
GAZİ ÜNİVERSİTESİ
Gazi
1926

SÖZCÜKSEL ANALİZ KULLANARAK KÖTÜ NİYETLİ URL'LERİ DERİN ÖĞRENME TEKNİKLERİ İLE TESPİT ETME

Cemile SARICAOĞLU

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

EYLÜL 2019

Cemile SARICAOĞLU tarafından hazırlanan "SÖZCÜKSEL ANALİZ KULLANARAK KÖTÜ NİYETLİ URL'LERİ DERİN ÖĞRENME TEKNİKLERİ İLE TESPİT ETME" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Dr. Öğr. Üyesi Mehmet DEMİRCİ

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. .....

Başkan: Prof. Dr. Ömer Faruk BAY

Elektrik Elektronik Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. .....

Üye: Dr. Öğr. Üyesi Adnan ÖZSOY

Bilgisayar Mühendisliği Ana Bilim Dalı, Hacettepe Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. .....

Tez Savunma Tarihi: 16/09/2019

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

.....

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

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.

Cemile SARICAOĞLU

16/09/2019

SÖZCÜKSEL ANALİZ KULLANARAK KÖTÜ NİYETLİ URL'LERİ DERİN ÖĞRENME TEKNİKLERİ İLE TESPİT ETME

(Yüksek Lisans Tezi)

Cemile SARICAOĞLU

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Eylül 2019

ÖZET

Günümüzde yeni geliştirilen teknolojiler insan hayatını kolaylaştırmaktadır. Fakat internet ve siber âlem siber saldırılara açık bir ortamdır. URL'ler saldırı için kullanılan temel araçlardan bir tanesidir. Kötü niyetli URL'ler phishing, spam, finansal dolandırıcılık ve malware gibi pek çok internet suç aktivitesi için temel bir mekanizmadır. İnternetteki kötü niyetli URL'leri etkili bir şekilde tespit etmek ve sınıflandırmak amacıyla araştırmacılar tarafından kara liste hizmetleri geliştirilmiştir. Kötü niyetli URL'leri kara listeye almak hem kötü niyetli URL'nin hem de yeni oluşturulan URL'nin varyasyonlarını bulmakta tamamen etkili olmadığı için bu durum sorunun yalnızca bir kısmını çözmektedir. Devamlı güncelleme yapılması gerektiği için de zaman alıcı bir yaklaşımdır. Kötü niyetli URL'lerin saldırı türlerine göre tespit edilmesi ve sınıflandırılması bu saldırıları engellemek için kritik öneme sahiptir. Bir tehdidin türünü bilmek, saldırının ciddiyetinin tahmin edilmesini sağlamakta ve etkili bir önlem alınmasına yardımcı olmaktadır. Bu tez çalışmasında, kötü niyetli URL'lerin saldırı türlerine göre tespit edilmesi ve sınıflandırılması için derin öğrenme kullanan bir yöntem önerilmiştir. Aynı zamanda literatürde çok sayıda örneği olan makine öğrenmesi yöntemi de kullanılmıştır. URL'lerin proaktif tespiti için sözcüksel analiz kullanılmıştır. İyi huylu, spam, phishing, malware ve defacement olmak üzere beş farklı URL türü hem makine öğrenmesi hem de derin öğrenme yöntemleri ile incelenmiştir. İki yöntemde de ikili ve çoklu sınıflandırma yapılmıştır. Sonuçlar kendi içlerinde karşılaştırılmıştır. Alınan sonuçlar literatür ve makine öğrenmesi algoritmaları ile karşılaştırıldığında daha başarılı sonuçlar elde edilmiştir.

Bilim Kodu : 92403
Anahtar Kelimeler : Kötücül URL, Siber Güvenlik, Sözcüksel Analiz, Derin Öğrenme
Sayfa Adedi : 63
Danışman : Dr. Öğr. Üyesi Mehmet DEMİRCİ

DETECTING MALICIOUS URLS USING LEXICAL ANALYSIS WITH DEEP LEARNING TECHNIQUES

(M. Sc. Thesis)

Cemile SARICAOĞLU

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

September 2019

ABSTRACT

Today, newly developed technologies make human life easier. But the internet and cyber realm is an open environment for cyber-attacks. URLs are one of the main tools used for cyber-attack. Malicious URLs are a key mechanism for many Internet criminal activity, such as phishing, spam, identity theft, financial fraud and malware. Blacklist services have been developed by researchers to effectively identify and classify malicious URLs on the Internet. Blacklisting malicious URLs solves only part of the problem, as it is not entirely effective in finding variations of both the malicious URL and the newly created URL. It is a time-consuming approach as continuous updating is required. Detecting and classifying malicious URLs by attack types is critical to preventing these attacks. Knowing the nature of a threat provides an estimate of the severity of the attack and helps to take effective action. In this thesis, a method that uses deep learning is proposed to detect and classify malicious URLs according to attack types. At the same time, machine learning method which has many examples in literature has been used. Lexical analysis was used for the proactive detection of URLs. Five different types of URLs, benign, spam, phishing, malware and defacement, have been examined through both machine learning and deep learning methods. Binary and multiple classification were performed in both methods. The results were compared in-house. Compared to literature and machine learning algorithms, the results were more successful.

Science Code : 92403
Key Words : Malicious URL, Cyber Security, Lexical Analysis, Deep Learning
Page Number : 63
Supervisor : Assist. Prof. Dr. Mehmet DEMİRCİ

TEŞEKKÜR

Bu çalışmanın gerçekleşmesine katkılarından dolayı ve danışmanım olarak tezin yazılmasında yol gösterdiği için sayın hocam Dr. Öğr. Üyesi Mehmet DEMİRCİ'ye içtenlikle teşekkür ederim. Maddi ve manevi desteklerini hiçbir zaman esirgemeyen, beni hiçbir zaman yalnız bırakmayan aileme teşekkür ederim. Çalışmalarım boyunca manevi desteği, motivasyon desteği ve değerli katkıları ile her zaman yanımda olan Mesut UĞURLU'ya teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

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

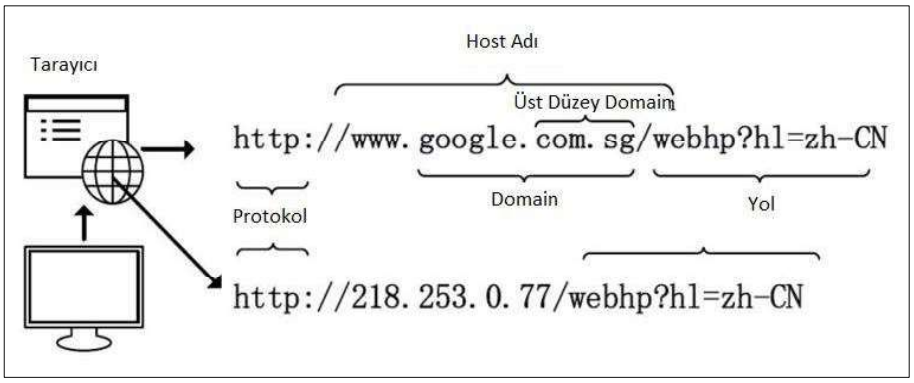
Kısaltmalar	Açıklamalar
CNN	Convolutional Neural Networks
CW	Confidence Weighted
DBN	Deep Belief Networks
DT	Decision Tree
ESA	Evrişimli Sinir Ağı
HTML	Hyper Text Markup Language
IP	Internet Protocol
KNN	K Nearest Neighborhood
MLP	Multilayer Perceptron
NB	Naive Bayes
PA	Passive Aggresive
RF	Random Forest
SVM	Support Vector Machine
URL	Uniform Resource Locator
YSA	Yapay Sinir Ağı

1. GİRİŞ

Yeni iletişim teknolojileri, çevrimiçi bankacılık, e-ticaret ve sosyal ağ gibi birçok uygulamanın büyümesini ve tanıtılmasını sağlamaktadır. Bu sebeple başarılı bir uygulamaya sahip olmak için çevrimiçi bir varlığa sahip olmak zorunlu hale gelmiştir. İnternetin tüm dünyaya yayılması yeni fırsatların yanında, interneti siber suçlar açısından bir platform haline getirmiştir. Bu sebeple güvenlik endişesi ön plana çıkmaktadır. Web, çeşitli kötü amaçlı etkinlikler için bir merkez olarak kullanılmaya başlanmıştır. Symantec'in 2016 İnternet Güvenlik Raporu [1], kurumsal veri ihlallerini, tarayıcılara ve web sitelerine yapılan saldırıları, ransomware ve diğer siber etkinlik türlerini içeren geniş ve kapsamlı küresel tehditleri ele almaktadır. Raporda ayrıca saldırganların kullanmış olduğu yöntemler ile ilgili püf noktalara da yer verilmektedir. Kullanıcıların kötü niyetli bir URL'yi tıklamaları ve bunun sonucunda sistemin tehlikeye atılması iyi bilinen ve şaşırtıcı derecede etkili bir yöntemdir.

Kötü niyetli URL'ler, siber saldırı amacı ile kullanılmaktadır. Bu tür URL'lerin ziyaretçileri, belirli saldırılara maruz kalma tehlikesi altındadır [2]. Kötü niyetli kullanıcılar, kötü niyetli URL'yi yaymak için e-posta veya başka yöntemler kullanmaktadır. Bu kötü niyetli URL'ler kişilerin gizlilik bilgilerini sızdırarak veya bilgisayarlarını botnet'lerin bileşenleri haline getirerek büyük zararlar verebilirler [3].

Uniform Resource Loader (URL)'in Türkçe karşılığı Tekdüzen Kaynak Bulucu olarak bilinmektedir ve internette ki bir kaynağın yerini bulmak amacı ile kullanılmaktadır. URL, web kaynaklarına referans olarak kullanılmaktadır. URL'ler, bilgisayarların sunucularla iletişim kurabilmek için kullandıkları sayıları (IP adreslerini) değiştirir ve insanların okuyabilmesi için birer metin haline getirir. URL'ler, istemci programları tarafından standart bir şekilde ayrıştırılabilmektedir. URL bunun dışında, ilgili web sitesinin dosya yapısını da belirtmektedir. Bir URL, bir protokol, bir domain adı ve bir yoldan oluşur Yapısı protocol://domain-adi.ust-duzey-domain/yol şeklindedir. Protokol, bir tarayıcının kaynak hakkında bilgi alması gerektiğini göstermektedir. Web standardı http:// , https://, mailto:, ftp: gibi kullanımları da mevcuttur. Bir domain adı, bir kaynağın (web sitesi) bulunduğu konumun insanlar tarafından okunabilen kısmıdır. Şekil 1.1'de URL örneği gösterilmiştir.

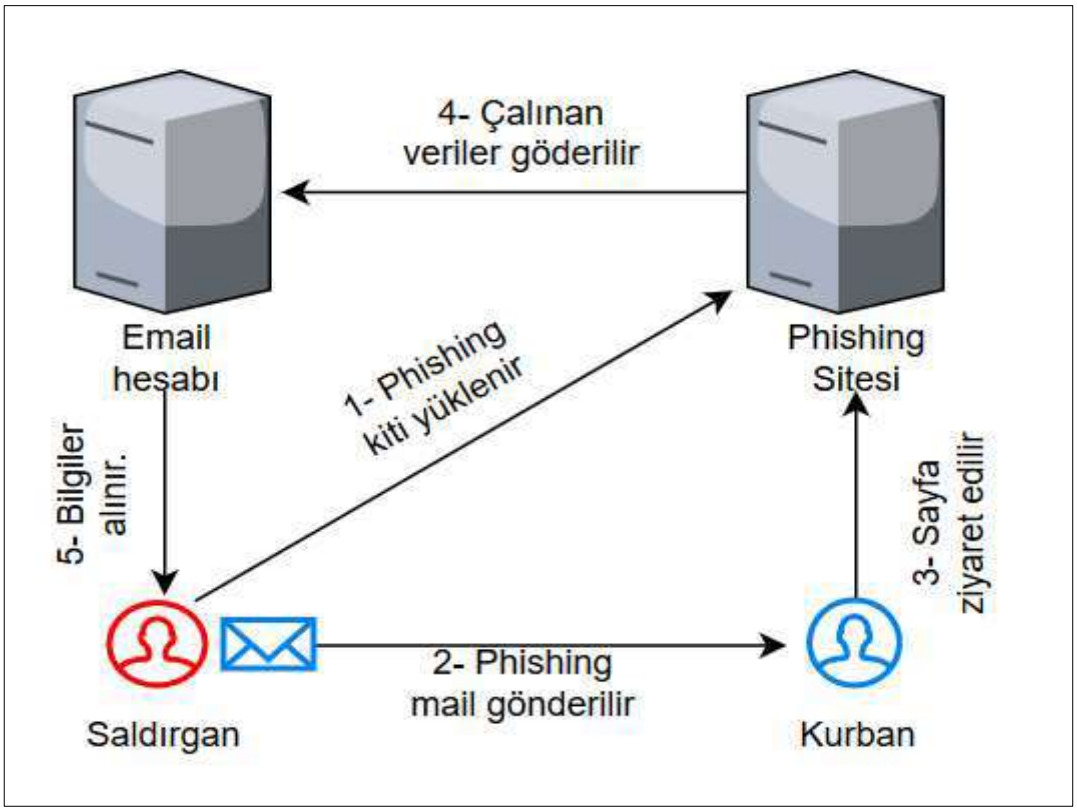


Şekil 1.1. URL örneği

URL'ler insan hayatını önemli derecede kolaylaştırmaktadır. Fakat insan hayatını kolaylaştıran bu adresleme yöntemi, siber saldırganlar tarafından kötü amaçlı kullanılmaktadır. En yaygın kullanılan saldırı tipleri phishing, spam, malware ve tahrif etme saldırılarıdır. Kötü niyetli URL'ler, phishing URL'leri, spam URL'leri, malware URL'leri ve tahrif URL'lerini içermektedir.

Phishing, internet dolandırıcılarının kullandığı ve orijinal bir web sayfasının kopyasını oluşturarak kişilerin bilgilerini çalmaya yönelik kullanılan bir aldatma tekniğidir [4]. Phishing, son zamanlarda internet kullanıcılarının güvenliğini risk altına sokan en popüler saldırı türlerinden biridir. Bu tür bir saldırıda kullanılan web siteleri kolayca çoğaltılır ve saldırı sırasında kullanılan sosyal mühendislik yöntemleri tespit edilmesini zorlaştırır. Phishing araçları internet ortamından kolay bir şekilde indirilebilmektedir. Bu durumun sonucu olarak, internette dolaşan herkes bu araçlara rahat bir şekilde ulaşabildiği için kendi phishing saldırılarını başlatabilmektedir. Phishing siteleri yasal orijinallerine çok benzeyip insanları aldatmaya yönelik amaçlarla oluşturulmaktadır. Bir phishing saldırganı bir web sitesinin HTML kodlarını kopyalayarak oluşturduğu sahte bir web sitesi yardımıyla sosyal mühendislik tekniklerini de kullanarak hedeflediği kişinin kişisel bilgilerini ele geçirmeye çalışır. Phishing web sayfası çok kısa bir sürede hazırlanabilmesi ve bunların önlenmesi için geliştirilen tekniklerin yetersiz olması bu saldırıların gün geçtikçe artmasına sebep olmuştur. Anti Phishing Working Group raporlarına göre 2004 yılının Mart ayında 402 phishing saldırısı belirlenmişken, 2016 yılının Kasım ayında bu sayı 118,928 olmuştur. 2016 yılında toplam saldırı sayısı ise 1,220,523 olarak belirlenmiş olup, 2016 yılında saldırılar bir önceki yıla göre %65 düzeyinde artış göstermiştir. Phishing saldırılarını başlatmak için sosyal mühendislik ve teknik bilgiler genellikle birlikte kullanılmaktadır. Bir phishing saldırısı,

hedef kişilere güvenilir kaynaktan görünen bir e-posta göndererek bu kişilerin gelen e-postadaki URL'i tıklamasıyla başlamaktadır. Phishing saldırılarının tespiti için yapılması gereken ilk işlem, internet kullanıcılarının güvenli internet kullanımı konusunda bilinçlendirilmesi ve farkındalık düzeylerinin arttırılmasını sağlamaktır. Fakat dünya üzerinde siber güvenlik bilgisi olamayan milyarlarca insan internete bağlanarak bilgi paylaşımı, internet bankacılığı ve e-ticaret gibi uygulamaları kullanmaktadır. Phishing saldırılarının kullanıcılara gelmeden tespit edilebilmesi için mail sunucularda filtreler veya yapay zeka çözümleri kullanılmaktadır. Şekil 1.2'de bir phishing saldırısına ait akış şeması gösterilmiştir.

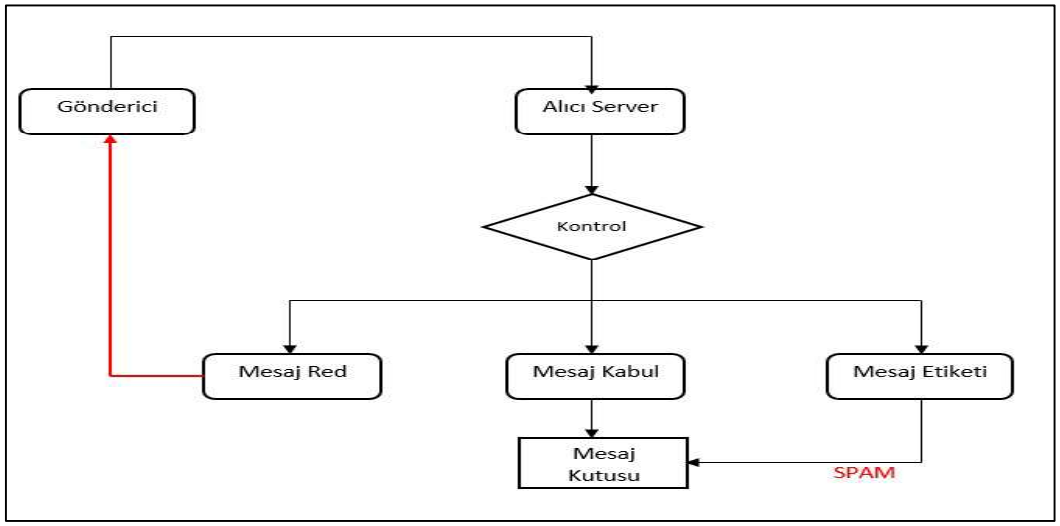


Şekil 1.2. Phishing saldırı akış şeması

İnternet üzerinde aynı mesajın yüksek sayıdaki kopyasının, kişilerin bilgisi olmadan zorlayıcı nitelikte mail olarak gönderilmesi genelde spam olarak adlandırılmaktadır [5]. Spam, istenmeyen mesajların toplu olarak gönderilmesi için dijital mesajlaşma sistemlerinin kötü niyetli kullanılmasıdır. Spam mesajlar reklam amaçlı kullanılabileceği gibi siber saldırganlar tarafından bilgisayar korsanlığı ve kimlik avı gibi amaçlarla kullanılabilir. Spam mail içerisine kötücül yazılımlar yerleştirilebilmektedir veya sahte zararlı sitelere

yönlendirme işlemi yapılabilmektedir. Bu yüzden spam veri güvenliği için büyük zaafiyetler taşımaktadır. Günümüzde spam tespiti için genellikle filtreler kullanılmaktadır fakat bu filtreler gerçek anlamda yeterli değildir. Çünkü spam yaratıcıları bu filtreleri kolaylıkla aşabilmektedir. Filtreleri aşan spamlari tespit etmek için kullanıcıların bilgili ve dikkatli olması gerekmektedir. Fakat dünya üzerinde siber güvenlik bilgisi olmayan milyarlarca insan internete bağlı durumdadır. Bu durum siber saldırganları teşvik etmektedir.

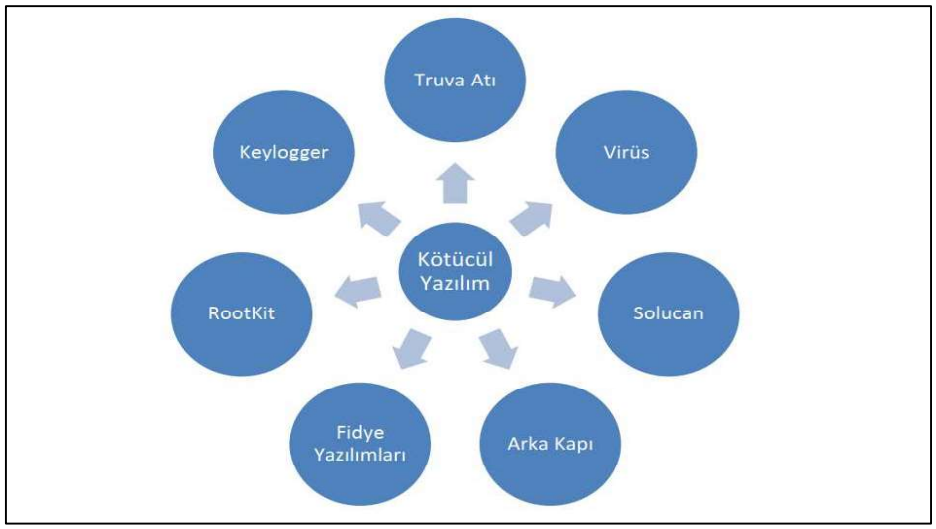
Bir spam mail gönderildiğinde ilk olarak siber saldırganın mail sunucusundan potansiyel kurbanın mail sunucusuna gelmekte ve eğer engellenmezse kullanıcıya mail olarak düşmektedir. Sunucu içerisinde bulunan filtreler mail içerik farklılıklarını kontrol ettikleri için yetersiz kalmaktadırlar. Sunucu üzerinde derin öğrenme yaklaşımı kullanılarak URL bazlı spam tespiti gerçekleştirilebilir. Mail akış diagramı Şekil 1.3'te gösterilmiştir.



Şekil 1.3. Spam mail akış diyagramı

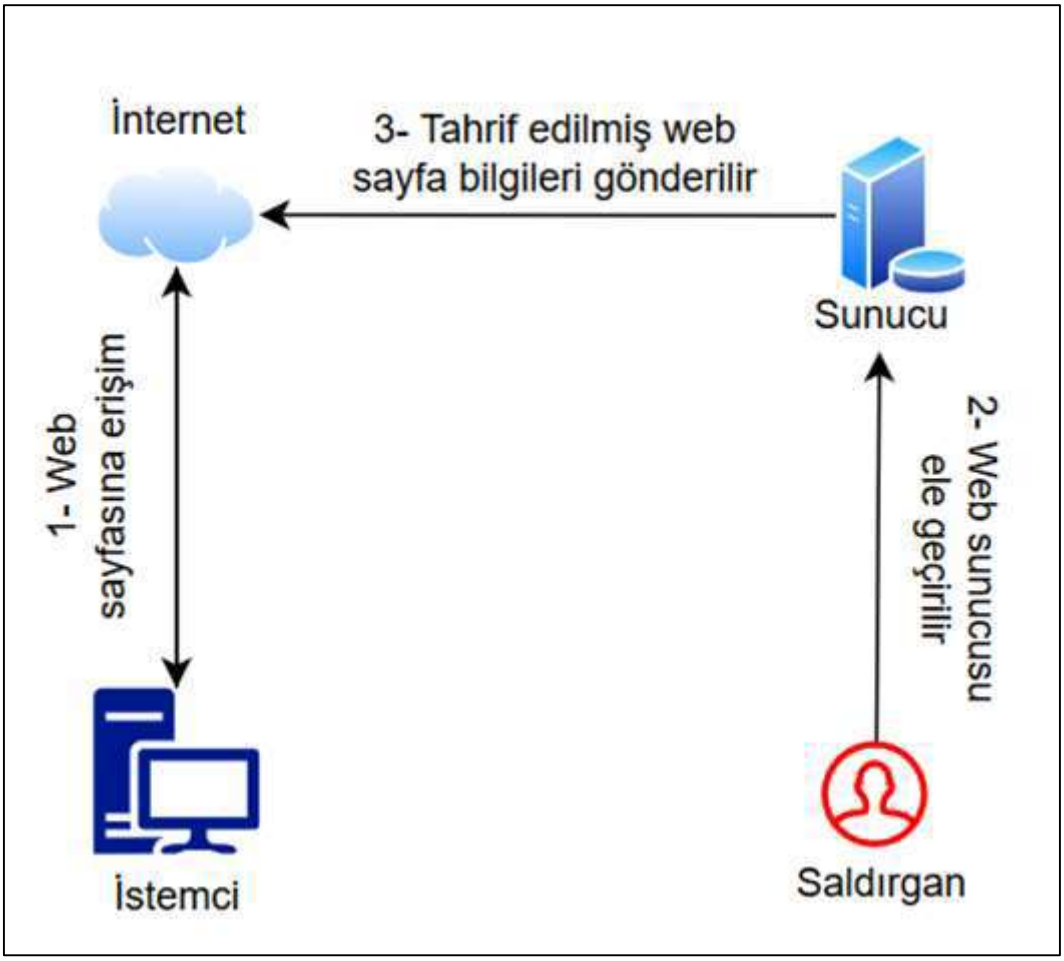
Malware kötü amaçlı yazılımlara verilen genel bir isimdir. İngilizce Malicious software (kötü amaçlı yazılım) ifadesinin kısaltmasıdır. Bilgisayar teknoljisiin gelişmesiyle birlikte bilgisayar güvenliği konusunda başta gelen tehditlerden biridir. Kötücül yazılım bulaştığı bir bilgisayar sistemine zarar vermek amacıyla hazırlanmış yazılımların adıdır. Truva atı, virüs, solucan, arka kapı, casus yazılım örnek olarak verilebilir. Bu yazılımlar her dilde yazılabilmektedir. Truva atı yazılımları zararsız bir kodun içine gizlenerek sisteme sızan yazılımlardır. Solucan yazılımlar ağ üzerinde kendi kendine dağılarak çoğalabilen yazılımlardır. Keylogger yazılımları kullanıcıların klavye hareketlerini kaydederek kullanıcı

bilgilerini çalan yazılımlardır. Fidye yazılımları, bilgisayara yüklendikten sonra tüm sistemi şifreleyen ve sadece ücret ödendiğinde saldırgan tarafından deşifreleme yapan bir yazılımdır. Kötücül yazılımlar ve çeşitleri her geçen gün artmakla birlikte kullanıcılar için büyük bir tehtit oluşturmaktadır.



Şekil 1.4. Örnek malware türleri

Defacement, bir web sayfasının görsel görünümünü değiştiren bir saldırı çeşitidir. Çoğu zaman defacement zararsızdır, ancak kötü amaçlı yazılım yükleme gibi daha kötü eylemleri gizlemek için bazen dikkat dağıtıcı olarak kullanılabilir. Doğru konfigüre edilmemiş veya güvenlik önlemleri alınmamış web sayfaları ele geçirilerek içerik bilgileri değiştirilebilir. Siber saldırganlar hem marka değeri anlamında kuruma zarar verebileceği gibi müşterileri de yanlış bilgi paylaşımı veya zararlı yazılım indirme işlemi gerçekleştirerek zarar verebilirler. 2016 Şubat ayında saldırganlar Linux Mint sayfasını ele geçirmiş ve indirilmek üzere hazırlanmış ISO dosyasını, arka kapı içeren başka bir ISO dosyası ile değiştirmişlerdir. Bu sayede Linux Mint resmi sayfasından zararlı yazılım dağıtım işlemi gerçekleştirilmiştir. Yapılan saldırıda Linux Mint marka değeri kaybettiği gibi, ISO dosyasını indirip kurulum gerçekleştiren insanların bilgileri çalınmıştır. Bunun haricinde defacement saldırıları terör, siyasi, askeri amaçlar için gerçekleştirilebilmektedir. Şekil 1.5'te bir defacement saldırısına ait akış şeması gösterilmiştir.



Şekil 1.5. Defacement saldırı akış diyagramı

İnternetten verilen hizmetler arttıkça web uygulamalarına yönelik saldırılar ve çeşitleri de her geçen gün artmaktadır. İnternetin yaygınlaşması ile beraber güvenlik riskleri de aynı ölçüde artmıştır[6]. Saldırıların çeşitliliği, potansiyel olarak yeni saldırı tipleri ve bu saldırıların ortaya çıkabileceği sayısız bağlam göz önüne alındığında, siber güvenlik ihlallerini tespit etmek için gelişmiş sistemler tasarlamak zordur. Bu durum geleneksel güvenlik yönetimi teknolojilerinin sınırlamaları, yeni güvenlik tehditlerinin her geçen gün büyümesi, yeni bilgi teknolojilerinin hızlı değişmesi gibi nedenlerle giderek ciddi bir hal almaktadır. Bu saldırı tekniklerinin çoğu, ele geçirilmiş URL'lerin yayılması yoluyla gerçekleştirilmektedir. Kaspersky Lab'ın web antivirüs yazılımı, zararlı URL'lerin tüm çevrimiçi saldırılarının % 75,76'sını oluşturduğunu tespit etmiştir [7].

Kötü niyetli URL'ler, siber suçları işleyen başlıca mekanizmalardan biri olması sebebiyle her yıl milyarlarca dolar değerinde zarara neden olmuştur [8]. Bu nedenle, kötü niyetli URL'leri zamanında tespit etmek için gelişmiş teknikler tasarlamak zorunlu hale gelmiştir.

Kötü niyetli URL'lerin tespiti ve tehdit türlerinin belirlenmesi, web kullanıcılarının kötü niyetli URL'leri ziyaret etmesini önleyerek web tabanlı saldırıları azaltmaktadır. Kötü niyetli URL'leri saptamanın en yaygın yöntemi kara liste yöntemidir. Google Güvenli Tarama hizmeti ve Microsoft Akıllı Ekran hizmeti istemci tarayıcılarına URL kara listelerini sunmaktadır. Kara liste, kullanıcıları zararlı web sitelerinden korumak için genel bir çözümdür; örnekler arasında PhishTank, SORBS ve URIBL bulunmaktadır. Kara listeler, aslında geçmişte zararlı olduğu doğrulanmış bir URL veritabanıdır. Bu veritabanı zaman içerisinde bir URL'nin kötü niyetli olduğu bilindiğinde derlenmektedir. Böyle bir teknik, basit bir sorgulama ile kullanılabilir. Bu sebeple de oldukça hızlı ve uygulanması çok kolaydır. Yanlış bir pozitif içermez. Ancak özellikle her gün yeni URL'ler oluşturulduğundan, ayrıntılı bir kötü niyetli URL listesi tutmak neredeyse imkânsızdır. Bu teknik, yalnızca listeler zamanında güncelleştirildiğinde ve kötü niyetli web sayfalarını bulmak için web siteleri kapsamlı bir şekilde ziyaret edildiğinde etkilidir. Ancak çevrimiçi kullanıcıların zamanında korunmasını sağlamada yetersiz kalmaktadır. Kara listeler (IP adresleri ve URL bilgileri) pahalı ve bazen de karmaşık filtreleme teknolojilerinden çıkarıldığından, şirketler güncellenmiş listelerini ücretsiz olarak piyasaya satmamaktadır. Ayrıca, web sayfalarına ve saldırganlara uygulanan gizlilik nedeniyle, kötü niyetli web sayfalarının URL'linin ve IP'lerinin sürekli değiştirilmesi, yeni web sayfalarının kara listelerde kontrol edilme olasılığını düşürmektedir. Bunun yanında, güvenilir web siteleri, kötü niyetli gizli URL'leri içerisinde barındırabilir. Bu URL'lerin tespiti zordur. Saldırganlar kara listeden kaçmak ve meşru görünen URL'yi değiştirerek kullanıcıları kandırmak için yaratıcı teknikleri kullanmaktadır. URL'ler meşru göründüğünde ve kullanıcı onları ziyaret ettiğinde bir saldırı başlatılabilir. Genellikle saldırganlar, imza temelli araçların kendilerini tespit etmesini önlemek için kodu gizlemeye çalışırlar. Aynı zamanda, saldırganlar çoğu zaman eşzamanlı olarak birden fazla saldırı başlatabilir. Bu durum saldırı imzasını değiştirir ve belirli imzalara odaklanan araçlar tarafından tespit edilemez hale getirir.

Kara liste modelinin en büyük sorunu, reaktif olmasıdır. Kullanıcıları daha iyi korumak için proaktif bir model olmalıdır. Bu kötü niyetli URL'lerin tespiti, gerçek zamanlı olarak gerçekleştirildiğinde, yeni URL'leri yüksek doğrulukta tespit ettiğinde ve özel saldırı türlerini özel olarak tanıdıklarında etkilidir.

İçerik tabanlı analiz hizmeti, kötü niyetli web sitelerini tespit etmek için iyi bilinen bir diğer çözümdür. Bu hizmet web sayfasının içeriğini indirmekte ve zararlı içerik için analiz

etmektedir. Bu yaklaşım kara listeye göre daha yüksek doğruluk sağlayabilir, ancak genel olarak çok daha fazla çalışma süresi gerektirir. Analiz için içeriğin indirilmesi zaman almakta ve bant genişliği tüketmektedir. Bununla birlikte, içerik tabanlı analiz yöntemleri, çok sayıda URL için ve yeni URL'lerin yaratılma hızı için pratik bir çözüm değildir.

Problem, mevcut yöntemlerin, kötü niyetli URL'leri tespit edip sınıflandırmada tamamen etkili çözümlerinin olmaması, reaktif olmaları ve bu sebeple çevrimiçi kullanıcıların zamanında korunmasını sağlamada yetersiz kalmalarıdır. Yapılan çalışma ile kötü niyetli URL'leri proaktif bir şekilde saldırı türlerine göre tespit edip sınıflandırarak internet kullanıcılarını bilgilendirmek ve onları çevrimiçi saldırılardan korumak amaçlanmaktadır.

Bu sorunların üstesinden gelmek için, hesaplama, makine öğrenme algoritmaları, çevrimiçi öğrenme teknikleri ve anomali tabanlı algılama motorunu kullanarak otomatikleştirilmiş zararlı URL tespiti ile ilgili çalışmalar yapılmıştır. Makine öğrenmesi, eğitim verisi olarak bir URL kümesi kullanmakta ve istatistiksel özelliklere dayanarak, bir URL'yi kötü niyetli veya iyi niyetli olarak sınıflandırmaktadır. Bu yöntem, kara liste yöntemlerinden farklı olarak yeni URL gelmesi durumunda genelleme yapma yeteneği sağlamaktadır. Bir makine öğrenme modelinin eğitimi için birincil gereksinim, eğitim verilerinin varlığıdır. Eğitim verileri toplandıktan sonra, bir sonraki adım URL'yi yeterince tanımlayacak şekilde bilgilendirici özellikler çıkarmak ve aynı zamanda, makine öğrenme modelleri ile matematiksel olarak yorumlayabilmektir. Tüm özellikler eğitim için verildiğinde düşük performans göstermektedir ve bu durum yüksek zaman ve karmaşıklığına neden olmaktadır. Bunun için özellik çıkarma teknikleri veya özellik seçimi teknikleri kullanılabilir. Özellik çıkarma, orijinal özellik alanını yeni bir özellik alanına dönüştürmektedir. URL'lerin özellik gösteriminin kalitesi, makine öğrenmesi tarafından öğrenilen kötü niyetli URL tahmin modelinin kalitesi için kritik öneme sahiptir. Özellikler çıkarıldıktan sonra bu özelliklerin, model eğitimi için uygun bir formatta (örneğin, sayısal bir vektör) işlenmesi gerekmektedir. Eğitim verileri üzerinde doğrudan kullanılabilecek birçok sınıflandırma algoritması mevcuttur (Naive Bayes, Destek Vektör Makinesi, Lojistik Regresyon, vb.). Ancak, URL verilerinin eğitimi zorlaştırabilecek bazı özellikleri vardır (ölçeklenebilirlik ve uygun kavramı öğrenme açısından). Örneğin, eğitim için uygun olan URL sayısı çok sayıda olabilir. Bu sebeple de geleneksel modellerin eğitim süresi yüksek olabilir. Ölçeklenebilir öğrenme teknikleri ailesi olan Çevrimiçi Öğrenme [9] bu görev için yoğun bir şekilde uygulanmıştır.

Sınıflandırma için özellik seçimlerinde sözlüksel özellikler, host tabanlı özellikler, içerik özellikleri ve hatta bağlam ve popülerlik özellikleri gibi çeşitli özellikler göz önünde bulundurulmuştur [10]. Bununla birlikte, en sık kullanılan özellikler, iyi performans gösterdikleri ve elde edilmeleri nispeten kolay olduğu için sözcüksel özelliklerdir [11, 12]. Bunlar, URL'nin uzunluğu, nokta sayısı vb. gibi istatistiksel özellikleri içermektedir. Aynı zamanda, kelime torbası gibi özellikler de sıklıkla kullanılmaktadır. Kelime torbası, belirli bir kelimenin veya dizenin URL'de görünüp görünmediğini göstermektedir. Bunun sonucu olarak, eğitim veri setindeki benzersiz her bir kelime özellik haline gelmektedir. Ancak eğitimli modeller URL ile ilgili faydalı bilgileri bu kelimelerden çıkaramamaktadır. Ayrıca, URL'lerdeki benzersiz kelimelerin sayısı, eğitim modellerinde ciddi bellek kısıtlamalarına neden olacak şekilde çok büyük olabilir.

Son zamanlarda, derin öğrenme olarak da adlandırılan derin yapay sinir ağları önem kazanmıştır. Derin öğrenmenin son zamanlarda önem kazanmasının çeşitli nedenleri vardır [13]. Bunlardan biri işlem kabiliyetlerinde keskin bir artış olmasıdır. Bir diğeri ise derin öğrenme teknikleri, öznitelikteki verilerden özellik gösterimlerini öğrenen bir özellik çıkarıcı veya sınıflandırıcı olabilmektedir [14]. Bu sebeple yukarıdaki sorunları ele almak ve kötü niyetli URL tespiti için Derin Öğrenme tabanlı bir çözüm önerilmiştir. Çalışmada URL'yi iyi huylu ve kötü niyetli olarak sınıflandırmanın yanı sıra, kötü niyetli URL'lerin saldırı türlerinin sınıflandırılması için çoklu etiket sınıflandırması yapılmıştır. İkili sınıflandırma ve çoklu sınıflandırma için Evrişimsel Sinir Ağı (ESA) kullanılmıştır. Sık kullanılan özellikler, iyi performans göstermeleri ve elde edilmeleri nispeten kolay olması sebebiyle sözcüksel özellikler kullanılmıştır. Veri kümesinde toplamda 79 özellik ve 1 sınıf etiketi bulunmaktadır. İlk olarak modele hem ikili sınıflandırma hem de çoklu sınıflandırma için 79 özelliğin hepsi verilmiştir. Saldırı tiplerinden spam, malware ve defacement için doğruluk oranı %99 üzerinde çıkmıştır. Fakat saldırı tiplerinden olan phishing saldırısının doğruluk oranı %50 oranında çıkmıştır. Çoklu sınıflandırmanın doğruluk oranı ise %80'dir. Ortaya çıkan bu problemin çözülmesi için yeni bir önişleme ihtiyacı duyulmuştur. Bu kapsamda random forest algoritması kullanılarak veri kümesi içerisindeki özellikler önem derecesine göre sıralandırılmıştır. Çalışmada en iyi performansı bulmak için ilk on, yirmi, otuz, kırk, elli, atmış ve yetmiş özellik ayrı ayrı seçilerek model eğitilmiş ve test edilmiştir. Yapılan testlerde 50 özellikten sonra sistemde kararsızlık olduğu ve doğruluk oranının %80'i geçemediği tespit edilmiştir. Bu kapsamda veri kümesi içerisindeki hangi özelliğin sistemi kararsızlık durumuna geçirdiğinin tespit edilebilmesi için 40 – 50 arasındaki özellikler sırası

ile test edilmiştir. Test sonucunda 44. özellik olan argPathRatio özelliğinin phishing için ikili sınıflandırmada ve çoklu sınıflandırmada sistem başarısını düşürdüğü görülmüştür. Yapılan tüm testlerde en iyi performans oranları ilk yirmi özellik seçilerek elde edilmiştir. İlk yirmi özelliğin ağrılık oranları 0,01'den büyüktür. Random-forest algoritması ile özellik seçimi yapılması sonucunda sistem performansı artırılmıştır. Eğitim ve test kümeleri, 0,2 oranında bölünmüştür. Aynı veri kümesinin 10 farklı kopyası alınarak model çalıştırılmıştır ve elde edilen sonuçların ortalaması alınarak doğruluk, kesinlik ve hassasiyet değeri hesaplanmıştır. 20 özellik seçilerek yapılan phishing test sonucunda doğruluk oranı % 99,93, kesinlik % 99,62 ve hassasiyet % 99,97 olarak bulunmuştur. Bu sayede phishing ikili sınıflandırma doğruluk oranı % 50,32 'den % 99,93'e yükseltilmiştir. Çoklu sınıflandırma doğruluk oranı % 80,00'dan % 99,76'e yükseltilmiştir.

Tez çalışması beş bölümden oluşmuştur. Tezin ikinci bölümünde konuyla ilgili olarak literatürde yapılmış olan çalışmalardan bahsedilmiş üçüncü bölümde ise teorik bilgi verilmiştir. Dördüncü bölümde önerilen çalışmadan bahsedilmiş ve geliştirilen uygulamaların sonucunda elde edilen çıktılar karşılaştırılmıştır. Son bölüm olan beşinci bölümde ise kötü niyetli URL tespitinin öneminden bahsedilmiştir. Aynı zamanda makine öğrenmesi ve önerilen derin öğrenme yönteminin neden bu alanda kullanıldığı açıklanmıştır. Yapılacak sonraki çalışmalar için öneride bulunulmuştur.

2. LİTERATÜR ÇALIŞMASI

URL sınıflandırılmasında sözcüksel özelliklerin kullanılması yüksek doğruluk elde edilmesini sağlamaktadır. Bunlar, URL'nin uzunluğu, nokta sayısı vb. gibi istatistiksel özellikleri içermektedir. Sözcüksel özellikler kullanılarak yapılan çalışmalar ile ilgili aşağıda bilgi verilmiştir.

Ma ve arkadaşları [12], yapmış oldukları çalışmada hem sözcüksel özellikleri hem de host tabanlı özellikleri kullanarak denetimli öğrenme gerçekleştirmiştir. URL'ler otomatik olarak hem sözcüksel hem de host tabanlı özelliklerinden faydalanarak denetimli öğrenme ile zararlı veya iyi huylu olarak sınıflandırılmıştır. Çalışma farklı kaynaklardan toplanan 20,000 ila 30,000 URL ile yapılmıştır. İyi huylu URL'ler için iki veri kaynağı (Yahoo ve DMOZ) kullanılmıştır. Kötü niyetli URL'ler için de iki kaynaktan (PhishTank ve Spamscatter) faydalanılmıştır. Çalışmada naive bayes, svm ve lojitik regresyon kullanılmıştır. Sonuçta ortaya çıkan sınıflandırıcılar % 95–%99 oranında doğruluk elde etmiştir.

Ma ve arkadaşları [15], phishing web sitelerindeki alan adlarının davranışlarını temel alarak şüpheli URL'leri tespit etmek üzerine başka bir çalışma yapmıştır. Bu çalışmayı yaparken sözcüksel özellikleri ve kara listeye alınmış host tabanlı özellikleri kullanmışlardır. Zararlı Web sitelerini tespit etmek için çevrimiçi öğrenme yaklaşımları incelenmiştir. Perceptron, lojistik regresyon, passive aggressive, confidence weighted algoritmaları kullanılmıştır. Yakın zamanda geliştirilen CW gibi çevrimiçi algoritmaların % 99'a kadar sınıflandırma doğruluğunu sağlayabilen son derece hassas sınıflandırıcılar olabileceği bu çalışma ile gösterilmiştir.

Choi ve arkadaşları [16], kötü niyetli URL'leri tespit etmek ve spam, phishing ve malware gibi saldırı türlerini tanımlamak için bir makine öğrenme yöntemi sunmuştur. Çalışmaları, metin özellikleri, bağlantı yapıları, web sayfası içeriği, DNS bilgileri ve ağ trafiği dâhil olmak üzere çeşitli ayırt edici özellikleri kullanarak birden fazla türde kötü niyetli URL türü içermektedir. 40,000 iyi huylu URL ve gerçek hayattaki internet kaynaklarından elde edilen 32,000 kötü niyetli URL ile yapılan deneysel çalışma sonucunda, kötü niyetli URL'leri tespit etmedeki doğruluk % 98'in üzerinde ve saldırı türlerini belirlemedeki doğruluk % 93'ün

üzerinde çıkmıştır. Kötü niyetli URL'leri tespit etmek için SVM saldırı tiplerini tanımlamak için KNN algoritması kullanılmıştır.

Lin ve arkadaşlarının yapmış olduğu çalışmada, sözcüksel özellikleri tamamlamak için URL'nin tanımlayıcı özellikleri kullanılmıştır [17]. Yazarlar, kötü niyetli URL'leri sınıflandırmak için URL dizisinin sözcüksel bilgisiyle statik özellikleri birleştirmiştir. Çalışma sonucunda sunucu ve içerik bazlı analiz olmadan, beş dakikada iki milyon URL ile başa çıkılmıştır ve önerilen yöntemleri kötü niyetli durumların sadece % 9'unu kaçırdığı gösterilmiştir.

Chu ve arkadaşları [18] bilinen korumalı web sitelerinde makine öğrenmesine dayalı phishing tespiti üzerine çalışmıştır. Sadece sözcüksel ve alan adı özellikleri temel alınarak yapılan çalışmada sınıflandırma sonucundaki doğruluk %91'in üzerinde çıkmıştır. Öğrenme algoritması olarak SVM kullanılmıştır. Çalışmada, 17,423 adet phishing url ve 28,722 adet iyi huylu URL kullanılmıştır.

Darling ve arkadaşları [21] yapmış oldukları çalışmada, yalnızca sözcüksel analizi kullanarak kötü amaçlı web sayfalarını sınıflandırmak için bir yaklaşım önermiştir. Çalışmanın amacı tamamen sözcüksel bir yaklaşımla sınıflandırma doğruluğunu incelemektir. Ayrıca, gerçek zamanlı bir sistemde kullanılabilecek ölçeklenebilir bir yaklaşım geliştirmeyi de hedeflemişlerdir. Çalışma, kötü amaçlı web sayfalarının URL'lerini % 99,1'lik bir oranla sınıflandırmıştır. Phishing verileri, Phishtank ve OpenPhish üzerinden toplanmıştır. Malware verileri ise MalwareDomainlist ve MalwareDomains olmak üzere iki farklı kaynaktan toplanmıştır. İyi huylu URL veri kümesi ise DMOZ ve Alexa'dan toplanmıştır. Algoritma olarak naive bayes, bayesian lojistik regresyon, lojistik regresyon, KNN ve J48 kullanılmıştır.

M. Mamun ve arkadaşları [19] tarafından kötü niyetli URL'lerin saldırı türlerine göre tespit edilmesi ve sınıflandırılmasına yönelik bir çalışma yapılmıştır. Çalışma farklı kaynaklardan alınan yaklaşık 110,000 URL üzerinde değerlendirilmiştir ve saldırı tipinin URL'lerini tespit etmede yaklaşık % 99, çok sınıflı sınıflandırıcı ile yapılan saldırıların tespitinde yaklaşık % 92-96'luk bir tahmin doğruluğu elde edilmiştir.

Vanhoenshoven ve arkadaşları [20] yapmış oldukları çalışmada, ikili sınıflandırma problemi olarak kötü niyetli URL'lerin tespitini ele alınmıştır. Algoritma olarak naive bayes, SVM, MLP, DT, RF ve KNN gibi bilinen birkaç sınıflandırıcı kullanılmış ve performansları karşılaştırılmıştır. Sonuçta, RF ve Çok MLP %97 olacak şekilde en yüksek doğruluğu elde etmiştir. 2,4 milyon URL (örnek) ve 3,2 milyon özellikten oluşan bir genel veri kümesi kullanılmıştır.

Selvaganapathy ve arkadaşları [13], kötü niyetli URL'lerin tespiti ve sınıflandırılması için kullanılan ikili ve çok sınıflı şemalar ile ilgili bir çalışma sunmaktadır. Bu makale, özellik seçimi ve sınıflandırma için derin öğrenme tekniklerinin kullanılmasını önermektedir. Önerilen metodolojide süreç veri kümesi toplama, sınıflandırma modelinin veri ön işleme ve inşası, kötü niyetli URL tespit süreci ve saldırı tiplerinin tanımlanmasını şeklindedir. Çeşitli kaynaklardan URL toplanarak, iyi huylu ve zararlı URL'lerden oluşan entegre bir veri kümesi oluşturulmuştur. Spam, phishing, malware ve APT URL'leri dahil olmak üzere toplam 17,700 kötü amaçlı URL toplanmıştır. Spam veri kümesi UCI deposundan Spambase'den toplanmıştır. Phishing veri kümesi UCI deposundan Phishing'den toplanmıştır. Malware ve APT veri kümeleri de Malware Domain'den toplanmıştır. İyi huylu URL'ler DMOZ'dan alınmıştır. Entegre edilmiş veri kümesi, URL'nin kötü niyetli olup olmadığını belirlemek için zararlı URL'ler ve iyi huylu URL'ler olarak işaretlenmiştir. İkili sınıflandırma yapıldıktan sonra, kötü niyetli URL'ler belirli saldırı türlerine göre kategorize edilmiştir Bu entegre veri kümesi, önerilen metodolojiyi değerlendirmek için kullanılmıştır. Giriş veri setinin kalitesini artırmak için ön işleme yapılmıştır. Çalışmada YSA, SVM, naive bayes ve DBN ikili sınıflandırma için kullanılmıştır. Doğruluk %55 ve %75 arasındadır. DBN en iyi sonucu vermiştir. Çoklu sınıflandırma için BR, LP ve KNN algoritmaları kullanılmıştır. %70 - %90 arasında doğruluk çıkmıştır.

Tez kapsamında makine öğrenmesinde KNN, SVM, DT, ve RF olmak üzere toplamda 4 tane algoritma kullanılmıştır. İkili sınıflandırmada minimum % 98,16 ve maksimum % 99,83 oranlarında doğruluk elde edilmiştir. Çoklu sınıflandırmada ise %96,66 oranında doğruluk elde edilmiştir. Derin öğrenme algoritması olarak da ESA kullanılmıştır. Yapılan çalışma sonucunda ikili sınıflandırma için doğruluk % 99,96 oranında, çoklu sınıflandırma doğruluk oranı % 99,98 oranında elde edilmiştir. ESA ile daha başarılı sonuçların elde edildiği uygulama ile gösterilmiştir.

Çizelge 1.1. Literatür çalışmasının özeti

Kaynak	Kullanılan Algoritma	Veri Kaynağı	Sonuç
Ma ve arkadaşları [12]	• NB, • SVM, • Lojistik Regresyon	İyi Huylu URL'ler: • Yahoo • DMOZ Kötü Niyetli URL'ler: • PhishTank • Spamscatter	% 95–99 arasında doğruluk elde edilmiştir.
Ma ve arkadaşları [15]	• Perceptron, • Lojistik Regresyon, • PA, • CW	İyi Huylu URL'ler: • Yahoo Kötü Niyetli URL'ler: • Web Posta Sağlayıcı	%99 oranında başarı elde edilmiştir.
Choi ve arkadaşları [16]	• SVM, • KNN	İyi Huylu URL'ler: • DMOZ Spam URL: • jwSpamSpy Phishing URL: • PhishTank Malware URL: • DNS-BH	İki sınıflandırmadaki doğruluk % 98, saldırı türlerini belirlemedeki doğruluk % 93 şeklindedir.
Lin ve arkadaşları [17]	• PA, • CW	Veri kümesi, güvenlik şirketi tarafından sağlanmaktadır. Sunucu 1 ve Sunucu 2 olarak adlandırılan iki farklı sunucu aracılığıyla veriler toplanır.	Beş dakikada iki milyon URL ile başa çıkılmıştır ve önerilen yöntem ile kötü niyetli vakaların sadece % 9'u kaçırılmıştır.
Chu ve arkadaşları [18]	• SVM	İyi Huylu URL'ler: • Yahoo Phishing URL: • Taoba	Sınıflandırmadaki doğruluk %91'in üzerinde çıkmıştır.
Darling ve arkadaşları [21]	• NB, • Bayesian Lojistik Regresyon, • Lojistik Regresyon, • J48 • KNN	İyi Huylu URL'ler: • DMOZ • Alexa Phishing URL: • Phishtank • OpenPhish Malware URL: • MalwareDomains • MalwareDomainlist	Kötü amaçlı web sayfalarının sınıflandırma doğruluğu %99 oranında çıkmıştır.

Çizelge 1.1. (Devam) Literatür çalışmasının özeti

Kaynak	Kullanılan Algoritma	Veri Kaynağı	Sonuç
M.Mamun ve arkadaşları [19]	• J48, • KNN, • RF	İyi Huylu URL'ler: • Alexa Spam URL: • WEBSPAM-UK2007 Phishing URL: • OpenPhish Malware URL: • DNS-BH Defacement URL: • Zone-H	İkili sınıflandırmalarda başarı %97 ile %99 arasında çıkmıştır. Çoklu sınıflandırmalarda başarı %92 ve %96 arasında çıkmıştır.
Vanhoenshoven ve arkadaşları [20]	• NB, • SVM, • MLP, • DT, • RF, • KNN	2,4 milyon URL ve 3,2 milyon özellikten oluşan bir genel veri seti kullanılmıştır.	RF ve MLP %97 olacak şekilde en yüksek doğruluğu elde etmiştir.
Selvaganapathy ve arkadaşları [13]	İkili Sınıflandırma: • YSA, • SVM, • NB, • DBN Çoklu Sınıflandırma: • BR, • LP, • KNN	İyi Huylu URL'ler: • DMOZ Spam URL: • UCI deposu Spambase Phishing URL: • UCI deposu Phishing Malware ve APT URL: • MalwareDomainlist	İkili sınıflandırmda doğruluk %55 ve %75 arasında çıkmıştır. Çoklu sınıflandırmada ise %70 ve %90 arasında doğruluk çıkmıştır.

3. TEORİK BİLGİ

3.1. Yapay Zeka

Yapay zekâ bilgisayar biliminin bir alt dalıdır [22]. Yapay zekânın geçmişi 1950'lere dayanmaktadır. 1950 yılında, Mind dergisinde yayımlanan "Computing Machinery and Intelligence" makalesinde Alan Turing "Makineler düşünebilir mi?" sorusuyla bu konunun öncülerinden biri olmuştur [23]. Zamanla beynin çalışma mantığı sorgulanmaya başlanmıştır. Bilgisayarları, insan beyni gibi çalıştırabilmek amaçlanmış ve bilgisayarlara akıl yürütmeyi öğretmek denenmiştir. Yapay zekâ, günümüzde oldukça popüler ve gelişmekte olan bir alandır. Birçok probleme etkili çözümler üretmiştir.

Yapay zekâ, insan davranışlarının makine tarafından taklit edilmesini sağlamaktadır. İnsan beyni gibi çalışmak için programlanmakta ve insan beyninin, hesaplama, öğrenme, genelleme yapma ve uyarlayabilme yeteneği gibi birçok özelliğini kullanmaktadır [24].

Yapay zekâda karar verme veya tahmin oluşturma süreçleri makine öğrenmesi ile gerçekleşmektedir. Makine öğrenmesi, bir veri içindeki örüntüleri otomatik olarak tespit etme ve bu örüntülerin gelecek verilerini tahmin etme amaçlı veya belirsiz bir konuda karar verme mekanizması oluşturma amaçlı kullanılan yöntemlerin tamamı olarak tanımlanmaktadır [25]. Makine öğrenmesinin geçmişi 1980'lere dayanmaktadır. Günümüzde de önemli bir teknolojidir. Makine öğrenmesi, tahmin, sınıflandırma, kümeleme gibi karmaşık modelleri ve algoritmaları tasarlamak için kullanılan bir yöntemdir [24].

3.2. Makine Öğrenmesi

Günümüzde oldukça popüler olan otomotiv, eğlence, fen bilimleri, tıp ve pazarlama gibi pek çok alanda kullanılan makine öğrenmesi, yapay zekanın bir alt koludur. Makine öğrenmesi mevcut verilerden çıkarımlar yaparak, bu çıkarımlarla bilinmeyene dair tahminlerde bulunan, bilgisayarlara karmaşık örüntüleri algılatma ve veriye dayalı akılcı kararlar verebilme yeteneği kazandıran yöntem örneklemleridir [26]. Çok büyük miktardaki ve çok özellikli verileri elle işlemek mümkün olamayacağı için makine öğrenmesi ve modelleri geliştirilmiştir. Makine öğrenmesi örnek veriler veya geçmiş deneyimleri kullanarak

performans kriterini optimize eden bilgisayar programıdır [27]. Makine öğrenmesi ile bilgisayara daha önceki örneklerden edinilmiş tecrübelerin öğretilmesi sağlanmaktadır. Bu sebeple bu olay, tecrübelerden öğrenme olarak nitelendirilebilmektedir [28]. Bu konuda önerilmiş birçok yaklaşım ve algoritma mevcuttur. Bu yaklaşımların bir kısmı tahmin ve kestirim bir kısmı da sınıflandırma yapabilme yeteneğine sahiptir.

Makine öğrenmesi, çeşitli algoritmalar ve yöntemler ile veride bazı kalıpları aramaktadır. Öğrenme işlemini de bu kalıplara karşılık gelen etiketlere bakarak gerçekleştirmektedir. Öğrenme işleminden sonra öğrendiklerine benzer durumlarla karşılaştığında deneyimlerinden yararlanarak çıkarım yapabilen sistemler geliştirmektedir. Bu işlemi, çeşitli matematiksel ve istatistiksel yöntemlerin kullanıldığı birçok algoritma ile sağlamaktadır.

Makine öğrenmesinde çözülmek istenen bir problem karşısında takip edilmesi gereken belli başlı temel adımlar bulunmaktadır. Var olan problemin istenilen zamanda ve başarılı bir şekilde çözümleyebilmek için bu adımları uygulamak önemlidir. Bu süreçte uygulanması gereken adımlar aşağıdaki gibidir [29]:

- Problemin Tanımlanması
- Veri Analizi
- Verilerin Hazırlanması
- Modelin Kurulması
- Modelin Değerlendirilmesi
- Modelin Kullanılması

Literatürde makine öğrenmesi yöntemleri birçok farklı şekilde sınıflandırılmıştır. Temel olarak denetimli ve denetimsiz öğrenme olmak üzere 2'ye ayrılmaktadır. Ancak zamanla bu 2 temel sınıfa ek sınıflar ortaya çıkmıştır. Bunlara örnek olarak yarı denetimli ve pekiştirmeli öğrenme verilebilir. Temel olarak denetimsiz yöntemler veriyi anlamak ve sonraki aşamalar için fikir vermeyi amaçlamaktadır. Bu yöntemde sonuç tanımlanmamış olup belirsizlik mevcuttur. Denetimli yöntem ise denetimsiz yöntemin aksine daha çok sonuç odaklı bir yöntemdir. Hedefleri bellidir ve veriden bilgi ve sonuç çıkarmayı amaçlamaktadır [30].

3.2.1. Denetimli öğrenme

Denetimli makine öğrenmesinde sistem etiketli veriler ile eğitilmektedir. Eğitimden sonra sistemin öğrenmesi sağlanmaktadır. Problem, sınıflandırma problemi olarak ele alınmaktadır. Denetimli öğrenme için sınıfları belli ve etiketli eğitim veri setine ihtiyaç duyulmaktadır. Denetimli öğrenmede süreç, verilen örneklerden yola çıkılarak her bir sınıfa ait özellikler belirlenmekte, önceden belirlenen kriterler temel alınarak sınıflara atamalar yapılmakta ve saptanan özellikler, kullanılabilir anlamlı kurallara dönüştürülmektedir [31].

Denetimli öğrenme genel olarak sınıflandırma ve regresyona dayalı tahmin problemlerinde kullanılmaktadır. Karar ağaçları, destek vektör makineleri, en yakın k-komşu, yapay sinir ağları, genetik algoritmalar, bayes sınıflandırıcılar denetimli öğrenme yöntemlerine örnek olarak verilebilir.

3.2.2. Denetimsiz öğrenme

Denetimsiz makine öğrenmesi modelinde, sistem eğitilirken etiketsiz veri kullanılmaktadır. Elde edilecek sonuç için özel bir tanımlama ve kategori olmadığı için bu yöntemde belirsizlik mevcuttur. Denetimsiz öğrenmede amaç, veriler arasındaki dolaylı ilişkiyi keşfedip, diğer değişkenlerde meydana gelebilecek en büyük nedensel etkiye sahip değişkenlerin durum değişikliğini belirleyecek fonksiyonların geliştirilmesidir [32].

Denetimsiz yöntemler, daha çok veriyi anlamaya, tanımaya, keşfetmeye yönelik olarak kullanılmaktadır ve sonraki uygulanacak yöntemler için fikir vermeyi amaçlamaktadır [30]. Denetimsiz öğrenme algoritması kullanımıyla elde edilen çıktıları denetimli öğrenme algoritmalarında da kullanmak mümkündür.

Makine öğrenmesi, veri setindeki örneklerin çıkışları bilinmediği için tanıma veya sınıflandırma yapmayıp, kümeleme, genellikle olasılık, yoğunluk tahmini, öznitelikler arasındaki ilişkilerin bulunması ve boyut indirgeme yapabilmektedir [31]. Korelasyon analizi, faktör analizi, kümeleme analizi, kendini düzenleyen haritalar, k-ortalama denetimsiz öğrenmeye örnek olarak verilebilir.

3.2.3. Yarı denetimli öğrenme

Yarı denetimli öğrenme, eğitim için etiketlenmemiş verilerin kullanımını da sağlayan denetlenen öğrenme görevleri ve tekniklerinin bir sınıfıdır. Genellikle etiketlenmemiş veri sayısı etiketli verilerden daha çoktur. Yarı denetimli öğrenme, denetimsiz öğrenme ve denetimli öğrenme arasındadır. Birçok makine öğrenmesi araştırmacısı, etiketlenmemiş verilerin küçük bir miktarla etiketlenmiş verilerle birlikte kullanıldığında, öğrenme doğruluğunda önemli ölçüde iyileşme sağlayabildiğini bulmuştur.

3.2.4. Pekiştirmeli öğrenme

Sisteme başarması için bir hedef verilerek denetimli öğrenme ve dinamik programlama alanlarını birleştirerek hedefe deneme yanılma yoluyla nasıl ulaşılacağını öğrenen bir öğrenme türüdür [33]. Sistem yaptığı her eylem sonunda bir ödül veya ceza alır ve yaptığı eylemleri dikkate alarak bir sonraki eylemde ödül kazanmayı hedeflediği öğrenme sistemidir. Pekiştirmeli öğrenmenin amacı, sistemin kazanacağı ödülü maksimize etmektir. Denetimli öğrenmeden farklı olarak denetçi direk olarak her bir girdi ve eylem için istenilen çıktıyı göstermemektedir. Denetçi sistemin ürettiği çıktının doğru olup olmadığı hakkında bilgi vererek sisteme yardımcı olmaktadır. Böylece sistemin en doğru çıktıları almasına yardımcı olmaya çalışılmaktadır.

3.3. Yapay Sinir Ağı

Yapay sinir ağları, insan beyninin öğrenme ve uygulama yapısını modellemeye çalışmaktadır. İnsan beyninin, öğrenme, eski bilgiye dayalı tahmin etme, eksik bilgiyi tamamlama gibi yeteneklerini makinelere kazandırmayı amaçlamaktadır. İnsandaki sinir hücresine benzediği için bu ismi almıştır. Yapay sinir ağları öğrenebilme yetenekleri sayesinde karmaşık problemleri çözebilmekte ve daha önce gördüğü örneklerden faydalanarak, hiç görmediği durumlar ile ilgili karar verebilmektedir. Öğrenme işlemi örnekler ile gerçekleştirilir [34] Ağ, ilgili olayın örnekleri ile eğitilerek genelleme yapabilmektedir. Örnekler ile eğitilen sistem, test değerleri ile test edilmekte ve sistemin doğruluk oranları belirlenmektedir.

Yapay sinir ağları, farklı zeki özellikleri bulundurmasından dolayı pek çok uygulamada kullanılmaktadır [35]. Günümüzde, tahmin, sınıflandırma, örüntü tanıma, veri ilişkilendirme, veri yorumlama ve veri filtreleme gibi işlemlerinde başarılı bir şekilde kullanılmaktadır.

Literatürde ortak yapay sinir ağı modeli bulunmamaktadır. Çünkü modeller problem çeşidine göre farklılık göstermektedir. Sistemin, giriş ve çıkışları arasındaki karmaşık ilişkiyi tanımlamada kullanılan matematiksel modellerdir. Yapay sinir ağı, yaygın kullanılan hesaplama yöntemlerinden farklı bir yöntem kullanmaktadır. Bulunduğu ortama uyum sağlayıp buna göre karar verebilen, eksik bilgi ile çalışabilen, belirsizliklerin olduğu durumlarda karar verebilen, hata toleransı yüksek olan bir hesaplama yöntemidir [36].

Yapay sinir ağında, veriler, girdi katmanından alınmaktadır. Sonrasında, gizli katmanda işlenmekte ve çıktı katmanına iletilmektedir. Gizli katmanda yapılan işlem, girdi verilerinin ağırlıklandırılmasıdır. Başarılı sonuçlar elde etmek için ağırlıklandırmanın doğru bir şekilde yapılması gerekmektedir. Çeşitli hesaplamalardan sonra doğru ağırlıkların bulunması, ağın eğitilmesi işlemidir. Yapay sinir ağlarında öğrenme işlemi ağırlıklara verilen değerler ile gerçekleştirilir. Ağırlıklandırmalar başlangıçta rastgele yapılmaktadır. Eğitim boyunca başarılı sonuçlar elde edilene kadar ağırlıklar sürekli olarak değiştirilmektedir. Ağırlıklar, öğrenme sürecinde kullanılan öğrenme algoritmaları ile değiştirilmektedir. Bu esnada test verileri ağa gönderilerek ve sonuçlarına bakılmaktadır. Sonuçlar başarılı ise ağın eğitimi tamamlanmıştır.

Yapay snir ağının temel görevi bilgisayarlarının öğrenmesini sağlamaktır. Olayları öğrendikleri için ağ kendisine gösterilen örneklerden genellemeler yaparak görmediği örnekler hakkında bilgiler üretebilmektedir. Ağın, örnekler ile kendisine gösterilen yeni durumlara adapte olması ve sürekli yeni olayları öğrenebilmesi mümkündür. Günümüzde endüstriyel uygulamalar, sağlık uygulamaları, finansal uygulamalar gibi birbirinden farklı pek çok alanda kullanılmaktadır. Yapay sinir ağının tarihçesi hakkında aşağıda bilgi verilmiştir [37].

- Walter Pitts ve Warren McCulloch tarafından 1943 yılında insan beyninin hesaplama yeteneğinden esinlenerek ilk yapay nöron modeli oluşturulmuştur.
- Donald Hebb tarafından 1949 yılında YSA'nın ağırlık değerlerini geliştiren Hebb

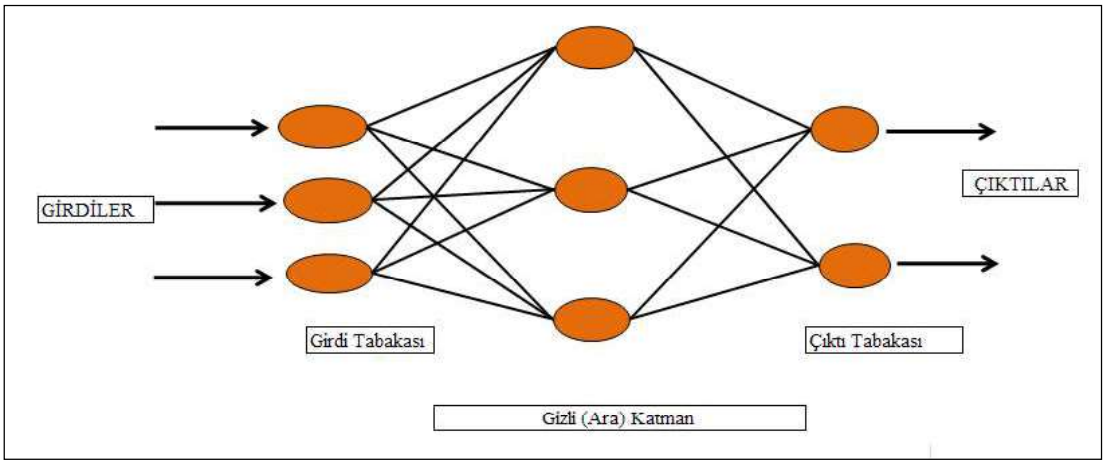
öğrenme kuralı geliştirilmiştir.

- Rosenblatt tarafından 1958 yılında ilk perceptron ve öğrenme metodu geliştirilmiştir.
- Widrow ve Hoff tarafından 1960 yılında Adaptive Linear Element (ADALINE) geliştirilmiştir. Widrow ve Hoff, türevlenebilir fonksiyonlarla gradient descent tabanlı öğrenme kuralını geliştirmiştir.
- Rosenblatt 1961 yılında backpropagation öğrenme şemasını önermiştir, ancak ağın eğitiminde başarılı olamamıştır.
- Minsky ve Papert 1969 yılında perceptron'un bazı basit mantıksal işlemlerde yetersiz olduğunu göstermiştir. Bir perceptron ile XOR probleminin çözülemediği gösterilmiştir. Çok katmanlı ağ ile bu tür problemlerin çözülebileceği düşünülmüş ancak nasıl eğitilebileceği konusunda çözüm bulunamamıştır.
- 1980'li yıllarda bu tür problemler çok katmanlı ve farklı ağ yapıları kullanılarak çözülmüştür.
- Günümüzde, YSA diğer yapay zekâ teknikleriyle birlikte kullanılarak çok daha etkin çözümler ortaya koymaktadır.

3.3.1. Yapay sinir ağının yapısı

Nöronlar birleşerek katmanları oluşturmaktadır. Katmanlar da birleşerek yapay sinir ağlarını oluşturmaktadır. Sinir hücreleri rastgele bir araya gelmemektedir. Nöronlar arasındaki bağlantılar ilk başlarda rastgele oluşturulmuştur ancak istenilen ve başarılı sonuçlar elde edilmemiştir. Zamanla girdi katmanı, gizli katman ve çıktı katmanı olmak üzere 3 katman halinde genelleşmiştir.

Girdi katmanı, dış dünyadan almış olduğu bilgileri gizli katmanlara iletmektedirler. Gizli Katman, girdi katmanından gelen bilgileri işlemekte ve sonuçları çıktı katmanına göndermektedir. Bilgilerin işlenmesi gizli katmanlarda gerçekleştirilmektedir. Bir ağ içinde girdi ve çıktı katmanı tek katmandan oluşurken, gizli katman birden fazla katmandan oluşabilir. Nöron sayıları da birden fazla olabilmekte ve bu nöronlar birbirleri ile bağlantılı olabilmektedir. Performans açısından gizli katmandaki nöron sayıları önemlidir. Sayı arttıkça ağın karmaşıklığı artmaktadır. Çıktı katmanında üretilmesi gereken çıktı üretilmektedir. Sonrasında üretilen çıktı dış dünyaya gönderilmektedir. Şekil 3.1'de yapay sinir ağının yapısı gösterilmiştir.

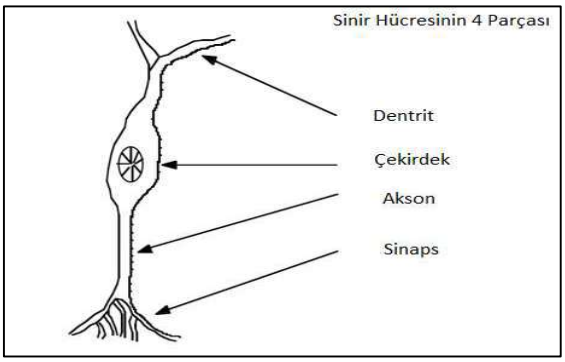


Şekil 3.1. Yapay sinir ağı yapısı [28]

3.3.2. Yapay sinir ağının mimarisi

Yapay sinir ağı, mimari olarak insandaki sinir hücrelerine benzemektedir. Sinir hücrelerindeki, çekirdek, nörona, dentrit, girdiye, akson, çıktıya sinaps ise ağırlığa karşılık gelmektedir.

Nöron bir sinir ağının temel işlem elemanıdır. Biyolojik açıdan bir nöron başka kaynaklardan girdileri almakta ve bunları birleştirerek sonuçta doğrusal olmayan bir işlem gerçekleştirmektedir. Ardından sonucu üretmektedir. Şekil 3.2 bu dört bölüm arasındaki ilişkiyi göstermektedir.

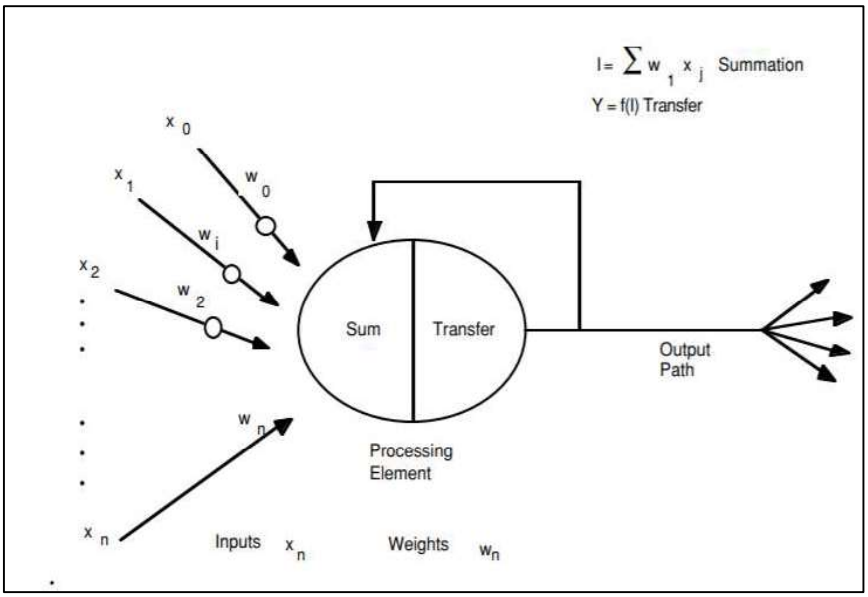


Şekil 3.2. Basit sinir hücresi yapısı [38]

Dentritler, giriş kanallarına karşılık gelmektedir. Buraya gelen bilgiler, diğer nöronların sinapsları aracılığıyla alınmaktadır. Sonrasında çekirdek, bu gelen bilgileri zaman içinde

işlemekte ve işlenmiş değeri akson ve sinapslar aracılığıyla diğer nöronlara gönderilen bir çıktıya dönüştürmektedir.

Yapay sinir ağı da mimari olarak insandaki sinir hücrelerine benzemektedir. Şekil 3.3 yapay sinir ağının temel bir gösterimini göstermektedir.



Şekil 3.3. Temel yapay sinir ağı [38]

Şekil 3.3'te girdi değerleri x(i) ve i=0,1,2,…,n, ağırlık değerleri ise w(j) j=0,1,2,…,n olmak üzere, matematiksel semboller ile gösterilmiştir. Bu girdi değerlerinin her biri bir bağlantı ağırlığıyla çarpılmaktadır. En basit yapıda, bu çarpımlar toplanarak bir aktivasyon (transfer) fonksiyonuna gönderilmekte ve sonuç üretilmektedir. Bu sonuç daha sonra bir çıktıya dönüştürülmektedir. Burada, girdi vektörü {x1,x2,…,xn} olarak ve uygun ağırlıklar vektörü ise {wj1, wj2,…,wjn} olarak gösterilmektedir.

Girdi, yapay sinir hücresine dış dünyadan veya diğer sinir hücrelerinden gelen bilgileri ifade etmektedir. Girdiler ağın öğrenmesi istenen örnekler tarafından belirlenmektedir. Her bir girdinin kendine özel ağırlığı mevcuttur. Ağ içinde yer alan nöronlar bir veya birden fazla girdiye sahip olabilmektedir. Ancak bu nöronlar, tek bir çıktı vermektedir. Bu çıktı, yapay sinir ağının dışına verilen bir çıktı ya da başka bir nörona verilen bir girdi olabilmektedir.

Ağırlık kavramı, yapay sinir ağına gelen bilginin önemini ve ağ üzerindeki etkisini göstermektedir. Ağırlıkların, pozitif veya negatif etkisi olabilmektedir. Ağırlığın sıfır olması demek herhangi bir etkisinin olmadığını ifade etmektedir.

Toplama fonksiyonu ile ağa gelen net bilgi hesaplanmaktadır. Bu işlem için en az, en çok, mod, çarpım, çoğunluk veya birkaç normalleştirme fonksiyonu gibi değişik fonksiyonlar kullanılmaktadır. Ancak bunlar içerisinde en yaygın olanı ağırlıklı toplamı kullanmaktır. Genellikle deneme yanılma yolu ile en iyi toplama fonksiyonu belirlenmektedir. Burada her gelen girdi değeri kendi ağırlığı ile çarpılarak toplanmaktadır. Sonuç olarak net bilgi bulunmaktadır.

Herhangi bir nöronunun toplam girdisi, diğer nöronlardan gelen bilgilerin ağırlıklı toplamı ile eşik değerlerin toplanmasıyla hesaplanmaktadır.

$$\text{Net Bilgi} = w_{ij} x_i + \theta_j \tag{3.1}$$

Toplama Fonksiyonu yukarıdaki gibi ifade edilmektedir.

w_{ij} : i ve j nöronları arasındaki bağlantının ağırlığını,

x_i : i nöronunun çıktısını,

θ_j : eşik değeri ifade etmektedir.

Aşağıda literatürde yapılan araştırmalarda kullanılan değişik toplama fonksiyonlarının matematiksel gösterimi yapılmıştır.

Toplam: Net Bilgi = $\sum_i^n w_{ij} x_i + \theta_j$

Çarpım: Net Bilgi = $\prod_i^n w_{ij} x_i$

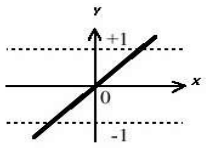
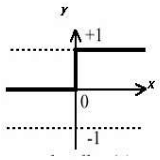
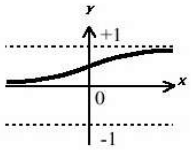
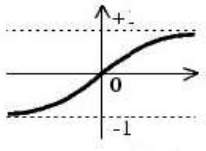
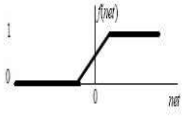
Minimum: Net Bilgi = $\min(w_{ij} x_i)$

Maksimum: Net Bilgi = $\max(w_{ij} x_i)$

Aktivasyon fonksiyonu, zaman söz konusu olduğunda toplama işlevinin çıkışının değiştirmektedir. Bu fonksiyon, toplama fonksiyonu ile elde edilen net bilgiyi işlemekte ve ağın bu bilgiye karşılık üreteceği çıktıyı belirlemektedir. Problem çeşidine göre farklı aktivasyon fonksiyonları kullanılmaktadır. Çok katmanlı algılayıcı modellerinde

öğrenmenin olabilmesi için aktivasyon fonksiyonunun türevi alınabilir bir fonksiyon olması gerekmektedir. Türevi alınamayan bir fonksiyonda öğrenme işlemi gerçekleşmemektedir.

Çizelge 3.1. En çok kullanılan aktivasyon fonksiyonları [39]

Aktivasyon Fonksiyonu	Matematiksel Gösterimi	Matematiksel Gösterimi
Lineer Fonksiyon	F(NET) = NET	
Step Fonksiyonu	$F(NET) = \begin{cases} 1 \text{ eğer } NET \geq \theta \\ 0 \text{ eğer } NET \leq \theta \end{cases}$	
Sigmoid Fonksiyonu	$F(NET) = \frac{1}{1+e^{-NET}}$	
Hiperbolik Tanjant Fonksiyonu	$F(NET) = \frac{e^{NET}+e^{-NET}}{e^{NET}-e^{-NET}}$	
Eşik Değer Fonksiyonu	$F(NET) = \begin{cases} 0 \text{ eğer } NET \leq 0 \\ NET \text{ eğer } 0 < NET < 1 \\ 1 \text{ eğer } NET \geq 1 \end{cases}$	

Aktivasyon fonksiyonunun seçimi, ağa ne öğretilmek isteniyorsa ona göre değişmektedir. Genelde doğrusal olmayan bir aktivasyon fonksiyonu ile işlemler yapılmaktadır. Çıktı, aktivasyon fonksiyonu tarafından belirlenmektedir. Üretilen çıktı başka bir nöronun girdisi olabilmektedir. Nöron kendi çıktısını kendisine yeniden girdi olarak da gönderebilmektedir.

3.3.3. Yapay sinir ağlarında öğrenme

Ağın eğitilmesi, örneklerin ağa gösterilerek örnekteki olaylar arasındaki ilişkilerin belirlenmesi demektir. Örnek, ağın eğitim ve test işlemleri için iki veri setine ayrılmaktadır.

Ağ ilk olarak eğitim örnekleri ile eğitilmekte ve sonrasında test edilmektedir. Ağ, hangi bilginin önemli olduğunu eğitim sırasında öğrenmektedir. Bu sebeple de eğitim tamamlandıktan sonra eksik bilgilerle bile çalışabilmektedir. Bu durum hatalara karşı toleranslı olmalarını sağlamaktadır. Çünkü bilgiler ağa yayılmış durumdadır. Bu sebeple eğitilmiş bir ağda hatanın olması veya bazı nöronların etkisiz olması ağın doğru bilgi vermesini etkilememektedir. Ağın performansını etkileyen bir durum değildir.

Öğrenme sürecinin başında ağırlıklar rastgele atanmaktadır. Örnekler ağa gösterildikçe ağırlıklar değişmektedir. Doğru ağırlık değerlerine ulaşan ağ, örneklerin temsil ettiği olay hakkında genellemeler yapabilmektedir. Bu durumda ağ öğrenme işlemini tamamlamıştır.

Literatüre göre geriye yayılım algoritması, eğitim için en çok kullanılan algoritmalardan birisidir. Bu algoritma çok katmanlı algılayıcılarda kullanılmaktadır. Verilen eğitim seti için en uygun sonucu üretecek ağırlıkların bulunmasını sağlamaktadır. Geriye yayılım algoritması, toplam hatayı minimize etmeye çalışmaktadır. Bu işlem için dereceli azaltma gibi çeşitli yaklaşımları kullanmaktadır. Geriye yayılım ağının öğrenme kuralı en küçük kareler yöntemine dayalı Delta öğrenme kuralının genelleştirilmiş halidir.

Geriye yayılım algoritması, ileri doğru hesaplama ve geriye doğru hesaplama olmak üzere iki aşamadan oluşmaktadır. İleri doğru hesaplama yapılırken ağın çıktısı hesaplanmaktadır. Geriye doğru hesaplamada ise hataları azaltmak için ağırlıklar yeniden düzenlenmektedir. En uygun ağırlık değerleri bulunana kadar bu süreç devam etmektedir.

3.3.4. Öğrenme kuralları

Literatürde konusu geçen çok sayıda öğrenme kuralı mevcuttur. Bunlardan en önemlileri hakkında aşağıda bilgi verilmiştir.

Hebb Kuralı, Donald Hebb tarafından geliştirilmiştir. En eski ve en ünlü öğrenme kuralıdır. Bir nöron kendisi aktif ise bağlı olduğu nöronu aktif, kendisi pasif ise bağlı olduğu nöronu pasif yapmaya çalışmaktadır [40].

Hopfield Kuralı'na göre girdi ve istenilen çıktının ikisi de aktif ise veya ikisi de pasif ise, bağlantı ağırlığı öğrenme katsayısı kadar arttırılmakta, aksi durumda ise öğrenme katsayısı

kadar azaltılmaktadır [41]. Ağırlıkların arttırılması veya azaltılması öğrenme oranı yardımı ile gerçekleştirilmektedir.

Delta Kuralı, Hebb kuralı gibi delta kuralı da en fazla kullanılan kurallardan biridir. Hebb kuralının genişletilmiş halidir. Beklenen çıktı ile sistemin bulmuş olduğu çıktı arasındaki farkın en az olmasını sağlar. En küçük kareler kuralı olarak da bilinmektedir.

Eğimli İniş Kuralı, Delta kuralına benzemektedir. Burada ağırlıklar, beklenen çıktı ile sistemin vermiş olduğu çıktı arasındaki farktan kaynaklanan hatanın birinci türeviyle orantılı bir şekilde değiştirilmektedir. Bu kuralın amacı, hata fonksiyonunu azaltarak yerel minimumdan kurtularak ve genel minimumu yakalamaktır [42].

Kohonen Öğrenme Kuralı'na Bu kurala göre ağdaki elemanlar, ağırlıklarını değiştirmek için birbirleri ile yarışmaktadır. En büyük çıktıyı üreten hücre çıktı olarak işleme alınmaktadır ve ağırlıkları değiştirilmektedir. Bu kural, hedef çıktıya gerek duymamaktadır. Bu sebeple de denetimsiz öğrenme yöntemini kullanmaktadır. Çeşitli sebeplerden dolayı yaygın olarak kullanılan bir kural değildir.

3.3.5. Yapay sinir ağının sınıflandırılması

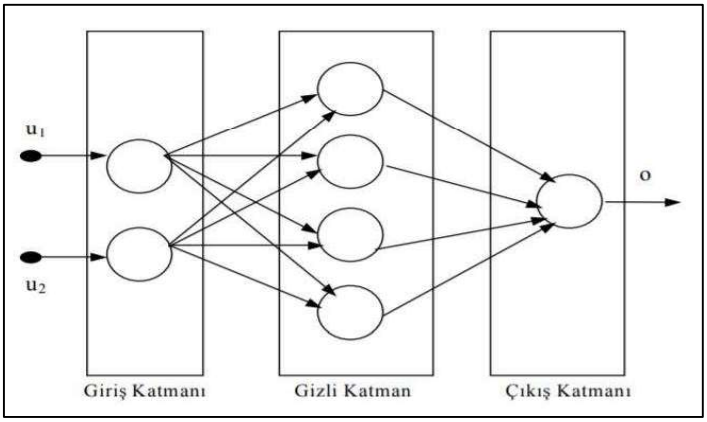
Yapay sinir ağları, bağlantı yapısına, öğrenme şekline ve katman sayısına göre sınıflandırılmaktadır. Yapay sinir ağının bağlantı yapısı, sinir hücreleri arasındaki bağlantıların yönlerine göre veya ağ içindeki işaretlerin akış yönlerine göre birbirlerinden ayrılmaktadır. Bazı ağlar ileri beslemeli bazıları ise geri beslemelidir. Hem ileri besleme hem de geri yayılma olarak tanımlanabilecek ağ yapıları da mevcuttur [42]. Öğrenme şekillerine göre denetimli, denetimsiz ve takviyeli öğrenme olarak üçe ayrılmaktadır. Son olarak da katman sayısına göre tek veya çok katmanlı olarak ikiye ayrılmaktadır.

- Bağlantı yapılarına göre sinir ağları hakkında aşağıda bilgi verilmiştir.

<u>İleri beslemeli sinir ağları</u>

Her bir katmandaki nöron bir sonraki katmandaki nöronlar ile bağlantılıdır. Ancak katmanlar içerisinde herhangi bir bağlantı yoktur. Nöronlar arasındaki bağlantılar bir döngü

oluşturmaz. Bilgi, girdi katmanından alındıktan sonra çıkış katmanına iletilmektedir. Herhangi bir geri besleme yoktur. İletim tek yönlü olacak şekildedir. Bu tür ağlar denetimli öğrenme yöntemini kullanarak eğitilmektedir. Şekil 3.4'te ağ yapısı gösterilmektedir.

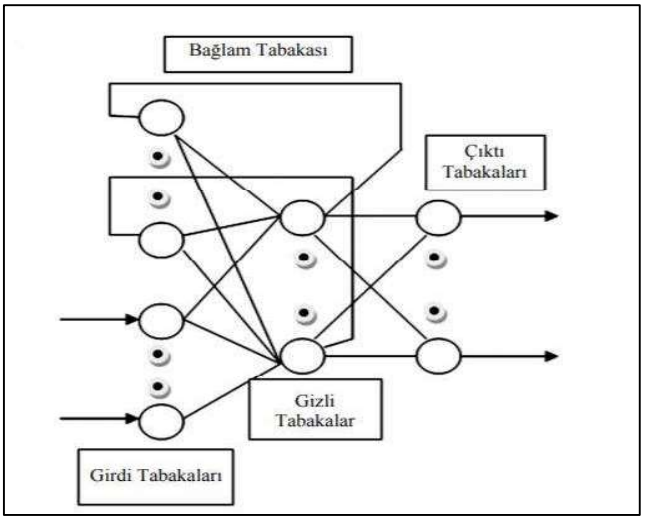


Şekil 3.4. İleri beslemeli sinir ağ yapısı [43]

Giriş katmanı almış olduğu bilgileri değişiklik yapmadan gizli katmandaki nöronlara iletmektedir. Gizli katmanların sayısı için herhangi bir kısıtlama yoktur. Geriye yayılım öğrenme algoritması, bu tip yapay sinir ağlarının eğitiminde etkin olarak kullanılmakta ve bazen bu ağlara geriye yayılım ağları da denilmektedir [42]. Bu tür ağlarda katman sayısı ve katmanlarda bulunan nöron sayısı performans açısından önemlidir. Bu ağlara, tek ve çok katmanlı algılayıcılar örnek olarak verilebilir.

Geri beslemeli sinir ağları

Gizli katmanındaki ve çıktı katmanındaki çıkışların giriş katmanına veya önceki ara katmanlara geri beslendiği ağ türüdür. Bağlantılar döngü içermektedir. Bu döngülerde yeni veriler kullanılmaktadır. Aynı katman içerisindeki nöronlar arasında bağlantı olabilmektedir. Geri beslemeli ağlar eğitim aşamasında çeşitli nedenlerden dolayı zordur. Aynı zamanda geri beslemeli ağların eğitimi uzun zaman almaktadır. Bu ağlara örnek olarak; Hopfield, Elman ve Jordan ağları verilebilir. Şekil 3.5'te ağ yapısı gösterilmektedir.



Şekil 3.5. Geri beslemeli ağ [44]

- Öğrenme yöntemlerine göre sinir ağları hakkında aşağıda bilgi verilmiştir.

Öğrenme işlemi olması için ağın eğitilmesi gerekmektedir. YSA ağlarının eğitilmesi demek, nöron bağlantılarının ağırlıklandırılması demektir. Ağırlık değerleri başlangıçta rastgele verilmektedir. Bu sayede öğrenme işlemi başlatılır. Örnekler ağa girdikçe ağırlık değerleri değişmektedir. Ağ istediği sonuçları elde edene kadar bu süreç devam etmektedir. Ağ öğrendikten sonra genelleme yapabilecek seviyeye gelmiştir. Denetimli, denetimsiz ve takviyeli olmak üzere 3 tip öğrenme yöntemi mevcuttur.

Denetimli öğrenme

Örnekler, sisteme girdi ve çıktı seti olarak verilmektedir. Her bir örnek için girdi ve girdiye karşılık gelen bir çıktı bulunmaktadır. Sistemden girdi ve çıktı arasındaki ilişkinin öğrenilmesi beklenmektedir. Ağın ürettiği çıktılar ve olması gereken çıktılar arasındaki fark hata olarak tanımlanmaktadır. Sistem, bu hatayı minimize etmek için çalışmaktadır. En uygun değer elde edilince bu süreç tamamlanmış olur. Bu süreçte bağlantı ağırlıkları da değişmektedir.

Denetimli öğrenme algoritmalarına örnek olarak; Wihrow-Hoff'un Delta Kuralı ve Rumelhart ve McClelland'ın Genelleştirilmiş Delta Kuralı veya geriye yayılım algoritması verilebilir.

Denetimsiz öğrenme

Sisteme sadece girdi değerleri verilmektedir. İlişkileri sistemin kendi kendisine öğrenmesi beklenmektedir. Daha çok sınıflandırma problemlerinde kullanılan öğrenme yöntemidir. Bu tür öğrenmeye örnek olarak, Hebb, Hopfield ve Kohonen öğrenme yöntemleri verilebilir.

Takviyeli öğrenme

Sistem, kendisine gösterilen girdiler sonucunda çıktı üretmektedir. Üretilen çıktının doğru veya yanlış olduğunu gösteren bir sinyal yayınlanmaktadır. Bu sinyal dikkate alınarak öğrenme işlemine devam edilmektedir. Bu öğrenme yönteminin kullanıldığı ağlara örnek olarak Doğrusal Vektör Parçalama Modeli verilebilir.

- Katman sayılarına göre sinir ağları hakkında aşağıda bilgi verilmiştir.

Tek katmanlı yapay sinir ağı

Sadece girdi ve çıktı katmanlarından oluşan sinir ağlarıdır. Doğrusal problemler için uygun bir ağdır. Ağın çıktısı, ağırlıklandırılmış girdi değerlerinin ve eşik değerin toplanması sonucu bulunmaktadır. Bu girdi değeri, bir aktivasyon fonksiyonundan geçirilerek ağın çıktısı hesaplanır. Bu işlemi matematiksel olarak aşağıdaki gibi gösterebiliriz [22].

$$Ç = f\left(\sum_i^m w_i x_i + \theta\right) \quad (3.2)$$

Burada;

x_i : i=1,2,…,n : girdiyi,

w_{ik} : i=1,2,…,n; k=1 : ağırlıkları,

θ : eşik değeri ifade etmektedir.

Çıktı doğrusal bir fonksiyon olup değeri bir veya sıfırdan oluşmaktadır. Çıktının değerinin hesaplanmasında eşik değer fonksiyonu kullanılmaktadır [45].

Çok katmanlı yapay sinir ağı

Birden fazla nöronun birbiri ile bağlanarak meydana getirdikleri sinir ağıdır. Birden fazla katmandan oluşmaktadır. Bu tür ağların, girdi katmanı, bir veya birden fazla gizli katmanı ve çıktı katmanı bulunmaktadır. Gizli katmandaki her nöron, doğrusal olmayan bir aktivasyon fonksiyonuna sahiptir. Bu fonksiyonlar aracılığıyla elde edilen sonuçlar bir sonraki nöronlara girdi olarak iletilmektedir. Karmaşık ve doğrusal olmayan problemlerin çözümünde kullanılmaktadır. Özellikle sınıflandırma, tanıma ve genelleme yapmayı gerektiren problemler için çok önemli bir çözüm aracıdır. Temel amacı, ağın beklenen çıktısı ile ürettiği çıktı arasındaki hatayı minimize etmektedir [45]

Bu tür ağlar, denetimli öğrenme yöntemini kullanmaktadır. Eğitim sırasında hem girdiler hem de o girdilere karşılık üretilmesi gereken çıktılar gösterilmektedir. Öğrenme kuralı ise en küçük kareler yöntemine dayalı Delta Öğrenme Kuralı'nın genelleştirilmiş halidir [45].

3.4. Derin Öğrenme

Yapay zeka teknolojisinin yeni yeni gelişmeye başladığı dönemlerde insanlar için zor ama bilgisayaralar için kolay olan problemler karmaşık matematiksel işlemlerle ve matematiksel formüllerle çözülebilmekteydi. Ancak insanlar için basit, bilgisayarlar için zor olan sezgisel problemlerin çözümü için yeni yaklaşımlar gerekmekteydi. Derin öğrenme bu yaklaşımlardan biridir. Bir nesnenin zihindeki soyut ve genel tasarımı anlamına gelen kavram terimi, merkezden genele, genelden de özele hiyerarşik bir yapıda bulunmaktadır. Kavramlar hiyerarşisi adı verilen bu yapı, derin öğrenme yöntemlerine karmaşık kavramları öğrenebilme imkânı sağlamaktadır [46]. Derin Öğrenme, bilgisayarların deneyimlerden öğrenmelerini ve dünyayı kavramlar hiyerarşisi bağlamında anlamalarını ve her kavramın daha basit kavramlarla olan ilişkileriyle tanımlanmasını sağlamaktadır. Tecrübelerden bilgi toplayan bu yaklaşım ile bilgisayarların ihtiyaç duyduğu tüm bilgilerin insanlar tarafından tanımlanmasına gerek yoktur. Kavram hiyerarşisi, bilgisayarın karmaşık kavramları öğrenerek daha basit olanları oluşturabilmesini sağlamaktadır.

Bugüne kadar derin öğrenmenin, literatürde farklı kaynaklarda değişik birçok tanımı yapılmıştır. Kaynaklardan birinde derin öğrenme, bilgisayarların, deneyimlerden öğrenmelerini ve dünyayı kavramların hiyerarşisi açısından anlamalarını sağlayan bir

makine öğrenimi olarak tanımlamıştır [47]. Başka bir kaynakta, denetimli veya denetimsiz özellik çıkarma, dönüştürme, desen analizi ve sınıflandırma için birçok doğrusal olmayan gizli katmandan yararlanan bir makine öğrenme teknikleri sınıfı olarak tanımlamıştır [48]. Diğer bir kaynakta, insan beyninin son derece karmaşık problemler için gözlemleme, analiz etme, öğrenme ve karar verme yeteneğini taklit etmeyi amaçlayan, büyük miktarda denetimsiz veri kullanan bir makine öğrenmesi olarak tanımlanmıştır [49]. Tanımlardan bir diğeri ise, insan beyninin karmaşık problemler için gözlemleme, analiz etme, öğrenme ve karar verme gibi yeteneklerini taklit eden, denetimli veya denetimsiz olarak özellik çıkarma, dönüştürme ve sınıflandırma gibi işlemleri büyük miktarlardaki verilerden yararlanarak yapabilen bir makine öğrenmesi tekniğidir. Burada kullanılan algoritmalar denetimli veya denetimsiz olabilmektedir. Derin öğrenme, özellik çıkarma ve dönüştürme için birçok doğrusal olmayan işlem birimi katmanını kullanmakta ve her ardışık katman, önceki katmandaki çıktıyı girdi olarak almaktadır [48].

Derin öğrenme modelinin genel yapısı, girdi, gizli ve çıktı katmanlarından oluşmaktadır. Derin öğrenme yapısının her bir katmanı farklı bir öznitelik kümesine karşılık gelmektedir. Bu sayede derin öğrenme modeli işlediği verilerden daha doğru çıkarımlarda bulunarak daha başarılı sonuçlar üretebilmektedir. Derin öğrenme süreci, sonuçtaki başarı oranı belirli bir seviyeye ulaşıncaya kadar tekrar etmektedir.

Derin öğrenmedeki katmanların her biri sayısal bir değer ile ağırlıklandırılmaktadır [50]. Gizli katmandaki nöronlar, giriş ve çıkış nöronlarından, öğrenme algoritmasından, ağ yapısından ve aktivasyon fonksiyonunun türünden etkilenmaktadir [51]. Gizli katmandaki nöronların sayısını belirlemek için, farklı ağları eğitmek ve test verisindeki hatayı araştırmak gerekmektedir [52].

Derin öğrenme temel olarak verinin temsilinden öğrenmeye dayanmaktadır [53].Temsili öğrenme, verinin işlenmesiyle elde edilen bazı bilgilerin veriyi kendisinden daha iyi temsil edebilme durumu olarak tanımlanmaktadır [54].

Eğitim verilerinin sayısı arttıkça derin öğrenme daha kullanışlı hale gelmiştir. Aynı zamanda büyük veri çağı ile birlikte makine öğrenmesi daha kolay uygulanmaya başlanmıştır. Derin öğrenme modelleri alanında yapılan çalışmalar, derin öğrenme için bilgisayar altyapısının (hem donanım hem de yazılım) gelişmesiyle birlikte artmıştır. Günümüzde, çok daha büyük

modelleri çalıştırmak için gerekli olan daha büyük belleğe sahip daha hızlı bilgisayarlar ve daha büyük veri kümeleri mevcuttur. Bu sebeple daha büyük ağlar daha karmaşık görevlerde daha yüksek doğruluk elde edebilmektedir [55].

Derin öğrenme çok büyük miktarda veriyi işlemek ve çeşitli bilim dallarında faydalı tahminler yapmak için faydalı araçlar sağlamaktadır. Bu sebeple derin öğrenme birçok alanda kullanılmaktadır. Günümüzde derin öğrenme, Google, Microsoft, Facebook, IBM, Baidu, Apple, Adobe, Net x ix, NVIDIA ve NEC dahil olmak üzere birçok teknoloji şirketi tarafından kullanılmaktadır [55].

3.4.1. Derin öğrenme tarihçesi

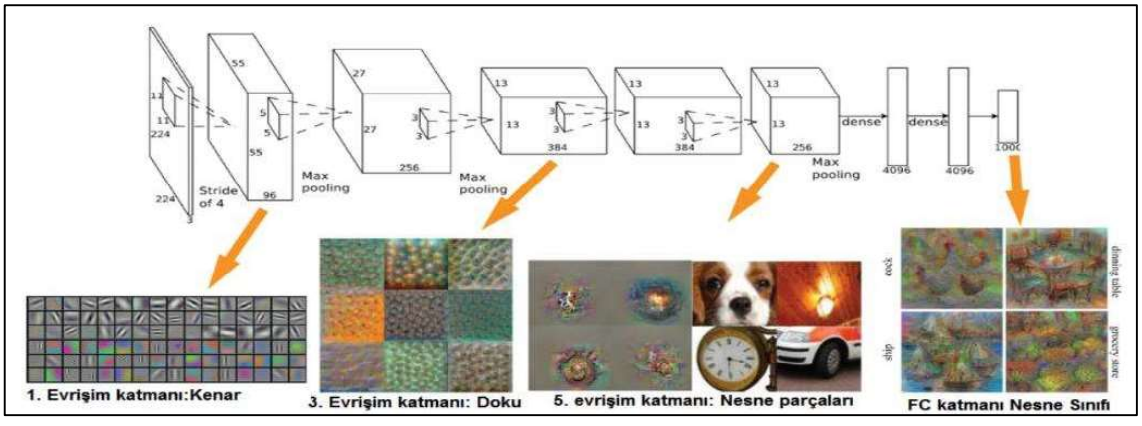
Derin Öğrenme yeni bir teknoloji gibi düşünülse de ortaya çıkışı eskiye dayanmaktadır. Sadece isimlendirmeleri farklı olmuştur ve zamanla çeşitli nedenlerden dolayı popülerliğini kaybetmiştir. Farklı araştırmacılar ve farklı bakış açılarının etkisine bağlı olarak, bu alan birçok kez yeniden gündeme gelmiştir.

Derin öğrenme, ilk olarak 1940 ve 1960 yılları arasında sibernetik olarak ortaya çıkmıştır. Sibernetik ile biyolojik öğrenme geliştirilmiş ve algılayıcı gibi ilk modeller uygulanmaya başlanmıştır. Tek bir nöronun eğitimi yapılmıştır. 1980 ve 1990 yılları arasında sinir ağları olarak ortaya çıkmıştır. Bir ya da iki gizli katmanı olan bir sinir ağını eğitmek için geri yayılım kullanılmıştır. 2006 yılı itibariyle de derin öğrenme adı ile ortaya çıkmıştır [55].

Aşağıda önemli olayları vurgulayan sinir ağlarının kısa bir geçmişi bulunmaktadır [56]. Derin öğrenmenin geçmişi 1940'lara dayanmaktadır.

- 1943 yılında, McCulloch ve Pitts, bir Turing makinesi oluşturmak için nöronların birleştirilebileceğini göstermiştir[57].
- 1958 yılında, Frank Rosenblatt tarafından yapay sinir ağının temeli olan perceptron(algılayıcı) kavramını tanımlanmıştır[58].
- 1969 yılında, Minsky ve Papert tarafından XOR probleminin tek katmanlı ağ yapısıyla çözülemeyeceğini göstererek algılayıcının sınırlarını çizmiştir. Sonraki on yıl boyunca sinir ağlarında yapılan araştırmalar durmuştur [59].

- 1985 yılında, Geoffrey Hinton ve arkadaşları ‘Geriye Yayılım’ (Backpropagation) teorisini ‘Çok Katmanlı Perseptron’ ağ yapısıyla birlikte paylaşmıştır [60].
- 1988 yılında, görsel örüntü tanıma yeteneğine sahip hiyerarşik bir sinir ağı oluşturulmuştur [61].
- 1998 yılında Yann LeCun, gradyan temelli yaklaşımla evrişimsel sinir ağlarını (convolutional neural network-CNN) kullanmıştır ve kendi ağ yapısına da LeNet adını verilmiştir. 0–9 arasındaki rakamları öğrenerek sınıflandıran model, sistemin bunlardan herhangi biriyle karşılaşması durumunda hangi sayı olduğunun ihtimallerini kullanıcıya sunmaktadır [62]. İlk başarılı sonucu veren evrişimli sinir ağı modelidir. Yann LeCun ve ekibi tarafından posta numaraları, banka çekleri üzerindeki sayıların okunması için geliştirilmiştir. MNIST (Modified National Institute of Standards and Technology) veri seti üzerinde deneyler gösterilmiştir.
- 2006 yılında, Hinton laboratuarı DNN'ler için eğitim problemlerini çözmektedir [63,64].
- 1998’den 2010’a kadar genellikle; sınıflandırma, örüntü tanıma, nesne tanıma alanlarındaki problemlere piksel tabanlı olarak görüntü işleme (image processing) yaklaşımları geliştirilmiştir [65].
- 2012 yılında Alex Krizhevsky, Ilya Sutskever ve Geoffrey Hinton’ın oluşturdukları evrişimsel sinir ağı modeli olan AlexNet büyük problemlere çözüm olmaktadır. Evrişimli sinir ağ modelleri, derin öğrenmenin tekrar popüler hale gelmesini sağlayan ilk çalışmadır. Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton tarafından geliştirilmiştir. Temel olarak LeNet modeline, birbirini takip eden evrişim ve pooling katmanları bulunmasından dolayı, çok benzemektedir.



Şekil 3.6. AlexNet evrişimsel ağ modeli katman çıktılarının görseli [66]

Derin öğrenme, insan beynini taklit ederek, istatistikleri ve uygulamalı matematiği, yoğun şekilde kullanan bir makine yaklaşımıdır. Son yıllarda, derin öğrenme, daha güçlü bilgisayarların, daha büyük veri setlerinin ve daha derin ağları eğitmek için kullanılan tekniklerin bir sonucu olarak gündeme gelmektedir ve kullanılma oranında belirgin bir büyüme kaydetmiştir. Önümüzdeki yıllarda, derin öğrenmenin daha da gelişeceği öngörülmektedir.

3.4.2. Derin öğrenme mimarileri

Problem tipine göre farklılık gösteren çok farklı türde derin öğrenme mimarileri mevcuttur. Bu bölümde literatürde en çok kullanılan derin öğrenme mimarisi olan Evrişimli Sinir Ağları (ESA) hakkında bilgi verilmiştir.

Evrişimli sinir ağları

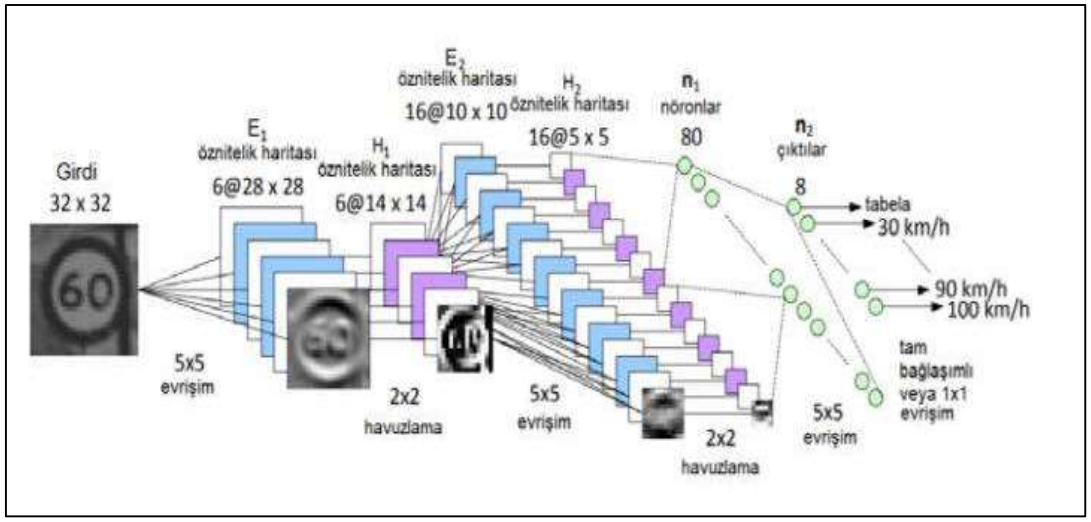
Evrişimli sinir ağları, son yıllarda farklı alanlar da kullanılsa da temel olarak görüntü sınıflandırma üzerinde yoğunlaşmıştır. Evrişimli Sinir Ağları çok katmanlı algılayıcıların bir türüdür. Geleneksel evrişimli sinir ağ mimarisi, girdi katmanı, evrişim katmanı, havuzlama (ortaklama) katmanı, tam bağlantı katmanı ve çıktı katmanı olmak üzere beş ana katmandan oluşmaktadır. Veriler, girdi katmanından girdikten sonra katman katman işlemler yapılarak eğitim süreci gerçekleştirilmektedir. Sonrasında sistemin vermiş olduğu çıktı ile istenen sonucun farkı kadar bir hata oluşmaktadır. Bu hata, geriye yayılım algoritması ile bütün ağırlıklara aktarılmaktadır. Her bir iterasyonla ağırlıkların güncellenmesi yapılarak hatanın azaltılması sağlanmaktadır.

Veriden özellik çıkarma işlemi evrişim katmanında gerçekleşmektedir. Özellik çıkarma işlemi esnasında filtreler kullanılmaktadır. Parametre azaltma işlemi bir diğer katman olan havuzlama katmanında yapılmaktadır.Bu işlem performans açısından önemlidir. Maksimum havuzlama ve ortalama havuzlama olmak üzere iki çeşittir. Bu iki yöntem ile çıktının boyutunun küçültülmesi amaçlanmaktadır. Maksimum ortaklamada, her havuzlama işlemi, geçerli matrisin maksimum değerini seçmektedir. Ortalama havuzlamada ise her havuzlama işlemi, geçerli matrisin değerlerinin ortalamasını almaktadır. Bir diğer katman olan tam bağlantı katmanı, yapay sinir ağı gibi çalışmaktadır. Evrişim ve havuzlama işlemleri

sonucunda üretilen değerler bu katman tarafından girdi olarak alınarak işleme sokulmakta ve çıkış katmanında sınıf sayısı kadar sonuç üretilmektedir.

Temel sinir ağlarında olduğu gibi evrişimli ağlarının da en iyi sonucu verebilmesi için hiper parametrelerin en iyi şekilde seçilmesi gerekmektedir. Hiperparametrelere örnek olarak, filtre derinliği, filtre boyutu, adım sayısı, dolgu, havuzlama sayısı verilebilir.

Evrişimli sinir ağları, görüntü ve ses işleme alanı başta olmak birçok farklı alanda uygulanmaktadır. Örneğin NLP, sınıflandırma, tahmin problemlerinde de kullanılmaktadır.



Şekil 3.7. Genel evrişimli sinir ağı mimarisi [67]

4. METODOLOJİ VE DENEYSEL SONUÇLAR

Çalışmada kötü niyetli URL'lerin saldırı türlerine göre tespit edilmesi ve sınıflandırılması için derin öğrenme kullanan bir yöntem önerilmiştir. Aynı zamanda literatürde çok sayıda örneği olan makine öğrenmesi yöntemi de kullanılmıştır. Daha önceden yapılmış olan çalışmalara, literatür çalışması bölümünde yer verilmiştir. URL'lerin proaktif tespiti için sözcüksel analiz kullanılmıştır. İyi huylu, spam, phishing, malware ve defacement olmak üzere beş farklı URL türü hem makine öğrenmesi hem de derin öğrenme yöntemleri ile incelenmiştir. İki yöntemde de ikili ve çoklu sınıflandırma yapılmıştır. Sonuçlar kendi içlerinde ve birbirleri ile karşılaştırılmıştır.

Hem makine öğrenmesi hem de derin öğrenme modellerini geliştirirken python programla dili kullanılmıştır. Makine öğrenmesi kullanarak geliştirilen modeller tümleşik geliştirme ortamlarından biri olan Anaconda üzerinde oluşturulmuştur. Sklearn, pandas, numpy kütüphaneleri kullanılmıştır. Derin öğrenme modelleri için ücretsiz bulut ortamı olan Google Colab kullanılmıştır. Colab'ta ücretsiz GPU desteği bulunmaktadır ve herhangi bir kuruluma gerek duymadan çalışma imkanı sağlamaktadır. Kütüphane olarak keras, pandas numpy, sklearn, scipy ve matplotlib kullanılmıştır.

4.1. Sözcüksel Analiz

Kötü niyetli URL'lerin saldırı türlerine göre tespit edilmesi ve sınıflandırılmasına yönelik bir çalışma yapılmıştır. Çalışma kapsamında URL'lerin proaktif tespiti için sözcüksel özelliklerin analizi yapılmıştır. Sözcüksel özellikler, hostname uzunluğu, URL uzunluğu, URL de bulunan belirteçler gibi URL'nin metinsel özellikleridir. URL'lerden toplanan özellikler hiçbir uygulamaya bağlı değildir. Pek çok kötü amaçlı URL'nin ömrü kısa olduğundan, kötü amaçlı web sayfaları kullanılamadığında bile sözcüksel özellikler kullanılabilir durumda kalır. Analiz için URL'lerin beş ana bileşeni (URI, domain, yol, argümanlar ve dosya adı) incelenmektedir. Analiz için dikkate alınan tüm özelliklerin kısa açıklaması aşağıdaki gibidir.

Entropy domain ve uzantısı, kötü amaçlı web siteleri genellikle meşru görünmek için URL'ye ek karakterler eklemektedir. İngilizce metinler oldukça düşük bir entropiye sahiptir,

yani tahmin edilebilmektedir. Karakter eklenmesi ile entropi normalden daha fazla değişmektedir. Bu sebeple rasgele oluşturulmuş kötü amaçlı URL'leri tanımlamak için alfabe entropisi kullanılmaktadır.

Karakter sürekliliği oranı, domainde yer alan her karakter türünün en uzun belirteç uzunluğunun toplamını bulmak için kullanılmaktadır. Kötü amaçlı web siteleri, değişken sayıda karakter türüne sahip URL'leri kullanmaktadır. Karakter sürekliliği oranı, harf, rakam ve sembol karakterlerinin sırasını belirlemektedir. Bir karakter türünün en uzun belirteç uzunluğunun toplamı, URL'nin uzunluğuna bölünmektedir.

Anormal parçaları bulmak için URL bölümlerinin uzunluk oranı hesaplanmaktadır. URL bölümünün birleşimi argüman, yol, domain ve URL'den oluşmaktadır.

- argPathRatio: Argüman ve yol oranı,
- argUrlRatio: Argüman ve URL oranı,
- argDomainRatio: Arguman ve alan adı oranı,
- domainUrlRatio: Alan adı ve URL oranı,
- pathUrlRatio: Yol ve URL oranı,
- PathDomainRatio: Yol ve Alan adı oranı

Harf, belirteç ve sembol sayıları, URL'deki karakterlerin sıklığı harf, belirteç ve sembol ile hesaplanmaktadır. Bu karakterler, URL'lerin bu bileşenlerinden kategorilere ayrılmakta ve sayılmaktadır.

- Symbol Count Domain: Alan adındaki : /.: /? = ,; ()] + gibi sınırlayıcıların alan adından hesaplanması. Örneğin phishing URL'lerinde iyi huylu URL'lere göre daha fazla nokta vardır.
- Domain token count: URL dizisinden alınan belirteçleri ifade etmektedir. Kötü amaçlı URL'ler birden çok alan adı belirteci kullanmaktadır. Burada alan adında yer alan belirteç sayısı hesaplanmaktadır.
- Query Digit Count: URL'nin sorgu bölümündeki basamak sayısı
- Tld: Bazı phishing URL'leri bir alan adı içinde birden fazla üst düzey alan adı kullanabilir.

Domain sayısı, dizin adı, dosya adı, URL, yol, sayı oranı, dizin adının, alan adının, dosya adının, URL'nin kendisinin ve yolun sonundaki URL bölümlerindeki basamakların oranını hesaplamaktadır.

Uzunluk ile ilgili özellikler, değişkenlerin eklenmesi veya yönlendirilen URL'den dolayı URL'nin uzunluğu uzamaktadır.

- url Len,
- Domain Len
- file Name Len,
- Arguments' LongestWordLength: Sunucu tarafında kodlanmış sayfalardan gelen URL'lerin genellikle argümanları vardır. URL argümanlarından en uzun değişken uzunluğu hesaplanır.
- Longest Path Token Length
- Avgpathtokenlen

ldl getArg, kimlik avı URL'lerinde, maskeleme harflere rakam ekleyerek yapılmaktadır. Bu aldatıcı URL'lerin tespiti için URL ve yoldaki harf rakamlı harflerin sırası hesaplanmaktadır.

spcharUrl, URL'ler // gibi şüpheli özel karakterler kullanmaktadır ve yeniden yönlendirme riski daha yüksektir.

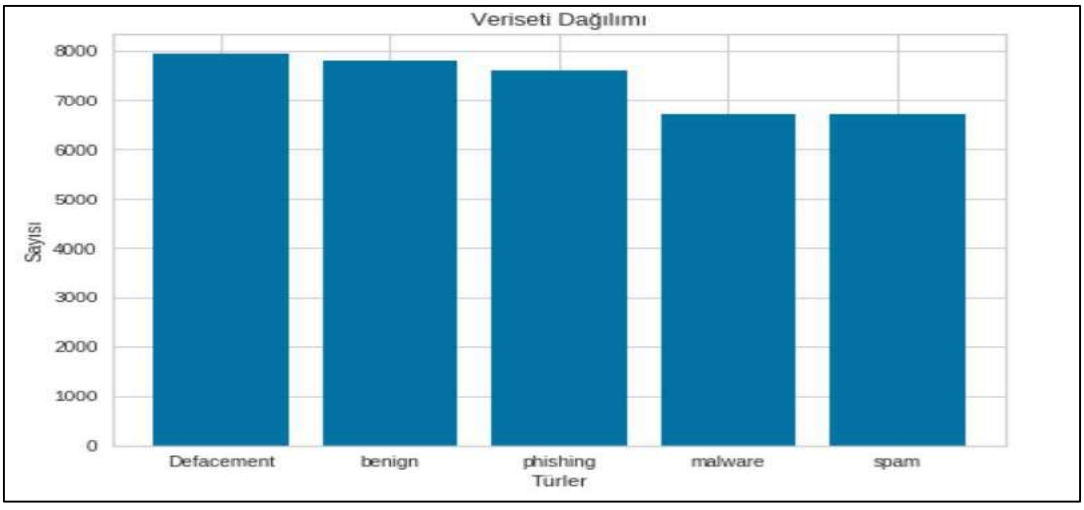
4.2. Kullanılan Veri Kümesi

Çalışma kapsamında M. Mamun ve arkadaşları tarafından 114,250 adet URL incelenerek oluşturulmuş veri kümesi kullanılmıştır. Veri kümesi içerisinde 5 farklı sınıfa ait veri bulunmaktadır. Bu sınıflar benign (iyi huylu), spam, phishing, malware ve defacement'dır. Sırasıyla benign için 35,000, spam için 12,000, phishing için 10,000, malware için 11,500 ve defacement için 45,450 URL incelenmiştir. Veri kümesinde ikili sınıflandırma ve çoklu sınıflandırma yapalabilecek şekilde veriler birleştirilmiştir. Veri kümesi içerisinde 79 özellik ve 1 sınıf etiketi bulunmaktadır. Çizelge 4.1'de veri kümesi içerisinden küçük bir kesit gösterilmiştir.

Çizelge 4.1. Veri örnekleri

Querylen gth	domain_token_ count	path_token_ count	longdomaintok enlen	…	NumberRate_Ext ension	URL_Type_obf _Type
0	4	5	14	…	1	Defacement
0	4	5	14	…	NaN	Spam
0	4	5	14	…	NaN	bening
0	4	12	14	…	NaN	Defacement
0	4	6	14	…	NaN	bening
0	4	5	14	…	NaN	malware
0	4	6	14	…	NaN	malware
0	4	5	14	…	1	Defacement
0	4	7	14	…	NaN	Defacement
0	4	9	14	…	NaN	Spam
0	4	4	14	…	NaN	Spam
0	4	5	14	…	NaN	bening
0	4	8	14	…	NaN	bening
0	4	5	14	…	NaN	Defacement

Veri kümesi içerisindeki URL'lerden çıkarılan özellikler sonucunda Defacement için 7,930, benign için 7,781, phishing için 7,586, malware için 6,712 ve spam için 6,698 adet kayıt bulunmaktadır. Veri kümesi içerisindeki örnekleme sayılarının birbirlerine yakın olması sistem performansı açısından önem arz etmektedir. Şekil 4.1'de kullanılan veri kümesi içerisindeki verilere ait sınıf dağılımları gösterilmiştir.



Şekil 4.1. Veri kümesi içerisindeki sınıf dağılımları

4.3. Değerlendirmede Kullanılan Metrikler

Literatürede, sistemlerin performansını ölçmek için kullanılan birçok metrik bulunmaktadır. Bunlar; doğruluk (accuracy), hassasiyet (recall), kesinlik (presicion) ve f-ölçütü (f-measure) şeklindedir. Bu metrikler doğru pozitif (TP, True Positive), yanlış pozitif (FP, False Positive), doğru negatif (TN, True Negative) ve yanlış negatif (FN, False Negative) değerleri üzerinden hesaplanmaktadır.

Doğru pozitif, gerçekte anormal olan bir veri model tarafından anormal olarak sınıflandırılıyorsa TP olarak kabul edilmektedir. Yanlış pozitif, gerçekte normal olan bir veri model tarafından anormal olarak sınıflandırılıyorsa FP olarak kabul edilmektedir. Doğru negatif, gerçekte normal olan bir veri model tarafından normal olarak sınıflandırılıyorsa TN olarak kabul edilmektedir. Yanlış negatif, gerçekte anormal olan bir veri model tarafından normal olarak sınıflandırılıyorsa FN olarak kabul edilmektedir.

Doğruluk: Başarılı sınıflandırılmış verilerin sayısının tüm verilerin sayısına oranıdır.

$$\text{Doğruluk} = \frac{TP+TN}{Tüm\ Veri\ Seti} \tag{4.1}$$

Kesinlik: Anormal olarak başarılı sınıflandırılmış verilerin sayısının, tüm anormal olarak sınıflandırılmış verilerin sayısına oranıdır.

$$\text{Kesinlik} = \frac{TP}{TP+FP} \tag{4.2}$$

Hassasiyet: Anormal olarak başarılı sınıflandırılmış verilerin tüm anormal verilerin sayısına oranıdır.

$$\text{Hassasiyet} = \frac{TP}{TP+FN} \tag{4.3}$$

F-Ölçütü: Kesinlik ve duyarlılık metriklerinin harmonik ortalamasıdır. Sistemin genel başarısını göstermek için kullanılmaktadır.

$$\text{F-Ölçütü} = \frac{2*Doğruluk*Kesinlik}{Doğruluk+Kesin} \quad (4.4)$$

4.4. Makine Öğrenmesi Yöntemi ve Sonuçları

Bu çalışma kapsamında zararlı URL'lerin tespit edilip sınıflandırılması için derin öğrenme tabanlı bir yaklaşımın yanında literatürde çok sayıda örneği olan makine öğrenmesi yöntemi de kullanılmıştır. Daha önceden yapılmış olan çalışmalara, literatür taraması bölümünde yer verilmiştir.

Makine öğrenmesi algoritmaları hem ikili sınıflandırma hem de çoklu sınıflandırma için kullanılmıştır. İkili sınıflandırmalarda veriler, kötü niyetli (defacement, malware, phishing ve spam) ve İyi huylu URL (benign) olarak etiketlenmiştir. Çoklu sınıflandırmada ise veriler 5 etiketten defacement, malware, phishing, spam ve benign olmak üzere oluşmaktadır. Hem ikili sınıflandırma da hem de çoklu sınıflandırmada etiketler encoding yöntemi ile sayısallaştırılmıştır. Veri setinde nan ve infinity değerlerin bulunması sebebiyle veri ön işleme işlemi gerçekleştirilmiştir. Bu işlemlerden sonra En Yakın k Komşusu (KNN), Destek Vektör Makinesi (SVM), Karar Ağacı (DT), Rassal Orman (RF) gibi 4 farklı makine öğrenmesi algoritması hem ikili hem de çoklu sınıflandırma yapmak için işleme alınmıştır. Algoritmaların performansları hakkında aşağıda bilgi verilmiştir.

4.4.1. İkili sınıflandırma

Çizelge 4.2'de algoritmaların ikili sınıflandırma sonucundaki doğruluk ve performans oranları gösterilmektedir. Random Forest algoritması tüm veri kümelerinde en iyi doğruluğu ve performansı elde etmiştir.

Çizelge 4.2. İkili sınıflandırma sonuçları

Veri Kümesi	Etiket	Algoritma	Doğruluk	Kesinlik	Hassasiyet	F1 Ölçütü
İkili Sınıflandırma	Defacement	KNN	0,9805	0,9836	0,9771	0,9803
		SVM	0,9753	0,9778	0,9725	0,9751
		DT	0,9967	0,9972	0,9961	0,9967
		RF	0,9971	0,9980	0,9961	0,9970
	Malware	KNN	0,9690	0,9650	0,9685	0,9668
		SVM	0,9795	0,9958	0,9600	0,9775
		DT	0,9870	0,9826	0,9896	0,9861
		RF	0,9918	0,9950	0,9874	0,9912
	Phishing	KNN	0,9593	0,9663	0,9509	0,9585
		SVM	0,9426	0,9808	0,9014	0,9394
		DT	0,9739	0,9763	0,9708	0,9735
		RF	0,9816	0,9866	0,9760	0,9813
	Spam	KNN	0,9794	0,9770	0,9787	0,9779
		SVM	0,9974	0,9977	0,9968	0,9972
		DT	0,9964	0,9977	0,9945	0,9961
		RF	0,9983	0,9995	0,9968	0,9981

4.4.2. Çoklu sınıflandırma

Çoklu sınıflandırmada veriler, defacement, malware, phishing, spam ve benign olmak üzere 5 etiketten oluşmaktadır. Bu etiketler önişleme kısmında sayısal değerlere atanmıştır. Sırası ile defacement için 0 değeri, benign için 1 değeri, malware için 2 değeri, phishing için 3 değeri ve spam için 4 değeri atanmıştır. 24,593 adet özellik eğitim için 12,114 adet özellik de test için kullanılmıştır. Çoklu sınıflandırmanın doğruluk ve performans oranları Çizelge 4.3., Çizelge 4.4., Çizelge 4.5. ve Çizelge 4.6'da gösterilmiştir. En yüksek doğruluk oranı ve en iyi performans değerleri Random Forest algoritması sonucunda elde edilmiştir.

Çizelge 4.3. KNN çoklu sınıflandırma sonuçları

Veri Kümesi	Etiket	KNN			
		Doğruluk	Kesinlik	Hassasiyet	F1
Çoklu Sınıflandırma (KNN)	Defacement	0,9195	0,9012	0,9540	0,9269
	Benign		0,9182	0,9403	0,9291
	Malware		0,9325	0,9281	0,9303
	Phishing		0,9058	0,8518	0,8780
	Spam		0,9467	0,9247	0,9355
	Ortalama		0,9197	0,9195	0,9192

Çizelge 4.4. SVM çoklu sınıflandırma sonuçları

Veri Kümesi	Etiket	SVM			
		Doğruluk	Kesinlik	Hassasiyet	F1
Çoklu Sınıflandırma (SVM)	Defacement	0,9174	0,9698	0,9096	0,9387
	Benign		0,7843	0,9824	0,8722
	Malware		0,9936	0,9437	0,9680
	Phishing		0,9122	0,8710	0,8911
	Spam		0,9875	0,8886	0,9299
	Ortalama		0,9271	0,9175	0,9190

Çizelge 4.5. DT çoklu sınıflandırma sonuçları

Veri Kümesi	Etiket	DT			
		Doğruluk	Kesinlik	Hassasiyet	F1
Çoklu Sınıflandırma (DT)	Defacement	0,9481	0,9624	0,9716	0,9670
	Benign		0,9637	0,9675	0,9656
	Malware		0,9481	0,9662	0,9571
	Phishing		0,9160	0,8844	0,8999
	Spam		0,9620	0,9660	0,9640
	Ortalama		0,9501	0,9504	0,9502

Çizelge 4.6. RF çoklu sınıflandırma sonuçları

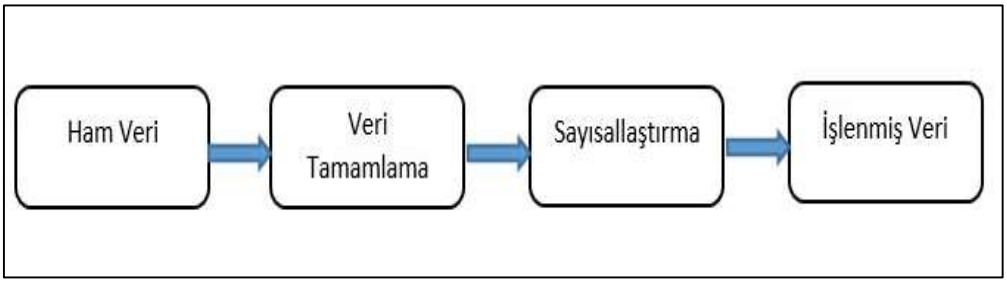
Veri Kümesi	Etiket	RF			
		Doğruluk	Kesinlik	Hassasiyet	F1
Çoklu Sınıflandırma (RF)	Defacement	0,9666	0,9741	0,9789	0,9765
	Benign		0,9614	0,9876	0,9743
	Malware		0,9867	0,9632	0,9748
	Phishing		0,9308	0,9381	0,9344
	Spam		0,9857	0,9647	0,9751
	Ortalama		0,9668	0,9666	0,9666

4.5. Önerilen ESA Tabanlı Yöntem ve Sonuçları

Zararlı URL'lerin tespit edilip sınıflandırılması için derin öğrenme tabanlı bir yaklaşım önerilmiştir. Derin öğrenme mimarisi olarak evrişimsel sinir ağları (ESA) seçilmiştir.

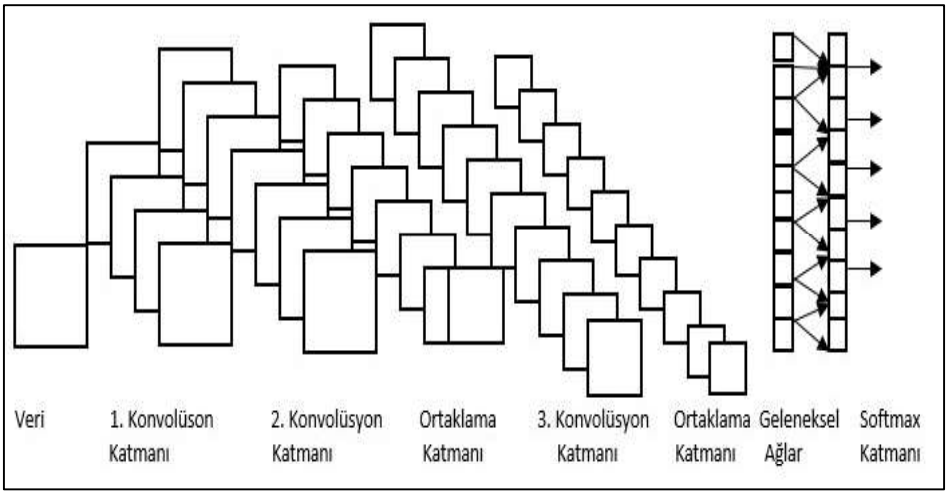
Evrişimsel sinir ağları görüntü işlemede yoğun olarak kullanılmasına rağmen siber güvenlik alanında da yaygın olarak kullanılmaktadır.

Veri kümesi içerisinde bulunan ham veriler derin öğrenme algoritmalarına verilmeden önce önişleme tabi tutulması gerekmektedir. Çünkü veri kümesi içerisinde NaN değerler ve sözcüksel veriler bulunmaktadır. Derin öğrenme algoritmalarının verimli çalışabilmesi için NaN değerler yerine 0 değeri veya aritmetik ortalama değeri verilebilir. Yapılan çalışmada NaN değerler yerine 0 değeri atanmıştır. Veri kümesi içerisinde bulunan 'URL_Type_obf_Type' alanında veriler defacement, benign, phishing, malware ve spam olarak etiketlenmiştir. Bu etiketler önişleme kısmında sayısal değerlere atanmıştır. Sırası ile defacement için 0 değeri, benign için 1 değeri, phishing için 2 değeri, malware için 3 değeri ve spam için 4 değeri atanmıştır. Şekil 4.2'de önişleme adımları gösterilmiştir.



Şekil 4.2. Veri önişleme adımları

İşlenmiş veri Evrişimsel Sinir Ağlarında kullanılmak için hazır hale getirilmiştir. Çalışmaya başlamadan önce veriler eğitim ve test olarak iki parçaya bölünmüştür. Bölme oranı 0.2 olarak belirlenmiştir. Çalışma kapsamında oluşturulan mimaride 3 adet konvolüsyon katmanı ve 2 adet ortaklama katmanı kullanılmıştır. Veri ilk aşamada 8 ve 16 adet filtrelere sahip iki konvolüsyon katmanından geçirildikten sonra maksimum ortaklama özelliğine sahip ortaklama katmanından geçirilmiştir. Daha sonra 32 adet filtreye sahip konvolüsyon katmanı ve max ortalama özelliğine sahip ortaklama katmanından geçirilmiştir. Bu aşamalardan sonra 160 adet düğümlere sahip 2 adet geleneksel yapay sinir ağından geçirilmiştir. Son katmanda softmax kullanılarak sınıflandırılma işlemi yapılmıştır. Epoch sayısı 10 olarak belirlenmiştir. Şekil 4.3'te oluşturulan mimari gösterilmiştir.



Şekil 4.3. ESA mimarisi

Oluşturulan mimaride aktivasyon fonksiyonu olarak Rectified Linear Unit (Relu) kullanılmıştır. Relu fonksiyonu negatif değerler için 0 değerini alan ve pozitif değerler için kendi değerlerini kabul eden bir fonksiyondur. Maksimum ortaklama katmanı için ortaklama boyutu 2x2 seçilmiştir. Geleneksel yapay sinir ağı modelinde düğüm sayısı 160 olarak belirlenmiştir. Optimizasyon algoritması stochastic gradient descent (sgd) seçilmiş ve öğrenme oranı 0,001 olarak belirlenmiştir. Sistem en iyi performansı 0,001 öğrenme oranında vermektedir. Yapılan çalışmada ikili ve çoklu sınıflandırma yapılmıştır.

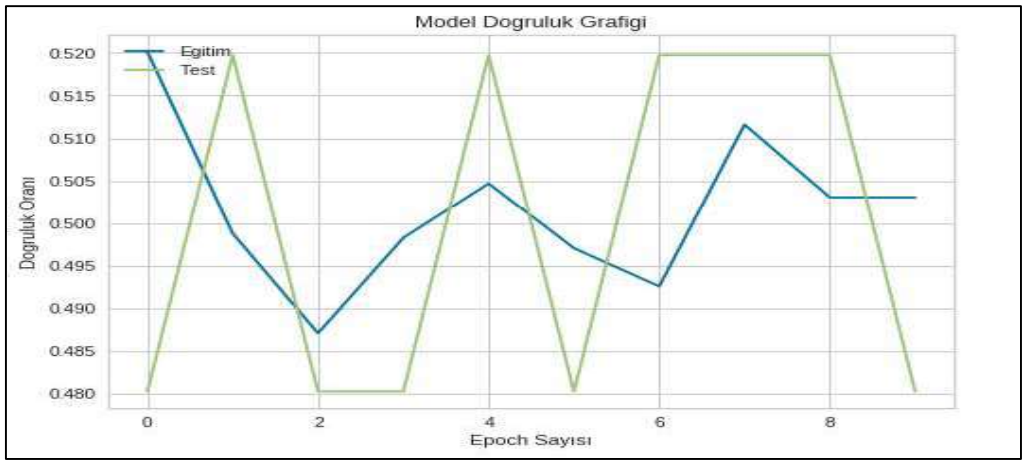
4.5.1. İkili ve çoklu sınıflandırma

Veri kümesinde toplamda 79 özellik ve 1 etiket bulunmaktadır. İkili sınıflandırmanın sonuçları Çizelge 4.7'de gösterilmiştir.

Çizelge 4.7. 79 özellik kullanılarak oluşan ikili sınıflandırma sonuçları

Veri Kümesi	Etiket	ESA			
		Doğruluk	Kesinlik	Hassasiyet	F1 Ölçütü
İkili Sınıflandırma	Defacement	0,9980	0,9980	0,9974	0,9977
	Malware	0,9991	0,9993	1,0000	0,9996
	Phishing	0,5032	0,5033	1,0000	0,6696
	Spam	0,9987	0,9985	1,0000	0,9993

Phishing saldırı tipinde 79 özellik verilerek elde edilen sonuçta doğruluk oranı % 50,32 oranında olmaktadır. 79 özellik kullanılması phishing için sistemde kararsızlık oluşturmaktadır. Phishing sınıflandırması doğruluk – epoch grafiği Şekil 4.4'te gösterilmiştir.



Şekil 4.4. Doğruluk – epoch sayısı grafiği

Ortaya çıkan bu problemin çözülmesi için yeni bir önişleme ihtiyacı duyulmuştur. Bu kapsamda random forest algoritması kullanılarak veri kümesi içerisindeki özellikler ağırlıklarına göre sıralandırılmıştır. Çizelge 4.8'de özellikler ve ağırlıkları gösterilmiştir.

Çizelge 4.8. Özellik ağırlık sırası

Sıra	Özellikler	Ağırlıklar
1	URL_Type_obf_Type	0,309943
2	pathDomainRatio	0,050437
3	tld	0,048161
4	domain_token_count	0,047325
5	SymbolCount_Domain	0,045423
6	domainUrlRatio	0,044519
7	pathurlRatio	0,039719
8	NumberofDotsinURL	0,031381
9	LongestPathTokenLength	0,030221
10	argDomanRatio	0,028586
11	host_letter_count	0,025258
12	domainlength	0,024710
13	path_token_count	0,021469
14	pathLength	0,020543

Çizelge 4.8. (Devam) Özellik ağırlık sırası

Sıra	Özellikler	Ağırlıklar
15	urlLen	0,020020
16	subDirLen	0,019987
17	delimeter_path	0,016664
18	CharacterContinuityRate	0,014389
19	ldl_url	0,010869
20	Entropy_Domain	0,010571
21	charcompvowels	0,009759
22	ldl_path	0,008794
23	Entropy_Filename	0,007258
24	longdomaintokenlen	0,006416
25	URL_Letter_Count	0,006159
26	Extension_LetterCount	0,005796
27	Directory_LetterCount	0,005705
28	avgdomaintokenlen	0,005623
29	SymbolCount_URL	0,005245
30	SymbolCount_Directoryname	0,005241
31	dld_path	0,005019
32	sub-Directory_LongestWordLength	0,004797
33	dld_url	0,004520
34	avgpathtokenlen	0,004302
35	ArgUrlRatio	0,004041
36	URL_DigitCount	0,003104
37	Filename_LetterCount	0,002730
38	charcompace	0,002719
39	NumberRate_URL	0,002676
40	Domain_LongestWordLength	0,002659
41	spcharUrl	0,002590
42	Extension_DigitCount	0,002539
43	fileNameLen	0,00215
44	argPathRatio	0,002095
45	Directory_DigitCount	0,001961
46	ldl_getArg	0,001959
47	delimeter_Domain	0,001664
48	ldl_filename	0,001639
49	Entropy_DirectoryName	0,001479
50	NumberRate_DirectoryName	0,001436
51	NumberRate_FileName	0,001353
52	this,fileExtLen	0,001347
53	File_name_DigitCount	0,001313
54	Entropy_Extension	0,001263
55	SymbolCount_Extension	0,001241
56	Path_LongestWordLength	0,001210
57	Entropy_URL	0,001159
58	ArgLen	0,000968
59	SymbolCount_Afterpath	0,000913
60	dld_getArg	0,000898
61	LongestVariableValue	0,000817

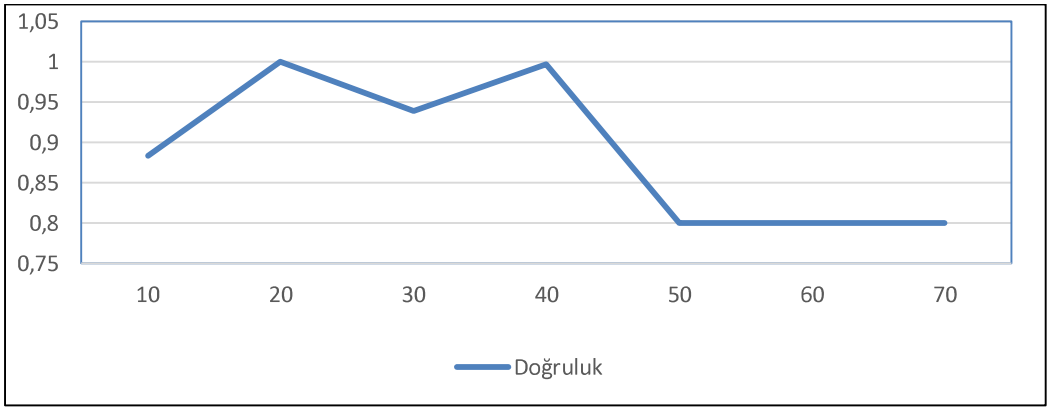
Çizelge 4.8. (Devam) Özellik ağırlık sırası

Sıra	Özellikler	Ağırlıklar
62	SymbolCount_FileName	0,000676
63	Querylength	0,000673
64	NumberRate_AfterPath	0,000587
65	Query_DigitCount	0,000536
66	Arguments_LongestWordLength	0,000526
67	Query_LetterCount	0,000399
68	URLQueries_variable	0,000368
69	NumberRate_Extension	0,000317
70	Entropy_Afterpath	0,000284
71	host_DigitCount	0,000239
72	delimeter_Count	0,000209
73	NumberRate_Domain	0,000149
74	ldl_domain	0,000111
75	dld_filename	0,000096
76	URL_sensitiveWord	0,000075
77	dld_domain	0,000000
78	ISIpAddressInDomainName	0,000000
79	isPortEighty	0,000000
80	executable	0,000000

Çalışmada en iyi performansı bulmak için ilk on, yirmi, otuz, kırk, elli, atmış ve yetmiş özellik ayrı ayrı seçilerek model eğitilmiş ve test edilmiştir.

Seçilen özellikler kullanılarak elde edilen çoklu sınıflandırma sonuçları Çizelge 4.9'da gösterilmiştir.

Yapılan testlerde 50 özellikten sonra sistemde kararsızlık olduğu ve doğruluk oranının %80'i geçemediği tespit edilmiştir. Bu durum Şekil 4.5'te gösterilmiştir.



Şekil 4.5. Seçilen özelliklerin doğruluk oranları

Çizelge 4.9. Seçili özelliklere göre çoklu sınıflandırma sonuçları

Veri Kümesi	Özellik Sayısı	Etiket	ESA			
			Doğruluk	Kesinlik	Hassasiyet	F1 Ölçütü
Çoklu Sınıflandırma	10	Defacement	0,8835	0,3947	0,9405	0,5561
		Benign		0,5547	0,9775	0,7078
		Phishing		0,3211	0,9472	0,4797
		Malware		0,4510	0,9214	0,6056
		Spam		0,5364	0,9048	0,6735
	20	Defacement	0,9976	1,0000	1,0000	1,0000
		Benign		1,0000	1,0000	1,0000
		Phishing		1,0000	0,9992	0,9996
		Malware		0,9987	1,0000	0,9993
		Spam		0,9979	1,0000	0,9989
	30	Defacement	0,9387	0,6068	0,9747	0,7480
		Benign		0,7916	0,9764	0,8744
		Phishing		0,4836	0,9589	0,6430
		Malware		0,6265	0,9418	0,7524
		Spam		0,6820	0,9709	0,8012
	40	Defacement	0,9968	0,9832	1,0000	0,9915
		Benign		0,9981	0,9994	0,9987
		Phishing		0,8865	0,9992	0,9395
		Malware		0,9495	0,9974	0,9729
		Spam		0,9992	0,9954	0,9973
	50	Defacement	0,8000	0,2174	1,0000	0,3571
		Benign		0,2170	1,0000	0,3566
		Phishing		0,1773	1,0000	0,3012
		Malware		0,2065	1,0000	0,3423
		Spam		0,1818	1,0000	0,3077
	60	Defacement	0,8000	0,2136	1,0000	0,3520
		Benign		0,2093	1,0000	0,3462
		Phishing		0,1784	1,0000	0,3028
		Malware		0,2078	1,0000	0,3442
		Spam		0,1908	1,0000	0,3205
	70	Defacement	0,8000	0,2178	1,0000	0,3577
		Benign		0,2093	1,0000	0,3462
		Phishing		0,1749	1,0000	0,2977
		Malware		0,2108	1,0000	0,3483
		Spam		0,1871	1,0000	0,3153

Bu kapsamda veri kümesi içerisindeki hangi özelliğin sistemi kararsızlık durumuna geçirdiğinin tespit edilebilmesi için 40 – 50 arasındaki özellikler sırası ile test edilmiştir. Test sonucunda 44. özellik olan argPathRatio özelliğinin phishing için ikili sınıflandırmada

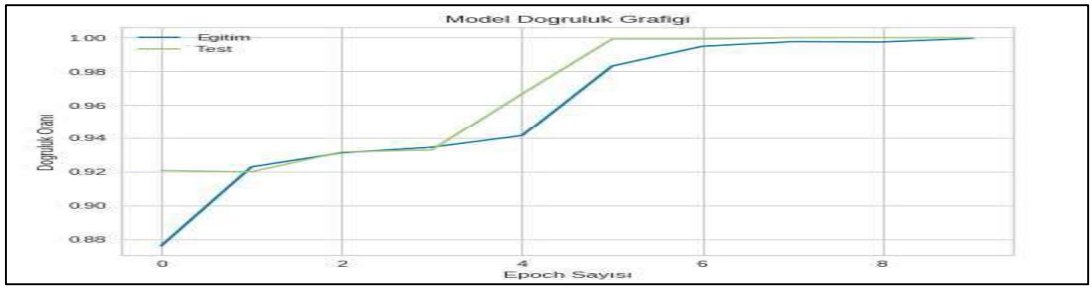
ve çoklu sınıflandırmada sistem başarısını düşürdüğü görülmüştür. argPathRatio özelliği URL'nin argüman ve yol oranı demektir. Bu özellik çıkarılarak 78 özellik ile yapılan test sonucu Çizelge 4.10'da gösterilmiştir.

Çizelge 4.10. 78 özelliğe göre çoklu sınıflandırma sonuçları

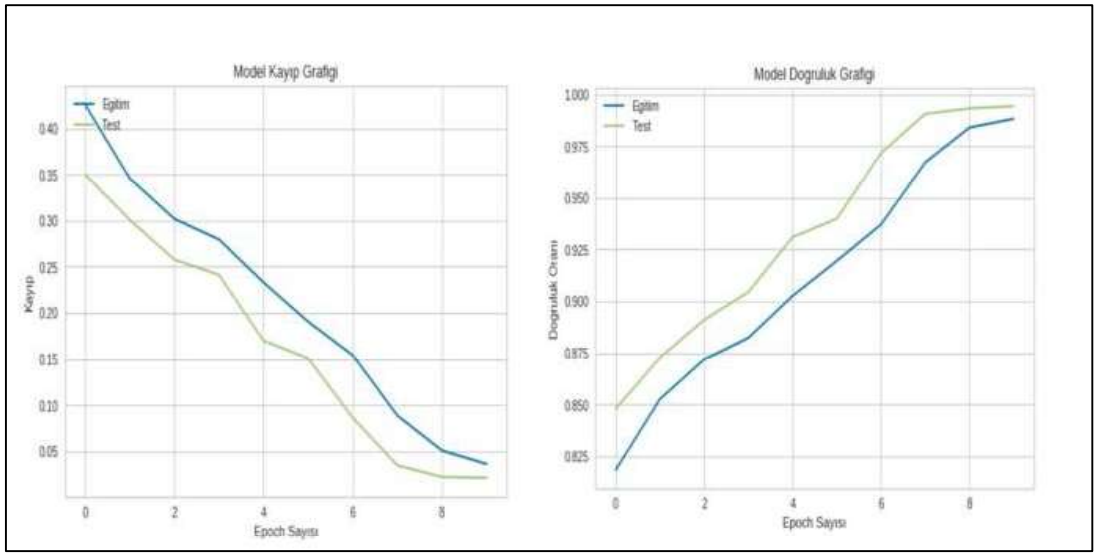
Veri Kümesi	Özellik Sayısı	Etiket	ESA			
			Doğruluk	Kesinlik	Hassasiyet	F1 Ölçütü
Çoklu Sınıflandırma	78	Defacement	0,9746	0,7529	0,9836	0,8529
		Benign		0,8584	0,9874	0,9184
		Phishing		0,7811	0,9763	0,8679
		Malware		0,7748	0,9738	0,8630
		Spam		0,8884	0,9821	0,9329

Yapılan tüm testlerde en iyi performans oranları ilk yirmi özellik seçilerek elde edilmiştir. İlk yirmi özelliğin ağrılık oranları 0,01'den büyüktür. Random-forest algoritması ile özellik seçimi yapılması sonucunda sistem performansı artırılmıştır. Eğitim ve test kümeleri, 0,2 oranında bölünmüştür. Aynı veri kümesinin 10 farklı kopyası alınarak model çalıştırılmıştır ve elde edilen sonuçların ortalaması alınarak doğruluk, kesinlik ve hassasiyet değeri hesaplanmıştır.

20 özellik seçilerek yapılan phishing test sonucunda doğruluk oranı % 99,93, kesinlik % 99,62 ve hassasiyet % 99,97 olarak bulunmuştur. Bu sayede phishing ikili sınıflandırma doğruluk oranı % 50,32 'den % 99,93'e yükseltilmiştir. Çoklu sınıflandırma doğruluk oranı % 80,00'dan % 99,76'e yükseltilmiştir. Şekil 4.5'te phishing ikili sınıflandırması için 20 özellik doğruluk – epoch grafiği gösterilmiştir. Şekil 4.6'da çoklu sınıflandırma için 20 özellik doğruluk – epoch ve kayıp – epoch grafikleri gösterilmiştir.



Şekil 4.6. 20 özellikli doğruluk – epoch grafiği



Şekil 4.7. Seçili 20 özellik ile yapılan test sonuç grafikleri

4.6. Makine Öğrenmesi ve Önerilen ESA Tabanlı Yöntemin Karşılaştırılması

Literatürde yapılmış olan çalışmalar incelendiği zaman elde edilen doğruluk oranlarının %55,00 ve %99,00 arasında değiştiği görülmektedir. Çalışma kapsamında literatürde çokça kullanılması sebebi ile makine öğrenmesi ve önerilen yöntem olarak da derin öğrenme kullanılmıştır.

Makine öğrenmesinde KNN, SVM, DT, ve RF olmak üzere toplamda 4 tane algoritma kullanılmıştır. Bu algoritmalar içerisinde en iyi doğruluk ve performans değerini RF algoritması vermiştir. İkili sınıflandırmada minimum % 98,16 ve maksimum % 99,83 oranlarında doğruluk elde edilmiştir. Çoklu sınıflandırmada ise %96,66 oranında doğruluk elde edilmiştir.

Derin öğrenme algoritması olarak da ESA kullanılmıştır. Yapılan çalışma sonucunda ikili sınıflandırma için doğruluk % 99,93 oranında, çoklu sınıflandırma doğruluk oranı % 99,76 oranında elde edilmiştir. ESA ile daha başarılı sonuçların elde edildiği uygulama ile gösterilmiştir.

5. SONUÇ VE ÖNERİLER

Teknolojinin gelişmesiyle birlikte verinin çeşitlenmesi, büyümesi, hızının artması gibi nedenlerden dolayı siber güvenlik ihlallerini tespit etmek için gelişmiş sistemler tasarlamak zorlaşmıştır. Bu durum geleneksel güvenlik yönetimi teknolojilerinin sınırlamaları, yeni güvenlik tehditlerinin her geçen gün büyümesi, yeni bilgi teknolojilerinin hızlı değişmesi gibi nedenlerle giderek ciddi bir hal almaktadır. Bu saldırı tekniklerinin çoğu, kötü niyetli URL'lerin yayılması yoluyla gerçekleştirilmektedir. Kötü niyetli URL'leri zamanında tespit etmek için gelişmiş teknikler tasarlamak zor olsa da zorunlu hale gelmiştir.

Kötü niyetli URL'leri tespit etmenin en yaygın yöntemi kara liste yöntemidir. Bu yöntem basit bir sorgulama ile kullanılabilir. Bu sebeple de oldukça hızlı ve uygulanması çok kolaydır. Ancak reaktif bir yöntem olması sebebiyle ayrıntılı kötü niyetli URL listesi tutmak neredeyse imkânsızdır. Bu listeler güncel olmadığı için çevrimiçi kullanıcıların zamanında korunmasını sağlamada yetersiz kalmaktadır.

Bir diğer yöntem olan içerik tabanlı analiz hizmeti, web sayfasının içeriğini indirmekte ve zararlı içerik için analiz etmektedir. Bu yaklaşım kara listeye göre daha yüksek doğruluk sağlayabilir, ancak genel olarak çok daha fazla çalışma süresi gerektirir. Analiz için içeriğin indirilmesi zaman almakta ve bant genişliği tüketmektedir. Bununla birlikte, içerik tabanlı analiz yöntemleri, çok sayıda URL için ve yeni URL'lerin yaratılma hızı için pratik bir çözüm değildir.

Yukarıdaki nedenlerden dolayı proaktif bir çözüme ihtiyaç duyulmaktadır. Bu tez kapsamında derin öğrenme kullanan bir yöntem önerilmiştir. Aynı zamanda literatürde çok sayıda örneği olan makine öğrenmesi yöntemi de kullanılmıştır. Her iki yöntem için de ikili ve çoklu sınıflandırma yapılmıştır. Sonuçlar kendi içlerinde karşılaştırılmıştır. Alınan sonuçlar makine öğrenmesi algoritmaları ile karşılaştırıldığında önerilen yöntemden elde edilen sonuçların daha başarılı olduğu görülmüştür.

Çalışma kapmasında derin öğrenme algoritması olarak da ESA kullanılmıştır. İkili sınıflandırmada minimum %50,00 ve maksimum %99,91 oranlarında doğruluk elde edilmiştir. Çoklu sınıflandırma da ise %80,00 oranında bir doğruluk elde edilmiştir. Elde

edilen %50,00'lik doğruluk phishing etiketinden kaynaklanmaktadır. Bu durum çoklu sınıflandırmanın da doğruluk oranını düşürmektedir. Özelliklerin hepsinin kullanılması phishing için sistemde kararsızlık oluşturmaktadır. Ortaya çıkan bu problemin çözülmesi için yeni bir önişleme ihtiyacı duyulmuştur. Bu kapsamda random forest algoritması kullanılarak veri kümesi içerisindeki özellikler ağırlıklarına göre sıralandırılmıştır. Çalışmada en iyi performansı bulmak için ilk on, yirmi, otuz, kırk, elli, atmış ve yetmiş özellik ayrı ayrı seçilerek model eğitilmiş ve test edilmiştir. Yapılan testlerde 50 özellikten sonra sistemde kararsızlık olduğu ve doğruluk oranının %80'i geçemediği tespit edilmiştir. Bu kapsamda veri kümesi içerisindeki hangi özelliğin sistemi kararsızlık durumuna geçirdiğinin tespit edilebilmesi için 40 – 50 arasındaki özellikler sırası ile test edilmiştir. Test sonucunda 44. özellik olan argPathRatio özelliğinin phishing için ikili sınıflandırmada ve çoklu sınıflandırmada sistem başarısını düşürdüğü görülmüştür. Yapılan tüm testlerde en iyi performans oranları ilk yirmi özellik seçilerek elde edilmiştir. Özellik seçimi yapılması sonucunda sistem performansı artmıştır. Aynı veri kümesinin 10 farklı kopyası alınarak model çalıştırılmıştır ve elde edilen sonuçların ortalaması alınarak doğruluk, kesinlik ve hassasiyet değeri hesaplanmıştır. 20 özellik seçilerek yapılan phishing test sonucunda doğruluk oranı % 99,93, kesinlik % 99,62 ve hassasiyet % 99,97 olarak bulunmuştur. Bu sayede phishing ikili sınıflandırma doğruluk oranı % 50,32 'den % 99,93'e yükseltilmiştir. Çoklu sınıflandırma doğruluk oranı % 80,00'dan % 99,76'e yükseltilmiştir.

Önerilen çalışma, ağ üzerinde güvenlik duvarının arkasına yerleştirilebilir. Kullanıcı tarafından gelen URL istekleri ilk olarak önerilen sistem üzerinde analiz edilir. Eğer sistem URL için normal dışında bir sınıflandırma yapar ise kullanıcıya bilgilendirme yaparak kullanıcının farkındalığının ve güvenliğinin artırılması sağlanır. Sistem başarı oranının çok yüksek olması sayesinde güvenlik duvarı ile entegrasyonu sağlanarak saldırı sınıfına giren URL'lerin engellenmesi sağlanmış olur.

Önerilen çalışmada derin öğrenme algoritmalarından olan ESA kullanılmıştır. Daha sonraki yapılacak çalışmalarda derin öğrenmenin farklı algoritmaları kullanılabilir. Aynı zamanda yapmış olduğumuz çalışma kapsamında veriseti olarak sözcüksel analiz kullanılmıştır. Yapılacak sonraki çalışmalarda host tabanlı özellikler ile sözcüksel özelliklerin birleşiminden oluşan hibrit veri kümesi derin öğrenme algoritmaları ile gerçeklenebilir.

KAYNAKLAR

1. İnternet: Symantec, 2016 Internet Security Threat Report. URL: https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf, Son Erişim Tarihi: 12.07.2019.

2. Huang, D., Xu, K., and Pei, J. (2014). Malicious URL detection by dynamically mining patterns without pre-defined elements. *World Wide Web*, 17(6), 1375-1394.

3. Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. (2009). *Beyond blacklists: learning to detect malicious web sites from suspicious URLs*. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. San Diego.

4. Milletary, J. (2005). Technical trends in phishing attacks. *US-CERT Report*,1-17.

5. İnternet: Spam Nedir. URL: http://web.deu.edu.tr/sss/spam.html, Son Erişim Tarihi: 10.07.2019.

6. Tekerek, A., Gemci, C., ve Faruk Bay, Ö. (2016). Web tabanlı saldırı önleme sistemi tasarımı ve gerçekleştirilmesi: yeni bir hibrit model. *Journal of the Faculty of Engineering and Architecture of Gazi University*, 31(3), 645-653.

7. Garnaeva, M., Wiel, J., Makrushin, D., Ivanov, A., and Namestnikov, Y. (2015). Kaspersky Security Bulletin 2015*, Kaspersky Security Bulletin Report,* 1-15.

8. İnternet: McCall, T. (2007). Gartner survey shows phishing attacks escalated in 2007. URL: http://www.gartner.com/it/page.jsp?id=565125, Son Erişim Tarihi: 02.05.2015.

9. Hoi, S. C., Wang, J., and Zhao, P. (2014). Libol: A library for online learning algorithms. *The Journal of Machine Learning Research*, 15(1), 495-499.

10. İnternet: Sahoo, D., Liu, C., and Hoi, S. C. (2017). Malicious URL detection using machine learning: a survey. URL: https://arxiv.org/pdf/1701.07179.pdf, Son Erişim Tarihi: 12.06.2019.

11. Blum, A., Wardman, B., Solorio, T., and Warner, G. (2010). *Lexical feature based phishing URL detection using online learning.* In Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*,* 54-60.

12. Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. (2009). *Beyond blacklists: learning to detect malicious web sites from suspicious URLs.* In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 1245-1254, Paris.

13. Selvaganapathy, S., Nivaashini, M., & Natarajan, H. (2018). Deep belief network based detection and categorization of malicious URLs. *Information Security Journal: A Global Perspective*, 27(3), 145-161.

14. Wang, Y., Cai, W. D., and Wei, P. C. (2016). A deep learning approach for detecting malicious JavaScript code. *Security and Communication Networks*, 9(11), 1520-1534.

15. Ma, J., Saul, L. K., Savage, S., and Voelker, G. M. (2009). *Identifying suspicious URLs: an application of large-scale online learning*. In Proceedings of the 26th annual international conference on machine learning, 681-688, Montreal.

16. Choi, H., Zhu, B. B., and Lee, H. (2011). *Detecting Malicious Web Links and Identifying Their Attack Types*. WebApps'11:2nd USENIX Conference on Web Application Development, 11(11), 218, Portland.

17. Lin, M. S., Chiu, C. Y., Lee, Y. J., and Pao, H. K. (2013). *Malicious URL filtering—A big data application*. In 2013 IEEE International Conference on Big Data, 589-596, Santa Clara.

18. Chu, W., Zhu, B. B., Xue, F., Guan, X., and Cai, Z. (2013). *Protect sensitive sites from phishing attacks using features extractable from inaccessible phishing URLs*. In 2013 IEEE International Conference on Communications, 1990-1994, Budapest.

19. Mamun, M. S. I., Rathore, M. A., Lashkari, A. H., Stakhanova, N., and Ghorbani, A. A. (2016). *Detecting malicious urls using lexical analysis*. In International Conference on Network and System Security, 467-482, Taipei.

20. Vanhoenshoven, F., Nápoles, G., Falcon, R., Vanhoof, K., and Köppen, M. (2016). *Detecting malicious URLs using machine learning techniques*. IEEE Symposium Series on Computational Intelligence, 1-8, Athens.

21. Darling, M., Heileman, G., Gressel, G., Ashok, A., and Poornachandran, P. (2015). *A lexical approach for classifying malicious URLs*. In 2015 international conference on high performance computing and simulation, 195-202, Amsterdam

22. Pannu, A. (2015) Artificial intelligence and its application in different areas. *International Journal of Engineering and Innovative Technology*, 4(10), 79-84.

23. Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236), 433.

24. Internet: Copeland, M. (2016). What's the difference between artificial intelligence, machine learning, and deep learning. URL: https://blogs.nvidia.com/blog/2016/07/29/whats-differenceartificial-intelligence-machinelearning-deep-learning-ai, Son Erişim Tarihi: 06.05.2019.

25. Robert. C. (2014). Machine learning, a probabilistic perspective. *Chance*, 62-63.

26. Türkmenoğlu, C. (2016). *Türkçe metinlerde duygu analizi*, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 1-75.

27. Alpaydin, E. (2004). *Introduction to machine learning*. Cambridge: The MIT Press, 1-45.

28. Öztemel E. (2003). *Yapay Sinir Ağları*. İstanbul: Papatya Yayıncılık, 82-118.

29. Shearer, C. (2000). *The CRISP-DM model: the new blueprint for data mining*. Journal of Data Warehousing, 5(4), 13-22.

30. Koyuncugil, A., ve Özgülbaş, N. (2009). Veri madenciliği: Tıp ve sağlık hizmetlerinde kullanımı ve uygulamaları. *Bilişim Teknolojileri Dergisi*, 2(2), 1-12.

31. Çoban, O. (2016). *Metin sınıflandırma teknikleri ile türkçe twitter duygu analizi*, Yüksek Lisans Tezi, Atatürk Üniversitesi Fen Bilimleri Enstitüsü, Erzurum, 1-119.

32. Ardıl, E. (2009). *Esnek hesaplama yaklaşımı ile yazılım hata kestrimi*, Yüksek Lisans Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü, Edirne, 1- 86.

33. Kartal, E. (2015). *Sınıflandırmaya Dayalı Makine Öğrenmesi Teknikleri Ve Kardiyolojik Risk Değerlendirmesine İlişkin Bir Uygulama*, Doktora Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 1–150.

34. Cinsdiki, M. (1997). *Neural Network Solutions for ATM Routing & Multicasting Problems*, Yüksek Lisans Tezi, Ege Üniversitesi Fen Bilimleri Enstitisü, İzmir, 1-23

35. Sağıroğlu, Ş., Beşdok, E. ve Erler, M. (2003). *Mühendislikte yapay zekâ uygulamaları*. Kayseri: Ufuk Kitabevi, 1-15.

36. Burak, T. (2018). *Ankara İli Doğal Gaz Tüketiminin Yapay Sinir Ağları İle Öngörüsü*, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Enerji Enstitüsü, İstanbul, 1-89.

37. İnternet: Akcayol, M. (2018). Derin Öğrenme. URL: http://w3.gazi.edu.tr/~akcayol/DL_L4ANN.pdf, Son Erişim Tarihi: 06.08.2019

38. Anderson, D., and McNeill, G. (1992). Artificial neural networks technology, *Kaman Sciences Corporation*, 258(6), 1-83.

39. Demuth, H. and Beale, M. (2004). Neural Network Toolbox User's Guide, *Neural Network Toolbox*, 1-7. Kamruzzman, J., Beggand, R. K. and Sarker, R. A. (2006). *Artificial Neural Networks in Finance and Manufactoring*. London: İdea Group Publishing, 4.

40. Hebb, D. O. (1949). *The first stage of perception: growth of the assembly*. New York: Psychology Press, 1-365.

41. Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8), 2554-2558.

42. Yurtoğlu, H. (2005). *Yapay Sinir Ağları Metodolojisi ile Öngörü Modellemesi: Bazı Makroekonomik Değişkenler İçin Türkiye Örneği*, Uzmanlık Tezi, Devlet Planlama Teşkilatı, Ankara, 1-110.

43. Fırat, M., ve Güngör, M. (2004). Askı madde konsantrasyonu ve miktarının yapay sinir ağları ile belirlenmesi. *İMO Teknik Dergi*, 15(73), 3267-3282.

44. Erilli, N. A., Eğrioğlu, E., Yolcu, U., Aladağ, Ç. H., ve Uslu, V. R. (2010). Türkiye'de enflasyonun ileri ve geri beslemeli yapay sinir ağlarının melez yaklaşımı ile öngörüsü. *Doğuş Üniversitesi Dergisi*, 11(1), 42-55.

45. Öztemel E. (2006). *Yapay Sinir Ağları*. İstanbul: Papatya Yayıncılık, 23-45.

46. Kim, K.G. (2016). Deep Learning Book Review. *Healthcare Informatics Research*, 22(4), 351–354.

47. Gu, X., Zhang, H., Zhang, D., and Kim, S. (2016). *Deep API learning*. In Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, 631-642, Seattle.

48. Deng, L., and Yu, D. (2014). *Deep learning: methods and applications*. Foundations and Trends in Signal Processing, 197-387, Boston.

49. Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., and Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data*, 2(1), 1.

50. Wang, S. C.(2003). *Artificial neural network*. Boston: Springer, 88-100.

51. Alsugair, A. M. and Al-Qudrah, A. A. (1998). Artificial neural network approach for pavement maintenance. *Journal of Computing in Civil Engineering*, 12(4), 249–255.

52. İnternet: Sarle, W. S. (1997). Neural networks frequently asked questions. URL: francky.me/aifaq/FAQ-comp.ai.neural-net.pdf, Son Erişim Tarihi: 13.02.2019.

53. Song, H. A. and Lee, S. Y. (2013). *Hierarchical Representation Using NMF*, in International Conference on Neural Information Processing, 466–473, Daegu.

54. Seker, A., Diri, B., ve Balık, H.H. (2017). Derin Ögrenme Yöntemleri ve Uygulamaları Hakkında Bir İnceleme. *Gazi Mühendislik Bilimleri Dergisi*, 3(3), 47–64.

55. Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep learning. Cambridge: The MIT Press, 161-217.

56. İnternet: Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., and Asari, V. K. (2018). The history began from AlexNet: a comprehensive survey on deep learning approaches. URL: https://arxiv.org/abs/1803.01164, Son Erişim Tarihi: 23.05.2019.

57. McCulloch, W. S., and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4), 115-133.

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

59. Minsky, M., and Papert, S. A. (1969). *Perceptrons*. Cambridge: The MIT Press, 8-25.

60. Ackley, D. H., Hinton, G. E. and Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science* ,9(1), 147-169.

61. Kunihiko, F. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks,* 1(2), 119-130.

62. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE,* 86(11), 2278-2324.

63. Hinton, G. E., Simon O., and Yee-Whye T. (2006). A fast learning algorithm for deep belief nets. *Neural Computation,* 18(7), 1527-1554.

64. Hinton, G. E., and Ruslan R. S. (2006). Reducing the dimensionality of data with neural networks. *Science,* 313(5786), 504-507.

65. İnternet: Motivasyon, Yapay Zeka ve Derin Öğrenme. URL: https://medium.com/deep-learning-turkiye/motivasyon-yapay-zeka-ve-derin-%C3%B6%C4%9Frenme-48d09355388d, Son Erişim Tarihi: 17.03.2019.

66. İnternet: Project 6 Deep Learning. URL: https://www.cc.gatech.edu/~hays/compvision/proj6/, Son Erişim Tarihi: 25.04.2019.

67. Kurt, F. (2018). *Evrişimli Sinir Ağlarında Hiper Parametrelerin Etkisinin İncelenmesi*, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 1-95.

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: SARICAOĞLU, Cemile
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 11.06.1990, Trabzon
Medeni hali	: Bekar
Telefon	: 0 (553) 187 50 10
e-mail	: saricaoglucemile@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi/Bilgisayar Mühendisliği	Devam ediyor
Lisans	Esk. Osmangazi Üniversitesi/Bilgisayar Mühendisliği	2014
Lise	Kırıkkale Fen Lisesi	2008

İş Deneyimi

Yıl	Yer	Görev
2015-Halen	TÜBİTAK BİLGEM Kamu Sertifikasyon Merkezi	Araştırmacı

Yabancı Dil

İngilizce

Yayınlar

1. Demirci, M., Sarıcaoğlu, C. (2019). *Siber Tehdit İstihbaratı Alanında Makine Öğrenmesi Algoritmalarının Kullanılması*, IDSES'19 International Data Science and amp; Engineering Symposium, Karabük.

Hobiler

Yüzme, Gezme, Kitap Okuma

GAZİ GELECEKTİR...