IDS Dataset
- NSL-KDD IDS Dataset

Elde edilen sonuçlar aşağıdaki tabloda gösterilmektedir:

Çizelge 2.25. Her bir veri seti için elde edilen oranlar

Veri Seti	Algoritma	Doğruluk (%)	TP (%)	FP (%)
Veri Seti - 1	SVM	95,6	96,8	0,7
	Karar Ağacı	82,0	79,2	2,9
	Bulanık Mantık	92,3	94,5	0,2
	Bayes Ağları	73,0	65,0	3,5
	Naive Bayes	70,0	62,0	3,0
Veri Seti - 2	SVM	94,2	97,0	0,5
	Karar Ağacı	85,0	81,1	1,9
	Bulanık Mantık	94,0	92,0	1,6
	Bayes Ağları	71,2	63,0	5,1
	Naive Bayes	71,0	65,0	6,0
Veri Seti - 3	SVM	95,4	97,3	3,1
	Karar Ağacı	81,2	78,0	10,0
	Bulanık Mantık	94,0	95,0	4,0
	Bayes Ağları	74,0	69,0	8,0
	Naive Bayes	79,0	70,0	7,6

Yukarıdaki tabloda, doğruluk oranları incelendiğinde, en iyi sonucu SVM algoritmasının verdiği görülmektedir. Snort aracının SVM, bulanık mantık ve karar ağacı eklentileri ile birlikte kullanılmasıyla, aşağıda gösterilen sonuçlar elde edilmiştir:

Çizelge 2.26. Snort aracının SVM, bulanık mantık ve karar ağacı eklentileri ile birlikte kullanılmasıyla elde edilen sonuçlar

Metot	FP Oranı	FN Oranı
Snort - SVM Eklentisi	16,9	4,1
Snort - Bulanık Mantık Eklentisi	39,4	3,7
Snort - Karar Ağacı Eklentisi	51,9	7,2

Çalışmada son olarak, SVM ve bulanık mantık hibrid eklentisi ve optimize edilmiş SVM ile birlikte firefly (ateş böceği) algoritması eklentileri kullanılarak, elde edilen sonuçlar, aşağıdaki tabloda sunulmuştur:

Çizelge 2.27. SVM ve bulanık mantık hibrid eklentisi ve optimize edilmiş SVM ile birlikte firefly (ateş böceği) algoritması eklentileri kullanılarak elde edilen sonuçlar

Metot	FP (%)	FN (%)
Snort - SVM ve Bulanık Mantık Hibrid Eklentisi	13,0	3,2
Snort - Optimize Edilmiş SVM ve Ateş Böceği Algoritması Eklentisi	8,6	2,2

Chen, Yan, Han, S. Wang, Peng, L. Wang ve Yang mobil cihazlara yönelik geliştirilen zararlı yazılımların tespiti üzerine bir çalışma gerçekleştirmişlerdir [36]. Çalışmada kullanılan öznitelikler aşağıdaki tabloda gösterilmiştir:

Çizelge 2.28. Zararlı yazılım tespitinde kullanılan öznitelikler

Öznitelik	Açıklama
UpBytes	Uplink trafik akışındaki byte miktarı
DownBytes	Downlink trafik akışındaki byte miktarı
UpPkgNum	Upstream trafik akışındaki paket sayısı
DownPkgNum	Downstream trafik akışındaki paket sayısı
EveryUpBytes	Uplink trafik akış paketlerindeki ortalama byte miktarı
EveryDownBytes	Downlink trafik akış paketlerindeki ortalama byte miktarı

Çalışmada WEKA programı üzerinde SVM, C4.5 karar ağacı, Grading, NaiveBayes, BayesNet ve Adaboost algoritmaları denenmiştir.

Dengesizlik oranına (Imbalanced ratio - IR) göre yukarıda belirtilen algoritmalardan elde edilen doğruluk oranları, aşağıdaki tabloda gösterilmiştir:

Çizelge 2.29. Dengesizlik oranına göre elde edilen doğruluk oranları (%)

IR	Bayes Ağları	SVM	C4.5	Grading	Adaboost	Naive Bayes
100	99,5	99,5	99,9	99,0	99,9	81,8
200	99,7	99,7	99,9	99,5	99,8	42,2
300	99,8	99,7	99,6	99,6	99,8	52,1
400	99,8	99,7	99,9	99,7	99,9	58,9
500	99,8	99,8	99,9	99,8	99,8	85,6
600	99,8	99,8	99,9	99,8	99,8	95,6
700	99,8	99,9	99,9	99,8	99,9	84,9
800	99,9	99,9	99,9	99,8	99,9	64,8
900	99,8	99,8	99,9	99,8	99,9	51,8
1000	99,9	99,9	99,9	99,9	99,9	93,5

Wang, Yang, Wu ve Abawajy tarafından, sistemlere zarar vermeden önce saldırının tespit edilmesine yönelik, “two seed expanding” kümeleme algoritmasının kullanılabilirliği araştırılmış ve k-means kümeleme algoritması ile karşılaştırılması yapılmıştır [37]. Yapılan çalışma genel olarak 5 aşamadan oluşmaktadır:

- Akış (flow) toplama
- Akıştan öznitelik çıkarımı
- Öznitelik üzerinde veri önişleme
- Denetimsiz öğrenme
- Kümeleme

Akış seviyesinde 14 tane öznitelik belirlenmiş olup; kullanılan öznitelikler aşağıdaki tabloda gösterilmiştir:

Çizelge 2.30. Saldırı tespitinde kullanılan öznitelikler

Öznitelik	Açıklama
pkts	Toplam paket sayısı
pkt-nopayload	Yüksüz paket sayısı
bytes	İletilen toplam byte miktarı
pay-bytes	Tüm yüklerdeki toplam byte miktarı
duration	Akış süresi
maxsz	Maksimum paket boyutu
minsz	Minimum paket boyutu
avgsz	Ortalama paket boyutu
stdsz	Paket boyutunun standart sapması
maxpy	Maksiumum yük boyutu
minpy	Minimum yük boyutu
avgpy	Ortalama yük boyutu
stdpy	Yük boyutunun standart sapması
flag	Bayraklar (ack, fin, reset, push vb.)

Kullanılan veri seti üzerinde, yukarıdaki tabloda belirtilen öznitelikler kullanılarak aşağıdaki tabloda sunulan sonuçlar (F-Ölçütü) elde edilmiştir:

Çizelge 2.31. Elde edilen sonuçlar

Algoritma	F-Ölçütü (%)
Seed Expanding	98,0
K-means - k = 1	92,1
K-means - k = 2	95,3
K-means - k = 3	95,3
K-means - k = 4	95,4
K-means - k = 5	95,5
K-means - k = 6	95,5
K-means - k = 7	96,4
K-means - k = 8	96,4

Yukarıdaki tablo incelendiğinde, seed expanding algoritmasının k-means algoritmasına göre daha iyi sonuçlar verdiği görülmektedir.

Niu, Zhang, Yang, Zhu ve Ren tarafından, mobil cihazlardaki DNS kayıtları üzerinde, APT zararlı yazılımlarının tespitine yönelik bir çalışma yapılmıştır [38]. Çalışma kapsamında;

- Mobil cihazlardan elde edilen yaklaşık 3000.000 tane DNS kaydından, 15 tane öznitelik belirlenmiştir.
- Alan adının Alexa sıralaması ve VirusTotal sonuçlarına göre her bir alan adına skor verilmiştir.
- Son olarak, komuta kontrol alan adlarının tespiti için, elde edilen veri seti üzerinde global anormal ormanı (Global Abnormal Forest - GAF), yerel aykırı faktörü (Local Outlier Factor - LOF), kNN ve izolasyon ormanı (Isolation Forest - iForest) algoritmaları çalıştırılarak; elde edilen sonuçlar arasında karşılaştırma yapılmıştır.

Çalışmada belirlenen öznitelikler aşağıdaki tabloda sunulmuştur:

Çizelge 2.32. APT tespitinde kullanılan öznitelikler

Öznitelik Seti	Öznitelik
DNS istek ve cevap tabanlı öznitelikler	Tekil kaynak IP adresi sayısı
	Aynı alan adının kullandığı tekil IP adresi sayısı
	Aynı ülkeye ait IP adresleri
Alan adı tabanlı öznitelikler	Alexa sıralaması
	Alan adı uzunluğu
	Alan adı seviyesi
Zaman tabanlı öznitelikler	İstek (request) sıklığı
	Reaksiyon süresi
	Tekrar eden desenler
Whois tabanlı öznitelikler	Kayıt süresi
	Aktiflik süresi
	Güncelleme süresi
	DNS sayısı

Yukarıda bahsedilen algoritmaların kullanımı neticesinde; Çizelge 2.33'te görüldüğü üzere GAF algoritmasının en kısa sürede çalıştığı ve en yüksek genel tanıma oranına sahip olduğu görülmüştür:

Çizelge 2.33. Elde edilen sonuçlar ve çalışma süreleri

Algoritmalar	Genel Tanıma oranı (%)	FP (%)	FN (%)	Süre (sn)
iForest	88,3	5,2	6,5	17
LOF	76,5	16,9	10,9	973
kNN	67,4	20,0	12,6	4573
GAF (information entropi ile birlikte)	98,3	1,3	0,4	18
GAF	93,9	5,6	0,4	15

Cao, Drabeck ve He tarafından, zararlı yazılımların komuta kontrol sunucularına tespit etmek için çeşitli öznitelik kombinasyonlarıyla 8 farklı durum üzerinde Adaboost algoritması denenmiştir [39]. Çalışmada kullanılan 8 farklı durumda kullanılan öznitelikler setleri şu şekildedir:

- Tüm öznitelikler
- DNS öznitelikleri
- HTTP öznitelikleri

- Zaman/sunucu tabanlı öznitelikler
- Başarısız (failure) istek öznitelikleri
- Seyrek gidilen sunucu öznitelikleri
- Sık gidilen sunucu öznitelikleri
- Seçilmiş öznitelikler (Korelasyon analizi sonucu elde edilen öznitelikler)

Veri seti için, içinde virus, worm, spyware, adware, trojan, backdoor vb. türde zararlı yazılım yer aldığı pcap trafikleri kullanılmıştır. Yukarıda bahsedilen 8 durum için elde edilen false positive ve false negative oranları aşağıdaki tabloda sunulmuştur:

Çizelge 2.34. Her bir durum için elde edilen FP ve FN oranları

Durum	FP (%)	FN (%)
Tüm Öznitelikler	0,7	3,4
DNS Öznitelikleri	2,8	10,8
HTTP Öznitelikleri	3,6	10,1
Zaman/Sunucu Öznitelikleri	1,6	4,3
Başarısızlık Öznitelikleri	4,9	60,3
Seyrek Öznitelikler	1,7	6,3
Seyrek Olmayan Öznitelikler	3,7	26,2
57 tane Seçilmiş Öznitelik	1,6	4,0

Tabloda görüldüğü üzere en iyi sonuç, özniteliklerin tamamının kullanılmasıyla elde edilmiştir. Ayrıca incelenen makalede, bir takım gözlemlere yer verilmiştir:

- Birçok zararlı yazılım, periyodik trafik desenine sahiptir.
- Zararlı yazılımlar, yüksek oranda başarısız sonuçlanan trafik üretmektedir.
- Zararlı yazılım trafiği ve normal trafik, faklı zaman dinamiklerine sahip olmakla beraber, kullanılan protokoller de genellikle farklıdır.
- Normal trafikler genellikle sık ziyaret edilen popüler web sitelerine yönelik olmaktadır.

Hodo, Bellekens, Iorkyase, Hamilton, Tachtatzis ve Atkinson tarafından, “UNB-CIC (University of New Brunswick- Canadian Institute for Cybersecurity) Tor Network Traffic” veri seti kullanılarak; SVM ve YSA algoritmaları ile ağ trafiğinin Tor trafiği olup olmadığı tespit edilmeye çalışılmıştır [40].

Toplamda 28 tane öznitelik belirlenerek, söz konusu öznitelikler üzerinde CFS algoritması uygulanmış ve çalışmada kullanılmak üzere 10 tane öznitelik belirlenmiştir. Sınıflandırma işlemi yapılırken, hem 28 tane niteliğin tamamı, hem de CFS algoritmasıyla belirlenen 10 tane öznitelik kullanılmış ve elde edilen sonuçlar karşılaştırılmıştır.

Çizelge 2.35. Tor trafiği tespitinde kullanılan öznitelikler

Öznitelik	Açıklama
Source IP	Kaynak IP adresi
Source Port	Kaynak port
Destination IP	Hedef IP adresi
Destination Port	Hedef port
Protocol	Protokol
Flow Duration	Akış süresi (saniye)
Flow Bytes/s	Akıştaki byte miktarı
Flow Packets/s	Akıştaki paket sayısı
Flow IAT Mean	Paketlerin ortalama iletim süresi
Flow IAT Std	Paketlerin iletim süresinin standart sapma miktarı
Flow IAT Max	Paketlerin maksimum iletim süresi
Flow IAT Min	Paketlerin minimum iletim süresi
Fwd IAT Mean	İleri yönde iletilen paketlerin ortalama iletim süresi
Fwd IAT Std	İleri yönde iletilen paketlerin ortalama iletim süresinin standart sapma miktarı
Fwd IAT Max	İleri yönde iletilen paketlerin maksimum iletim süresi
Fwd IAT Min	İleri yönde iletilen paketlerin minimum iletim süresi
Bwd IAT Mean	Geri yönde iletilen paketlerin ortalama iletim süresi
Bwd IAT Std	Geri yönde iletilen paketlerin ortalama iletim süresinin standart sapma miktarı
Bwd IAT Max	Geri yönde iletilen paketlerin maksimum iletim süresi
Bwd IAT Min	Geri yönde iletilen paketlerin minimum iletim süresi
Active Mean	Bir akışın boşta (idle) olmadan önce aktif olduğu ortalama süre
Active Std	Bir akışın boşta (idle) olmadan önce aktif olduğu sürenin standart sapma miktarı
Active Max	Bir akışın boşta (idle) olmadan önce aktif olduğu maksimum süre
Active Min	Bir akışın boşta (idle) olmadan önce aktif olduğu minimum süre
Idle Mean	Bir akışın aktif olmadan önce boşta olduğu ortalama süre
Idle Std	Bir akışın aktif olmadan önce boşta olduğu sürenin standart sapma miktarı
Idle Max	Bir akışın aktif olmadan önce boşta olduğu maksimum süre
Idle Min	Bir akışın aktif olmadan önce boşta olduğu minimum süre

CFS algoritmasıyla, yukarıdaki tabloda belirtilen 28 öznitelikten;

- Destination Port
- Bwd IAT Mean

- Idle Max
- Fwd IAT Min
- Source Port
- Idle Min
- Flow Bytes/s
- Flow IAT Std
- Source IP
- Destination IP

öznitelikleri seçilmiştir.

Uygulanan algoritmalarla birlikte aşağıdaki tabloda gösterilen sonuçlar elde edilmiştir:

Çizelge 2.36. Her bir algoritma için elde edilen sonuçlar

	YSA (%)	CFS - YSA (%)	SVM (%)	CFS – SVM (%)
Tespit Oranı (Tor)	93,7	98,8	67,0	98,4
FP Oranı (Tor)	0,2	<0,1	2,3	1,8
Tespit Oranı (NonTor)	99,2	100,0	98,0	99,0
FP Oranı (NonTor)	1,6	1,2	32,8	2,6
Toplam Doğruluk	99,1	99,8	94,0	88,1

Genel doğruluk oranına bakıldığında, en iyi sonucu CFS algoritmasıyla seçilen 10 tane öznitelikle birlikte YSA algoritmasının verdiği gözlemlenmiştir.

Lashkari, A.Kadir, Gonzalez, Mbah ve Ghorbani tarafından, Android cihazlarda, yalnızca 9 tane öznitelik kullanılarak; zararlı yazılım trafiğinin tespitine ilişkin bir çalışma yapılmıştır [41]. Söz konusu çalışmada Random Forest, kNN, Karar Ağaçları, Regresyon ve Random Tree algoritmalarından yararlanılmıştır. Çalışmanın öznitelik çıkarımı aşamasında, Çizelge 2.37'de görüldüğü gibi CFS algoritması kullanılarak, öznitelik sayısı 9'a düşürülmüştür.

Çizelge 2.37. Çalışmada değerlendirilen öznitelikler

Davranışsal Tabanlı	
Öznitelik	Açıklama
Duration	Akış süresi
Byte Tabanlı	

Çizelge 2.37. (devam) Çalışmada değerlendirilen öznitelikler

Öznitelik	Açıklama
Total Forward bytes	Giden paket boyutu
Total Backward bytes	Gelen paket boyutu
Forward Header length	Giden paketlerdeki başlık boyutu
Backward Header length	Gelen paketlerdeki başlık boyutu
Paket Tabanlı	
Öznitelik	Açıklama
Total Foward Packets	Giden toplam paket sayısı
Total Backward Packets	Gelen toplam paket sayısı
Forward packet length (min, mean, max, std)	Giden paket boyutlarının minimum, maksimum, ortalama ve standart sapma değerleri
Backward packet length (min, mean, max, std)	Gelen paket boyutlarının minimum, maksimum, ortalama ve standart sapma değerleri
Zaman Tabanlı	
Öznitelik	Açıklama
Forward Arrival Time (min, mean, max, std)	Giden paket iletim sürelerinin minimum, maksimum, ortalama ve standart sapma değerleri
Forward Arrival Time (min, mean, max, std)	Gelen paket iletim sürelerinin minimum, maksimum, ortalama ve standart sapma değerleri
Idle Time (min, mean, max, std)	Bir akışın aktif olmadan önce boşta olduğu sürenin minimum, maksimum, ortalama ve standart sapma değerleri
Active Time (min, mean, max, std)	Bir akışın boşta olmadan önce aktif olduğu sürenin minimum, maksimum, ortalama ve standart sapma değerleri
Akış Tabanlı	
Öznitelik	Açıklama
Flow Packet Length (min, mean, max, std)	Bir akış boyutunun minimum, maksimum, ortalama ve standart sapma değerleri
Flow Forward Bytes	Bir akıştaki giden paketlerin ortalama boyutu
Flow Backward Bytes	Bir akıştaki gelen paketlerin ortalama boyutu
Akış Tabanlı (Önerilen)	
Öznitelik	Açıklama
Backward Variance Data Byte	Gelen paketlerin varyansı
Forward Variance Data Byte	Giden paketlerin varyansı
Flow FIN	FIN paketi sayısı
Idle (max, min)	Bir akışın aktif olmadan önce boşta olduğu sürenin minimum/maksimum değeri
Initial Window Forward	Giden paketlerdeki ilk pencere boyutu
Initial Window Backward	Gelen paketlerdeki ilk pencere boyutu
Segment Size Forward (Max, Min)	Giden paketlerdeki maks/min segment boyutu
Segment Size Backward (Max, Min)	Gelen paketlerdeki maksimum/minimum segment boyutu

Yukarıdaki tabloda belirtilen öznitelikler arasından, çalışmada kullanılan 9 tane öznitelik şu şekildedir:

Çizelge 2.38. Çalışmada kullanılmak üzere seçilen öznitelikler

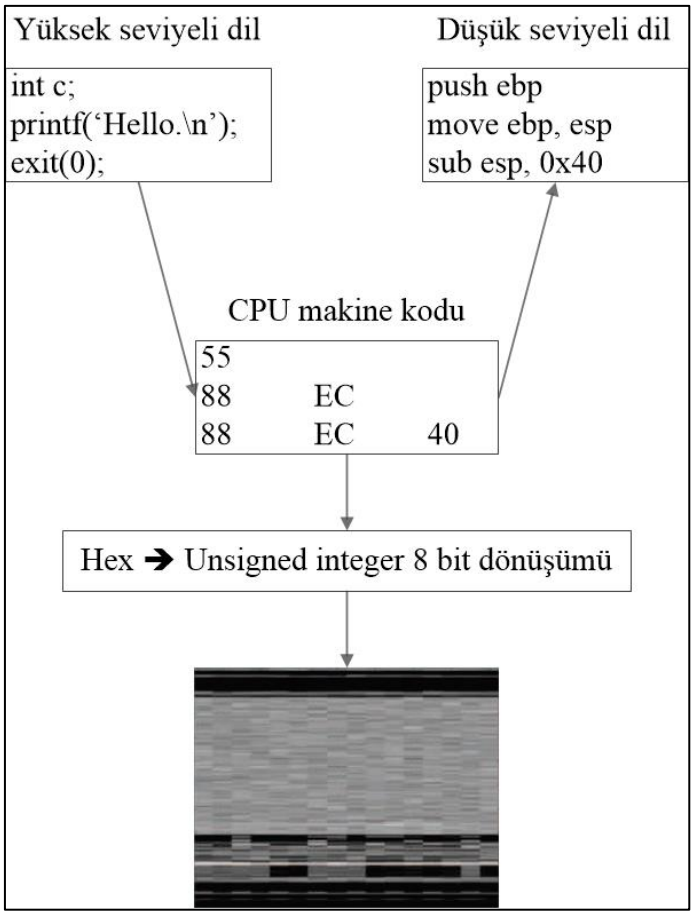
Öznitelik	Açıklama
Maximum flow packet length	Bir akıştaki maksimum byte miktarı
Minimum flow packet length	Bir akıştaki minimum byte miktarı
Backward variance data bytes	Geri yönde iletilen toplam byte miktarının varyansı
Flow FIN	FIN paketi sayısı
Flow forward bytes	Bir akıştaki ileri yönde iletilen ortalama byte miktarı
Flow backward bytes	Bir akıştaki geri yönde iletilen ortalama byte miktarı
Maximum Idle	Bir akışın, aktif olmadan önce boşta kaldığı maksimum süre
Initial window forward	İleri yönde ilk window içinde gönderilen toplam byte miktarı
Minimum segment size forward	İleri yönde gözlenen minimum segment boyutu

Sınıflandırma işleminin sonucunda, her bir algoritmada birbirine yakın doğruluk oranları gözlemlenmiş ve ortalama %93 doğruluk oranına ulaşılmıştır. Elde edilen sonuçlar aşağıdaki çizelgede sunulmuştur:

Çizelge 2.39. Elde edile sonuçlar

Algoritma	Doğruluk (%)	FP (%)	Precision (%)	Recall (%)
Random Forest	92,1	7,8	92,1	92,2
kNN	91,5	8,5	91,4	91,5
Karar Ağacı	91,6	8,4	91,5	91,6
Random Tree	91,6	8,4	91,5	91,6
Regresyon	90,4	9,6	90,3	90,4

Kebede, Djaneye-Boundjou, Narayanan, Ralescu ve Kapp tarafından, “Microsoft Malware Classification Challenge (BIG 2015)” veri seti üzerinde, zararlı yazılımların sınıflandırılmasına yönelik, geleneksel makine öğrenmesi algoritmaları ile deep learning (derin öğrenme) mimarisinin karşılaştırılması yapılmıştır [42]. Bunun için zararlı yazılım kodları makine kodlarına çevrilmiş ve her bir zararlı yazılım için makine kodlarıyla gri ölçekli resimler oluşturulmuştur.



Şekil 2.7. Zararlı yazılım kodlarının resme dönüştürülmesi

Daha sonra elde edilen resimler üzerinde "Auto Encoder" modeli kullanılarak, derin öğrenme metodu uygulanmıştır. Derin öğrenme, sinir ağları ve temel bileşen analizi (Principle Component Analysis – PCA) ile kNN kümeleme algoritmasından elde edilen sonuçlar aşağıda gösterilmiştir:

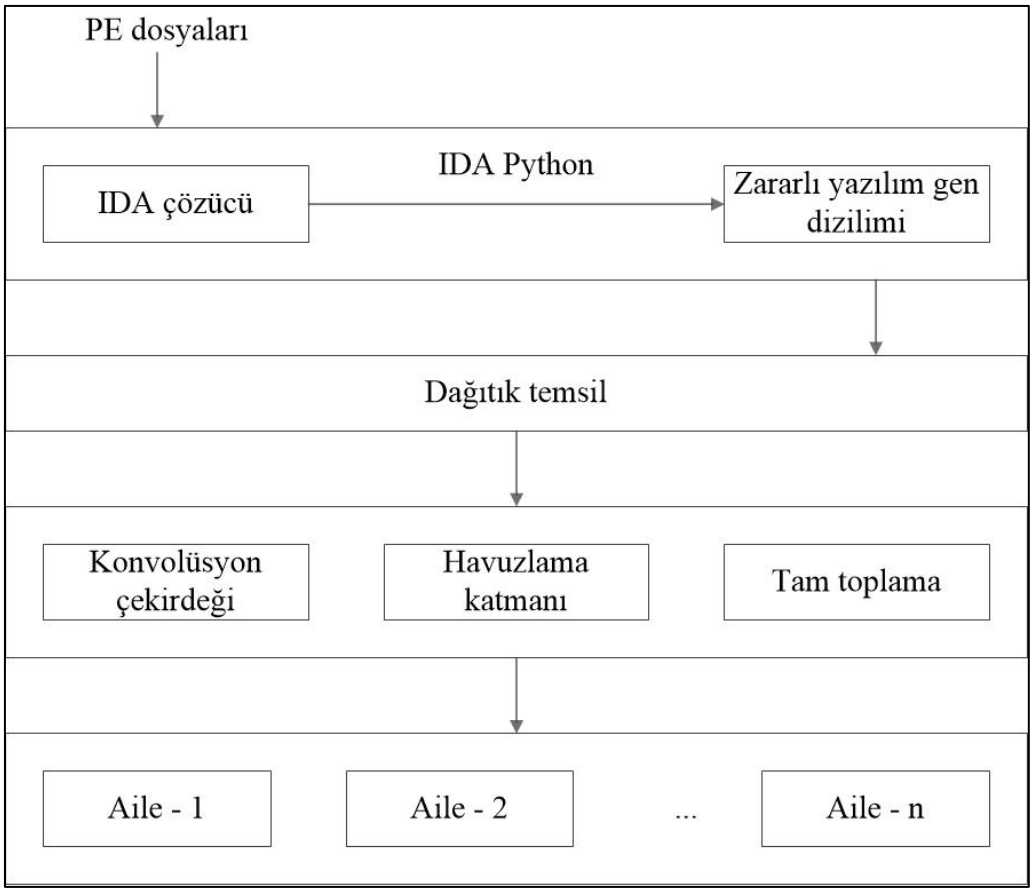
Çizelge 2.40. Elde edilen sonuçlar

Algoritma	Doğruluk (%)
Derin Öğrenme (AutoEncoder modeli ile)	99,1
PCA ve kNN sınıflandırıcı	96,6
Sinir Ağları	96,1

Bu çalışmada önerilen derin öğrenme yaklaşımının, diğer algoritmalara göre daha başarılı sonuç verdiği görülmektedir.

Meng, Shan, Liu, Zhao, Han, J. Wang ve H. Wang tarafından, "Statik Zararlı Yazılım Gen Sıralamaları Tabanlı Sınıflandırma - Malware Classification Based on Static Malware Gene

Sequences (MCSMGS)" olarak adlandırılan bir sistem geliştirilmiştir [43]. Söz konusu sistem ile "IDA Pro" programı kullanılarak; zararlı yazılımların gen sıralamaları çıkarılmış ve konvolüsyonel sinir ağlar (Convolutional Neural Networks - CNN) modeli ile analiz işlemi yapılmıştır. Ayrıca SVM algoritması ile de sınıflandırma işlemi yapılarak, elde edilen sonuçlar karşılaştırılmıştır. Geliştirilen sistemin şeması aşağıdaki şekilde gösterilmektedir:



Şekil 2.8. Önerilen sisteme ait şema

Veri setinde trojan ve arka kapılardan (backdoor) oluşan 5647 tane zararlı yazılım örneği kullanılmıştır:

Çizelge 2.41. Veri setinde kullanılan örnekler ve sayısı

Zararlı Yazılım	Örnek Sayısı
Backdoor.Win32.Bifrose	1422
Trojan-Downloader.Win32.Small	1074
Trojan.Win32.Obfuscated	1828
Trojan.Win32.Agent	1323

Veri seti eğitim ve test kümesi olarak, sırasıyla %90 ve %10 oranında ayrılmıştır. Sinir ağlarının oluşturulması için "TensorFlow Framework", SVM algoritması için ise "Matlab R2013a" platformu kullanılmıştır. Derin öğrenme tabanlı önerilen yaklaşım ile %98 oranında başarı elde edilirken; SVM algoritması ile %94,7 oranında başarıya ulaşılmıştır.

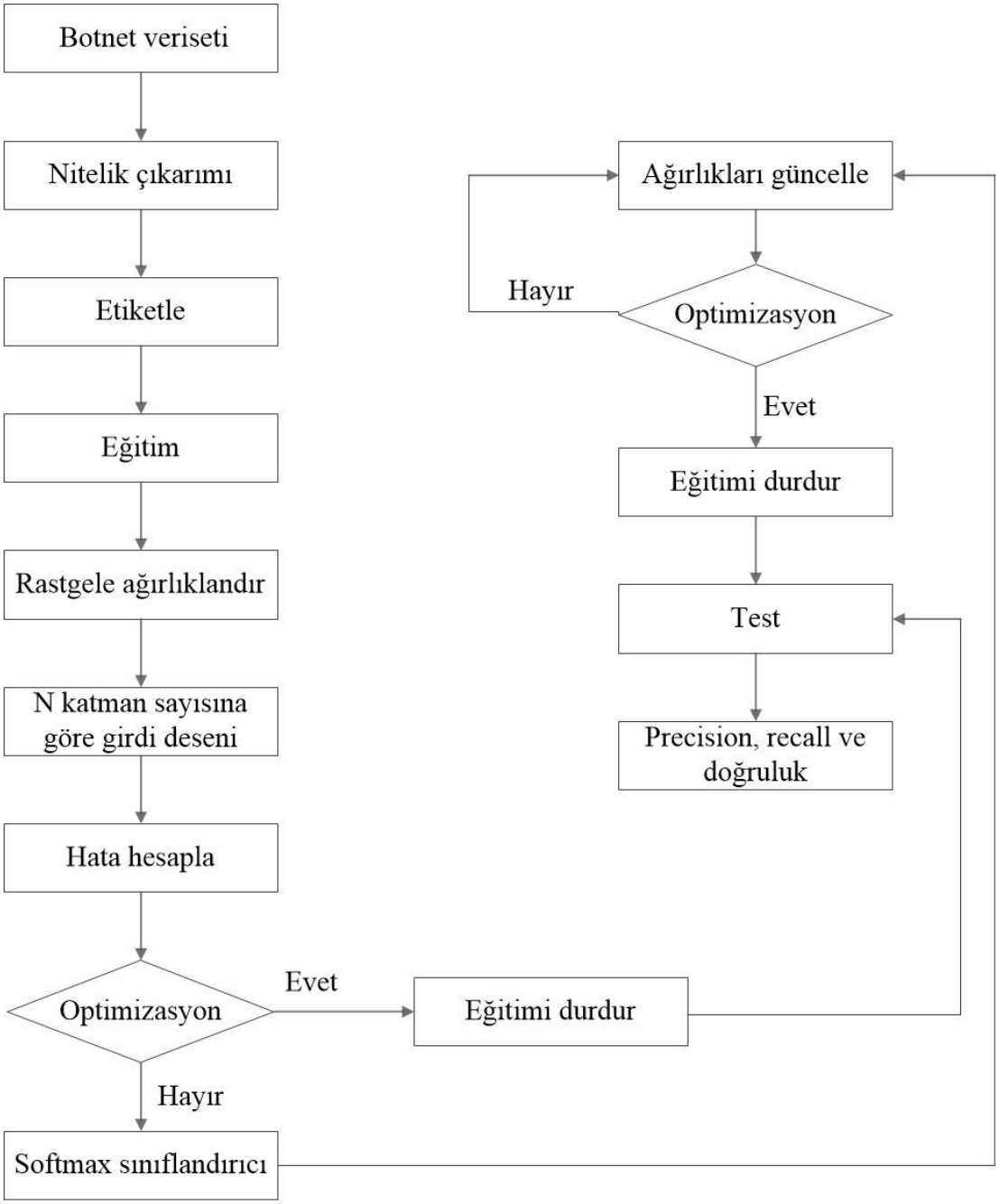
Bulut ve Yavuz tarafından, derin öğrenme yöntemleri kullanılarak; Android işletim sistemlerine yönelik geliştirilen zararlı yazılımların, çalıştırılmasına gerek kalmadan tespit edilmesi amaçlanmıştır [44]. Öznitelik olarak, mobil uygulamaların ihtiyaç duydukları izinler kullanılmış olup; bu izinler "AndoridManifest.xml" dosyasından çıkarılmaktadır. Veri seti için 3.229 tane zararsız, 1.668 tane zararlı mobil uygulama verisi kullanılmıştır. Bu verilerin %70'i eğitim, %15'i test, %15'i doğrulama amacı ile kullanılmıştır:

Derin öğrenmenin gerçekleştirilmesi aşamasında "Denoising Autoencoder (Gürültü Giderici Otomatik Kodlayıcı - GGOK)" modeli kullanılmıştır. Derin öğrenme metodu ile önceden tanınmayan zararlı yazılımların, %93,67 oranıyla tespit edildiği görülmüştür. Ayrıca elde edilen hassasiyet (precision), geri çağırma (recall) ve F1 değeri, aşağıdaki tabloda sunulmuştur:

Çizelge 2.42. Elde edilen hassasiyet (precision), geri çağırma (recall) ve F1 değerleri

Veri Sınıfı	Sonuçlar		
	Precision (%)	Recall (%)	F1-Ölçütü (%)
Zararsız Uygulama	91,9	97,1	94,4
Zararlı Uygulama	93,7	83,6	88,3
Ağırlıklı Ortalamalar	92,5	92,5	92,3

Kant, Singh ve Ojha tarafından, CNN modeli kullanılarak, akış tabanlı bir botnet sınıflandırma sistemi geliştirilmiştir [45]. Derin öğrenme yöntemiyle elde edilen sonuçlar ile Naive Bayes, SVM ve Random Forest algoritmalarından elde edilen sonuçlar karşılaştırılmıştır. Geliştirilen sistem modeli aşağıda gösterilmiştir Şekil 2.6'da gösterilmiştir.



Şekil 2.9. Önerilen sisteme ait akış şeması

Yukarıdaki şekilde de görüldüğü üzere, botnet veri setinden 39 tane öznitelik çıkarılmakta ve etiketlendirme yapılmaktadır. Daha eğitim aşamasına geçilerek ağırlıklar belirlenmektedir. Optimum sonuç elde edilinceye kadar ağırlıklar güncellenerek, test verisi ile precision, recall ve accuracy sonuçları çıkarılmaktadır.

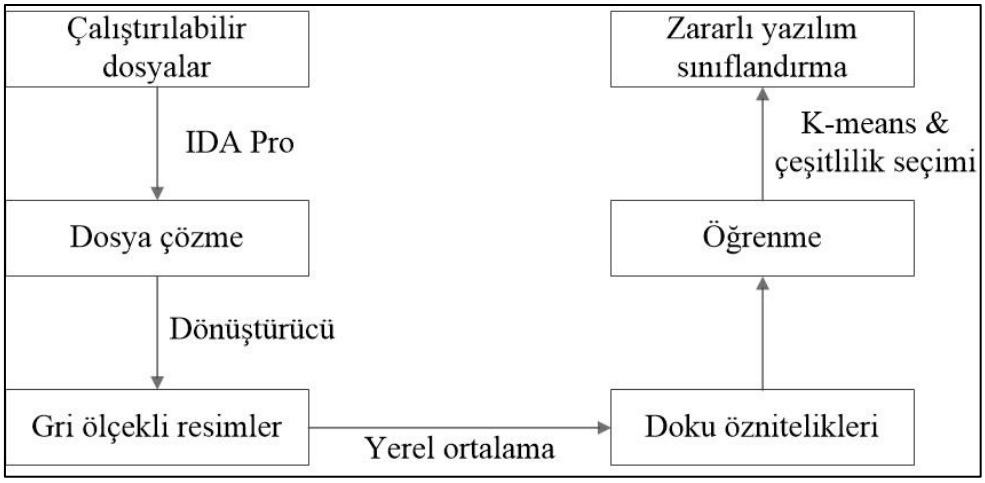
Önerilen derin öğrenme metodu ile diğer algoritmalardan elde edilen sonuçlar aşağıdaki tabloda paylaşılmıştır.

Çizelge 2.43. Her bir algoritma için elde edilen sonuçlar

Algoritma	Doğruluk (%)	Precision (%)	Recall (%)
Random Forest	71,0	77,9	65,0
Naive Bayes	83,1	80,9	82,0
SVM	89,1	92,7	95,9
CNN	93,3	94,3	91,3

Yukarıdaki tabloda da görüldüğü üzere en yüksek doğruluk oranını CNN, en düşük doğruluk oranını Ranfom Forest algoritmasının verdiği görülmektedir.

Liu ve Wang tarafından yapılan çalışmada, girdi olarak verilen yazılımlar, muhtelif algoritmalarla sınıflandırılmaya çalışılmıştır [46]. Bunun için exe dosyalar "IDA Pro" programı ile disassemble edilerek; gri tonlamalı resimlere dönüştürülmüştür. Şekil 2.7'de görüldüğü üzere buradan öznitelik çıkarımı yapılarak sınıflandırma işlemi yapılmıştır.



Şekil 2.10. Önerilen sisteme ait akış şeması

Yukarıdaki şekilde de görüldüğü üzere, her bir zararlı exe dosyası gri ölçekli resim haline dönüştürülmüştür. Yani exe dosyalarının öznitelikleri, pikseller üzerinden çıkarılmaktadır. Resim boyutlarını modifiye etmek maksadıyla "local mean" metodu kullanılmıştır. Bu metodla, gri ölçekli resimler bloklara bölünmüş ve her bir bloğun ortalama değeri hesaplanmıştır (0 – 255 aralığında).

$$f_{11} = (p_{11} + p_{12} + p_{21} + p_{23}) / 4 \tag{2.1}$$

$$\begin{pmatrix} p_{11} & \cdots & p_{16} \\ \vdots & \ddots & \vdots \\ p_{51} & \cdots & p_{5n6} \end{pmatrix} \xrightarrow{F=mean(piksel)} \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{pmatrix} \tag{2.2}$$

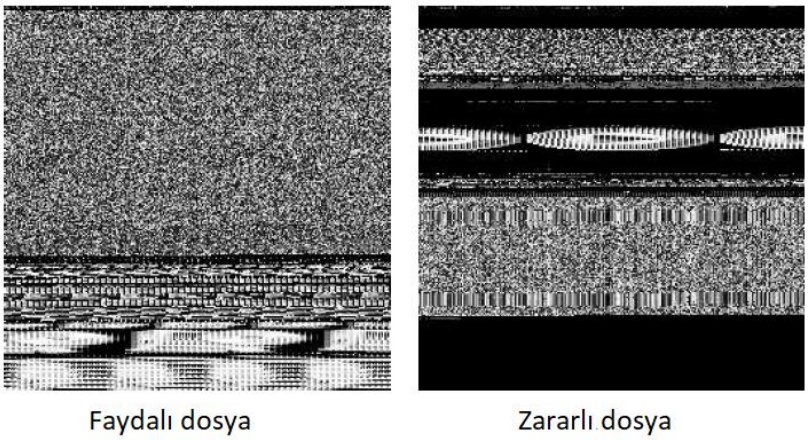
Bu metodla her bir resim 512x512 piksel boyutuna dönüştürülmüştür.

Sınıflandırma aşamasında, yazarlar tarafından bagging ve k-means algoritmaları birlikte kullanılmış ve ayrıca karşılaştırma amacıyla Random Forest, SVM, Karar Ağaçları, k-means ve lineer regresyon algoritmalarının sonuçları da incelenmiştir. Sonuçlar incelendiğinde, en başarılı doğruluk oranının, k-means ve bagging algoritmalarının birlikte kullanılarak elde edildiği görülmüştür:

Çizelge 2.44. Elde edilen doğruluk oranları

Algoritma	Doğruluk Oranı (%)
Bagging ve K-means	98,2
Random Forest	96,2
SVM	94,0
Karar Ağacı	91,0
K-means	84,3
Lineer Regresyon	93,5

Choi, Jang, Y. Kim ve J. Kim tarafından, faydalı ve zararlı yazılım dosyalarından resimler üretilerek; deep learning metoduyla bir yazılımın zararlı olup olmadığı tespit edilmeye çalışılmıştır [47]. Oluşturulan resimlerin 64 KB boyutunda olması sağlanmış ve resimler 32x32 boyutunda piksellerden oluşturulmuştur. Aşağıdaki resimde örnek birer faydalı ve zararlı yazılım dosyalarının resim hali gösterilmektedir:



Şekil 2.11. Faydalı ve zararlı yazılım dosyalarının resim halinde görünümü

Veri seti olarak 10 000 tane normal dosya, 2 000 tane zararlı yazılım dosyası kullanılmış ve eğitim ile test verisi %90 - %10 oranlarında ayrılmıştır. Zararlı yazılım tespit sisteminin doğruluk oranını ölçmek amacıyla "NVidia Cuda 8.0" aracı kullanılmıştır. Sınıflandırma aşamasında CNN modeli tercih edilmiştir. Sınıflandırma neticesinde %95,66 oranında doğruluk elde edilmiştir.

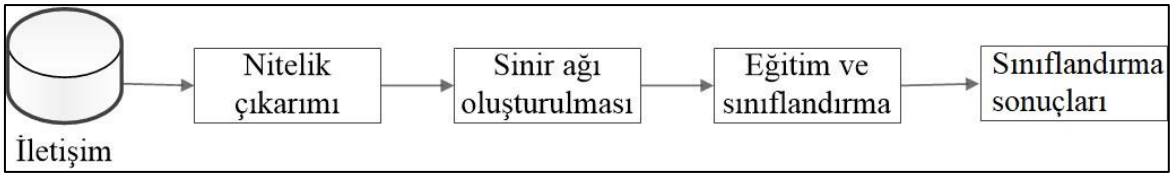
Luo ve Lo'nun yapmış olduğu çalışmada, çeşitli muhtelif makine öğrenmesi algoritmaları ve resim tanımlayıcı (image descriptor) metodları kullanılarak; zararlı yazılımların tespit edilmesi amaçlanmıştır [48]. Bunun için zararlı yazılım dosyaları gri ölçekli resimlere çevrilmiş ve resimler üzerinde deep learning, SVM ve kNN algoritmaları kullanılarak sınıflandırma işlemi yapılmıştır. Resim tanımlayıcı olarak ayrıca global resim tanımlayıcısı (Global Image Descriptor – GIST) ve yerel ikili desen (Local Binary Pattern – LBP) metodları kullanılmıştır.

Veri setinde 32 farklı zararlı yazılım ailesinden yaklaşık 12.000 tane zararlı yazılım kullanılmıştır. Söz konusu zararlı yazılımlar genellikle trojan, password stealer ve viruslerden oluşmaktadır. Veri seti, eğitim ve test verisi olarak, %80'e %20 oranında bölünmüştür. Çizelge 2.45'te görüldüğü üzere sınıflandırma işlemi neticesinde en iyi sonucun deep learning algoritması ve LBP metodunun birlikte kullanılmasıyla elde edildiği görülmüştür.

Çizelge 2.45. Elde edilen doğruluk oranları

Sınıflandırma Metodu	Doğruluk Oranı (%)
Deep learning + LBP	93,1
SVM + LBP	87,8
kNN + LBP	85,9
Deep learning + GIST	87,8
SVM + GIST	81,2
kNN + GIST	82,8

Shibahara, Yagi, Akiyama, Chiba ve Yada tarafından, derin öğrenme metoduyla tekrarlayan sinir ağları (Recurrent Neural Networks - RNN) modeli kullanılarak; dinamik zararlı yazılım analizi üzerine bir çalışma yapılmıştır [49]. Önerilen yöntem ile ağ trafiği dinlenerek öznitelik çıkarımı yapılmaktadır. Daha sonra RNN modeli ile eğitim ve sınıflandırma işlemleri yapılmaktadır.



Şekil 2.12. RNN modeline ait şema

Öznitelik çıkarımı aşamasında;

- DNS, HTTP gibi uygulama protokolleri,
- Durum kodu (100, 101, 200, 201, 202 vb.),
- Dosya türü (javascript, image, php, html, css ve diğerleri),
- Sorgu parametreleri sayısı,
- Dizin derinliği (path depth),
- İletilen ve alınan byte miktarları,
- HTTP metodları (GET, POST, HEAD ve diğerleri)
- İletişim türü (zararlı URL, faydalı URL, zararlı protokol, faydalı protokol)

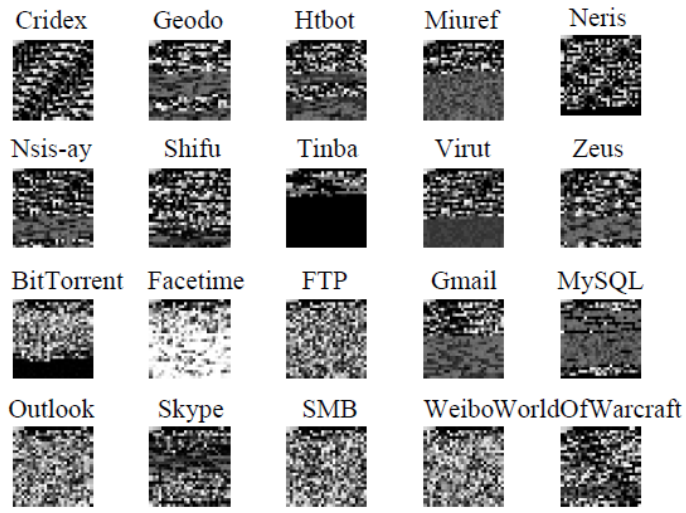
öznitelik olarak kullanılmıştır.

VirusTotal platformundan toplanan zararlı yazılım örnekleri üzerinde yapılan sınıflandırma işlemi neticesinde;

- % 97,6 Precison (TP/(TP + FP))
- % 97,5 Recall (TP/(TP + FN))
- % 96,9 F-measure (2xPrecisionxRecall/(Precision+Recall))

oranları elde edilmiştir.

Wang, Zhu, Zeng, Ye ve Sheng'in yapmış olduğu bu çalışmada, CNN kullanılarak zararlı yazılımlara ait trafiğin tespit edilmesi amaçlanmıştır [14]. USTC-TFC2016 adında, pcap formatında bir veri seti hazırlanmıştır. Pcap halindeki raw datalar CNN'e girdi olarak verilmek üzere, gri ölçekli resim dosyalarına dönüştürülmektedir. Her bir resmin boyutu 784 (28x28 boyutunda) byte olarak ayarlanmaktadır. Dosya boyutu 784 bytetan büyük ise resim kırpılmakta; küçük ise sonlarına "0x00" eklenmektedir ("0x00" siyah rengi, "0xff" ise beyaz rengi temsil etmektedir). Son olarak resimler idx formatına çevrilmektedir. Pcap dosyalarından oluşturulan örnek resimler aşağıda gösterilmektedir:

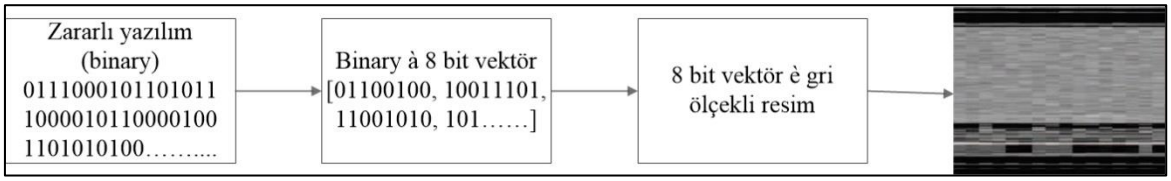


Şekil 2.13. Zararlı yazılımların resim halinde görünümü

Yukarıdaki resimde de görüldüğü üzere zararlı yazılım ve uygulama trafikleri, görsel olarak birbirinden ayırt edilebilmektedir.

CNN modelinin oluşturulma ve eğitilme aşamasında, Google tarafından geliştirilen "TensorFlow" aracı kullanılmıştır. Sınıflandırma neticesinde her bir uygulama ve zararlı yazılımın doğru tespit edilme oranı ortalama %99,41 çıkmıştır.

Kalash, Rochan, Mohammed, Bruce, Wang ve Iqbal tarafından, CNN tabanlı zararlı yazılım sınıflandırılmasına yönelik bir çalışma yapılmıştır [17]. Veri seti olarak "Microsoft" ve "Malimg" veri setleri ayrı ayrı kullanılmıştır. Yapılan çalışmada zararlı yazılım örnekleri binary şekliyle 8 bit 8 bit ayrılmıştır. Sonrasında elde edilen binary değerler matrise decimal haliyle yerleştirilmiş ve gri ölçekli resimler elde edilmiştir:



Şekil 2.14. Zararlı yazılımların resim haline dönüştürülmesi

Sınıflandırma işlemi sırasında Torch kütüphanesinden yararlanılmıştır. Microsoft veri setinin kullanıldığı sınıflandırmada %99,97; Malimg veri setinin kullanıldığı sınıflandırmada ise %98,52 oranında doğruluk elde edilmiştir.

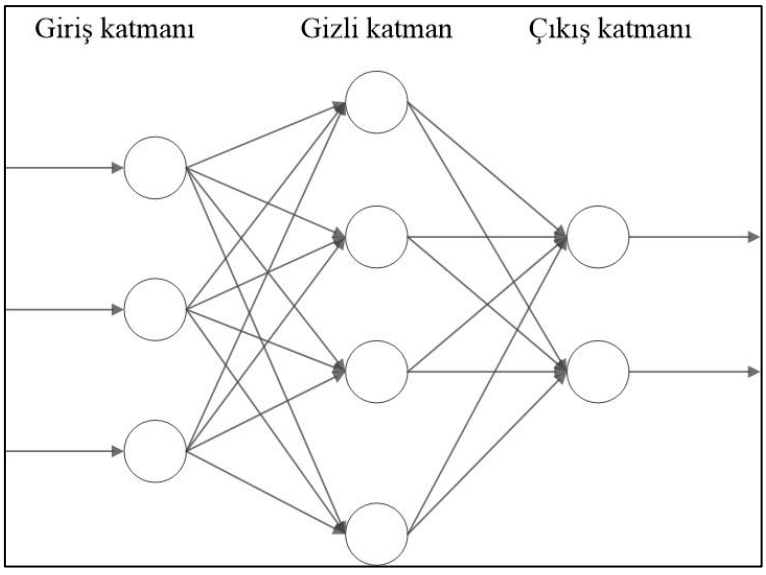
3. AĞ TRAFİĞİNDE ANORMALLİK TESPİTİNDE KULLANILAN MAKİNE ÖĞRENMESİ YÖNTEMLERİ

Anormallik tespitine yönelik yapılan çalışmalarda makine öğrenmesi algoritmaları çözüm üretmede sıklıkla kullanılmaktadır [10]. Son yıllarda derin öğrenme teknikleri başta olmak üzere yapay zeka çözümleri, gelecekte karşılaşabilecek karmaşık problemler için uygulanabilir çözümler olarak öne çıkmaktadır [10]. Bu bölümde, anormallik tespiti çalışmalarında sıklıkla kullanılan YSA ve son yıllarda kullanımı giderek yaygınlaşan derin öğrenme metodu ile birlikte karar ağacı, random forest, naive bayes, adaboost, kNN ve SVM algoritmaları ve temel özellikleri sunulacaktır.

3.1. Yapay Sinir Ağları

YSA, insan beyninden esinlenerek geliştirilen ve sinir sisteminin çalışma şeklini örnek alan bir teknolojidir. Bir yapay sinir ağı, işlem elemanları ve bunlar arasındaki bağlantı olmak üzere en az iki bileşene sahiptir. İşlem elemanları nöron olarak adlandırılır ve her bir nöronun arasında bir bağlantı vardır. Her parametrenin de onunla ilişkili bir ağırlık parametresi bulunmaktadır [6, 7, 9].

İşlem elemanları "girdi", "ara", "çıktı" olmak üzere toplam 3 katman halinde bir araya gelerek bir ağ oluştururlar. Ara katman sayısı birden fazla olabilmektedir. Örnek bir yapay sinir ağı, Şekil 3.1'de gösterilmiştir [8]:



Şekil 3.1. Örnek bir yapay sinir ağı

Eldeki veri seti girdi olarak ağa verilir ve ara katmanlarda işlenerek çıktı üretilir. Burada, ağırlık değerlerinin doğru olarak belirlenmesi gerekir. Doğru ağırlık değerlerinin bulunması işlemi ağın eğitilmesi olarak adlandırılmaktadır. Ağırlık değerleri ilk başta rastgele belirlenmektedir. Optimum değer bulununcaya kadar yani test setindeki örneklerden doğru çıktılar alınana kadar ağırlık değerleri değiştirilerek, eğitim işlemi yapılmaktadır [7]. Her girdi birimi kendi aktivasyon değerini, bağlanılan her bir gizli birime gönderir. Bu gizli birimlerin her biri, kendi aktivasyon değerini hesaplar ve bu değer daha sonra çıkış ünitesine aktarılır [50]. Literatüre bakıldığında çok sayıda yapay sinir ağı modelinin geliştirildiği görülmektedir. Oluşturulan ağın bir olayı en iyi şekilde öğrenmesi için en doğru yapay sinir ağı modelinin seçilmesi gerekmektedir. Bir yapay sinir ağı modeli oluşturulurken, aşağıda belirtilen özellikler önemlidir [7]:

- Ağ topolojisi
- Toplama fonksiyonu
- Aktivasyon fonksiyonu
- Öğrenme stratejisi ve kuralı

3.2. Derin Öğrenme

YSA uzun yıllardır çeşitli problemlere karşı kullanılmış ve kullanılmaya devam etmektedir. Son yıllarda grafiksel işlem birimlerinin (Graphics Processing Unit - GPU) gelişimiyle birlikte makine öğrenmesi tabanlı yapay zeka uygulamalarının kullanımında büyük bir artış yaşanmış ve yapay sinir ağı temelli derin öğrenme metodunun ortaya çıkması sağlanmıştır [9, 12]. Derin öğrenme, insan beyninin, içgüdüsel olarak deneyerek öğrenme kabiliyetinden esinlenmiştir [12]. Derin öğrenme için çok sayıda gizli katmandan oluşan YSA denebilmektedir [9, 10]. Görüntü tanıma, ses tanıma, doğal dil işleme ve diğer benzer karmaşık işlemler için kullanılabilmektedir [9, 11, 13, 14, 15, 18].

Bir veri setinden öznitelik çıkarımı yapmak, zaman alıcı ve maliyetli bir işlemdir. Klasik makine öğrenmesi yaklaşımlarında, bir girdinin önemli öznitelikleri manuel olarak seçilirken; derin öğrenme yaklaşımında bu öznitelikler, otomatik olarak belirlenir [11, 15]. İşlemlerin paralel olarak yapabilmesine imkan sağlayan karmaşık modeller sayesinde, çok karmaşık problemler hızlı ve iyi bir şekilde çözülebilmektedir [11]. Derin öğrenme metodu verilerde otomatik olarak korelasyon bulma yeteneğine sahiptir ve bu yüzden yeni nesil

saldırı tespit sistemleri için umut vaat eden bir yöntemdir. Ayrıca sıfırıncı gün (zero-day) açıklıklarına karşı da etkili bir şekilde kullanılabilmekte ve yüksek seviyede doğruluk oranı elde edilebilmektedir [15].

Yapay sinir ağlarında olduğu gibi derin öğrenme metodu için de çeşitli modeller geliştirilmiştir. En sık kullanılan derin öğrenme modelleri şu şekildedir [10]:

- Derin Sinir Ağları (Deep Neural Network - DNN)
- Konvolüsyonel Sinir Ağlar (Convolutional Neural Network - CNN)
- Tekrarlayan Sinir Ağları (Recurrent Neural Network - RNN)
- Derin İnanç Ağları (Deep Belief Network - DBN)

DNN bir giriş katmanı, bir çıkış katmanı ve gizli katmanlardan oluşmakla beraber, diğer sinir ağları için temel modeldir. CNN video ve ses uygulama alanlarında iyi performans göstermektedir. RNN yüksek tanıma oranları sağlayan dil modelleme ve el yazısı tanıma alanlarında etkilidir. DBN ise nesne tanımlama uygulamalarında kullanılmaktadır [10]. Yukarıda belirtilen derin öğrenme modelleri arasında, en çok CNN üzerine çalışmalar yapılmıştır [13].

CNN, yaşayan canlıların doğal görsel algılama mekanizmasından esinlenmiş bir derin öğrenme mimarisi olup; son zamanlarda resim, video, konuşma ve metin işleme alanlarında sıkça kullanılmıştır [13]. CNN, karmaşık işlemleri modellemek ve büyük miktarda veriye sahip uygulamalar üzerinde örüntü tanılamak için en güçlü tekniklerden biridir [9]. CNN şu ana kadar özellikle görüntü işleme alanında en iyi sonuçları vermiş olup; anlamsal ayrıştırma, arama sorgusu, cümle modelleme, sınıflandırma ve tahmin problemlerinde de iyi sonuçlar vermiştir [17]. Bir CNN'i sıfırdan eğitmek; yüksek seviyede hesaplama gücü, büyük miktarda bellek kaynağı ve zaman gerektirmektedir. CNN'lerin derinliği (katman sayısı) arttıkça, çok iyi performans gösterdiği görülmektedir [16]. Son zamanlarda görüntü işleme ve sınıflandırma gibi alanlar başta olmak üzere birçok alanda geleneksel öğrenme algoritmalarına göre üstün bir performans gösteren CNN'ler, aynı zamanda ileri beslemeli bir sinir ağıdır [17].

Bir veri seti üzerinde derin öğrenme metodunun uygulanması ve konfigürasyonlarının yapılması için çeşitli araçlar ve kütüphaneler geliştirilmiştir. Bunlardan bir kısmı aşağıda sunulmuştur:

- Keras
- Torch
- TensorFlow
- Caffe
- Microsoft Cognitive Toolkit
- DeepLearning4j
- Theano
- MXNet
- H2O.ai
- ConvNetJS
- DeepLearningKit
- Gensim
- DeepLearnToolbox

Yukarıda bahsedilen araçların kullanım kolaylığı, hız, dokümantasyon, arayüz tasarımı vb. konularda birbirlerine göre avantajları bulunmaktadır. Bu tez çalışmasında Keras aracı tercih edilmiştir.

3.3. Karar Ağacı

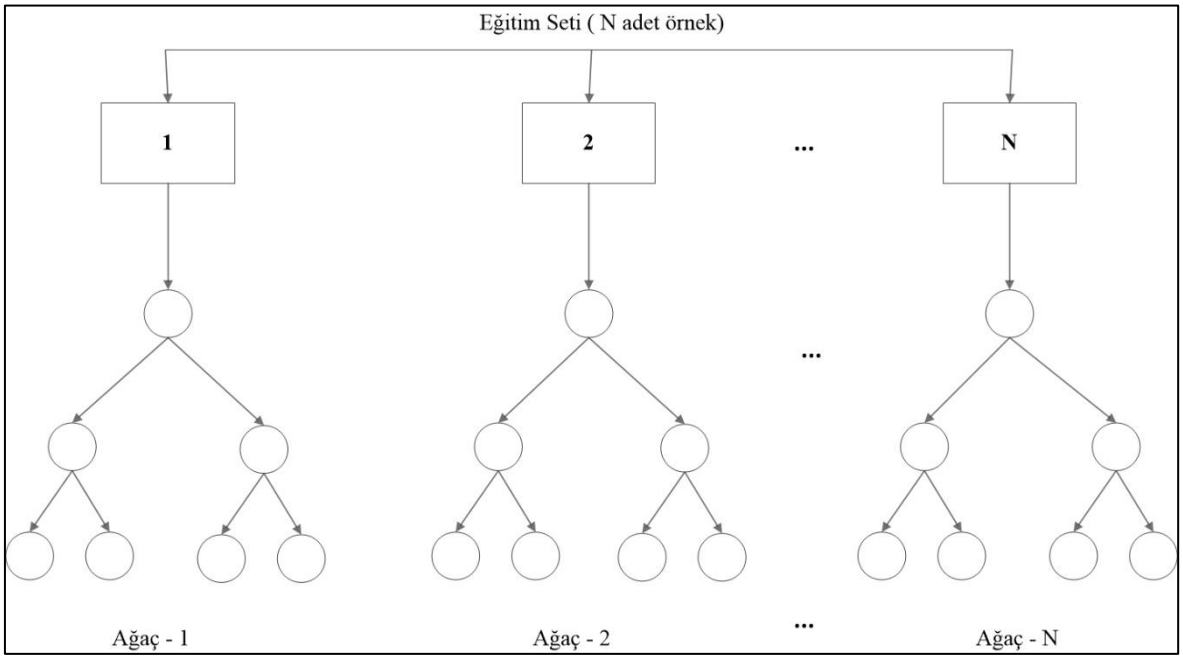
Karar ağaçları, her bir verinin özelliklerine göre kümelere ayrılıp, aynı sınıf etiketine sahip verilerin aynı kümede yer alarak karmaşıklığın minimum seviyede olmasını amaçlayan sınıflandırıcılardır [50]. Oluşturulan karar ağacında her bir düğüm bir niteliği tanımlar. Bu algoritma ile ağaç düzgün bir şekilde budanır ve tablolar oluşturulur. Test verisinin sınıf etiketi bu tablolar vasıtasıyla belirlenir [52].

Karar ağaçlarında, verilerin sınıf etiketlerini en iyi şekilde bölerek bulabilmek için, “gini index” ve “information gain” gibi çeşitli metotlar mevcuttur. Yapılan çalışmalar, bu metotlar arasında en iyi diye tek bir metot olmadığını, veri setine göre değişiklik gösterebileceğini göstermiştir.

Karar ağaçları kapsamlılık yönü ve insanların kolayca anlayabileceği yönüyle sınıflandırma için kullanılabilecek en iyi sınıflandırıcılardan biridir. Karar ağaçları, ayrık/kategorik özellikler söz konusu olduğunda, daha iyi performans göstermektedirler [50].

3.4. Random Forest

Random forest algoritması 2001 yılında Leo Breiman ve Adele Cutler tarafından geliştirilmiş olup; büyük verileri doğru bir şekilde sınıflandırmada kullanılan en iyi sınıflandırma algoritmlarından biridir. Şekil 3.2'de görüldüğü üzere bu algoritma ile birden fazla karar ağacı üretilerek; bir nevi orman oluşturulmaktadır [54].



Şekil 3.2. Birden fazla ağaçtan oluşan karar ormanı

Bu algoritmada girdi verilerinin alt kümelerinden birçok sınıflandırıcı üretilir ve sonrasında, her bir sınıflandırıcının sonuçları, girdi verilerinin istenen çıktısını üretmek için oylama mekanizmasına dayanarak toplanır [54].

Random forest algoritması saldırı tespit sistemleri, modern ilaç keşfi, kredi derecelendirme analizi, gen mikrodizileri, arazi analzi vb. dahil olmak üzere çeşitli alanlarda sıklıkla kullanılmaktadır [55, 56].

3.5. Naive Bayes

Naive Bayes algoritması denetimli sınıflandırma için kullanılan en basit modeldir [51] ve istatistiksel tabanlı bir makine öğrenmesi algoritmasıdır. Olasılık hesabı aşağıdaki şekilde yapılmaktadır:

$$P(i \mid X) = (P(X \mid i)P(i)) \mid P(X) \tag{3.1}$$

Burada $P(X)$, X olayının olasılık hesabı, $P(i)$ hipotezin gerçekleşme olasılığı, $P(X \mid i)$ ise i hipotezi doğrultusunda x olayının gerçekleşme olasılığını temsil etmektedir.

Naive Bayes sınıflandırıcının en büyük avantajı, öğrenme için kısa zaman harcanmasıdır. Ayrıca, model bir ürünün formuna sahip olduğundan, hesaplama avantajlarıyla logaritma kullanılarak toplama dönüştürülebilir [50].

3.6. Adaboost

Boosting algoritmasında, güçlü bir sınıflandırıcı elde etmek için zayıf sınıflandırıcıların kombine edildiği herhangi bir öğrenme algoritmasının doğruluğunu geliştirmek, temel amaçtır [52].

Adaboost algoritması, saldırı tespit sistemleri ve yüz tanıma gibi birçok örüntü tanıma problemlerinde kullanılan, güçlü bir teorik temel ve kolay uygulanabilen makine öğrenme algoritmalarından biridir. Bu algoritma, birden fazla hipotezler içerisinde eğitim örneklerini yönetir [52]. Adaboost algoritması, yüksek tespit oranı, düşük yanlış alarm oranı ve düşük hesaplama karmaşıklığı ile sızma tespit yaklaşımlarında uygulanmıştır [53].

3.7. K En Yakın Komşu (kNN)

kNN, bir veri kümesi içerisinde örnekleri genellikle benzer özelliklere sahip, diğer örneklerin yakın olacağı prensibine dayanmaktadır [50, 51]. Örnekleri bir sınıflandırma etiketi ile etiketlenmiş, sonra sınıflandırılmamış bir örneğinin etiketinin değeri, en yakın komşularının sınıfını gözlemleyerek tespit edilebilir [50].

kNN algoritmasında ideal olan, farklı sınıfların örnekleri arasındaki mesafeyi maksimize etmek iken, benzer sınıflandırılan örnekler arasındaki mesafeyi minimize etmektir. Bunun için farklı metodlar uygulanmaktadır ve örnekler arasındaki mesafeyi tanımlamada kullanılan metodlardan bazıları aşağıda gösterilmiştir [50].

Minkowsky

$$D(x,y) = (\sum_{i=1}^{m} | x_i - y_i |^p)^{1/p} \tag{3.2}$$

Manhattan

$$D(x,y) = \sum_{i=1}^{m} | x_i - y_i | \tag{3.3}$$

Chebychev

$$D(x,y) = \max_{i=1}^{m} | x_i - y_i | \tag{3.4}$$

Euclidean

$$D(x,y) = (\sum_{i=1}^{m} | x_i - y_i |^2)^{1/2} \tag{3.5}$$

Camberra

$$D(x,y) = \sum_{i=1}^{m} \frac{| x_i - y_i |}{| x_i + y_i |} \tag{3.6}$$

Kendall Sıra Korelasyonu

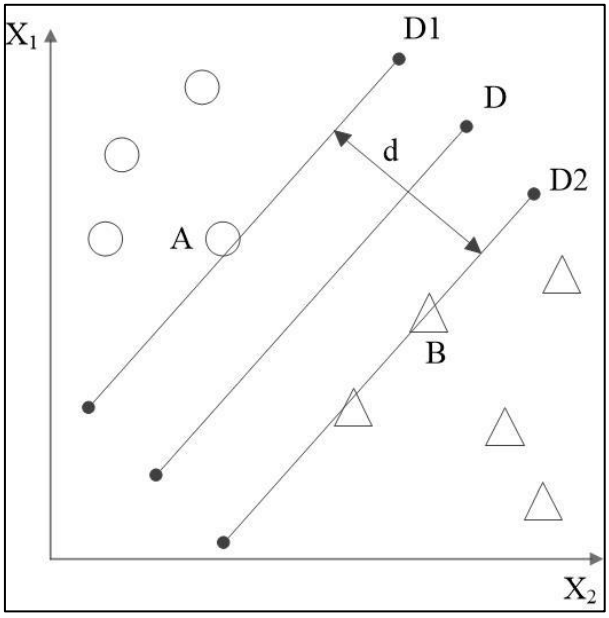
$$D(x,y) = 1 - \frac{2}{m(m-1)} \sum_{i=j}^{m} \sum_{j=1}^{i-1} sign(x_i - x_j) \, sign(y_i - y_j) \tag{3.7}$$

3.8. Destek Vektör Makineleri

Destek vektör makineleri hiper düzlem üzerindeki ayrıma ve örnekler arasındaki mesafenin maksimize edilmesine dayalı bir istatistiksel sınıflandırma yaklaşımıdır. Hem sınıflandırma hem de regresyon problemlerinde kullanılabilecek denetimli bir makine öğrenmesi algoritması olup; daha çok sınıflandırma problemlerinde kullanılmaktadır [54].

Bu algoritmada her bir veri, n (öznitelik sayısı) boyutlu uzayda bir nokta olarak belirtilir ve her niteliğin değeri belli bir koordinatın değerini alır. Daha sonra iki sınıfı iyi bir şekilde ayıran hiper düzlem bulunarak sınıflandırma işlemi gerçekleştirilir [54].

Aşağıdaki şekilde iki farklı sınıfa ait verilerin sınıflandırılması görsterilmiştir [57]:



Şekil 3.3. İki sınıfın doğrusal olarak ayrılabildiği durum

Şekilde de görüldüğü üzere veriler bir doğru ile ayrılarak; belirli koordinatlar arasında kalan bölgeler birer sınıf etiketine atanır. D doğrusu <w.x>+b=0 denklemi ile belirlenmekte olup, w ağırlık vektörü ve b de sabit sayı (bias) değeridir. D1 doğrusu <w.x>+b=1 denklemi ile D2 doğrusu ise <w.x>+b=-1 denklemi ile belirlenir. D1 ve D2 doğruları arasındaki uzaklık d ile gösterilmektedir [57].

4. VERİ SETİ OLUŞTURMA

Anormallik tespiti problemlerine karşı kullanılan makine öğrenmesi yöntemlerinin en önemli aşamalarından birisi veri seti oluşturma kısmıdır. Veri setinin gerçek hayata uygun olarak oluşturulması; çalışmanın başarı oranını doğrudan etkilemekte ve gerçek hayatta kullanılabilirliğini arttırmaktadır. Bu kapsamda veri setlerinde kullanılan veri miktarı ve çeşitliliğinin çok olması önemli bir hal almaktadır.

4.1. Ağ Trafiğindeki Anormallik Tespitine Yönelik Yapılan Çalışmalarda Veri Seti Oluşturma

Kumar ve arkadaşları Android tabanlı bir saldırı tespit sistemi geliştirmişlerdir [21]. Bu çalışmada veri seti oluşturmak amacıyla Facebook, Facebook Messenger, Gmail, Google Maps, Twitter, Youtube, Chrome, Skype gibi uygulamalar çalıştırılarak normal trafik verisi üretilmiş ve Wireshark aracı ile trafik kaydedilmiştir. 30'un üzerinde antivirüs firmasının verilerini depolayan VirusTotal platformu üzerinden, 600 civarı zararlı yazılım örneği indirilerek, zararlı yazılım trafiği verisi üretilmiştir. Söz konusu trafiğin zararlı trafikten oluşup oluşmadığını doğrulamak amacıyla; online olarak pcap dosyasını analiz eden "NetworkTotal – www.networktotal.com" platformunda test edilmiştir.

Azab ve arkadaşlarının botnet tespitine yönelik yapmış olduğu çalışmada [22], veri seti oluşturulurken, sisteme bulaşan zararlı yazılımın veri çalmadan önce tespit edilebilmesi için, botnet trafiğindeki "HTTP GET" bağlantıları, pcap formatında toplanmıştır. Botnet trafiği, Zeus'un 1 ve 2. versiyonu çalıştırılarak elde edilmiştir. Normal trafik için ise Alexa'da en çok ziyaret edildiği belirtilen 23 tane siteye yönelik trafik oluşturulmuş ve toplanmıştır. Ayrıca çeşitli sitelerden, farklı boyutlarda dosyalar indirilerek, ağ trafiği "Wireshark" aracı ile kaydedilmiştir.

McLaren ve arkadaşları tarafından zararlı yazılım trafiği tespitine yönelik bir çalışma yapılmıştır [26]. Bu çalışmada, normal trafik verileri oluşturulurken; Windows işletim sistemli bir bilgisayarda, sanallaştırılmış bir ağ üzerinden çeşitli İnternet adreslerine trafik oluşturularak; trafik verileri pcap formatında kaydedilmiştir. Zararlı trafik verisi olarak APT ve botnet trafik verilerinin paylaşıldığı Contagio veri seti kullanılmıştır. Bu veri setinde pcap formatındaki zararlı trafik verileri Tshark aracı ile filtrelenmiştir.

Boukhtouta ve arkadaşları tarafından, ağ trafiği üzerinden zararlı yazılımların sınıflandırılmasına yönelik bir çalışma yapılmış ve bu çalışmada WISNET (Wireless and Secure Networks Research Lab) veri seti kullanılmıştır [28]. Ayrıca özel şirketlerden de trafik verileri toplanmıştır. WISNET veri setinde, zararlı yazılımlar ThreatTrack sandbox üzerinde çalıştırılarak, ağ trafiği kaydedilmiştir. Öznitelik çıkarımı ve etiketleme aşamasında JNetPcap API'sinden faydalanılmıştır.

Radoglou-Grammatikis ve Sarigiannidis'in yapmış olduğu çalışmada Android tabanlı bir saldırı tespit sistemi geliştirilmiş ve CTU-13 (Czech Technical University) veri sti kullanılmıştır [30]. Bu veri setinde üniversitedeki ağındaki internet trafiği pcap formatında kaydedilmiş ve her bir veri IRC, spam, port taraması, DDoS, http vb. şeklinde etiketlenmiştir. Zararlı trafik üretmek için çeşitli zararlı yazılımlar üniversite ağında çalıştırılmıştır.

Cheng ve arkadaşları, Android işletim sistemine sahip akıllı telefonlardaki zararlı ağ akışlarını yakalamak ve zararlı uygulamaların davranışlarını anlamak amacıyla bir çalışma yapmış ve zararlı trafik verisi oluşturmak için "Moledroid" isimli bir uygulama geliştirilmiştir [33]. Söz konusu uygulama ile kullanıcılara ve telefona ait bilgilerin (rehber, çağrı geçmişi, kısa mesajlar, lokasyon bilgileri, telefon kimliği vb.) bir sunucuya aktarılması sağlanmaktadır. Bununla beraber normal veri toplamak amacıyla WeChat, Weibo, BaiduYun gibi çeşitli uygulamalar da telefona kurularak, uygulamaların gönderip aldığı paketler yakalanmış ve analiz edilmiştir. Akıllı telefonların oluşturduğu trafik Wireshark ile toplanarak; süre (Unix epoch formatında), hedef ve kaynak IP adresleri, portlar, paket boyutları, protokol ve TCP/IP flagleri gibi bilgiler ayıklanmıştır.

Verma ve Ranga, akış tabanlı bir saldırı tespit sistemi geliştirmiş ve bu çalışmada CIDDS-01 (Coburg Network Intrusion Detection Dataset) veri setini kullanmıştır [34]. Bu veri setini oluşturmak için e-posta ve web sunucularının yer aldığı küçük bir işletme ortamı taklit edilerek bir platform oluşturulmuştur. Burada bir Python scripti ile normal kullanıcı davranışları (İnternet'te gezinmek, e-posta göndermek/almak, dosya alışverişi yapmak vs.) taklit edilmiştir. Zararlı trafik için ağda servis aksatma, kaba kuvvet (brute force) saldırıları ve port taraması yapılarak internet trafiği kaydedilmiştir.

Chen ve arkadaşları zararlı yazılım trafiği tespitine yönelik bir çalışma yapmışlar [36] ve bu çalışmanın veri setinde kullanılmak üzere 5560 tane zararlı yazılım örneği, "Drebin Project" ten temin edilmiştir. Android cihazlardaki normal trafik örneği için ise çeşitli mobil uygulamalar kullanılmış ve bu uygulamaların zararlı olmadığı, VirusTotal platformu üzerinden teyit edilmiştir. Zararlı yazılım trafiği üretilirken, trafik kalitesini geliştirmek maksadıyla basit servis bulma protokolü (Simple Service Discovery Protocol – SSDP), dinamik sunucu yapılandırma protokolü (Dynamic Host Configuration Protocol – DHCP), adres çözümleme protokolü (Address Resolution Protocol – ARP), NetBIOS ad servisi (NetBIOS Name Service – NBNS), internet grup yönetimi protokolü (Internet Group Management Protocol – IGMP) ve sunucu mesaj bloğu (Server Message Block – SMB) paketleri dikkate alınmamıştır. Trafik kaydedilirken Tcpdump aracından faydalanılmıştır.

Hodo ve arkadaşları, Tor trafiğini tespit etmek için bir çalışma yapmış ve bu çalışmada UNB-CIC Tor-nonTor ve VPN-nonVPN veri setleri kullanılmıştır [40]. Bu veri setlerinde, kullanıcı hesapları oluşturularak;

- Tarama (Firefox ve Chrome tarayıcıları ile HTTP/HTTPS trafikleri),
- E-Posta (Gmail),
- Chat (Skype, AIM, ICQ),
- Audio-streaming (Spotify),
- Video-streaming (Youtube, Vimeo),
- FTP (Skype dosya transferi, SFTP, FTPS),
- VoIP (Facebook, Hangouts, Skype),
- P2P (BitTorrent)

trafik verileri Wireshark ve Tcpdump araçları ile kaydedilmiştir. Daha sonra Java ile geliştirilen bir uygulama ile pcap dosyaları okunmuş ve seçilen özniteliklerle beraber csv dosyaları oluşturulmuştur.

Lashkari ve arkadaşları tarafından Android tabanlı zararlı yazılım trafiği tespitine yönelik bir çalışma yapılmıştır [41]. Veri setine eklemek üzere normal trafik olarak, 2015 ve 2016 yıllarına ait Google Play Market'ten indirilen 1500 tane uygulamanın trafiği toplanmıştır. Bu uygulamaların zararlı olup olmadığı, VirusTotal platformu üzerinden teyit edilmiştir.

Zararlı yazılım trafiğine eklemek üzere ise Airpush, FakeAV, Penetho, Shuanet gibi 12 farklı kategoride, 400 tane zararlı yazılımın trafiği kullanılmıştır. Trafik toplanırken bir akıllı telefon bilgisayara bağlanmıştır. Geliştirici erişimi ile USB debugging aracı açılarak; uygulama ve zararlı yazılımlar çalıştırılırken bir yandan da trafik kaydedilmiştir. Symantec firması referans gösterilerek; gerçek dünyada normal yazılımların oranının %80, zararlı yazılımların oranının %20 olduğu belirtilmiş ve bu çalışmanın veri setinde de bu orana riayet edilmiştir.

Singh ve arkadaşları tarafından botnet tespitine yönelik bir çalışma yapılmış ve çalışmada CAIDA (Center for Applied Internet Data Analysis) veri seti kullanılmıştır [27]. Bu veri setinde, pcap dosyalarını analiz etmek ve filtrelemek için bir Perl scripti geliştirilerek Tshark aracından faydalanılmıştır.

Ichino ve arkadaşları tarafından zararlı yazılımların trafiğinin incelenmesine yönelik yapılan çalışmada CCC veri seti kullanılmıştır [20]. Söz konusu veri setinde, normal veri olarak bir iç ağdaki trafik verileri pcap formatında kaydedilmiştir. Zararlı trafik için ise balküpü (honeypot) trafiği kullanılmıştır. Elde edilen pcap formatındaki trafik verileri incelenmiş ve filtrelenmiştir.

Zhao ve arkadaşları tarafından botnet tespitine yönelik yapılan çalışmada The Honeynet Project ve Lawrence Berkeley National Laboratory (LBNL) veri setleri kullanılmıştır [19]. The Honeynet Project veri seti Apache, SysLog, Iptable gibi log dosyalarından oluşmaktadır. LBNL veri setinde ise pcap formatında, binlerce bilgisayardan elde edilen 100 saatten fazla internet trafiği kullanılmaktadır.

4.2. Bu Tez Kapsamında Oluşturulan Veri Seti

Bu tez kapsamında geliştirilen uygulama, Şekil 4.1'de görüldüğü üzere toplam 5 aşamadan oluşmakta olup; ilk 3 aşama veri toplama, öznitelik çıkarımı ve etiketleme işlemlerinden oluşmaktadır:

Şekil 4.1. Uygulama aşamalarına ait şema

Yukarıdaki şekilde de görüldüğü üzere öncelikle pcap formatında veri toplanmıştır.

Normal trafik verileri için;

- Bir bilgisayardan, ülkemizde ve dünyada en çok ziyaretçi toplayan ve zararlı bir içeriğe sahip olmadığı teyit edilmiş İnternet adreslerine yönelik trafik oluşturulurken; bir diğer bilgisayardan Wireshark aracı ile İnternet trafiği kaydedilmiştir.
- En çok ziyaretçi toplayan İnternet adresleri Alexa platformundan elde edilmiştir.
- Zararlı bir içeriğe sahip olmadığı doğrulanmış İnternet adresleri ise Google tarafından desteklenen Directory Mozilla (DMOZ) dizin sitesinden elde edilmiştir.

Zararlı trafik verileri için;

- Veri sızıntısı yapan zararlı/casus yazılımlar araştırılarak İnternet'ten, hali hazırda paylaşılmış olan söz konusu yazılımlara ait pcap dosyaları indirilmiş ve bu çalışmada kullanılmıştır.
- Ayrıca bazı zararlı/casus yazılım exe dosyaları da İnternet'ten temin edilmiştir. Exe dosyaları bir bilgisayarda çalıştırılırken, bir başka bilgisayar üzerinden İnternet trafiği kaydedilmiştir.
- Toplamda 12 farklı zararlı yazılım ailesine ait İnternet trafiği kullanılmıştır.

Toplanan veriler belirli bir filtreden geçirilerek; sınıf etiketleriyle beraber veri seti olarak kullanılacak şekilde son haline getirilmiştir. Toplanan verilerin filtrelenmesi için Tshark aracı kullanılarak bir Python scripti hazırlanmıştır. Çizelge 4.1'de görüldüğü gibi hazırlanan script ile pcap dosyasından 22 tane özellik ayıklanmıştır.

Çizelge 4.1. Python scripti ile pcap dosyalarından ayıklanan öznitelikler

	Tshark filtresi	Açıklama
1	tcp.stream	TCP stream indeks numarası
2	udp.stream	UDP stream indeks numarası
3	ip.src	Kaynak IP
4	ip.dst	Hedef IP
5	tcp.srcport	TCP kaynak port
6	tcp.dstport	TCP hedef port
7	udp.srcport	UDP kaynak port

Çizelge 4.1. (devam) Python scripti ile pcap dosyalarından ayıklanan öznitelikler

8	udp.dstport	UDP hedef port
9	tcp.flags.syn	SYN bayrağı
10	tcp.flags.ack	ACK bayrağı
11	tcp.flags.fin	FIN bayrağı
12	tcp.flags.push	PUSH bayrağı
13	tcp.flags.urg	URG bayrağı
14	tcp.flags.reset	RST bayrağı
15	tcp.window_size	Pencere boyutu
16	frame.len	Frame uzunluğu
17	data.len	Veri (payload) boyutu
18	ssl.record.length	SSL veri boyutu
19	frame.time_epoch	Paket gönderim zamanı
20	frame.time_delta	Paket iletim süresi
21	http.request.method	HTTP istek metodu
22	http.user_agent	HTTP user agent bilgisi

Daha sonra, aynı script ile yukarıda belirtilen özniteliklerden yeni öznitelikler oluşturulmuştur. Öznitelik oluşturma safhasında örnek olarak, bir akıştaki paketlerin frame boyutları kaydedilmiş; her bir akış için ortalama, minimum, maksimum, standart sapma, medyan, varyans gibi özellikler çıkarılmıştır. Çıkarılan özelliklerle beraber çalışmada kullanılan tüm öznitelikler, açıklamasıyla birlikte Çizelge 4.2'de sunulmuştur:

Çizelge 4.2. Türetilen özniteliklerle birlikte veri seti açıklamaları

	Öznitelik	Açıklama
1	Src IP	Kaynak IP
2	Dst IP	Hedef IP
3	Src Port	Kaynak Port
4	Dst Port	Hedef Port
5	SYN	SYN Bayrağı
6	ACK	ACK Bayrağı
7	FIN	FIN Bayrağı
8	PUSH	PUSH Bayrağı
9	URG	URG Bayrağı
10	RST	RST Bayrağı
11	synAckOran	SY-CK Oranı
12	finAckOran	FI-CK Oranı
13	pushAckOran	PUSH/ACK Oranı
14	rstAckOran	RST/ACK Oranı
15	windowSizeToplam (TCP)	Toplam Pencere Boyutu

Çizelge 4.2. (devam) Türetilen özniteliklerle birlikte veri seti açıklamaları

16	gidenWindowSizeToplam	Giden paketlerin pencere boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
17	gidenWindowSizeOrt	
18	gidenWindowSizeMin	
19	gidenWindowSizeMax	
20	gidenWindowSizeStd	
21	gidenWindowSizeVar	
22	gidenWindowSizeMed	
23	gelenWindowSizeToplam	Gelen paketlerin pencere boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
24	gelenWindowSizeOrt	
25	gelenWindowSizeMin	
26	gelenWindowSizeMax	
27	gelenWindowSizeStd	
28	gelenWindowSizeVar	
29	gelenWindowSizeMed	
30	frameLenToplam	Toplam Frame Boyutu (byte)
31	gidenGelenFrameOran	Giden paketlerin gelen paketlerin frame boyutuna oranı
32	gidenFrameLenToplam	Giden paketlerin frame boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
33	gidenFrameLenOrt	
34	gidenFrameLenMin	
35	gidenFrameLenMax	
36	gidenFrameLenStd	
37	gidenFrameLenVar	
38	gidenFrameLenMed	
39	gelenFrameLenToplam	Gelen paketlerin frame boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
40	gelenFrameLenOrt	
41	gelenFrameLenMin	
42	gelenFrameLenMax	
43	gelenFrameLenStd	
44	gelenFrameLenVar	
45	gelenFrameLenMed	
46	bytePerSecToplam	1 saniyedeki toplam paket boyutu (byte)
47	bytePerSecGiden	1 saniyedeki giden toplam paket boyutu (byte)
48	bytePerSecGelen	1 saniyedeki gelen toplam paket boyutu (byte)
49	paketSayisiToplam	Toplam paket sayısı
50	paketSayisiGiden	Giden paket sayısı
51	paketSayisiGelen	Gelen paket sayısı
52	paketSayisiPerSecToplam	1 saniyedeki toplam paket sayısı
53	paketSayisiPerSecGiden	1 saniyedeki giden toplam paket sayısı

Çizelge 4.2. (devam) Türetilen özniteliklerle birlikte veri seti açıklamaları

54	paketSayisiPerSecGelen	1 saniyedeki gelen toplam paket sayısı
55	dataLenToplam	Toplam veri boyutu (byte)
56	gidenDataLenToplam	Giden paketlerin veri boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
57	gidenDataLenOrt	
58	gidenDataLenMin	
59	gidenDataLenMax	
60	gidenDataLenStd	
61	gidenDataLenVar	
62	gidenDataLenMed	
63	gelenDataLenToplam	Gelen paketlerin veri boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
64	gelenDataLenOrt	
65	gelenDataLenMin	
66	gelenDataLenMax	
67	gelenDataLenStd	
68	gelenDataLenVar	
69	gelenDataLenMed	
70	dataChunk	Veri yığını var mı (evet/hayır)
71	sslDataLenToplam	Toplam SSL veri boyutu (byte)
72	gidenSSLDataLenToplam	Giden paketlerin SSL veri boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
73	gidenSSLDataLenOrt	
74	gidenSSLDataLenMin	
75	gidenSSLDataLenMax	
76	gidenSSLDataLenStd	
77	gidenSSLDataLenVar	
78	gidenSSLDataLenMed	
79	gelenSSLDataLenToplam	Gelen paketlerin SSL veri boyutlarının toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (byte)
80	gelenSSLDataLenOrt	
81	gelenSSLDataLenMin	
82	gelenSSLDataLenMax	
83	gelenSSLDataLenStd	
84	gelenSSLDataLenVar	
85	gelenSSLDataLenMed	
86	sslDataChunk	SSL veri yığını var mı (evet/hayır)
87	duration	Akış süresi (saniye)
88	gidenFrameTimeDeltaToplam	Giden paketlerin iletim sürelerinin toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (saniye)
89	gidenFrameTimeDeltaOrt	
90	gidenFrameTimeDeltaMin	
91	gidenFrameTimeDeltaMax	
92	gidenFrameTimeDeltaStd	
93	gidenFrameTimeDeltaVar	
94	gidenFrameTimeDeltaMed	

Çizelge 4.2. (devam) Türetilen özniteliklerle birlikte veri seti açıklamaları

95	gelenFrameTimeDeltaToplam	Gelen paketlerin iletim sürelerinin toplam, ortalama, minimum, maksimum, standart sapma, varyans ve medyan değerleri (saniye)
96	gelenFrameTimeDeltaOrt	
97	gelenFrameTimeDeltaMin	
98	gelenFrameTimeDeltaMax	
99	gelenFrameTimeDeltaStd	
100	gelenFrameTimeDeltaVar	
101	gelenFrameTimeDeltaMed	
102	httpRequestMethod	HTTP istek metodu
103	httpUserAgent	HTTP user agent bilgisi uzunluğu
104	sinif	Sınıf etiketi (normal/anormal)

Yukarıdaki çizelgede gösterilen özniteliklerden ilk 3 adedi (Src IP, Dst IP ve Src Port), ayırt edici bir özelliğe sahip olmadığı için veri setinde kullanılmamıştır. Sonuç olarak, bu çalışmanın veri setinde, 100 tane öznitelik ve normal/ anormal olmak üzere 2 tane sınıf etiketi kullanılmıştır. Kullanılan veri seti, 16221 tane akış verisinden oluşmaktadır. Bunlardan 14629 adedi normal trafik, 1592 adedi ise zararlı trafik verileridir.

Çalışmanın son aşamasında kullanılan makine öğrenmesi algoritmalarının sağlıklı çalışabilmesi ve gerçek hayatta uygulanabilirliğinin olması için veri seti gerçekçi ve düzgün dağılıma sahip olarak oluşturulmuştur.

Normal trafik verileri içerisinde arama motorları, DNS sorguları, forum siteleri, dosya yükleme siteleri, haber siteleri, blog siteleri vb. çeşitli kategorilerde İnternet site trafikleri bulundurulmuştur. Son kullanıcıların sıklıkla kullandığı 672 tane İnternet adresi ve bu İnternet adreslerine ait 676 tane IP adresine yönelik İnternet trafik verisi kullanılmıştır.

Zararlı trafik verileri içerisinde ise trojan ve keylogger tarzı, içlerinde Vontime, Ramnit, Azorult, Chithonic, Dridex, Formbook, Hancitor, Zeus, Gozi-ISFB ve Trickbot ile beraber ismi bilinmeyen zararlı yazılımların yer aldığı 12 farklı zararlı/casus yazılıma ait 67 tane İnternet adresi ve toplamda 72 tane IP adresine ait İnternet trafik verisi kullanılmıştır.

Ağ trafiğindeki anormal trafikleri tespit etme konusunda yapılan literatürdeki çalışmalarda ve bu tez çalışmasında kullanılan öznitelik sayıları Çizelge 4.3’te paylaşılmıştır:

Çizelge 4.3. Bu tez çalışmasında ve literatürde kullanılan öznitelik sayısı

Çalışma	Öznitelik Sayısı
Tez çalışması	100
[19]	13
[20]	36
[21]	10
[22]	9
[23]	29
[27]	32
[28]	67
[29]	8
[31]	14
[34]	12
[36]	6
[37]	14
[41]	9
[45]	39

Yukarıdaki tabloda da görüldüğü üzere, bu tez için oluşturulan veri setinde kullanılan öznitelik sayısı, diğer çalışmalara göre daha fazladır. Bu tez kapsamında oluşturulan veri setinde öznitelikler belirlenirken; ağ trafiğindeki anormal trafikleri tespit etme konusunda yapılan literatürdeki çalışmalar detaylıca incelenmiş ve söz konusu çalışmalarda kullanılan özniteliklerden de faydalanılmıştır. Literatürde kullanılan onlarca tane niteliğin kullanılmasının yanı sıra, bu tezin amacı olan sistemden veri sızıntısı yapan zararlı/casus yazılım trafiğinin tespit edilmesi amacı göz önüne alınarak; literatürde kullanılmayan yeni özniteliklere de yer verilmiştir. Örnek olarak bir akıştaki toplam paket boyutlarının ve toplam iletim sürelerinin minimum, maksimum, ortalama, standart sapma gibi değerlerini kullanmak yerine; bir akıştaki paketler gelen/giden olmak üzere ayrılarak; minimum, maksimum, ortalama, standart sapma gibi değerler ayrı ayrı kullanılmıştır. Ayrıca çoğu çalışmada yer almayan payload boyutları, pencere boyutları vb. özniteliklere ilişkin minimum, maksimum, ortalama, standart sapma, varyans, medyan gibi değerler bu çalışmada kullanılmıştır. Bu tez kapsamında incelenen literatür çalışmalarında yer almayıp, bu çalışmada kullanılan 59 tane öznitelik bulunmakta olup; bu öznitelikler Çizelge 4.4'te sunulmuştur:

Çizelge 4.4. Bu tez kapsamında incelenen literatür çalışmalarında yer almayıp, bu çalışmada kullanılan öznitelikler

	Öznitelik	Açıklama
1	gidenWindowSizeToplam	Giden ve gelen paketlerin pencere boyutlarına ilişkin değerler (byte)
2	gidenWindowSizeOrt	
3	gidenWindowSizeMin	
4	gidenWindowSizeMax	
5	gidenWindowSizeStd	
6	gidenWindowSizeVar	
7	gidenWindowSizeMed	
8	gelenWindowSizeToplam	
9	gelenWindowSizeOrt	
10	gelenWindowSizeMin	
11	gelenWindowSizeMax	
12	gelenWindowSizeStd	
13	gelenWindowSizeVar	
14	gelenWindowSizeMed	
15	gidenGelenFrameOran	Giden ve gelen paketlerin frame boyutlarına ilişkin değerler (byte)
16	gidenFrameLenToplam	
17	gidenFrameLenMed	
18	gelenFrameLenVar	
19	gelenFrameLenMed	
20	bytePerSecGiden	1 saniyede giden/gelen paketlerin boyutları ve sayıları
21	bytePerSecGelen	
22	paketSayisiPerSecGiden	
23	paketSayisiPerSecGelen	

Çizelge 4.4. (devam) Bu tez kapsamında incelenen literatür çalışmalarında yer almayıp, bu çalışmada kullanılan öznitelikler

24	dataLenToplam	Payload boyutlarına ilişkin değerler (byte)
25	gidenDataLenOrt	
26	gidenDataLenMin	
27	gidenDataLenMax	
28	gidenDataLenStd	
29	gidenDataLenVar	
30	gidenDataLenMed	
31	gelenDataLenOrt	
32	gelenDataLenMin	
33	gelenDataLenMax	
34	gelenDataLenStd	
35	gelenDataLenVar	
36	gelenDataLenMed	
37	sslDataLenToplam	
38	gidenSSLDataLenToplam	
39	gidenSSLDataLenOrt	
40	gidenSSLDataLenMin	
41	gidenSSLDataLenMax	
42	gidenSSLDataLenStd	
43	gidenSSLDataLenVar	
44	gidenSSLDataLenMed	
45	gelenSSLDataLenToplam	
46	gelenSSLDataLenOrt	
47	gelenSSLDataLenMin	
48	gelenSSLDataLenMax	
49	gelenSSLDataLenStd	
50	gelenSSLDataLenVar	
51	gelenSSLDataLenMed	
52	dataChunk	Veri yığını var mı (evet/hayır)
53	sslDataChunk	
54	gidenFrameTimeDeltaToplam	Gelen ve giden paketlerin iletim sürelerine ilişkin değerler (saniye)
55	gidenFrameTimeDeltaVar	
56	gidenFrameTimeDeltaMed	
57	gelenFrameTimeDeltaToplam	
58	gelenFrameTimeDeltaVar	
59	gelenFrameTimeDeltaMed	

Herhangi bir probleme karşı kullanılan makine öğrenmesi yöntemlerinde, oluşturulan veri setinde kullanılan özniteliklerin yanı sıra veri sayısı ve çeşitliliği de önemlidir. Bu tezde

kullanılan veri setindeki veri miktarına bakıldığında; toplam 16221 tane akış verisi kullanıldığı görülmektedir. Kısıtlı bir süre içerisinde normal verilerin tamamı, zararlı verilerin ise yaklaşık yarısı yazar tarafından manuel olarak oluşturulmuş olup; bu tür problemler için bu tez kapsamında oluşturulan veri setinde kullanılan veri miktarının az olduğu değerlendirilmektedir.

Fakat diğer çalışmalarla kıyaslandığında; bu veri seti oluşturulurken, veri setinin gerçekçi ve düzgün dağılıma sahip olmasına özen gösterilmiştir. Örnek olarak normal trafik verilerinde DNS sorguları, forum, haber, blog, dosya yükleme vb. sitelerin trafiği kullanılarak çeşitlilik bol tutulmuştur. Ayrıca bu tezin amacı olan sistemden veri sızıntısı yapan zararlı/casus yazılımların İnternet trafiğinin tespit edilmesi hususu göz önüne alındığında; bir bilgisayara bulaşan zararlı/casus yazılımın, komuta kontrol sunucularına veri aktarması ve dolayısıyla dışarıya aktarılan veri miktarının fazla olması beklenmektedir. Bu kapsamda, dosya yükleme sitelerine çeşitli boyutta ve formatta veriler yüklenerek İnternet trafiği kaydedilmiş olup; bu veriler normal olarak etiketlenmiştir.

Zararlı trafik verilerinde ise birbirinden tamamen farklı 12 farklı zararlı yazılım ailesine ait trafik verileri kullanılmıştır. Gerek yazar tarafından oluşturulan gerekse internetten hazır olarak temin edilen pcap formatındaki trafik verileri, Wireshark aracı ile belirli filtrelemeler (iletilen paketlerin boyutları, paketlerin iletim sıklığı, kriptolu/kriptosuz payload olup olmadığı, veri yığını yapılıp yapılmadığı vb.) yapılarak analiz edilmiş ve İnternet trafiğinde bir veri sızıntısının olup olmadığı teyit edilmiştir.

Ağ trafiğindeki anormallik tespiti çalışmalarında KDD, ISCX gibi hazır veri setleri, literatürde sıklıkla kullanılmaktadır. KDD veri setinde, temel TCP bağlantı özellikleri, bağlantı içerik özellikleri ve iki saniyelik window içinde kullanılan trafik özelliklerinin yer aldığı onlarca adet öznitelik kullanılmakta olup; saldırı tespit sistemi çalışmalarında kullanılmaktadır. ISCX veri setlerinde ise Tor, VPN, botnet, Android vb. internet trafiklerine yönelik veri kümeleri almaktadır. Bu tez için veri seti oluşturulurken; belirtilen hazır veri setlerinden fikir edinilmiş olup; bu tezde kullanılan birçok öznitelik, hazır veri setlerinde yer almadığı için kullanılamamıştır.

5. DENEYSEL SONUÇLAR

Veri seti toplandıktan sonra elde edilen veriler üzerinde çeşitli makine öğrenmesi algoritmaları uygulanmıştır. Algoritmaların başarıları ölçülürken, 10-fold cross-validation yöntemi kullanılmıştır.

Bu çalışmada öncelik olarak, normal ve anormal trafiklerden oluşan veriler arasında, anormal trafiğin tespit edilmesi amaçlanmıştır.

Çizelge 5.1. Tespit durumları

Durum	Açıklama
TP	Anormal Trafik, Anormal olarak tespit edildi
FP	Normal Trafik Anormal olarak tespit edildi
TN	Normal Trafik Normal olarak tespit edildi
FN	Anormal Trafik Normal olarak tespit edildi

5.1. Başarı Ölçütleri

Çalışmanın başarı durumu değerlendirilirken aşağıdaki metrikler kullanılmıştır:

TP Oranı / Recall (Duyarlılık): Anormal veriler içinde yüzde kaçının anormal olarak tahmin edildiği bilgisidir. Recall değerinin yüksek olması FP oranının yüksek olmasına yol açabilmektedir. Yani bütün anormal veriler tahmin edilse bile çok sayıda normal veri anormal olarak sınıflandırılabilir.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.1}$$

TN Oranı / Specificity: Normal veriler içinde yüzde kaçının normal olarak tahmin edildiği bilgisidir. Yani Recall ölçütünün tam tersidir. Bu çalışma için bakıldığında, veri setinin yaklaşık %90'lık bölümü normal verilerden oluştuğu için TN oranının diğer ölçütlere göre daha yüksek çıkması beklenmiştir ve Çizelge 5.2 incelendiğinde; beklenen sonucun elde edildiği görülmektedir.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{5.2}$$

FP Oranı: Normal veriler içinde yüzde kaçının anormal olarak tespit edildiği durumdur. Problemlere göre değişiklik göstermekle birlikte, çoğu problemlerde, FP oranının, FN oranından daha düşük olması beklenmektedir. Bu çalışma kapsamında değerlendirildiğinde; kullanıcının normal bir İnternet adresine gitmek isteyip; sistem tarafından anormal olarak görülmesi neticesinde, isteğin engellenmesi durumudur. Yani yanlış alarmı ifade etmektedir.

$$\text{FP Oranı} = \frac{FP}{FP + TN} \tag{5.3}$$

FN Oranı: Anormal veriler içinde yüzde kaçının normal olarak tespit edildiği durumdur. Bazı problemlerde, FN oranının çok düşük olması beklenmektedir. Bu çalışma için değerlendirildiğinde, FN oranının yüksek olması, anormal bir trafiğin normal olarak görülmesi ve sistem tarafından engellenmemesi durumu temsil edilmektedir.

$$\text{FN Oranı} = \frac{FN}{FN + TP} \tag{5.4}$$

Accuracy (Doğruluk): Model tarafından yapılan doğru tahminlerin, her tahmin sayısına oranıdır.

$$\text{Doğruluk} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.5}$$

Precision (Kesinlik): Anormal olarak tahmin edilen verilerin, gerçekte yüzde kaçının anormal olduğu bilgisi elde edilir. Precision değerinin yüksek olması FN oranın yüksek olmasına yol açabilmektedir. Yani anormal bir veri gerçekten anormal olarak tahmin edilebilir fakat çok sayıda anormal veri gözden kaçırılabilir.

$$\text{Kesinlik} = \frac{TP}{TP + FP} \tag{5.6}$$

F1 – Skoru: Precision ve Recall değerleri arasındaki tutarsızlıktan dolayı F1 Score ölçütü kullanılabilmektedir. Precision ile recall değerlerinin harmonik ortalaması alınarak F1 Score değeri hesaplanmaktadır.

$$F1 - Skoru = \frac{2 \text{ x Precision x Recall}}{\text{Precision + Recall}} \quad (5.7)$$

5.2. Elde Edilen Sonuçlar

Veri seti oluşturduktan sonra YSA, derin öğrenme, karar ağacı, random forest, naive bayes, adaboost, kNN ve SVM algoritmaları kullanılarak sınıflandırma işlemleri yapılmıştır. Bazı algoritmalar üzerinde çeşitili parametre değişiklikleri yapılarak; optimum sonuçlar elde edilmeye çalışılmıştır. Elde edilen sonuçlar, aşağıdaki çizelgelerde sunulmuştur:

Çizelge 5.2. Derin öğrenme ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
6600,0	94,0	39,3	0,0	99,9	60,6	56,1	98,2	99,7

Derin öğrenme ile doğruluk oranı %94,0 olsa da bu tezin amacı olan anormal verilerin tespit edilmesi konusunda %39,3 oranıyla başarısız bir sonuç elde edilmiştir. Derin öğrenme metodu uygulanırken Keras aracı kullanılmış ve çeşitli katman sayılarında sinir ağı oluşturularak denemeler yapılmıştır. Sinir ağı oluşturulurken çeşitli sayılarda katman ve düğüm ile beraber farklı aktivasyon fonksiyonları denenmiştir. Kullanılan sinir ağı 7 katman ve her bir katmanda 50 düğüm ile oluşturulmuştur. Aktivasyon fonksiyonu olarak relu fonksiyonu kullanılmış ve en iyi sonuç bu mimari ile elde edilmiştir.

Çizelge 5.3. YSA ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
519,1	99,4	95,9	0,1	99,8	4,0	97,1	98,4	99,3

YSA ile de çeşitli katman sayılarında sinir ağı oluşturulmuş ve en başarılı sonucun 3 katmanlı sinir ağı ile elde edildiği; katman sayısı arttıkça başarı oranının azaldığı görülmüştür. Anormal verileri tespit etmede %95'in üzerinde başarılı bir sonuç elde edilmiştir. Ayrıca %0,1'lik oranla yanlış alarm üretilmiştir.

Çizelge 5.4. SVM ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
6,1	98,6	89,6	0,3	99,6	10,3	93,0	96,8	94,7

SVM algoritması ile yaklaşık %90 oranında TP oranı elde edilmiş ve %0,3 ile oldukça başarılı sayılabilecek FP oranına ulaşılmıştır.

Çizelge 5.5. kNN ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
26,4	99,8	99,4	0,1	99,9	0,6	99,3	99,4	99,7

kNN algoritmasındaki k değeri (en yakın nokta sayısı) arttıkça başarı oranının azaldığı görülmüştür. K değeri 1 olarak belirlendiğinde, doğruluk, recall, TN oranı, F1 ölçütü ve precision ölçütlerinde, %99'un üzerinde gayet başarılı sonuçlara ulaşılmıştır. %0,1 FP oranı ile oldukça başarılı yanlış alarm oranı (FP oranı) elde edilmiştir.

Çizelge 5.6. Adaboost – Karar Ağacı ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
6,2	99,9	99,7	0,1	99,9	0,3	99,8	99,9	99,9

Adaboost algoritması Karar Ağacı ile birlikte kullanıldığında doğruluk, recall, TN oranı, F1 ölçütü ve precision ölçütlerinde %100'e yakın oranlarda başarılı sonuç elde edilmiştir. Ayrıca %0,1'lik FP oranı ile oldukça düşük yanlış alarm üretilmiştir.

Çizelge 5.7. Naive Bayes ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
0,1	98,9	98,9	1,0	98,9	1,0	94,8	91,1	99,2

Naive Bayes algoritmasında 0,1 saniye ile düşük eğitim süresine ulaşılmıştır. Yaklaşık %99 TP oranı ve %1 FP oranına ulaşılmış ve başarılı bir sonuç elde edilmiştir.

Çizelge 5.8. Random Forest ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
3,6	99,9	99,9	0,0	100,0	0,1	99,9	100,0	100,0

Random Forest ile doğruluk, recall, TN oranı, F1 ölçütü ve precision ölçütlerinin tamamında neredeyse %100 oranında başarılı sonuçlar elde edilmiştir. Bununla beraber %0 FP oranı ve %0,1 FN oranının elde edilmesi ile en başarılı sonuçlara ulaşılmıştır.

Çizelge 5.9. Karar Ağacı ile elde edilen sonuçlar

Süre (Sn)	Doğruluk (%)	TP Oranı /Recall (%)	FP Oranı (%)	TN Oranı/ Specificity (%)	FN Oranı (%)	F1 Ölçütü (%)	Precision (%)	ROC (%)
0,9	99,9	99,7	0,1	99,9	0,3	99,8	99,9	99,8

Karar Ağacı algoritması ile Adaboost (Karar Ağacı ile birlikte) algoritmasından elde edilen sonuçların yaklaşık olarak aynı oranlara sahip olduğu görülmektedir.

Çizelge 5.10. Tez çalışmasından elde edilen sonuçlar

Algoritma	Derin Öğrenme	YSA	SVM	kNN	Adaboost - Karar Ağacı	Naive Bayes	Random Forest	Karar Ağacı
Süre (Sn)	*6600,0*	519,1	6,1	26,4	6,2	0,1	3,6	0,9
Doğruluk (%)	*94,0*	99,4	98,6	99,8	99,9	98,9	99,9	99,9
TP Oranı/ Recall (%)	39,3	95,9	89,6	99,4	99,7	98,9	99,9	99,7
FP Oranı (%)	0,0	0,1	*0,3*	0,1	0,1	1,0	0,0	0,1
TN Oranı/ Specificity (%)	99,9	99,8	99,6	99,9	99,9	*98,9*	100,0	99,9
FN Oranı (%)	*60,6*	4,0	10,3	0,6	0,3	1,0	0,1	0,3
F1 Ölçütü (%)	*56,1*	97,1	93,0	99,3	99,8	94,8	99,9	99,8
Precision (%)	98,2	98,4	96,8	99,4	99,9	*91,1*	100,0	99,9
ROC (%)	99,7	99,3	*94,7*	99,7	99,9	99,2	100,0	99,8

Yukarıdaki tablo incelendiğinde, her bir algoritmadan başarılı sonuçlar elde edildiği görülmektedir. Doğruluk, recall, TN, F1 ölçütü, precision ve ROC oranlarının %100'e yakın, FP ve FN oranlarının ise %0'a yakın başarısıyla her bir ölçütte, Random Forest algoritmasının en başarılı sonucu verdiği görülmüştür.

Her ne kadar %94'lük bir doğruluk oranı elde edilse de bu tezin amacı olan anormal verilerin tespit edilmesi konusunda derin öğrenme metodunun en başarısız sonucu verdiği görülmektedir. 6600 saniye ile denenen algoritmalar arasında en yüksek eğitim süresine ulaşılmıştır.

3 katmanlı YSA ile çeşitli metriklere göre %95,9 ile %99,3 arasında başarı elde edilmiştir. Ayrıca %0,1'lik yanlış alarm oranı (FP oranı) elde edilmiştir. Derin öğrenme metodunun ardından 519,1 saniye ile en yüksek ikinci eğitim süresine ulaşılmıştır. Derin öğrenme metodundan sonra %89,6 ile en düşük TP oranı SVM algoritmasında görülmüştür. kNN algoritmasındaki k değeri (en yakın nokta sayısı) arttıkça başarı oranının azaldığı görülmüştür. K değeri 1 olarak belirlendiğinde, çeşitli metrikler üzerinde %99'un üzerinde başarı elde edilmiştir. Ayrıca %0,1 oranında yanlış alarm (FP) üretilmiştir. Karar ağacı algoritmasının tek başına ve adaboost algoritmasıyla birlikte kullanılması neticesinde aynı sonuçlar elde edilmiş ve çeşitli metriklere göre %99'un üzerinde bir başarı oranına ulaşılmıştır. Ayrıca %0,1 FP oranı elde edilmiştir. Eğitim süresi en kısa süren Naive Bayes algoritmasıyla da çeşitli ölçütlerde yaklaşık %91 - %99 aralığında başarı elde edilmiştir.

Algoritmaların çalışma süreleri incelendiğinde, beklendiği üzere YSA ve derin öğrenme metodunun daha fazla süre harcadığı görülmüştür. Özellikle son yıllarda yapılan çalışmalarda olduğu gibi derin öğrenme metodu için GPU'ların kullanılmasıyla bu sürenin belirgin derecede azalacağı tahmin edilmektedir.

6. SONUÇ VE ÖNERİLER

Bu tezde, makine öğrenmesi yöntemleriyle sistemden veri sızıntısı yapan zararlı/casus yazılımların uygulama katmanı üzerindeki İnternet trafiğinin tespit edilmesi ve bu yöntemlerin uygulanması için gerçekçi ve düzgün dağılıma sahip bir veri seti oluşturulması amaçlanmıştır. Bunun için pcap formatında normal ve zararlı trafik verileri toplanmış, daha sonra hazırlanan bir Python scripti ile 100 tane öznitelikten oluşan 16221 tane akış verisi elde edilmiştir.

Oluşturulan veri seti üzerinde karar ağacı, random forest, naive bayes, adaboost, kNN, SVM, YSA ve derin öğrenme yöntemleri denenmiştir. Her bir yöntem için doğruluk, TP oranı (recall), FP, TN (specificity), FN, F1 ölçütü, precision ve ROC ölçütleri ile değerlendirme yapılmıştır. Doğruluk, recall, TN, F1 ölçütü, precision ve ROC ölçütleri ile %100'e yakın, FP ve FN oranlarının ise %0'a yakın başarısıyla her bir ölçütte, random forest algoritması en başarılı sonucu vermiştir. Karar ağacı, adaboost ve kNN algoritmalarıyla da doğruluk, recall, TN, F1 ölçütü, precision ve ROC ölçütleri ile %99 - %100 aralığında oldukça başarılı sonuçlar elde edilmiştir. Ayrıca FP ve FN oranları incelendiğinde, %1'in altında oranların elde edildiği görülmektedir. Naive bayes, SVM ve YSA algoritmalarıyla derin öğrenme hariç diğer algoritmalara nispeten daha düşük oranda başarılı sonuçlar elde edilmiştir. En düşük başarı oranını ise derin öğrenme yöntemi vermiştir. Özellikle anormal veriler içinde yüzde kaçının anormal olarak tespit edildiğini gösteren %39,3'lük TP oranı (recall) ile oldukça düşük bir başarı oranı elde edilmiştir.

Bu tezde uygulanan makine öğrenmesi yöntemleri arasında, derin öğrenme yönteminin en başarısız sonucu vermesinin sebebinin veri sayısının az olmasından kaynaklandığı değerlendirilmektedir. Kısıtlı bir süre içerisinde normal verilerin tamamı, zararlı verilerin ise yaklaşık yarısı yazar tarafından manuel olarak oluşturulmuş olup; bu tür problemler için bu tez kapsamında oluşturulan veri setindeki veri miktarı, derin öğrenme yöntemi için yetersiz kalmıştır. Derin öğrenme yöntemi, literatürde milyonlarca veriden oluşan veri seti ile GPU'lar üzerinde uygulanmaktadır. Bu tezde ise 16221 tane akış verisinden oluşan veri seti ile kişisel bilgisayar üzerinde uygulanmıştır.

Bu tez kapsamında oluşturulan veri seti, literatürdeki diğer çalışmalarda kullanılan veri setine kıyasla daha az veriden oluşsa da daha fazla ayırt edici öznitelik kullanılmıştır. Çeşitli

ölçütlere göre elde edilen yüksek başarı oranlarının, bu tez için oluşturulan özniteliklerden kaynaklandığı değerlendirilmektedir. Bu çalışmanın gerçek hayatta uygulanabilirliğinin optimum seviyede olması için, gerek normal gerekse anormal İnternet trafiği verilerinin çeşitlendirilmesi önem arz etmektedir. Ayrıca veri sayısının arttırılmasıyla diğer algoritmalara göre daha düşük başarı oranı elde edilen derin öğrenme metodundan da daha başarılı sonuç elde edileceği değerlendirilmektedir.

Veri setindeki veri miktarını arttırmak ve daha kısa sürede oluşturmak için manuel olarak toplamak yerine literatürdeki bazı çalışmalarda olduğu gibi hazırlanacak bir script ile normal trafik verisi toplanmasının faydalı olacağı değerlendirilmektedir. Veri setinin gerçekçi olması için tüm veriler değil de haber, blog, forum siteleri, DNS sorguları gibi trafik verileri, bir script ile kolaylıkla toplanabilecektir. Fakat bir siteye yorum yapma, dosya yükleme sitelerine çeşitli format ve boyutlarda dosya yükleme, sitelere login olma gibi durumların, bir script ile otomatize edilmesinin zor olacağı değerlendirilmektedir.

Veri setinin gerçekçi olması ve düzgün dağılıma sahip olması için normal ve anormal verilere, sırayla yaklaşık %90 - %10 oranında yer verilmiştir. Bu nedenle sadece doğruluk oranı ile bir değerlendirme yapmak yanlış olacağı gibi özellikle bu tezin amacı olan anormal verileri tespit etme konusunda TP oranına bakmak gereklidir. Çünkü veri setindeki her bir veri normal olarak etiketlense bile %90 doğruluk oranı elde edilecektir. Bu tezde yüksek doğruluk oranı ile birlikte yüksek TP oranı ve diğer ölçütlere göre de birçok algoritma ile %95'in üzerinde başarı oranları elde edilmiştir. Bu durum da, bu tezin gayet başarılı bir sonuç verdiğini ve gerçek hayatta uygulanabilirliğinin olduğunu göstermektedir.

Veri setinin eğitilmesi süreleri incelendiğinde, beklendiği üzere YSA ve derin öğrenme yönteminin, diğer yöntemlere göre fazla zaman aldığı görülmüştür. En düşük eğitim süresi ise naive bayes algoritmasıyla elde edilmiştir.

Bu çalışmanın devam ettirilmesi durumunda, toplam akış verisi sayısının milyonlara ulaştırılarak; gerek normal gerekse anormal verilerin çeşitliliğinin artırılmasıyla çok daha başarılı sonuçlar elde edileceği ve gerçek hayatta uygulanabilirliğinin daha da artacağı umulmaktadır. Özellikle derin öğrenme metodu için GPU'ların da kullanımıyla hesaplama süresinin de kısalacağı beklenmektedir.

KAYNAKLAR

1. İnternet: Malware Statistics & Trends Report | AV-TEST. URL: http://www.webcitation.org/6yg3k7KOf, Son Erişim Tarihi: 14.04.2018.

2. Tounsi W., Rais H. (2018). A survey on technical threat intelligence in the age of sophisticated cyber attacks. *Computers & Security,* 72, 212-233.

3. İnternet: The Industrial Control System Cyber Kill Chain. URL: https://www.sans.org/reading-room/whitepapers/ICS/industrial-control-system-cyber-kill-chain-36297, Son Erişim Tarihi: 14.04.2018.

4. Bryant B. D., Saiedian H. (2017). A novel kill-chain framework for remote security log analysis with SIEM software. *Computers & Security,* 67, 198-210.

5. Miloslavskaya N. (2018). Remote Attacks Taxonomy and their Verbal Indicators. *Procedia Computer Science,* 123, 278-284.

6. Kim D. Y., Song H. Y. (2018). Method of predicting human mobility patterns using deep learning. *Neurocomputing,* 280, 56-64.

7. Lin S., Ying K., Lee C. and Lee Z. (2012). An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. *Applied Soft Computing,* 12(10), 3285-3290.

8. Öztemel E. (2012). *Yapay Sinir Ağları* (Üçüncü Baskı). İstanbul: Papatya Yayıncılık, 29-57.

9. Ferentinos K. P. (2018, February), Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture,* 145, 311-318.

10. İnternet: Artificial Neural Network (Yapay Sinir Ağı). URL: http://www.webcitation.org/6yqttOdcY, Son Erişim Tarihi: 21.04.2018.

11. Diro A. A., Chilamkurti N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Generation Computer Systems,* 82, 761-768.

12. Kamilaris A., Prenafeta-Boldu F. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture,* 147, 70-90.

13. Gu J., Wang Z., Kuen J., Ma L., Shahroudy A., Shuai B., Liu T., Wang X., Wang G., Cai J. and Chen T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition,* 77, 354-377.

14. Wang W., Zhu M., Zeng X., Ye X. and Sheng Y. (2017, January). *Malware Traffic Classification Using Convolutional Neural Network for Representation Learning.* Information Networking, Da Nang, Vietnam.

15. Tang T., Mhamdi L., McLernon D., Zaidi S. A. R. and Ghogho M. (2016, October). *Deep Learning Approach for Network Intrusion Detection in Software Defined Networking.* Wireless Networks and Mobile Communications, Fez, Morocco.

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

17. Kalash M., Rochan M., Mohammed N., Bruce N. D. B., Wang Y. and Iqbal F. (2018). *Malware Classification with Deep Convolutional Neural Networks.* IFIP International Conference on New Technologies, Mobility and Security, Paris.

18. Chougrad H., Zouaki H. and Alheyane O. (2018). Deep Convolutional Neural Networks for breast cancer screening. *Computer Methods and Programs in Biomedicine*, 157, 19-30.

19. Zhao D., Traore I., Sayed B., Lu W., Saad S., Ghorbani A. and Garant D. (2013). Botnet detection based on traffic behavior analysis and flow intervals. *Computers & Security,* 39*, 2-16.*

20. Ichino M., Kawamoto K., Iwano T., Hatada M. and Yoshiura H. (2015). Evaluating Header Information Features for Malware Infection Detection. *Journal of Information Processing*, 23(5), 603-612.

21. Kumar S., Viinikainen A. and Hamalainen T. (2016). *Machine Learning Classification Model For Network Based Intrusion Detection System.* The 11th International Conference for Internet Technology and Secured Transactions, Barcelona.

22. Azab A., Alazab M. and Aiash M. (2016). *Machine learning based Botnet Identification Traffic*. 2016 IEEE TrustCom/BigDataSE/ISPA, Tianjin, China.

23. Bou-Harb E., Lakhdari N., Binsalleeh H. and Debbabi M. (2014). Multidimensional investigation of source port 0 probing. *Digital Investigation,* 11(2), 114-123.

24. Mirza Q., Awan I. and Younas M. (2017). CloudIntell: An Intelligent Malware Detection System. *Future Generation Computer Systems*, 86, 1042-1053.

25. Buczak A., Guven E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, 18(2), 1153-1176.

26. McLaren P., Russell G. and Buchanan B. (2017). *Mining Malware Command and Control Traces.* 2017 Computing Conference, London.

27. Singh K., Guntuku S., Thakur A. and Hota C. (2014). Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests. *Information Sciences,* 278, 488-497.

28. Boukhtouta A., Mokhov S., Lakhdari N., Debbabi M. and Paquet J. (2015). Network malware classification comparsion using DPI and flow packet headers. *Journal of Computer Virology and Hacking Techniques*, 12(2), 69-100.

29. Mimura M., Otsubo Y., Hidehiko T. and Hidema T. (2017). *A Practical Experiment of the HTTP-Based RAT Detection Method in Proxy Server Logs.* 12th Asia Joint Conference on Information Security, Seoul, South Korea.

30. Radoglou-Grammatikis P., Sarigiannidis P. (2017). *Flow Anomaly Based Intrusion Detection System for Android Mobile Devices.* 6th International Conference on Modern Circuits and Systems Technologies, Thessaloniki, Greece.

31. Boero L., Marchese M. and Zappatore S. (2017). *Support Vector Machine meets Software Defined Networking in IDS domain.* 29th International Teletraffic Congress, Genoa, Italy.

32. Cuzzocrea A., Martinelli F., Mercaldo F. and Vercelli G. (2017). *Tor Traffic Analysis and Detection via Machine Learning Techniques.* IEEE International Conference on Big Data, Boston, MA.

33. Cheng Z., Chen X., Zhang Y., Li S. and Sang Y. (2017). *Detecting Information Theft Based on Mobile Network Flows for Android Users.* International Conference on Networking, Architecture, and Storage, Shenzhen, China.

34. Verma A., Ranga V. (2018). Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine Learning. *Procedia Computer Science*, 125, 709-716.

35. Shah S., Issac B. (2018). Performance comparison of intrusion detection systems and application of machine learning to Snort system. *Future Generation Computer Systems*, 80, 157-170.

36. Chen Z., Yan Q., Han H., Wang, S., Peng L., Wang L. and Yang B. (2018). Machine learning based mobile malware detection using highly imbalanced network traffic. *Information Sciences,* 346-364.

37. Wang J., Yang L., Wu J. and Abawajy J. (2017). *Clustering Analysis for Malicious Network Traffic. IEEE* ICC 2017 Communication and Information Systems Security Symposium, Paris.

38. Niu W., Zhang X., Yang G., Zhu J. and Ren Z. (2017). Identifying APT Malware Domain Based on Mobile DNS Logging. *Mathematical Problems in Engineering*, 1-9.

39. Cao J., Drabeck L. and He R. (2017). *Statistical Network Behavior Based Threat Detection.* 2017 IEEE Conference on Computer Communications Workshops, Atlanta, GA.

40. Hodo E., Bellekens X., Iorkyase E., Hamilton A., Tachtatzis C. and Atkinson R. (2017). *Machine Learning Approach for Detection of nonTor Traffic.* Proceedings of the 12th International Conference on Availability, Reliability and Security, Regio Callabria, ITALY.

41. Lashkari A., Kadir A., Gonzalez H., Mbah K. and Ghorbani A. (2017). *Towards a Network-Based Framework for Android Malware Detection and Characterization.* 15th International Conference on Privacy, Security and Trust, San Juan, Puerto Rico.

42. Kebede T., Djaneye-Boundjou O., Narayanan B., Ralescu A. and Kapp D. (2017). *Classification of Malware Programs using Autoencoders based Deep Learning Architecture and its Application to the Microsoft Malware Classification Challenge (BIG 2015) Dataset.* Aerospace and Electronics Conference, Dayton, OH.

43. Meng X., Shan Z., Liu F., Zhao B., Han J., Wang J. and Wang H. (2017). *MCSMGS: Malware Classification Model Based on Deep Learning.* International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Nanjing, China.

44. Bulut İ., Yavuz A. (2017). *Zararlı Mobil Uygulamaların Derin Yapay Sinir Ağı ile Tespit Edilmesi.* 25th Signal Processing and Communications Applications Conference, Antalya.

45. Kant V., Singh M. and Ojha N. (2017). *An Efficient Flow based Botnet Classification using Convolution Neural Network.* International Conference on Intelligent Computing and Control Systems, Madurai, India.

46. Liu L., Wang B. (2016). *Malware classification using gray-scale images and ensemble learning.* 3rd International Conference on Systems and Informatics, Shanghai, China.

47. Choi S., Jang S., Kim Y. and Kim J. (2017). *Malware Detection using Malware Image and Deep Learning.* Information and Communication Technology Convergence, Jeju, South Korea.

48. Luo J., Lo D. (2017). Binary Malware Image Classification using Machine Learning with Local Binary Pattern. *Big Data,* 4664-4667.

49. Shibahara T., Yagi T., Akiyama M., Chiba D. and Yada T. (2016). *Efficient Dynamic Malware Analysis Based on Network Behavior Using Deep Learning.* Global Communications Conference, Washington, DC.

50. Kotsiantis S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica,* 31(3), 249-268.

51. Ibanez A., Bielza C. And Larranaga P. (2014). Cost-sensitive selective naive Bayes classifiers for predicting the increase of the h-index for scientific journals. *Neurocomputing,* 135, 42–52.

52. Gowrison G., Ramar K., Muneeswaran K. and Revathi T. (2012). Minimal complexity attack classification intrusion detection system. *Applied Soft Computing,* 13(2), 921–927.

53. Zhang, Q., Xi C., Wang F. and V. K. Devabhaktuni (1999). *Neural Network Structures for RF and Microwave Applications.* IEEE AP-S Antennas and Propagation, Orlando, FL.

54. Al Amrani Y., Lazaar M. and El Kadiri K. E. (2018). Random Forest and Support Vector Machine based Hybrid Approach to Sentiment Analysis. *Procedia Computer Science,* 127, 511–520.

55. Rodríguez-Galiano V.F., Abarca-Hernández F., Ghimire B., Chica-Olmo M., Akinson P.M. and Jeganathan C. (2011). Incorporating Spatial Variability Measures in Land-cover Classification using Random Forest, *Procedia Environmental Sciences*, 3, 44-49.

56. Elbasiony R. M., Sallam E. A., Eltobely T. E. and Fahmy M. M. (2013). A hybrid network intrusion detection framework based on random forests and weighted k-means. *A in Shams Engineering Journal*, 4(4), 753-762.

57. Güran A., Uysal M. ve Doğrusöz Ö. (2014). Destek Vektör Makineleri Parametre Optimizasyonunun Duygu Analizi Üzerindeki Etkisi. *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Mühendislik Bilimleri Dergisi,* 16(48), 86-93.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: ERGİNAY, Emrullah
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 19.04.1990, Ankara
Medeni hali	: Evli
Telefon	: 0 (506) 423 19 73
E-mail	: emrullaherginay@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi /Bilgisayar Mühendisliği	Devam Ediyor
Lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	2013
Lise	Fethiye Kemal Mumcu Anadolu Lisesi	2008

İş Deneyimi

Yıl	Yer	Görev
2015	Türksat	Bilgisayar Mühendisi
2015-Halen	Cumhurbaşkanlığı	Bilgisayar Mühendisi

Yabancı Dil

İngilizce

Yayınlar

1. Erginay E., Akcayol M. A. (2019). *Kurumsal Ağ Trafiğinde Derin Öğrenme Tabanlı Anormallik Tespiti Uygulaması*. Uluslararası Veri Bilimi ve Mühendisliği Sempozyumu, Karabük.

Hobiler

Futbol, Kitap

GAZİ GELECEKTİR...