
GAZİ ÜNİVERSİTESİ
Gazi
1926

IOT'DE MEYDANA GELEN DDOS SALDIRILARININ DERİN ÖĞRENME YÖNTEMLERİ KULLANILARAK BÜYÜK VERİ ORTAMINDA ANALİZİ VE TESPİTİ

Sami YARAŞ

YÜKSEK LİSANS TEZİ
BİLGİ GÜVENLİĞİ MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OCAK 2024

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.

Sami YARAŞ
22/01/2024

IOT'DE MEYDANA GELEN DDOS SALDIRILARININ DERİN ÖĞRENME YÖNTEMLERİ KULLANILARAK BÜYÜK VERİ ORTAMINDA ANALİZİ VE TESPİTİ

(Yüksek Lisans Tezi)

Sami YARAŞ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Ocak 2024

ÖZET

IoT'de kurulan ağların karşılaşabileceği en büyük tehlikelerin başında kötücül saldırılar gelmektedir. Bu saldırılardan en çok karşılaşılan saldırı ise DDoS ataklarıdır. Saldırılar sonrasında ağın haberleşme trafiği engellenebilir, sensör düğümlerin enerjisi hızlıca azabilir. Bu yüzden, oluşan saldırıların tespit edilmesi büyük önem taşımaktadır. Kurulan ağ içerisinde birçok sensör düğüm olduğu düşünüldüğünde oluşan ağ trafik datası geleneksel yöntemlerle analiz edilmesi imkânsız hale gelebilmektedir. Ağdaki bu trafiğin büyük veri ortamında analiz edilmesi gerekmektedir. Bu çalışmada, elde edilen ağ trafik veri setinin büyük veri ortamında analiz edilmesi ve oluşturulan derin öğrenme algoritması ile ağda meydana gelen saldırıların tespit edilmesi amaçlanmaktadır. Bu çalışma, Google Colaboratory (Colab) ortamında PySpark ile Apache Spark kullanılarak gerçekleştirilmiştir. Çalışmada Keras ve Scikit-Learn kütüphaneleri kullanılmıştır. Modelin eğitiminde ve test edilmesinde 'CICIoT2023' ve 'TON_IoT' verisetleri kullanılmıştır. Verisetlerindeki özellikler korelasyon yöntemi kullanılarak azaltılmış, önemli özelliklerin testlere dahil edilmesi sağlanmıştır. Bir boyutlu CNN ve LSTM kullanılarak oluşturulan hibrit bir Derin Öğrenme Algoritması tasarlanmıştır. Geliştirilen yöntem, 10 farklı makine öğrenmesi ve derin öğrenme algoritmalarıyla karşılaştırılmıştır. Geliştirilen model Doğruluk, Kesinlik, Geri çağırma ve F1 parametreleri kullanılarak değerlendirilmiştir. Yapılan çalışma sonrasında, 'CICIoT2023' veri setinde binary sınıflandırma olarak %99,995 ve çoklu sınıflandırma olarak ise %99,96 doğruluk oranına ulaşılmıştır. 'TON_IoT' veri setinde ise binary sınıflandırma başarısında %98,75'e ulaşılmıştır.

Bilim Kodu	:	92403
Anahtar Kelimeler	:	Büyük Veri, Anamoli Tespit Sistemleri, Derin Öğrenme, Makine Öğrenimi, IoT, DDoS, CICIoT2023, TON_IOT
Sayfa Adedi	:	85
Danışman	:	Prof. Dr. Murat DENER

ANALYSIS AND DETECTION OF DDOS ATTACKS IN IOT BY USING DEEP LEARNING METHODS IN BIG DATA ENVIRONMENT

(M. Sc. Thesis)

Sami YARAŞ

GAZI UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

January 2024

ABSTRACT

The most significant threat that networks established in IoT may encounter is malicious attacks. The most commonly encountered attack among these threats is DDoS attacks. After attacks, the communication traffic of the network can be disrupted and the energy of sensor nodes can quickly deplete. Therefore, the detection of occurring attacks is of great importance. Considering that there are numerous sensor nodes in the established network, analyzing the network traffic data through traditional methods can become impossible. Analyzing this network traffic in a big data environment is necessary. In this study, the aim is to analyze the obtained network traffic dataset in a big data environment and detect attacks in the network using a deep learning algorithm. This study is conducted using PySpark with Apache Spark in the Google Colaboratory (Colab) environment. Keras and Scikit-Learn libraries are utilized in the study. CICIoT2023 and TON_IOT datasets are used for training and testing the model. The features in the datasets are reduced using the correlation method, ensuring the inclusion of significant features in the tests. A hybrid Deep Learning Algorithm is designed using one-dimensional CNN and LSTM. The developed method was compared with 10 different machine learning and deep learning algorithms. The model's performance is evaluated using Accuracy, Precision, Recall, and F1 parameters. Following the study, a accuracy rate of 99.995% for binary classification and 99.96% for multi-classification is achieved in the CICIoT2023 dataset. In the TON_IOT dataset, a binary classification success rate of 98.75% is reached.

Science Code	:	92403
Key Words	:	Big Data, Intrusion Detection Systems, Deep Learning, Machine Learning, IoT, DDoS, CICIoT2023, TON_IOT
Page Number	:	85
Supervisor	:	Prof. Dr. Murat DENER

TEŞEKKÜR

Benim yetişmemde büyük emek ve sabır gösteren her daim yanımda olan ve haklarını ödeyemeceğim annem Hatice YARAŞ ve babam Mehmet YARAŞ teşekkürlerimi sunuyorum. Tez çalışmamda bana destek olan ve sabır gösteren hayat arkadaşım Merve YARAŞ'a teşekkür ederim. Tez çalışmamda ve bitirmemde beni teşvik eden ve yönlendiren danışman hocam Prof. Dr. Murat DENER'e teşekkür ederim.

Kızım Asya'ya...

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

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

Kısaltmalar	Açıklamalar
Ack	Acknowledgement
ADA	AdaBoost
ADC	Analog to Digital Converter
ANN	Artificial neural network
AUC	Area under the curve
CNN	Convolutional Neural Network
DoS	Denial of Service
DDoS	Distributed Denial of Service
DNN	Deep neural networks
DT	Decision Tree
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GB	Gradient Boost
GRU	Gated Recurrent Unit
GPU	Graphics Processing Unit
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IoT	Internet of Things
KNN	K-Nearest Neighbors
KSA	Kablosuz Sensör Ağları
LR	Logistic Regression
LSTM	Long Short Term Memory
MITM	Man In The Middle
MLP	Multilayer Perceptron
NB	Naive Bayes
PCA	Principal Component Analysis

Kısaltmalar	Açıklamalar
PCC	Pearson Correlation Coefficient
ReLu	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
SMO	Sequential Minimal Optimization
SVM	Support Vector Machine
Syn	Synchronization
TCP	Transmission Control Protocol
TN	True Negative
TP	True Positive
TPR	True Pozitif Rate
TPU	Tensor Processing Unit
UDP	User Datagram Protocol
WSN	Wireless Sensor Network

1. GİRİŞ

Kablosuz sensör ağlar, gerçek dünya ile dijital dünya arasında bir köprü işlevi görmektedir. Gerçek dünyayı algılamak ve bu verileri dijital dünyaya göndermek için sensörler birbirlerine bağlanarak oluşturdukları ağa genel olarak kablosuz sensör ağlar adı verilir. Kablosuz Algılayıcı Ağlar (KSA), askeri gözetleme, savaş alanları, orman yangını takibi, bina güvenliği izleme ve sağlık hizmetleri gibi çok çeşitli alanlarda gerçek zamanlı veri akışı sağlar. Kablosuz veri ağları, IoT'nin bir parçasıdır ve toplanan veriler; işlenir, analiz edilir ve baz istasyonu yardımıyla kullanıcıya sunulurlar. KSA'larda ağ ile dış dünya arasında bağlantı görevi gören en az bir tane geçit düğümü vardır [1].

Bu sensörler ile geçit düğümü arasındaki iletişim enerji bakımında en verimli yol ile yapılmalıdır; çünkü sensör düğümlerinin enerjileri kısıtlıdır ve bataryalarının yeniden dolum imkânı yoktur. Kablosuz sensör ağlarının karakteristik özellikleri sebebiyle basit, verimli ve farklı senaryolara kolayca adapte edilebilmelidir. Bunun sonucu olarak kullanabilecekleri kaynaklar sınırlıdır. Düşük güç kullanımı, işlemci sınırı, bazı cihazların maliyet sebebiyle eklenememesi sebebiyle kablosuz sensör ağlar saldırılara açık haldedir. Bu sistemlerde güvenlik, gizlilik önemli konulardandır. Bunları geliştirmek, yeni fonksiyonlar eklemek enerji kısıtı, işlemci sınırı gibi engeller ile baş etmeyi gerektirecektir. Kriptografi gibi geleneksel güvenlik önlemlerini bu tür ağlara uygulanması çok zordur; çünkü KSA'lar, açık ve dağıtılmış yapıları ve sensör düğümlerinin sınırlı kaynakları nedeniyle saldırılara karşı oldukça savunmasızdır. Ayrıca, KSA paketlerinde yayının sık sık yapılması gerekebilir, sensör düğümleri bir ortamda rastgele konuşlandırılabilir, böylece kötücül bir saldırı KSA'ya kolayca enjekte edilebilir [2].

Saldırgan bir sensör, ağı tehlikeye atabilir, mesajları gizlice dinleyebilir, sahte mesajlar enjekte edebilir, verilerin bütünlüğünü değiştirebilir ve ağ kaynaklarını boşa harcayabilir. Hizmet Reddi (DoS) saldırısı, KSA güvenliğini tehdit eden en genel ve tehlikeli saldırılardan biri olarak kabul edilir. DoS saldırıları, günümüzde de ana zorluklardan biridir. Bu saldırının çeşitli biçimleri vardır ve asıl amacı KSA'lar tarafından sağlanan hizmetleri kesintiye uğratmak veya askıya almaktır [2]. DoS saldırılarının yıkıcı etkisi, düğümlerin güç kaynaklarını tüketmesi ve kullanım ömürlerini önemli ölçüde kısaltmasıdır. Böylece DoS saldırıları nedeniyle sensörler oldukça hızlı ölebilir. Gücü

tükenen düğümler işe yaramaz hale gelir; dolayısıyla WSN'nin kullanım amacı tehlikeye girer.

DoS saldırlarının günümüzde oldukça tehlikeli ve dikkate alınması gereken bir saldırı türü olmasının sebebi ise modern teknolojinin veri üzerine kurulu olmasından kaynaklanmaktadır. Veri, işlenmemiş bilgi olarak bilinir ve işlendikten sonra anlamlı hale gelmektedir. Bilgisayar çağının gelmesiyle kullanılan veri miktarı ciddi oranda artmıştır. Ağ trafiğinin, sistem olaylarının ve sistem elemanlarının bıraktığı loglar büyük veri kapsamına girebilmektedir. Büyük veri analitiği ve ilgili teknolojileri sayesinde, veri akışları sürekli izlenebilmekte ve ağda meydana gelebilecek anormallikler ve değişiklikler tespit edilerek ağ güvenliği sağlanabilmektedir. Büyük veri ve yapay zekâ algoritmaları birlikte çalışarak ağ durumunun geçmiş ve şimdiki verilerinin analizi yapılabilir ve anlık ağ trafiğinin normal mi yoksa saldırı mı olduğu belirlenebilir [3].

Saldırılara karşı kısıtlı kaynaklarından dolayı zayıf olan ağlarda, bu saldırıların meydana geldiği anda anlayıp, ilgili sensör düğümü uyarabilen sistemler mevcuttur. Bu sistemlere Saldırı Tespit Sistemi (IDS) adı verilmektedir. IDS, izinsiz girişleri, saldırıları veya güvenlik politikalarının ihlallerini zamanında tespit etmek ve sınıflandırmak için kullanılan proaktif bir saldırı tespit aracıdır [4]. Bu saldırı sistemleri, sensör düğümlerinin kısıtlı kaynaklarından dolayı, yüksek doğruluğa sahip olmalıdır. Ayrıca ağın kaynak tüketimi üzerine ek bir yük getirmemelidir.

Bu çalışmada hibrit bir derin öğrenme algoritması ile büyük veri ortamında yeni bir saldırı tespit sistemi geliştirilmiştir. Algoritma Google Colabs ortamı kullanılarak, Apache Spark'ın Python desteği olan Pyspark ile yazılmıştır. Apache Spark'ın kullanılmasının sebebi hızlı olması ve algoritmada kullanılan verilerin büyük veri kapsamına girmesidir. Bu saldırı tespit algoritması CICIoT2023 ve TON_IOT verisetleri kullanılarak eğitilmiş ve test edilmiştir. Geliştirilen sistem hem binary olarak hem de çoklu sınıf olarak değerlendirilmiştir. Değerlendirme parametresi olarak doğruluk (accuracy), kesinlik (precision), geri çağırma (recall) ve F1-Skor kullanılmıştır. Geliştirilen model, 10 farklı geleneksel makine ve derin öğrenme algortimasıyla (Random Forest, Decision Tree, Gradient Boost, Ada Boost, Naive Bayes, Logistic Regression, K-Nearest Neighbour, CNN, MLP ve LSTM) karşılaştırılmıştır.

Yapılan çalışmanın literatüre katkıları şu şekildedir:

- Büyük veri ortamında, hibrit derin öğrenme algoritması ile yeni bir IDS geliştirilerek bunun verimli şekilde çalıştığı gösterilmiştir.
- Geliştirilen algoritma hem çoklu sınıf hem de binary olarak test edilmesi sağlanmıştır. İki durumda da yüksek doğruluk oranına ulaşılmıştır.
- Geliştirilen hibrit algoritma, 10 farklı popüler makine ve derin öğrenme algoritması ile karşılaştırılmıştır. Elde edilen sonuçlar, hibrit yöntemin geleneksel yöntemlerden daha iyi doğruluğa sahip olduğu göstermiştir.
- Derin öğrenme algoritmalardan CNN ve LSTM'in tek başlarına test edilmesi sağlanmıştır. CNN ve LSTM kullanılarak oluşturulan hibrit algoritmanın bunların ayrı ayrı kullanılmasından daha iyi sonucu varıldığı gözlemlenmiştir.
- CICIoT2023 gibi büyük ve değerlerin dengesiz dağılım gösterdiği bir veri setinde herhangi bir dengeleme yöntemi kullanılmadan yüksek doğruluk oranına ulaşılmıştır.
- Çalışmaya ikinci bir veri seti eklenerek farklı veri kümesinde de yüksek saldırı tespit oranına ulaşıldığı gözlemlenmiştir.

Çalışma yedi bölümden oluşmaktadır. Bölüm 2, çalışmanın kavramsal çerçevesini anlatır. Bölüm 3, litaratür taramasını sunmaktadır. Bölüm 4, kullanılan araçları ve verisetlerini açıklar. Bölüm 5, kullanılan derin öğrenme algoritmalarını tanımlar. Bölüm 6, geliştirilen algoritmayı açıklar. Modelin değerlendirme sonuçları ve bunların karşılaştırılması ise Bölüm 7'de bahsedilmiştir. Bölüm 8 ise tartışma, sonuç ve gelecek çalışmalar kısmından oluşmaktadır.

2. KAVRAMSAL ÇERÇEVE

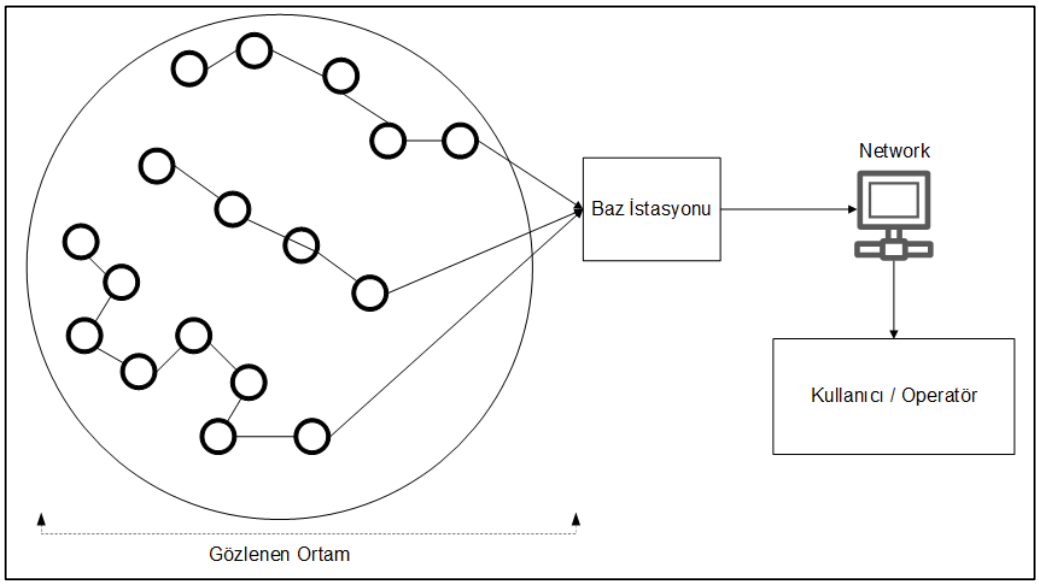
Bu bölümde Nesnelerin İnterneti, Büyük Veri, DDoS ve Anamoli Tespit Sistemleri hakkında bilgiler sunulmaktadır.

2.1. Nesnelerin İnterneti

Geçtiğimiz son on yılda popülerliği artan Nesnelerin İnterneti (IoT) kavramı, dünyada var olan teknolojik cihazların birbirleriyle bağlanması ve haberleşmesi sağlanarak hayatımızı daha verimli ve kolay hale getirmeye yarayan uygulamalara dayanmaktadır [5]. Kavramın içinde geçen "nesne" kelimesi ise akıllı saat, telefon, yağmur algılayan sensör, akıllı priz, akıllı trafik lambası, otonom araçlar gibi çok geniş bir yelpazeyi ifade eder [6]. Diğer bir açıdan Nesnelerin İnterneti, eşsiz bir biçimde tanımlanabilen nesnelerin belirli bir protokol aracılığıyla birbirleriyle iletişim kurduğu bir sistemdir ve ağdır. Birbiriyle bağlanan nesnelerin bilgi alışverişi yapabilmesi ve senkron sekilde çalışması gerekmektedir. Tüm bu sistem ve kurulacak algoritmalardaki amaç insanın yaşam standardının yükseltilmesi ve insan hayatını kolaylaştırılmaya çalışılmasıdır [7].

IoT ağları genel olarak Makineden Makineye İletişim(M2M), Düşük güçlü Kablosuz Kişisel Bölge Ağları(LoWPAN), FANET(Tasarsız Hava Taşıt Ağları), VANET(Araçlar arası Tasarsız Ağlar) ve Kablosuz Sensör Ağlar(KSA) gibi ağlardan oluşmaktadır [8]. Kablosuz sensör ağlar ise sensör düğümü denilen aygıtlardan oluşmaktadır. Hedef ortama bırakılan ve birbirleriyle bağlanarak bir kablosuz sensör ağını oluşturan sensör düğümler, ortamdaki verileri üzerinde bulunan sensörler yardımıyla toplayıp, işleyerek gerekli bilgileri ortaya cıkarıp dijital dünyada kullanıma sunarlar.

Şekil 2.1'de Kablosuz sensör ağların genel mimarisi verilmiştir. Hedef ortama konulan sensör düğümler, üzerinde amacına uygun sensörler yardımıyla gerekli algılamaları yaparak baz istasyonuna gönderirler. Kablosuz sensör ağlarında en az bir tane baz istasyonu olmalıdır. Baz istasyonu ise kullanıcı ve sensör düğümler arasında köprü işlevi görerek iletişimi sağlar.



Şekil 2.1. Kablosuz sensör ağların genel yapısı

Kablosuz sensör ağları düşük maliyetli olması, küçük hacimde olmaları, birçok farklı sensörün desteklenebilmesi ve kolay entegre edilmeleri sebebiyle birçok alanda uygulama yapılabilmektedir. Bunun yanı sıra, kablosuz algılayıcı ağlar, kendi kendini düzenleyen bir yapıya sahiptir; bu özellik, işleme ve iletişim kapasitesini paylaşma gibi avantajları beraberinde getirmektedir [6]. Kablosuz sensör ağlar; hava durumu tahminleme, habitat takibi, çevresem gözlem, hasta takibi, enerji tüketim ve izleme uygulamaları, endüstride makinelerin performanslarının değerlendirilmesi, yapıların güvenliği ve deprem tespiti, akıllı ev sistemleri, ofis uygulamaları, canlı trafik takibi, otopark uygulamaları, tarımsal üretim uygulamaları, askeri uygulamalar vb. gibi birçok farklı sektörde kullanılmaktadır [9].

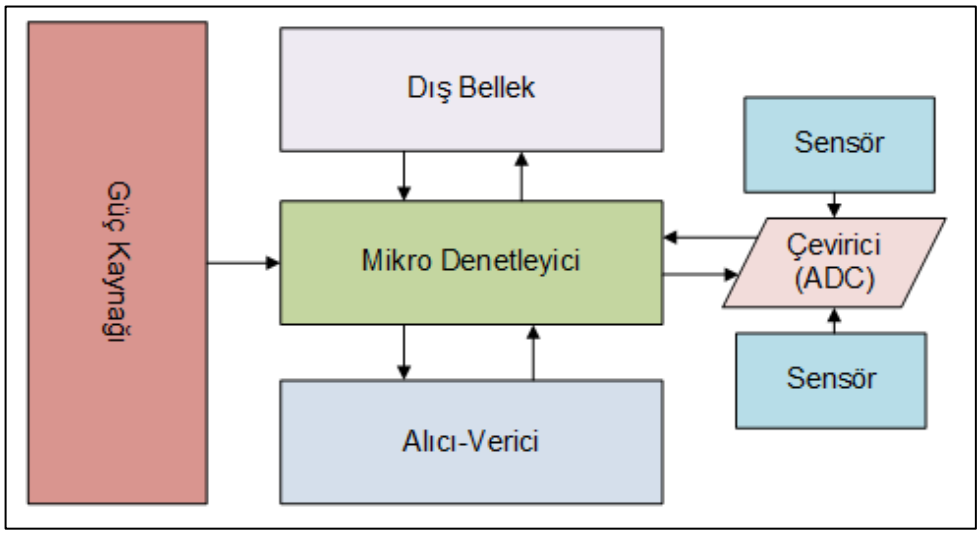
Kablosuz sensör ağların bazı gereksinimleri mevcuttur. Bunlar aşağıda bahsedilmiştir [9]:

• Ölçeklenebilirlik: Ekonomik ve küçük boyutlu sensörler kullanılarak sensör ağları, binlerce sensör düğümünü içerebilir. Ölçeklenebilirlik ve bu çok sayıdaki sensörleri yönetme zorluğu ağ tasarımını etkilemektedir.
• Düşük enerji tüketimi: Çoğu uygulamada sensör düğümleri uzak alanlara yerleştirilir, bu nedenle düğümlerin bakımı zor olabilmektedir. Düğüm ömrü, pil ömrüne bağlıdır, bu nedenle enerji tüketimini minimize etmek ve pilin en etkili şekilde kullanılmasını sağlamak önem arz etmektedir.
• Verimli bellek kullanımı: Maliyet sebebiyle sensör düğümün içerisine kısıtlı kaynaklar

konulabilir. Bu sebeble sensör düğümde yapılacak tüm işlemler, kısıtlı sensör düğüm belleğine sığacak şekilde planlanmalıdır.

• Veri toplama: Çok sayıdaki sönserlerden gelen bilgiler karmaşık ve dublike olabilmektedir. Bu sorunu çözmek için bazı düğümler veriyi toplaması ve belirli hesaplamalar yaparak özet bilgileri yayınlaması gerekebilir.
• Ağ özörgütlenmesi: Ağı oluşturan bazı sensör düğümler çökebilir, yeni düğümler ağa katılabilir, bu nedenle ağın kendini belirli aralıklarla yeniden yapılandırabilmesi ve işlevine devam edebilmesi için özörgütlenme oldukça önemlidir.
• Sorgulama yeteneği: Veri merkezli ve adres merkezli olmak üzere iki tür adresleme kullanılan sensör ağları, sorguların belirli bir bölgeye veya belirli bir düğüme gönderilmesi gerekebilir.
• Düşük maliyet: Binlerce düğüm içeren ağlar için sensör düğümlerinin maliyeti düşük olmalıdır.

Şekil 2.2'de bir sensör düğümün mimarisi verilmiştir [7]. Sensör düğüm genel olarak alıcı/verici, mikro işlemci, bellek, sensörler, dönüştürücü (ADC) ve güç kaynağı gibi komponentlerden oluşmaktadır. Mikroişlemci, düğüme gelen verileri işleyerek düğümün asli görevini yapmasını sağlar. Mikroişlemci, düğümün beynidir. Alıcı/verici ile düğüme gelen sinyallerin alınıp mikroişlemciye gönderilmesini sağlar. Aynı şekilde, düğümden yayınlanacak verileri de mikroişlemciden alarak ortama verilmesini sağlar. Bellek bölümü ise sensörlerden gelen verilerin ve diğer bilgilerin hafıza da tutulması görevini yapar. Sensör düğüm içerisinde birden fazla ve bir çok farklı çeşitte sensör bulunabilir. Sensörler, ortamdaki fiziksel bir özelliği algılayarak bunu dijital ortama aktarırlar. Sensörlerden gelen veriler analog bir yapıda olduğundan bunun mikroişlemci tarafından işlenebilmesi için dijital veriye dönüştürülmesi gerekir. Bu işlem ise "Analog to digital converter (ADC)" adı verilen dönüştürücü cihaz sayesinde yapılır. Düğüm üzerinde bulunan tüm bu komponentlerin çalışabilmesi için gerekli enerji ise sensör düğümde bulunan batarya veya pil üzerinden sağlanır. Uygulamanın türüne göre, sensör düğümde bulun güç kaynağı şarj edilebilir veya edilemez olarak iki farklı çeşitte bulunabilir.



Şekil 2.2. Sensör düğüm genel mimarisi

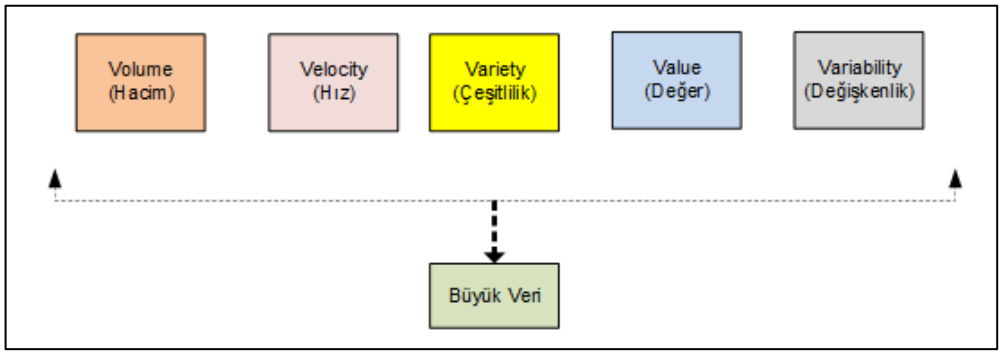
Bir kablosuz sensör ağ donanımsal olarak; yukarıda bahsedilen komponentlerden oluşur. Kablosuz sensör ağların ve kısıtlı batarya süreli IoT cihazların kaçınılmaz olarak bazı dezavantajları vardır. Sınırlı güç kaynağı, kısıtlı işlemci gücü, limitli bellek ve sınırlı bant genişliği ilk karşılaşılan negatif kısımlardır. Bu kısıtlamalardan dolayı kablosuz sensör ağlarda, klasik ağ teknolojilerinden farklı olarak, özel yönlendirme protokolleri, hafif şifreleme algoritmaları ve bu ağlar için özel olarak geliştirilmiş güvenlik protokolleri uygulanmalıdır Bu KSA'ların güvenliğinin sağlanması oldukça önem arz etmektedir. Bu ağların genel olarak ulaşılabilir bir alana bırakılması sebebiyle kötücül saldırılara daha açık haldedirler [6].

En önemli noktalardan biri bilgi güvenliğidir. Ağda işlenen verilerin bütünlüğünün ve gizliliğinin ifşa edilmemesi ve sadece yetkili kişiler tarafından ulaşılabilir olması gerekmektedir. Bu yüzden, kablosuz sensör ağlarda güvenliği sağlayacak sistemlerin tasarlanması oldukça önemlidir. Bu güvenlik sistemleri gerçek zamanlı tepki verebilmesi, yüksek doğruluk ve performansa sahip olması ve herhangi bir insan müdahalesine veya minimum şekilde müdahale ile çalışacak şekilde tasarlanması gerekmektedir [7]. Bu kapsamda Anamoli tespit sistemleri geliştirilmiştir. Bu sistem hakkında bilgiler ise Bölüm 2.4'te verilmiştir.

2.2. Büyük Veri Kavramı

Büyük veri kavramı genel olarak, geleneksel veri işleme yöntemleriyle analiz edilemeyecek kadar büyük, hızlı ve karmaşık veriler olarak tanımlanabilir. Büyük veri, Şekil 2.3'de belirtilen ve 5-V olarak adlandırılan hacim (volume), çeşitlilik (variety), hız (velocity), değişkenlik (variability) ve değer (value) olmak üzere beş temel özellik taşıyan ve geleneksel yöntemlerle işlenemeyen verileri ifade eder.

- Hacim: Verinin büyüklüğünü gösterir.
- Çeşitlilik: Çeşitli veri kaynaklarından farklı türlerdeki verileri ifade eder.
- Hız: Verinin üretilme frekansını tanımlar.
- Değişkenlik: Eldeki verinin ne kadar doğru ve gerçek olduğunu belirtir.
- Değer: Toplanan verilerden anlamlı bilgilerin elde edilmesini ifade eder.



Şekil 2.3. Büyük verinin özellikleri

Bir verinin ne zaman büyük olarak adlandırılacağına dair tanımlar, teknolojinin gelişmesiyle birlikte değişmektedir. Örneğin; 70'li yıllarda megabayt seviyesine büyük veri denirken, günümüzde ise terrabayt ve üzerindeki verilere büyük denilebilmektedir [10]. Hacim, büyük verinin en önemli sorunlarından biridir. Zira veri depolama ve analiz süreçlerinde yenilikçi yazılımlara ihtiyaç duyulmaktadır. Büyük veri artık geleneksel veri analiz yöntemleri tarafından işlenemeyen ve klasik veritabanlarında barınamayan boyutlara erişmiştir [11]. Kullanılan giyilebilir teknolojik aletlerden otonom sürüş yeteneği olan otomobillere kadar tüm teknolojik aletler kullandığı sensörler veya çalışma sırasında ürettikleriyle fazla miktarda veri oluşturmaktadırlar. Bunun sonucu olarak üretilen ve tüketilen veri miktarının artması büyük veri bilimine olan ihtiyacı arttırmaktadır. IBM'e göre, 2014 yılından bu yana dünya genelindeki verinin yaklaşık %90'ı son iki yılda oluşturulmuş olup; her gün 2,5 eksabayt büyüklüğünde veri üretimi gerçekleştirilmiştir.

2003 yılına kadar insanlık tarihi boyunca üretilen veri miktarı ise sadece iki gün içerisinde üretilmiştir [12]. Bu durum veri büyüklüğünün ne denli arttığını gözler önüne sermektedir.

Çeşitlilik, bir veri kümesindeki yapısal heterojenliği belirtir ve bu çeşitliliği ise %95 oranla yapısal olmayan veriler oluşturur [13]. Verilerin farklı formatlarda ve yerlerde bulunması, büyük veri kavramının diğer bir özelliğidir. Büyük veri, sadece veri tabanındaki kayıtlı girdilerle sınırlı olmamakla birlikte e-postalar, resim formatındaki faturalar, bilgisayarların ve akıllı cihazların bıraktığı loglar, iletişim merkezlerindeki ses kayıtları, toplantılarda veya eğitimlerde oluşturulan video dosyaları gibi farklı formatlarda ve yerlerde bulunabilir [14]. Tüm bu çeşitli veri türlerinin incelenmesi, büyük veri biliminin ilgi alanına girmektedir.

Büyük veri karakteristik özelliğinde yer alan bir diğer özellik ise hızdır. Verinin sürekli olarak hareket halinde olduğu kabul edilmektedir. Verinin üretilme hızı her geçen gün katlanarak artmaktadır [11]. Veri hızı kavramı, verilerin oluşma ve işlenme süresiyle ilişkilidir. Örnek olarak CERN'de gerçekleştirilen deneyde sensörler aracılığıyla 1 petabayt veri üretilmiştir [12]. Bu durum verinin ne kadar hızlı bir şekilde üretildiğini açıklayabilmektedir. Bir diğer açıdan bakıldığında bir alışveriş sitesinde, sitede gerçekleşen satış kayıtları önemli olarak kabul edilir. Bunun yanında oldukça önemli olan bir diğer bilgi ise müşterilerin alışveriş sırasındaki incelediği ürünler, aynı ürünle ilgili sayfada geçirilen süre gibi parametrelerdir. Bu veriden çıkarılan bilginin hemen değerlendirilip kullanılması gerekmektedir. Bu durumu aşmak için verilerin senkron şekilde analiz edilmesi gerekir. Analizler sonucunda müşteriye alternatifler sunan öneri modellerinin geliştirilmesi sağlanabilir [14].

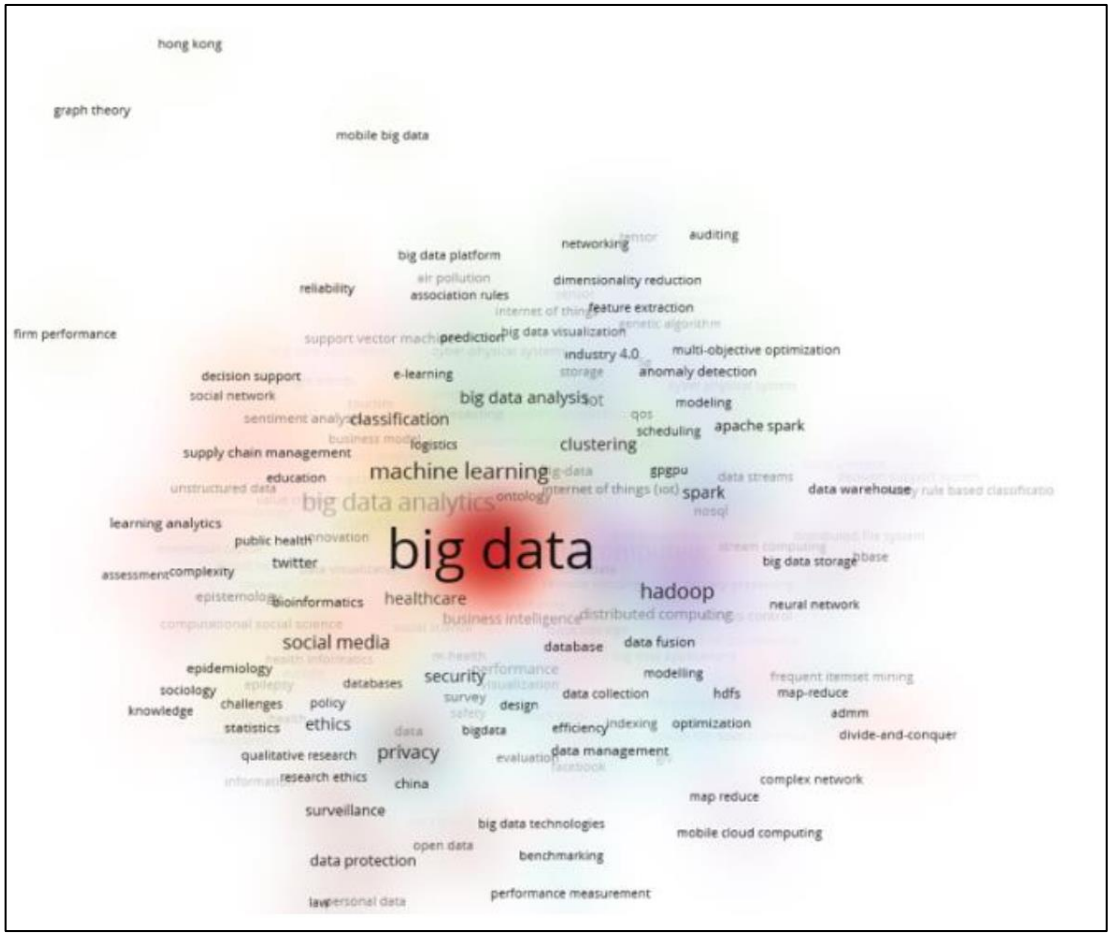
Değişkenlik, büyük verinin güvenilirlik ve doğruluk düzeyini gösterir. Veri, bir amaç ve analiz için kullanılacaksa güvenilir olması gerekir. Ancak, büyük verinin çok çeşitli format ve kaynaklardan oluşması sebebiyle güvenilirlik kavramı sorgulanabilir. Veri kalitesi, verinin güvenilirliği ile değerlendirilebilir, çünkü yüksek kaliteli veri sadece güvenilir modellerle üretilebilir [11]. Kesin olmayan ve belirsizlik içeren verilerle baş etme ihtiyacı, belirsizlik içeren verinin yönetimi, veri madenciliği için özel olarak tasarlanmış araçların ve analiz metotlarının kullanılması, bu konuda başka bir önemli perspektifi de ortaya koyar [13]. Bu noktada mevcut verinin doğruluğu ve geçerliliği büyük bir öneme sahiptir. Analiz

için sağlıklı bir temel oluşturmayan, doğru ve geçerli olmayan büyük veriler yanlış yorumlamalara sebep olabilmektedir [15].

Büyük verinin değeri, kuşkusuz en önemli özelliğini temsil eder. Çünkü, büyük verinin zahmetli bir şekilde depolanması ve analiz edilmesinin ardında ortaya bir değerin konma ihtimali yatar. Bu işlemlerden sonra ortaya anlamlı bir ilişkinin, bilginin ortaya çıkması büyük veriyi anlamlı hale getirir. Ortaya çıkarılan bu değer sayesinde alınacak kararlardan ve çıkarılacak sonuçlardan fayda sağlanması hedeflenir.

Büyük veri kullanarak verilerin içerisinde bulunan anlamların, ilişkilerin ortaya çıkarılarak değer üretmesi hedeflenir. Bu değerlendirme farklı biçimlerde yapılabilir. Kullanılan methodlardan birisi de veri içerisinde bulunan parametreler arasındaki korelasyonu ortaya çıkartmaktır. Korelasyon, veri içinde bulunan farklı iki parametre arasındaki ilişkiyi ifade eder. Eğer korelasyon değeri 1 ise bu iki parametre %100 ilişkili olduğu söylenebilir. Aynı şekilde korelasyon değeri 0 ise bu parametreler arasında ilişki yoktur. Örneğin, bir ürünün satışı artmasıyla birlikte belli oranda artmış bir başka ürünün varlığı korelasyon yöntemi ile ortaya çıkarılabilir [14]. Bu sayede işletmelerin buna uygun pozisyon alarak fayda sağlaması hedeflenebilir. Benzer şekilde makine öğrenimi kullanılarak yapılacak bir algoritma eğitiminde kullanılacak verisetindeki özellikler korelasyon yöntemi ile analiz edilebilir. Bu analiz sonucunda birbirleriyle yüksek oranda korele olmuş özellikler verisetinden çıkarılarak eğitim ve test maliyeti azaltılabilir. Bir diğer yöntem ise görselleştirme yöntemidir. Veri görselleştirme ise verinin içerisinde bulunan birçok istatiksel özelliğin herkes tarafından rahatça anlaşılabilecek şekilde görsel olarak ortaya konmasıdır. İlişkisel verilerin ortaya çıkarılıp bunları pasta diyagramlar, histogramlar, mum grafikleri, infografikler gibi görsel yöntemlerle sunulması herkes tarafından rahatça anlaşılmasına ve veriye dayanarak daha sağlıklı karar verilmesine katkı sağlayabilir [14]. Şekil 2.4'de büyük veri çalışmalarında kullanılan anahtar kelimelerin kullanım sıklığının görselleştirilmesi sunulmuştur [11]. Bir başka yöntem ise büyük veri kullanılarak yeni gelecek veriler için tahmin yapılabilmesidir. Bu yöntem makine öğrenimi algoritmalarının kullanılmasıyla yapılabilir. Büyük veri analiz araçlarının gelişmesiyle makine öğrenimi algoritmaların doğruluk oranları da artmıştır. Makine öğreniminde ne kadar çok ve değerli bilgi ile eğitim yapılırsa o denli yüksek doğruluk oranlarına ulaşılabilir. Makine öğrenimi ile veri içerisindeki yapılar ve ilişkiler ortaya çıkarılarak yeni olarak gelecek parametreler hakkında tahmin yapabilecek duruma getirilir. Makine öğrenimi algoritmalarının yüksek

değer verebilmesi içim kullanılacak veri kümesinin hatalardan arındırılmış ve algoritmanın eğitimine uygun olacak şekilde temizlenmiş ve uyumlaştırılmış olması gerekmektedir.



Şekil 2.4. Büyük verinin görselleştirilmesi

Büyük veri, bir dizi fırsat sunarken aynı zamanda yeni ve karmaşık zorluklar da beraberinde getirir. Büyük verilerin depolanması ve bu verilerden çeşitli analizler yapılması, geleneksel yöntemlerle oldukça zor veya neredeyse imkansız bir süreçtir. Ancak, bu zorlukların üstesinden gelmek için Hadoop, Apache Spark gibi özel büyük veri teknolojileri kullanılarak büyük veri üzerinde etkili bir şekilde çalışmak ve önemli verileri çıkarmak mümkün hale gelmiştir [3].

2.3. DDoS Kavramı ve Veri Setinde Bulunan DDoS Türleri

DDoS saldırısı, hedef sistemin normal trafiğini bozmaya yönelik kötü niyetli girişimdir. DDoS saldırıları isteklere yanıt vermek için hedeflenen kaynak makinelerin kapasitesini aşan paketler göndererek hizmeti kesintiye uğratmayı amaçlar. Saldırganlar, büyük miktarda paket göndermek için kurbanların bilgisayarlarına yerleştirilen kötücül amaçlı

yazılımlarla oluşturulmuş zombi bilgisayarları kullanırlar. DDoS saldırıları ağ üzerinden gönderilen paketlerle yüksek ağ trafiğine neden olarak, sistemin hizmet almak isteyen normal kullanıcıların isteklerine yanıt vermemesine sebep olurlar [4]. Bir DDoS saldırısı, internet tabanlı uygulamalara ve kaynaklarına yönelik en büyük tehdittir. Bu saldırının amacı, büyük miktarda saldırı trafiği ileterek internet tabanlı hizmetleri yetersiz kılmaktır [16]. TON_IOT veri setinde saldırı tipleri ana kategori içinde verildiği için bu saldırıların detayı bilinmemektedir. Örneğin DDoS ana kategorisi altında hangi türlerin bulunduğu verilmemiştir. CICIoT2023 veri setinde ise saldırı türlerinin alt kategorileri de verilmiştir. Bu veri setinde flood ve fragmentation başlıkları altında farklı saldırı türlerini içermektedir. Bu saldırılar aşağıda tanımlanmıştır.

Syn Flood saldırısında, saldırgan TCP bağlantı isteğini için art arda yarı açık senkronizasyon paketleri göndererek IoT cihazların kaynaklarını tüketir. Bu bağlantılar, daha ilerideki iletişim için açık bırakılır [4]. Tüm kullanılabilir bağlantı noktalarını kullanan kurban makine, yasal trafiğe yavaş yanıt veya hiç yanıt veremeyebilir. UDP flood saldırısı, hızlı veri paylaşım protokolü olan User Datagram Protocol (UDP) ile saldırganlar tarafından herhangi bir izin alınmadan büyük paketlerin gönderildiği bir saldırıdır [4]. ICMP (Internet Control Message Protocol), IP kontrolü/hata raporlaması için kullanılan ağ protokolüdür. ICMP Flood saldırısında ise, saldırgan cok fazla sayıda ICMP isteği yolayarak, ağın cevrimdışı kalmasını amaçlar [17]. TCP protokolünde FIN ve RST paketleriyle yapılan RSTFIN Flood saldırıları mevcuttur. FIN paketi, TCP bağlantısı mevcut client-sunucu arasındaki bağlantıyı güvenli bir şekilde sonlandırmak için gönderilir. RST paketi ise normal olmayan durumlarda sunucu tarafından gönderilen ve bağlantıyı zorla kapatmak için kullanılır. RSTFIN Flood saldırında ise saldırgan, hedef ağa ait olmayan FIN ve RST paketleri göndererek sistemin tıkanmasına sebep olmaktadır [17]. HTTP flood saldırısının ana odağı, normal ağ trağine yakın benzerlikte simüle eden saldırı trafiği oluşturmaya yöneliktir. Böylece kurbanın, meşru trafik ile saldırı trafiği arasında ayrım yapması zorlaşmaktadır. HTTP flood saldırısında oturum bağlantı istek oranları, meşru kullanıcılar tarafından oluşturulandan daha yüksek olması sağlanarak sunucunun kaynakları tüketmesi hedeflenir [18]. Slowloris saldırısında, HTTP isteği parçalar halinde ve yavaş olacak şekilde gönderir ve oluşturulan istek tamamlanmaz. Bunun sonucunda, sunucu bağlantıyı tamamlamak ve gerekli veriyi alabilmek için ilgili bağlantıyı bekleme aşamasında tutar. Bu şekilde zaman geçtikçe acık kalan bağlantı istekleri çoğalarak sistemin tıkanmasına sebep olur [18]. PSHACK flood saldırında kullanılan PSH paketi, bu

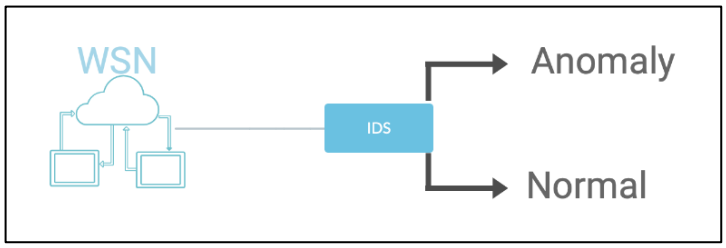
komutu alan istemcinin tüm verileri belirlenen bir uygulamaya göndermesini ve işlenmesini sağlamak için kullanılır. PSH ve ACK kombinasyonlarına sahip paketler genellikle normal gelen trafikte görülür [19]. Saldırgan bu paket kombinasyonlarını yoğun şekilde hedef sunucuya yönlendirerek PSHACK flood saldırı oluşturabilir. Synonymous IP Flood saldırısında, var olmayan bir etki alanı için yüksek hacimli istek gönderilerek DNS sunucularının kaynaklarını tüketmeyi amaçlayan bir DDoS saldırı türüdür. TCP protokolünü kullanan bu saldırı da yüksek hızda paketler kullanılır [20].

Fragmentation saldırılarında, ağ girişinde tranferin yapılabileceği MTU (Maksimum Transfer Unit) limitinden daha yüksek boyutta paket gönderilerek bunların parçalanarak gönderilmesini sağlamaktır. Bu boyut ethernet ağlarında 1500 byte'dır. Fragmentation saldırılarından bu değerden yüksek çerceve gönderilerek saldırı yapılır. ACK Fragmentation saldırısı ACK ve PUSH-ACK Flood saldırısının bir versiyonudur. Parçalanmış paketler switch'ler, güvenlik duvarları, IDS ve IPS'lerden geçer çünkü yönlendirici parçalanmış çerçeveleri yeniden birleştirmez. Bu paketler çok sayıda rastgele ve birbirinden alakasız bilgiler içerebilmektedir. Bu yöntemle kurbanun kaynakları tüketmesi hedeflenir [21]. ICMP Fragmentation saldırısında, parçalanmış ICMP paketleri kullanılır. Kurban, yeniden birleştirilemeyen ICMP paketlerine maruz kalır. Bu paketler rastgele ve alakasız bilgiler içirdiği için kurbanın kaynakları bunları birleştirmeye calısarak tüketilir [21]. UDP Fragmentation saldırısı, UDP flood saldırısının bir uyarlamasıdır. Parçalanmış UDP paketleri sahte ve birbirleriyle ilişkisiz olduğu için hedef sunucu, bunları yeniden birleştirmeye calısarak kaynaklarını boş yere tüketmektedir. Bu saldırı türünde kurbanın CPU'larının aşırı ısınması ve kaynaklarının gereksiz yere tüketilmesine sebep olmaktadır [21].

2.4. Anamoli Tespit Sistemleri

İnternet çağının gelişerek devam etmesi sonucu kullanılan cihazlar ve bunların oluşturduğu ağın güvenlik konusu çok önemli hale gelmiştir. Kötücül saldırılar, bilginin gizliliğini, bütünlüğünü ve erişilebilirliğini tehlikeye atabilecek girişimlerin tümüdür şeklinde tanımlanmaktadır [22]. Burada saklanan verilerin bütünlüğü ve gizliliği bu saldırılardan korunmalıdır. Bunu gerçekleşmesi için kullanılan sistemlerden biri de Intrusion detection sistemleridir. Bu sistemler, herhangi bir saldırıyı normal trafikten ayırmakta ve bir saldırı durumunda kullanıcıyı uyarabilmektedir. Sekil 2.5’te genel çalışma mekanizması verilen

IDS sistemler, normal trafikten farklı olan tüm aktiviteleri anomali olarak kaydeder. Bu sistemler yanlış alarm vermeye de eğilimlidirler. Bu yüzden, geliştirilecek IDS'lerin doğruluk değerinin yüksek olması çok önemli bir konudur. Bu çalışmada sadece saldırı tespiti değil, saldırı türünün ne olduğu tespiti de yüksek doğruluk oranı ile yapılmıştır.



Şekil 2.5. Anamoli tespit sistemleri

3. LİTERATÜR TARAMASI

Anomali tabanlı izinsiz giriş tespit sistemleri, saldırıyı normal davranıştan herhangi bir sapma olarak tanımlar. WSN'lerin yapısı gereği bazı kısıtlamaları mevcuttur. Bu yüzden kullanılacak IDS'lerin bu kısıtlamaları dikkate alınarak tasarlanması gerekmektedir. Litaratürde WSN'lere yönelik sınıflandırma, kümeleme, makine öğrenme ve istatistiksel öğrenme algoritmalarını kullanarak geliştirilen IDS sistemleri mevcuttur [1]. Bu bölümde WSN'de DDoS tespit etmeye yönelik geliştirilen IDS çalışmaları özetlenmiştir.

Cil ve arkadaşları [4] klasik derin öğrenme algoritması kullanarak DDoS tespit etme sistemi geliştirmişlerdir. Bu algoritma giriş tabakasında 69 ünite, giriş ve çıkış arasında bulunan 50'şer ünite içeren 3 gizli katman olarak tasarlanmıştır. CICDDoS2019 veri seti kullanılarak iki farklı veri seti elde edilmiştir. Bunlardan ilkinde trafiğin saldırı veya normal olarak etiketlenen veriler, diğerinde ise saldırı türleri mevcuttur. Saldırı tespitinde %99,97'lik, saldırı türü tespitinde ise %94,57'lik doğruluk oranına ulaşılmıştır.

Almaraz-Rivera ve arkadaşları [23], Makine Öğrenimi ve Derin Öğrenme modellerine dayalı yeni bir Saldırı Tespit Sistemi oluşturmak için sınıf dengesizliği sorununu ele alan ve 2019 yılında yayımlanan Bot-IoT veri kümesini kullanmışlardır. Bu veri seti kullanılarak farklı özellik setleri seçilerek üç farklı veri kümesi oluşturulmuştur. İlk veri setinin diğer iki kümeden farkı zaman damgaları (stime, ltime) ve Argus sıra numarasıdır (seq). İkinci veri kümesinde ise zaman damgaları, modelin bu özellikleri ezberleyeceği varsayılarak kaldırılmıştır. Aynı şekilde sıra numarası da kaldırılmıştır. Son veri kümesinde ise zaman damgalarının etkisini değerlendirmek için stime ve ltime özellikleri kaldırılmıştır. Oluşturulan bu veri kümeleri Random Forest, Decision Tree, LSTM, GRU, MLP, RNN ve SVM makine öğrenme algoritmaları tarafından ikili ve çoklu sınıflandırma olacak şekilde ayrı ayrı teste tabi tutulmuştur. Değerlendirme sonucunda ortalama %99 üzerinde doğruluk elde edilmiştir. Decision Tree ve MLP modellerinin en iyi performans gösteren yöntemler olduğu görülmüştür.

Jia ve ekibi [24] tarafından tasarlanan Flow Guard algoritması, Akış filtresi (Flow Filter) ve akış işleyicisi (Flow Handler) olarak iki componentten oluşmaktadır. Akış filtresi, akış işleyicisi tarafından oluşturulan filtreleme kurallarına dayalı olarak kötü amaçlı akışları

filtrelemekten ve trafik değişikliklerine dayalı olarak tanımlanamayan kötü amaçlı akışları algılamaktan sorumludur. Akış işleyici, geliştirilen LSTM ve CNN'nin iki makine öğrenimi modeline göre kötü amaçlı akışları tanımlama ve sınıflandırma sorumluluğunu üstlenmektedir. CICDDoS2019 veri seti kullanılarak FlowGuard algoritmasının performansı ölçülmüştür. Algoritmaların saldırı ve saldırı türü tespit doğruluk oranları sırasıyla %99,9 ve %98,9 olarak ölçülmüştür.

Alghazzawi ve ekibi [25], öznitelik belirleme yaklaşımıyla hibrit bir derin öğrenme algoritması geliştirmişlerdir. CNN ve BiLSTM derin öğrenme algoritmaları kullanılarak oluşturulan model, filtre sayısı, filtre büyüklüğü ve BiLSTM ünitelerinin büyüklüğü gibi değişkenlerinin farklı kombinasyonları ile 10 farklı şekilde test edilmiştir. Oluşturulan algoritmalarının eğitim, test ve doğrulama sırasında CIC-DDoS2019 veri kümesini kullanarak yüzde 94,52'ye varan bir doğruluk oranı elde edilmiştir.

Chartuni ve Márquez [26], yapay sinir ağı kullanılarak DDoS tespit etme mekanizması geliştirmişlerdir. Algoritma geliştirme sırasında parametrelerin hangi kombinasyonda en iyi sonucu vereceğini tespit etmek için 10 farklı seçenek ile test edilmiştir. Bu parametreler, gizli katman derinliği, optimizasyon fonksiyonunun öğrenme katsayısı ve dropout katmanının oranıdır. Sinir ağının derinliğini, yani katman sayısını seçilmesi sağlanmıştır. Bu seçim için gizli katman bloğunda 2, 3 ve 4'lük bir derinlik seçenekleriyle test edilmiştir. Sonuca göre, model doğruluğu ve eğitiminde harcanan zaman nedeniyle derinlik üç katman olacak şekilde belirlenmiştir. Adam optimizasyon fonksiyonunun öğrenme oranı, $1 \times 10{-}4$, $1 \times 10{-}5$ ve $1 \times 10{-}6$ olmak üzere üç aday oranına göre yapılandırılarak değerlendirilmiştir. Bu değerlendirme sonucunda, Adam optimizasyon konfigürasyonu için $1 \times 10{-}4$'e karşılık gelen öğrenme oranının seçilmesine karar verilmiştir. Dropout katmanında, yok sayılacak toplam nöron oranı farklı kombinasyonlarla test edilmiştir. 0.3 oranının en iyi sonucu verdiğini görülmüştür. Oluşturulan modelin eğitim ve test edilmesinde CIC-DDoS2019 veri seti kullanılmıştır. Bu veri seti ile yapılan önceki çalışmalar ikili bir sınıflandırma yaklaşımı önermiştir, bu çalışma ise çoklu sınıflandırmaya dayanmaktadır. Test sonucunda modelin %94'lük bir doğruluk oranına ulaştığı tespit edilmiştir.

Ferrag ve arkadaşları [27], CNN, RNN ve DNN olmak üzere üç farklı modele dayalı derin öğrenme tabanlı bir DDoS saldırı tespit sistemi önermişlerdir. Her bir modelin performansı

CIC-DDoS2019 ve TON_IoT verisetlerini kullanılarak ikili ve çoklu sınıflandırma türünde eğitilmiş ve test edilmiştir. Eldeki veri setleri ikili sınıflandırma ve çok sınıflı sınıflandırma verimliliğini analiz etmek için üç farklı veri setine bölünmüştür (Dataset_2_class, Dataset_7_class ve Dataset_13_class). Yapılan deneyler sonucunda CIC-DDoS2019 veri setiyle CNN ile yapılan algoritmanın doğruluk oranları ikili sınıflandırma için %99,95, yedili sınıflandırma için %95,90 ve on üçlü çoklu sınıflandırma için ise %95,12 olarak ölçülmüştür.

Bhati ve arkadaşları [28], DNN kullanarak DDoS saldırılarını tespit etme algoritması geliştirmişlerdir. Oluşturulan derin öğrenme algoritmasının eğitiminde ve testlerinde ISCX2017, ISCX2018 ve CIC-DDOS2019 veri setleri kullanılmıştır. Veri setlerinde bulunan belirli özelliklerin seçimi ile optimum başarı elde edilmeye çalışılmıştır. ISCX2017 veri setinden 20, CIC-DDOS2019 veri setinden ise 22 özellik seçilerek algoritma eğitilmiştir. İki adet gizli katman kullanılarak algoritma oluşturulmuştur. Algoritma, DDoS saldırı tespitinde %96,9 başarı oranına ulaşmıştır.

Wei ve arkadaşları [29], DDoS saldırı tespiti ve sınıflandırması için AE-MLP isimli hibrit bir yaklaşım önermişlerdir. Otomatik Kodlayıcı (AE) kısmı ile ağ trafiğindeki en gerekli özellikleri otomatik olarak bularak kullanılacak özellikleri belirler. MLP kısmı ise, AE tarafından belirlenen ve indirgenmiş özellik kümelerini girdi olarak kullanarak atak tespiti yapılmasını sağlar. AE-MLP algoritması sayesinde uygun özellikler seçilerek, eğitim maliyeti düşürülür ve daha doğru tespit yapılmasına olanak sağlar. MLP algoritması ile sadece atak tespiti değil, atak türlerinin de sınıflandırılması yapılır. Kullanılan veri setinden 6 farklı alt veri seti oluşturularak testler yapılmıştır. LDAP, MSSQL, NetBIOS, SYN, UDP ve BENIGN türlerindeki network trafiğinin sınıflandırılması amaçlanmıştır. Testler sonucunda önerilen algoritmanın %98,34 doğruluk oranına ulaştığı görülmüştür.

Rehman ve arkadaşları [30], RNN ve GRU (Gated Recurrent Unit) algoritmalarını kullanarak DDoS saldırılarına karşı koruma sağlamak için DIDDDOS adlı bir yaklaşım önermişlerdir. Çalışmada ayrıca geleneksel makine öğrenme algoritmaları olan NB ve SMO da kullanılmıştır. Algoritmaların eğitim ve testlerinde CIC-DDOS2019 veri setinden yararlanılmıştır. Kullanılan veri seti çok büyük olduğu için atakların hepsi ayrı ayrı test edilmiştir. Yapılan deneyler sonucunda GRU algoritması kullanılarak SSDP tespit oranı %99,91 ve diğer atak türleri için ortalama %99,7 olarak ölçülmüştür.

Susilo ve arkadaşı [31], DDoS algılamak için Derin öğrenme yöntemlerini kullanan bir algoritma geliştirmişlerdir. Çalışmada scikit-learn, Tensorflow ve Seaborn kütüphanelerini Python programlama vasıtasıyla kullanmışlardır. Saldırıların sınıflandırılması için CNN ve MLP algoritmaları geliştirilmiştir. Geliştirilen bu derin öğrenme algoritmaları, makine öğrenmesi algoritması olan Random Forest (RF) ile karşılaştırılmıştır. Derin öğrenme algoritmaları, batch size 32, 64 ve 128; epoch sayısı 10, 30 ve 50 kombinasyonları olacak şekilde BoT-IoT veri setinde eğitilmiş ve test edilmiştir.

Kumar ve arkadaşları [32], Decision Trees, Random Forest, KNN, Naive Bayes ve bunların birlikte kullanıldığı hibrid bir makine öğrenmesi algoritmalarıyla, geliştirilen Yapay Sinir Ağı algoritmasını Bot-IOT veri setiyle eğitilmiştir. Daha sonra ise 20 IoT cihazdan oluşan testbed kurularak elde edilen trafik verisiyle test edilmesi sağlanmıştır. Oluşturulan veri setinde 3M DoS, 2.5M DDoS UDP, 2.5M DDoS TCP ve 2M Normal trafik verisi mevcuttur. Bu veri seti ile yapılan testlerde en iyi sonuca KNN (Acc. %99,466) ve hibrit (%99,611) algoritma ile ulaşılmıştır. Yapay Sinir Ağı algoritması ise farklı aktivasyon fonksiyonları ile ayrı ayrı test edilmiştir. En iyi sonucu, %99,529 doğruluk oranı ile ReLU fonksiyonu kullanılarak alınmıştır. Bu karşılaştırmalardan sonuç olarak daha az kaynağa sahip sistemlerde makine öğrenmesinin, daha fazla veri ve kaynak kullanılabileceği sistemlerde ise derin öğrenme algoritmalarının daha iyi sonuç verebileceği vurgulanmıştır.

Alzahrani R.J ve Alzahrani A. [33], CICDDoS2019 veri setini kullanarak WEKA tool'u aracılığıyla makine öğrenmle algoritmalarından KNN, SVM, NB, DT, RF ve LR olmak üzere altı farklı algoritmanın değerlendirilmesini sunmuşlardır. Veri setinde özellik seçimnde, saldırı trafiğini tespit etmede ML yöntemlerinin doğruluğunu artırmasına katkıda bulunduğu tespit edilen random forest regressor (RFR) öznitelik seçim yöntemi kullanılmıştır. Değerlendirmede DT ve RF algoritmaları sırasıyla %99 ve %99 olmak üzere en yüksek doğruluk oranlarına ulaşmışlardır. DT algoritması, RF'e göre daha hızlı hesaplama süresi olduğu da tespit edilmiştir. Çalışma da ayrıca ML ve DL tabanlı IDS'lerin artılarını, eksilerini ve tespit yöntemlerini içeren bir inceleme de sunulmuştur.

Batchu and Seetha [2], veri setinde özellik seçimine dayanan hibrid özellik seçimi ve hiperparametre tuning algoritmalarını kullanan anomali tespit sistemi geliştirmişlerdir. Bu yöntem ile makine öğrenme algoritmalarından LR, DT, Gradient boost (GB), KNN and

SVM kullanılarak değerlendirilmiştir. Değerlendirmede CICDDoS2019 veri seti üzerinde 4 farklı durum ve çeşitli senaryolar ile denenmiştir. En başarılı sonuca hibrit özellik seçimi ve hiperparametre tuning algoritmalarını kullanılarak hazırlanan veri setinde GB algoritmasıyla elde edilmiştir, %99,97 oranında doğruluk değerine ulaşılmıştır.

Patil ve arkadaşları [16], DDoS saldırı türlerini gerçek zamanlı sınıflandırmak için SSK-DDoS olarak adlandırılan Spark Streaming ve Kafka tabanlı sınıflandırma sistemi önermişlerdir. Sistem, CICDDoS2019 veri setinde bulunan altı saldırı türü (DDoS-DNS, DDoS-LDAP, DDoS-MSSQL, DDoS-NetBIOS, DDoS-UDP ve DDoS-SYN) ve normal trafiğin sınıflandırılması için eğitilmiş ve test edilmiştir. Geliştirilen algoritma, formüle edilmiş özellikleri değerlendirilen sınıflarıyla birlikte HDFS'ye depolamakta ve yeniden eğitim sırasında tekrar kullanılabilmektedir. Deneyler sonucunda sunulan tespit sisteminin ağ trafiğini %89,05 doğruluk oranı ile yedi sınıfa ayırdığı gösterilmiştir.

Al ve Dener [3], büyük veri ortamında ağ trafiğinde sınıflandırma tabanlı yeni bir anomali tespit sistemi olan STL-HDL yöntemini sunmuşlardır. Geliştirilen algoritma, CNN ve LSTM'den oluşan hibrit bir algoritmadır. Ayrıca verisetlerinde bulunan dengesiz dağılımın etkisini azaltmak için SMOTE ve Tomek-Links teknikleri kullanılmıştır. Geliştirilen model binary sınıflandırma değerlendirilmesi UNS-NB15 ve çoklu sınıflandırma değerlendirilmesi ise CIDDS-001 verisetleri üzerinde yapılmıştır. Önerilen yöntem dokuz farklı makine ve derin öğrenme algoritmalarıyla da karşılaştırılmıştır. Sistemin başarı oranı ise binary olarak %99,17, çoklu sınıflandırma olarak ise %99,83 olmuştur.

Haq ve arkadaşları [34], IoT cihazlarda saldırı durumlarını engellemek özellik boyutunu azaltmaya yönelik Principal Component Analysis (PCA) tekniği ile saldırı sınıflandırması için kullanılan 13 katmanlı CNN algoritmasını birleştirerek PCCNN yöntemini sunmuşlardır. Bu yöntem NSL-KDD veri seti kullanılarak değerlendirilmiştir. Yöntemin doğruluk oranı binary ve çoklu sınıflandırma için sırasıyla %99,34 ve %99,13 olarak ölçülmüştür.

Iwendi ve ekibi [35], IoT cihazların DDoS saldırılarını tespit edebilmek için derin öğrenme tabanlı anamoli tespit sistemi sunmuşlardır. Bu sistem derin öğrenme algoritması olan LSTM kullanılarak gerçeklenmiştir. CICDDOS2019 veri seti kullanılarak yapılan bu deneyler sonucunda SNMP saldırı tespitinde %99,97'lik doğruluk oranına ulaşılmıştır.

Gamal ve arkadaşları [36], çalışmada CNN-IDS olarak adlandırılan yeni bir IDS yöntemi sunmuşlardır. Önerilen sistem iki aşamadan oluşur: İlk aşamada Information Gain yöntemi olarak adlandırılan veri setindeki özellik seçme, ikinci aşama ise tek boyutlu CNN algoritması ile ağ trafiğini sınıflandırma aşamasıdır. Önerilen model, UNSW-NB15 ve Bot-IoT verisetleri kullanılarak eğitilmiş ve doğrulanmıştır. Bot-IoT veri setinde %99,9 oranında algılama oranına ulaşılmıştır.

Gad ve arkadaşları [37], TON_IOT veri setini temel alan IDS sunmuşlardır. TON_IOT veri setindeki eksik değerler ve sınıf dengesizliği gibi konular ele alınmıştır. Sınıf dengeleme için SMOTE tekniği kullanılarak veri setinin sınıf dengesizliği ve aşırı öğrenme problemini önlemeye çalışmışlardır. Öznitelik seçimi olarak Chi2 tekniği kullanılmıştır. Özellik sayısını 20'ye düşürülerek daha hızlı eğitim süresi sağlanmış ve modelin komleksliği azaltılmıştır. Ön işlemeye tabii tutulan veri seti, LR, NB, DT, RF, AdaBoost, KNN, SVM ve XGBoost algoritmalarıyla test edilmiştir. En iyi sonuca, XGBoost algoritmasıyla binary olarak %99,3 ve çoklu sınıflandırma olarak %98,6 doğruluk oranları ile ulaşılmıştır.

Disha ve Waheed [38], özellik seçimi algoritmasına dayalı bir anomali tespit sistemi geliştirmişlerdir. Özellik seçim algoritması olarak Gini Impurity-Based Weighted Random Forest (GIWRF) şeklinde adlandırılan algoritma geliştirilmiştir. Bu kapsamda yapılacak deneylerde iki farklı dataset kullanılmıştır. Bunlar UNSW-NB 15 ve TON_IoT verisetleridir. UNSW-NB 15 veri setinde 15 ve TON_IoT veri setinde ise 10 özellik seçimi yapılarak, seçim yapılmayan duruma göre performans karşılaştırılması yapılmıştır. Deneyler DT, AdaBoost, GBT, MLP, LSTM ve GRU algoritmaları kullanılarak test edilmiştir. Özellik seçimi kullanılarak TON_IoT veriseetinde en yüksek doğruluk oranına %99,98 değeri ile AdaBoost ve GBT algoritmaları ulaşmıştır. Özellik seçimi yapılmamasına göre DT algoritmasında 0.4, AdaBoost algoritmasında ise 0.1 daha yüksek başarı elde edilmiştir. MLP, LSTM ve GRU algoritmalarında ise doğruluk değerinde düşme meydana gelmiştir.

Kaur ve arkadaşları [39], gizlilik odaklı olan man-in-the- middle (MiTM) ve DoS/DDoS saldırılarını tespit edebilmek için iki aşamalı çalışan ve P2ADF olarak adlandırılan anamoli tespit sistemi geliştirmişlerdir. İlk aşamada kullanılan verisetlerindeki baskın özellikleri çıkarmak için özellik azaltma mekanizması vardır. İkinci aşamada ise model eğitimi için

üç temel öğrenici (AdaBoost, LR ve KNN) ve bir meta sınıflandırıcı (XGBoost) kullanılmıştır. Deneyler için IoTID20, TON_IoT, N-BaIoT, UNSW-NB15 ve CICDoS19 verisetleri kullanılmıştır. DDoS tespiti için CICIDS2019DDoS LDAP ile %99,98 ve TON_IoT DDoS ile %99,95 doğruluk değerlerine ulaşılmıştır.

Verma ve Chandra [40], IoT alanı için bu alanda popüler olan DoS/DDoS ve Sybil saldırılarının tespit edilmesini sağlayan RepuTE adı verilen algoritmayı geliştirmişlerdir. Sunulan RepuTe algoritmasında IoT katmanındandan gelen canlı trafik Fog katmanına aktarılarak buraya gelen trafik verisi öncelikle ön işlemden geçmektedir. Extra Tree, KNN ve Quadratic Discriminant Analysis algoritmalarından oluşan model, gelen bu canlı trafik verisini değerlendirmektedir. Modelin en yüksek doğruluk oranına ulaşabilmesi için bu algoritmalar arasında soft-voting denilen yöntem uygulanmıştır. Bu çalışmanın NSL-KDD, CICDDOS2019, IoTID20, NBaIoT2018, TON_IoT ve UNSW_NB15 verisetlerinde değerlendirilmesi yapılmıştır. Saldırı tespitinde CICDDoS2019_NTP ile %99,9988 ve TON_IoT_ DDoS ile %99,9851 doğruluk değerine ulaşılmıştır.

Neto ve arkadaşları [41], yaptıkları çalışma ile IoT operasyonlarının güvenliğinin geliştirilmesi için kapsamlı bir IoT saldırı veri seti önermektedir. Bunu sağlamak 105 IoT cihazdan olusan bir topoloji kurulmuştur. Bu düzenek ile 7 kategoride 33 farklı saldırı türü trafiğinin kaydı yapılmıştır. Bu saldırı kategorileri ise DDoS, DoS, Recon, Web-based, brute force, spoofing, and Mirai'dir. Oluşturulan veri setine ise CICIoT2023 adı verilmiştir. Geliştirilen veri seti kullanılarak, çeşitli makine öğrenimi algoritmalarının doğruluk değerleri test edilmiştir.

WSN'de yapılmış IDS çalışmalarına ait çalışmalar Çizelge 3.1'de sunulmuştur. Bu tabloda yıl, yazar, kullanılan algoritma, kullanılan veri seti ve doğruluk bilgileri yer almaktadır. Literatürde makine öğrenmesi ve derin öğrenme modellerini kullanan birçok farklı algoritma geliştirilmiştir. Gelişen teknoloji ile DDoS saldırılarının yapısı değişmiş ve kullanılan verisetleri geçerliliğini yitirmiştir. Bu yüzden bu çalışma da güncel CICIoT2023 veri seti kullanılmıştır. Algoritmanın güvenirliğini test etmek için yine güncel bir veri seti olan TON_IOT veri seti de kullanılmıştır.

Çizelge 3.1. DDoS tespit edilmesine yönelik yapılan çalışmalar

Yıl	Yazarlar	Model	Veri seti	Doğruluk
2021	Cil ve diğerleri	DNN	CICDDoS2019	%99,97 (b), %94,57 (m)
2022	Almaraz-Rivera ve diğerleri	DT, MLP, RNN, RF, GRU, LSTM and SVM	Bot-IoT	%99,972 (b), %99,945 (m)
2022	Jia ve diğerleri	DNN, LSTM	CICDDoS2019	%99,9 (b), %98,9 (m)
2021	Alghazzawi ve diğerleri	CNN, BiLSTM	CICDDoS2019	%94,52 (b)
2021	Chartuni and Márquez	ANN	CICDDoS2019	%94 (m)
2021	Ferrag ve diğerleri	CNN, RNN and DNN	CICDDoS2019, TON_IoT	%99,95 (b), %95,12 (m) %99,92 (m)
2020	Bhati ve diğerleri	DNN	CICDDOS2019, ISCX2017 and ISCX2018	%96,9 (b)
2021	Wei ve diğerleri	MLP	CICDDOS2019	%99,96 (b) %98,34 (m)
2021	Rehman ve diğerleri	RNN, GRU, NB and SMO	CICDDOS2019	%99,91 (b)
2020	Susilo ve diğerleri	CNN, MLP and RF	BoT-IoT	%91,27 (m)
2022	Kumar ve diğerleri	DT, RF, KNN, NB and ANN	Bot-IOT	%99.611 (m)
2021	Al and Dener	CNN, LSTM	UNS-NB15 and CIDDS-001	%99,17 (b), %99,83 (m)
2021	Alzahrani and Alzahrani	SVM, KNN, DT, NB, RF and LR.	CICDDOS2019	%99 (m)
2021	Batchu and Seetha	LR, DT, GB, KNN and SVM	CICDDOS2019	%99,97 (b)
2022	Patil ve diğerleri	DT, MLR, NB, and RF	CICDDOS2019	%89,05 (m)
2021	Haq ve diğerleri	CNN	NSL-KDD	%99,34 (b), %99,13 (m)
2021	Iwendi ve diğerleri	LSTM	CICDDOS2019	%99,97 (b)
2021	Gamal ve diğerleri	CNN	UNSW-NB15 and Bot-IoT	%99,9 (b)

Çizelge 3.1. (devam) DDoS tespit edilmesine yönelik yapılan çalışmalar

2021	Gad ve diğerleri	LR, NB, DT, RF, AdaBoost, KNN, SVM and XGBoost	TON_IoT	%99,3 (b), %98,6 (m)
2022	Disha and Waheed	DT, AdaBoost, GBT, MLP, LSTM and GRU	TON_IoT, UNSW-NB 15	%99,98 (b)
2023	Kaur ve diğerleri	AdaBoost, LR and KNN	CICDDOS2019, TON_IoT, IoTID20, N-BaIoT, UNSW-NB15	%99,98 (b) (LDAP), %99,95 (b) (DDoS)
2023	Verme and Chandra	Extra Tree, KNN and Quadratic Discriminant Analysis	CICDDOS2019, TON_IoT, NSL-KDD, IoTID20, NBaIoT2018, UNSW_NB15	%99,9988 (b) (NTP), %99,9851 (b) (DDoS)
2023	Neto ve diğerleri	RF, DNN, MLP, LR, AdaBoost	CICIoT2023	%99,68 (b), %99,43(m) (8 classes)

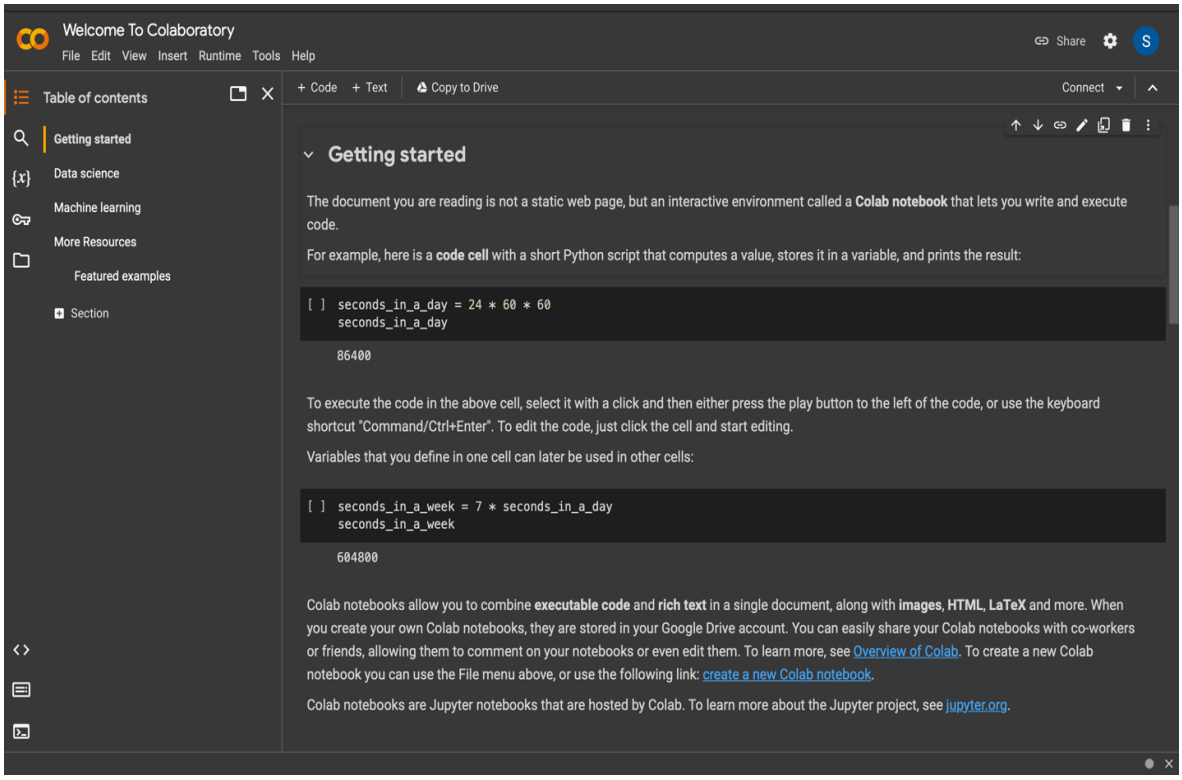
Yukarıdaki çalışmalar incelendiğinde, makine ve derin öğrenme modellerini kullanarak ağ trafiğinin analizi ve DDoS saldırılarının tespiti yapabilen IDS'lerin yüksek başarı oranına ulaşmış olduğu gözlemlenmiştir. Sunulan çalışmada makine öğrenme ve derin öğrenme algoritmaları karşılaştırılmıştır. Ayrıca CNN ve LSTM derin öğrenme algoritmaları birleştirilerek geliştirilen hibrit algoritmanın değerlendirilmesi yapılmıştır. Tüm bu çalışmalar büyük veri ortamında geliştirilmiştir. Geliştirilen hibrit algoritmanın hem binary hem de çoklu olarak sınıflandırma performansı ölçülmüştür.

4. KULLANILAN ARAÇLAR ve VERİ SETLERİ

Çalışmada kullanılan CICIoT2023 ve TON_IOT verisetlerine dair bilgilere bu bölümde yer verilmiştir.

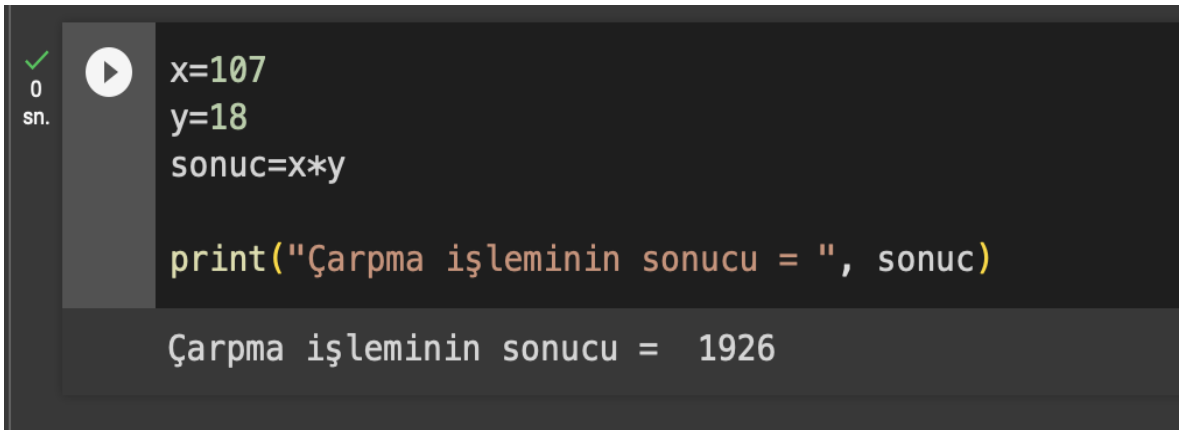
4.1. Google Colaboratory

Google Colaboratory [42], kısa ismiyle Colab, web üzerinde çalışan bir geliştirme ortamıdır. Python dilinde kodlama yapılmasına imkan sağlayan ve aynı zamanda zengin metin, resimler, HTML, LaTeX ve diğer öğeleri tek bir dokümanda birleştirmenizi sağlayan Colab not defteri adında etkileşimli bir ortamdır. Şekil 4.1'de Colab ortamının genel görümü sunulmuştur. Yapay zekâ çalışmaları için sıklıkla kullanılabilen Colab, arka planda da güçlü bir sunucu hizmeti sunar. GPU ve TPU donanımlarını ücretsiz bir şekilde sunarak kullanıcıların güçlü makinelere ulaşmasını sağlar. Bu donanımların kullanımı hiçbir yapılandırma gerekmeksizin kullanılır. Yapay zekâ ve büyük veri çalışmalarında kullanılacak donanım önem arz etmektedir. Çünkü, büyük veri ile çalışmak ve bu verileri yapay zekâ algoritmalarında kullanmak yüksek donanım kaynağı tüketen işlerdir. Google, bu ortamı sunarak araştırmacılara önemli bir katkı yapmaktadır. Sunulan bu ortamlarda, çalışmalarda kullanılabilecek Pandas, Keras, TensorFlow, Scikit-Learn, Numpy, Matplotlib gibi birçok kütüphane yüklü olarak kullanmaktadır.



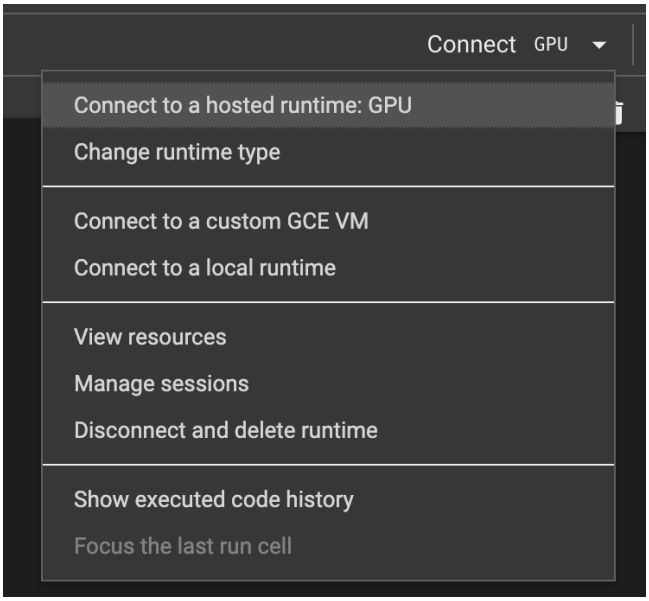
Şekil 4.1. Google colaboratory genel görüntüsü

Colab, not defteri formatında bir ortama sahiptir. İçerisinde Jupyter not defterleri barındırarak araştırmacıların hücreye eklenecek kod satırını bölüm bölüm çalıştırma imkânı sunar. Şekil 4.2'de Colab not defterinin kod hücresi verilmiştir. Kod hücresinin sadece kendisi çalıştırılabilmektedir. Kodun çıktısı ise hemen altında yazdırılır. Böylece çalışma sırasında küçük kod parçalarında çalışma imkânı sunarak, işlerin adım adım kontrollü bir şekilde yapılmasını sağlar.



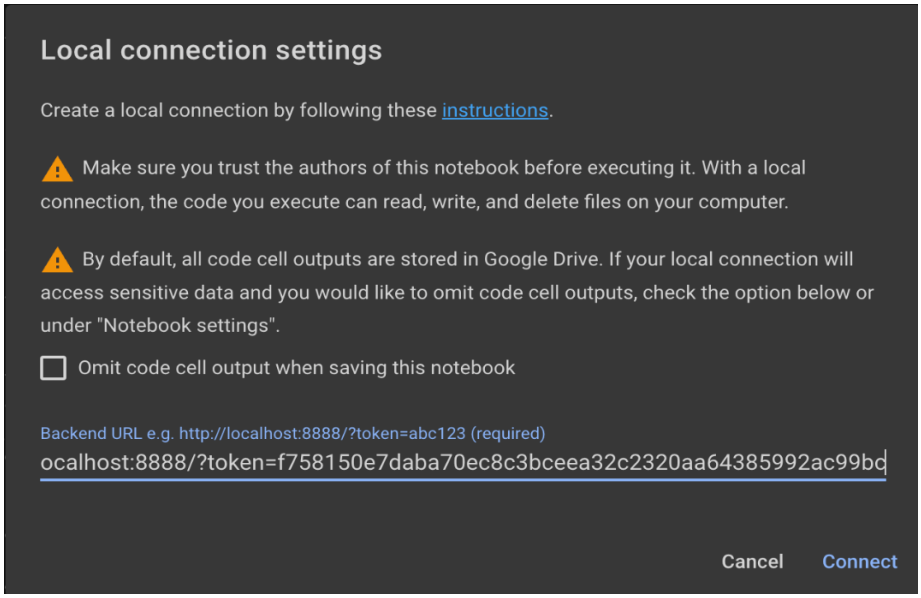
Şekil 4.2. Colab not defteri kod hücresi

Colab, yazılacak kod parçalarının çalıştırmak için üç farklı seçenek sunmaktadır. Bunlar, herkesin ücretsiz olarak bağlanabileceği sınırlı kaynaklara sahip seçenektir. Şekil 4.3'te gösterilen "Connect to a hosted runtime" seçeneği ile ücretsiz Google sanal makinelerine bağlanılabilir. Bir diğer seçenek ise daha güçlü GPU ve TPU'ları sunulduğu, uzun çalışma süresi sunan ücretli makinelerin kullanılmasıdır. Bu iki seçenekte de tam olarak ne kadar kaynak ayrıldığı ve bu kaynaklardan yararlanma süresi belirtilmediği için uzun süren bir çalışmanın ardından, çalışma sonlanmadan kaynak kullanımı kesilebilmekte ve sonuca ulaşılmadan tekrar başlatmaya ihtiyaç duyulabilmektedir. Büyük veri ile çalışma sırasında ise yüksek kaynak tüketimi oluşabileceğinden, Colab sanal makinelerin kaynakları yetmeyebilmektedir.



Şekil 4.3. Colab sanal makinelere bağlanılması

Colab'ın bir diğer çalıştırma seçeneği ise yerel kaynaklara bağlanabilme imkânı sunmasıdır. Çalıştırılacak makinenin kaynaklarını kullarak kodun yürütülmesi sağlanabilir. "Connect to a local runtime" seçeneği tıklanarak, Colab'in yerel kaynağa bağlanması sağlanabilir. Kullanıalcak local makinede ise Jupyter Notebook kurulması ve gösterilen bağlanma talimatları takip edilerek Colab'e bağlanması sağlanmalıdır. Şekil 4.4'te yerel kaynaklara bağlanma ekranı gösterilmiştir.



Şekil 4.4. Colab yerel kaynaklara bağlanılması

4.2. Python

Python programlama dili, veri bilimi alanında en yaygın olarak tercih edilen, nesne yönelimli ve dinamik semantiğe sahip bir üst düzey programlama dilidir. Veri çıkarımı, veri ön işleme, sınıflandırma algoritmalarının eğitimi, modelin dışa aktarılması ve başka ortamlarda kullanılması gibi birçok konuda Python'un zengin bir kütüphane ekosistemine sahip olması büyük avantaj sağlar. Bu özellikler, çalışmalarda hem model geliştirme aşamasında hem de uygulama aşamasında etkili bir şekilde kullanılmasına imkan tanır [43].

Python'un dünya çapında araştırmacılar ve yazılımcılar tarafından kullanılan en popüler kodlama dillerinden biri olmasının nedenleri şöyle açıklanabilir. İlk ve temel özellik, Python'un anlaşılır ve okunabilir bir sözdizimine sahip olmasıdır. Bu özellik, geliştiricilerin kod yazma ve anlama süreçlerini kolaylaştırarak projelerin daha hızlı ve etkili bir şekilde geliştirilmesine imkan tanır. Python'un bir diğer önemli avantajı geniş kütüphane desteğidir. Pandas, NumPy, Matplotlib gibi kütüphaneler, güçlü araçlar sunarak veri hazırlama ve analizi konusunda destek sağlar. Scikit-Learn, Keras ve TensorFlow gibi kütüphaneler, araştırmacıların makine öğrenimi ve derin öğrenme algoritmalarının kolayca kullanmasını sağlar. Bu hazır kütüphaneler, geliştiricilere zaman kazandırır ve karmaşık işlemleri basitleştirir. Gelişmiş topluluk desteği de Python'u diğer dillere göre avantajlı kılar. Son olarak, Python'un çapraz platform desteği, farklı işletim sistemlerinde sorunsuz

çalışabilmesine olanak tanır, bu da geliştiricilere projelerini çeşitli platformlarda kullanma esnekliği sağlar. Tüm bu özellikler bir araya geldiğinde, Python'un genel amaçlı bir programlama dilinden öte, özellikle veri bilimi ve yapay zeka alanlarında vazgeçilmez bir araç olduğunu görülebilir. Python, öğrenmesi kolay olmasıyla yeni başlayanlar için uygun olmanın yanı sıra, deneyimli geliştiricilere güçlü bir araç seti sunmasıyla da programlama dünyasında önemli bir konuma sahiptir.

Bir sonraki bölümde ise çalışmada kullanılan Python kütüphaneleri olan Pandas, Scikit-Learn, Keras, Matplotlib, Seaborn ve Numpy hakkında bilgi verilecektir.

4.3. Kullanılan Kütüphaneler

Yapılan çalışmada Pandas, Scikit-Learn, Keras, Matplotlib, Seaborn ve Numpy gibi kütüphaneler kullanılmıştır. Bu bölümde, bunlar hakkında bilgi verilecektir.

4.3.1. Pandas ve Numpy

Pandas, Python programlama dilinde yaygın olarak kullanılan ve özellikle veri işlemleri ve analizi için tasarlanmış açık kaynaklı bir kütüphanedir. Pandas'da Series ve DataFrame olmak üzere iki temel veri yapısı bulunur. Series, tek boyutlu dizileri temsil ederken, DataFrame çok boyutlu bir tabloyu ifade eder. Bu yapılar, büyük veri setleriyle çalışanlar için esnek ve güçlü bir çözüm sunar. DataFrame yapısı, kullanılacak verileri tablo halinde düzenleyerek işlemlerin sade ve etkili bir şekilde yapılmasına olanak tanır. DataFrame, veri setinin genel özelliklerini ön inceleme ve anlama fırsatı sağlar. Ayrıca, bu yapının içerdiği fonksiyonlar sayesinde, veri setindeki eksik değerleri ele alma, filtreleme, sıralama ve gruplama gibi çeşitli işlemler kolaylıkla gerçekleştirilebilir. Pandas, veri manipülasyonu ve analizi konularında büyük veri ile çalışanlar için vazgeçilmez araçlarından biridir. Pandas kütüphanesinin kullanımına ait örnek Şekil 4.5’de verilmiştir. Burada bir veri setinin okunması ve bunu DataFrame olarak alma işlemi gösterilmiştir. Ayrıca head() komutu ile DataFrame'in ilk 5 satırının gösterimi yapılmıştır.

```
dataset = pd.read_csv("./Desktop/dataSet.csv")
dataset.head()
```

	ts	Processor_DPC_Rate	Processor_pct_ Idle_Time	Processor_pct_ C3_Time	Processor_pct_ Interrupt_Time
0	1554206309	4	29.90817156	0	0.078240397
1	1554206319	9	31.75168186	0	0.312520973
2	1554206329	5	29.49516707	0	1.16822183
3	1554206339	12	18.22437505	0	1.097191902
4	1554206349	12	14.86118688	0	1.562431019

5 rows × 127 columns

Şekil 4.5. Pandas kütüphanesinin kullanımı

Numpy, Python diline ait veri işleme ve matematik kütüphanesidir. Özellikle matris ve dizi işlemlerinde kullanışlı ve performanslı çözümler sunmaktadır. Numpy ile, dizi işlemleri, veri işlemleri, matris işlemleri, ikili işlemler, rasgele sayı üretimi, istatistiki işlemler ve benzeri birçok matematiksel işlem gerçekleştirilebilmektedir. Numpy aracı Google Colab ortamında kendiliğinden bulunduğundan, kurulum işlemi gerekmemektedir [44]. Numpy aracının kullanımına ait örnek Şekil 4.6'da gösterilmektedir.

```
import numpy as np

array = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print("Oluşturulan Array'in boyutu= ",array.shape)
```

```
Oluşturulan Array'in boyutu=  (2, 4)
```

Şekil 4.6. Numpy kütüphanesinin kullanımı

4.3.2. TensorFlow ve Scikit-Learn

TensorFlow, Google çalışanları tarafından geliştirilen açık kaynaklı bir uygulama kütüphanesidir ve makine öğrenmesi ile derin öğrenme çalışmalarında geniş bir kullanım alanına sahiptir. TensorFlow, sistem ve grafik belliğini en verimli bir şekilde kullanacak veri setleri oluşturabilme yeteneğine sahiptir. Bu veri setleri oluşturulan model tarafından

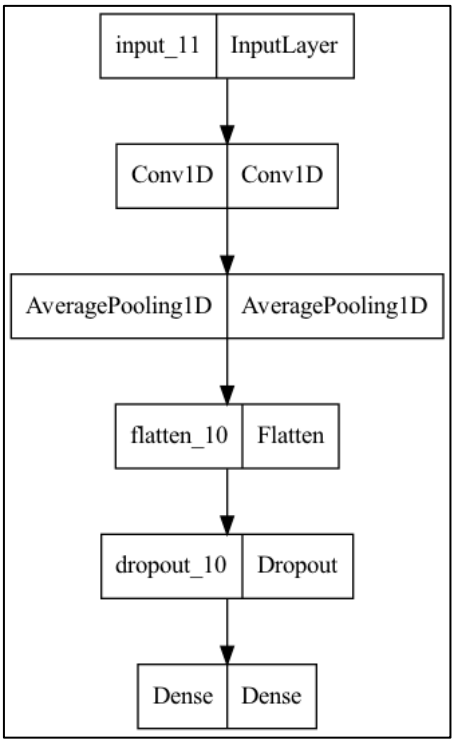
kullanılmadan önce, ön işleme kütüphanesinde bulunan fonksiyonlar aracılığıyla veri setleri işlenebilir [44]. TensorFlow'un sunduğu bu esneklik, çeşitli veri işleme ihtiyaçlarını karşılamak için kullanıcılara geniş bir yelpaze ortaya koymaktadır. Temelde Python kullanarak oluşturulan TensorFlow kütüphanesi, şimdilerde R, Java, Javascript, C++ ve C# gibi popüler birçok dili desteklemektedir.

TensorFlow'un güncel sürümlerinde, Keras kütüphanesi de entegre bir şekilde çalışmaktadır. Keras, bir derin öğrenme algoritmaları kütüphanesidir ve TensorFlow'un üzerinde çalışır. Python aracılığı ile yazılmış ve hemen hemen tüm derin öğrenme modelleri oluşturulabilmektedir. Keras, derin öğrenme algoritmaları oluştururken tekrar eden ve karmaşık arka plan kodları daha basit ve kullanıcı dostu bir hale getirmektedir. Geliştirici sadece katmanları birbirine bağlamakta ve arka planda gerçekleşen birçok kompleks sürece dahil olması gerekmemektedir. Google Colab ortamında Tensorflow ve Keras kütüphaneleri hazır bulunduğundan kuruluma ihtiyaç yoktur.

```
import tensorflow as tf
from tensorflow import keras
from keras import layers
from keras.layers import Dropout


def keras_model():
  model = keras.Sequential(name="keras_model")
  model.add(keras.layers.Conv1D(filters=128, kernel_size=2, activation='relu', name="Conv1D"))
  model.add(keras.layers.AveragePooling1D(pool_size=2, name="AveragePooling1D"))
  model.add(keras.layers.Flatten())
  model.add(Dropout(0.3))
  model.add(keras.layers.Dense(2, activation='softmax', name="Dense"))
  model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['Accuracy', 'Recall', 'Precision'])
  return model
```

Şekil 4.7. Tensorflow ile derin öğrenme modelinin oluşturulması

Şekil 4.7'de Tensorflow ve Keras kütüphaneleri kullanılarak oluşturulan basit bir derin öğrenme modeli gösterilmektedir. Öncelikle Tensorflow ve kullanılacak Keras kütüphaneleri yazılıma dahil edilmelidir. Daha sonra Keras'ın içerisindeki alt kütüphanlerin (layers, dropout vb) kullanılabilmesi için gerekli fonksiyonlar da yazılıma dahil edilmelidir. Keras'ın sunduğu kolaylık sebebiyle sıralı bir şekilde model oluşturulabilir. Şekil-Z10'da, oluşturulan mıdel CNN, pooling (havuzlama), flatten (düzleştirme), dropout (yoksayma) ve dense katmanlarından oluştuğu görülmektedir. Tüm bu katmanlar sıralı bir şekilde kolayca birbirlerine bağlanabilirler. Şekil 4.8'de ise yukarıda oluşturulan modelin görselleştirilmiş hali sunulmuştur.



Şekil 4.8. Keras ile oluşturulan modelin görselleştirilmesi

Scikit-learn, Python programlama dilinde yazılmış ve ücretsiz olarak kullanılabilen bir makine öğrenmesi kütüphanesidir. Bu kütüphane, karar ağacı, rastgele orman ve lojistik regresyon gibi çeşitli makine öğrenmesi algoritmalarını içermektedir. Scikit-learn, sadece model eğitimi değil aynı zamanda modelin performansını değerlendirmek ve veri setinde eksik değerleri doldurmak gibi veri seti temizleme ve analiz işlemlerde de kullanılır. Şekil 4.9'da verilen örnekte Scikit-Learn kütüphanesinin veri seti üzerinde yapabildiği bazı işlemler gösterilmiştir. Scikit-Learn kütüphanesi ile verisetinin kategorik değişkenleri sayısallaştırılabilir, tüm değerler belli bir değer aralığına oranlanabilir ve veri setleri farklı oranlarda bölünme işlemi yapılabilir. Bu kütüphane, kullanıcılarına esneklik ve geniş bir özellik yelpazesi sunarak, çeşitli makine öğrenimi algoritmalarının daha kolay ve etkili bir şekilde gerçekleştirmelerine olanak tanır. Scikit-learn, açık kaynak olması ve geniş bir topluluk tarafından desteklenmesi sayesinde kullanıcıların en son makine öğrenimi yöntemlerine erişimini sağlar.

```
[10] # X veri kümesi üzerinde Label Encode işleminin yapılması
     for i in X.columns:
         X[i] = LabelEncoder().fit_transform(X[i])

[ ]  # X veri kümesi üzerinde Scale işleminin yapılması
     scaler = StandardScaler()
     scaler.fit(X)
     X_N = scaler.transform(X)

[ ]  # Oluşturulan datasetleri eğitim ve test olarak ayrılması
     X_train, X_test, y_train, y_test = train_test_split(X_N, y, test_size = 0.2)
```

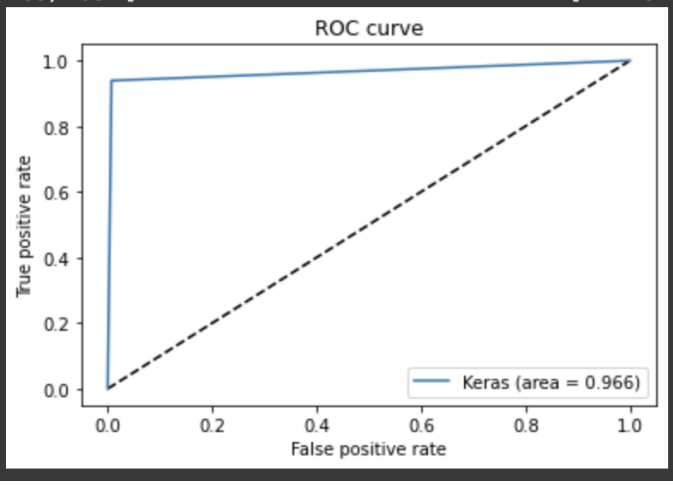
Şekil 4.9. Scikit-Learn kütüphanesinin kullanımı

4.3.3. Matplotlib ve Seaborn

Matplotlib açık kaynaklı bir Python 2D çizim kütüphanesidir ve farklı formatlarda grafik çizimleri üretebilir. Bu kütüphane sayesinde grafikler, histogramlar, güç spektrumları, çubuk grafikler, hata grafikleri, dağılım grafikleri gibi birçok görselleştirme işlemi birkaç satır kod ile kolayca oluşturulabilir [45]. Şekil 4.10'da MatPlotLib ile oluşturmuş grafik örneği gösterilmektedir. MatPlotLib kütüphaneleri, Colab ortamında hazır bulunduğundan kurulum işlemi gerektirmemektedir.

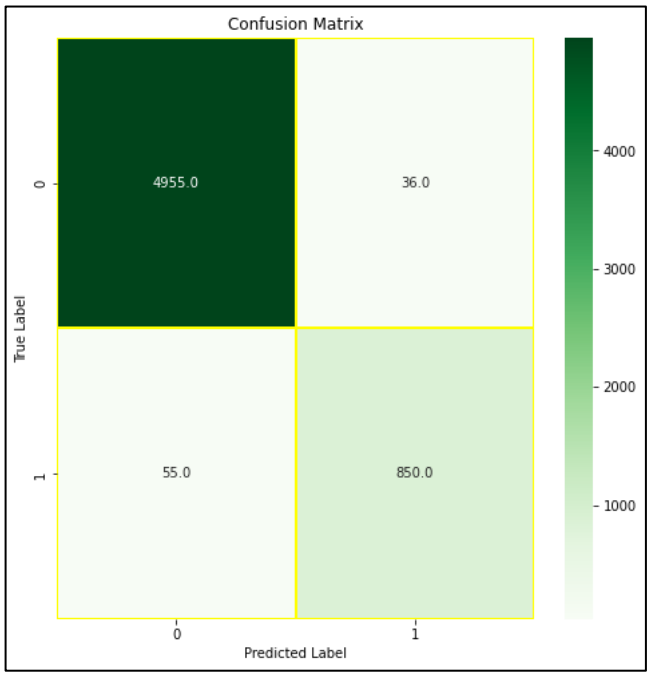
```
import matplotlib.pyplot as plt

y_pred = hybrid_model.predict(X_test_R)
y_pred_classes = np.argmax(y_pred,axis = 1)
y_true = np.argmax(y2_test,axis = 1)
fpr_keras, tpr_keras, thresholds_keras = roc_curve(y_true, y_pred_classes)
auc_keras = auc(fpr_keras, tpr_keras)
plt.figure(1)
plt.plot([0, 1], [0, 1], 'k--')
plt.plot(fpr_keras, tpr_keras, label='Keras (area = {:.3f})'.format(auc_keras))
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('ROC curve')
plt.legend(loc='best')
plt.show()
```

```
185/185 [==============================] - 2s 8ms/step
```



Şekil 4.10. MatPlotLib ile oluşturulmuş grafik örneği

Seaborn kütüphanesi ise görselleştirmede kullanılan bir diğer Python kütüphaneleridir. Seaborn kütüphanesi oluşturulurken MatPlotLib kütüphanesi baz alınmıştır. Bu sebeple, her iki kütüphane de büyük oranda birbirine bütünleşik ve birlikte çalışabilmektedir. Seaborn kütüphanesi de, Colab ortamında hazır bulunur ve kurulum işlemi gerektirmez [44]. Şekil 4.11'de Seaborn aracıyla oluşturulmuş ısı haritası gösterilmektedir.



Şekil 4.11. Seaborn kullanılarak oluşturulmuş ısı haritası

4.4. CICIoT2023 Veri Seti

Bu veri seti Neto ve arkadaşları [41] tarafından üretilerek Univercity of New Brunswick (UNB) - Canadian Institute for Cybersecurity (CIC) database'inde yayınlanmıştır. 105 cihazdan oluşan IoT topolojisi kurularak, 33 farklı saldırı türü gerçekleştirilmiştir. Bu saldırılar DDoS, DoS, Recon, Web-based, Brute Force, Spoofing ve Mirai olmak üzere yedi kategoride sınıflandırılmıştır. Veri seti 47 özellik içermektedir. Çizelge 4.1'de bu özelliklerin detayı verilmiştir.

Çizelge 4.1. CICIoT2023 verisetideki özelliklerin tanımları

Özellik	Açıklama
ts	zaman damgası
flow_duration	Paket akışının süresi
Header_Length	Başlık Uzunluğu
ProtocolType	IP, UDP, TCP, IGMP, ICMP, Bilinmeyen (Tamsayı)
Duration	Yaşam Süresi (ttl)
Rate	Bir akıştaki paket aktarım hızı
Srate	Bir akışta giden paket iletim hızı
Drate	Bir akışta gelen paketlerin iletim hızı

Çizelge 4.1. (devam) CICIoT2023 verisetideki özelliklerin tanımları

fin_flag_number	Fin flag değeri
syn_flag_number	Syn flag değeri
rst_flag_number	Rst flag değeri
psh_flag_number	Psh flag değeri
ack_flag_number	Ack flag değeri
ece_flag_number	Ece flag değeri
cwr_flag_number	Cwr flag değeri
ack_count	Aynı akışta Ack flag ayarlanmış paketlerin sayısı
syn_count	Aynı akışta syn flag ayarlanmış paketlerin sayısı
fin_count	Aynı akışta fin flag ayarlanmış paketlerin sayısı
urg_count	Aynı akışta urg flag ayarlanmış paketlerin sayısı
rst_count	Aynı akışta rst flag ayarlanmış paketlerin sayısı
HTTP	Uygulama katmanı protokolünün HTTP olup olmadığını gösterir.
HTTPS	Uygulama katmanı protokolünün HTTPS olup olmadığını gösterir.
DNS	Uygulama katmanı protokolünün DNS olup olmadığını gösterir.
Telnet	Uygulama katmanı protokolünün Telnet olup olmadığını gösterir.
SMTP	Uygulama katmanı protokolünün SMTP olup olmadığını gösterir.
SSH	Uygulama katmanı protokolünün SSH olup olmadığını gösterir.
IRC	Uygulama katmanı protokolünün IRC olup olmadığını gösterir.
TCP	Uygulama katmanı protokolünün TCP olup olmadığını gösterir.

Çizelge 4.1. (devam) CICIoT2023 verisetideki özelliklerin tanımları

UDP	Uygulama katmanı protokolünün UDP olup olmadığını gösterir.
DHCP	Uygulama katmanı protokolünün DHCP olup olmadığını gösterir.
ARP	Uygulama katmanı protokolünün ARP olup olmadığını gösterir.
ICMP	Uygulama katmanı protokolünün ICMP olup olmadığını gösterir.
IPv	Uygulama katmanı protokolünün IPv olup olmadığını gösterir.
LLC	Uygulama katmanı protokolünün LLC olup olmadığını gösterir.
Totsum	Akıştaki paket uzunluklarının toplamı
Min	Akıştaki minimum paket uzunluğu
Max	Akıştaki maksimum paket uzunluğu
AVG	Akıştaki ortalama paket uzunluğu
Std	Akıştaki paket uzunluğunun standart sapması
Totsize	Paket uzunluğu
IAT	Önceki paketle saat farkı
Number	Akıştaki paket sayısı
Magnitue	(Akıştaki gelen paketlerin uzunluklarının ortalaması + Akıştaki giden paketlerin uzunluklarının ortalaması) **0,5
Radius	(Akıştaki gelen paketlerin uzunluklarının varyansı + Akıştaki giden paketlerin uzunluklarının varyansı) **0,5
Covariance	Gelen ve giden paketlerin uzunluklarının kovaryansı
Variance	Akışta gelen paketlerin uzunluklarının değişimi / Akışta giden paketlerin uzunluklarının değişimi
Weight	(Gelen paket sayısı) * (Giden paket sayısı)

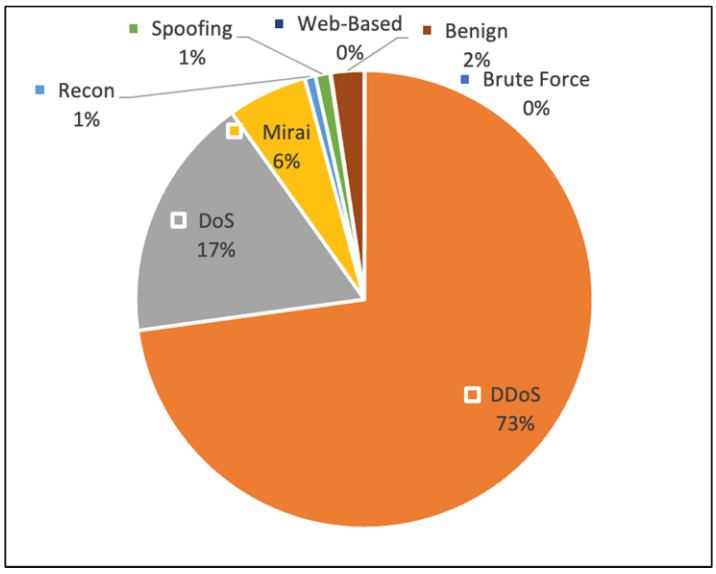
Çizelge 4.2’de veri setinde bulunan saldırıların miktarı ve ait oldukları saldırı sınıfları verilmiştir. Şekil 4.12'de ise saldırı sınıfına göre atakların dağılım grafiği verilmiştir. Şekilde görüleceği üzere DDoS sınıfı %73 ile veri setinde en fazla olan sınıftır. Bunu %17 ile DoS sınıfı takip etmektedir. Web-based ve Brute Force saldırı sınıfları ise %1'in altındadır.

Çizelge 4.2. CICIoT2023 verisetindeki atakların miktarı

Etiket	Sınıf	Miktar
DDoS-ICMP_Flood	DDoS	7200047
DDoS-UDP_Flood	DDoS	5411768
DDoS-TCP_Flood	DDoS	4497763
DDoS-PSHACK_Flood	DDoS	4094563
DDoS-SYN_Flood	DDoS	4059403
DDoS-RSTFINFlood	DDoS	4045410
DDoS-SynonymousIP_Flood	DDoS	3598454
DoS-UDP_Flood	DoS	3318467
DoS-TCP_Flood	DoS	2671471
DoS-SYN_Flood	DoS	2028995
BenignTraffic	Normal	1098282
Mirai-greeth_flood	Mirai	991846
Mirai-udpplain	Mirai	890708
Mirai-greip_flood	Mirai	751891
DDoS-ICMP_Fragmentation	DDoS	452557
MITM-ArpSpoofing	Spoofing	307598
DDoS-UDP_Fragmentation	DDoS	286968
DDoS-ACK_Fragmentation	DDoS	285089
DNS_Spoofing	Spoofing	178902
Recon-HostDiscovery	Recon	134375
Recon-OSScan	Recon	98269
Recon-PortScan	Recon	82267
DoS-HTTP_Flood	DoS	71844

Çizelge 4.2. CICIoT2023 verisetindeki atakların miktarı

VulnerabilityScan	Recon	37379
DDoS-HTTP_Flood	DDoS	28795
DDoS-SlowLoris	DDoS	23414
DictionaryBruteForce	Brute Force	13048
BrowserHijacking	Web-Based	5858
CommandInjection	Web-Based	5419
SqlInjection	Web-Based	5253
XSS	Web-Based	3852
Backdoor_Malware	Web-Based	3221
Recon-PingSweep	Recon	2262
Uploading_Attack	Web-Based	1253
	Toplam	46686691



Şekil 4.12. CICIoT2023 verisetindeki atakların oranları

4.5. TON_IOT Veri Seti

Güncellenmiş bir IoT veri kümesi olan TON_IoT [46] veri seti, 2019 yılında UNSW Canberra Cyber kurumunun IoT Laboratuvarında gerçekçi ve büyük ölçekli bir test ortamında toplanmıştır. Veri kümesi, scanning, DoS, DDoS, ransomware, backdoor, injection, cross-site scripting (XSS), password cracking, and Man-In-The-Middle (MITM)

saldırıları gibi bir dizi modern IoT saldırısını içerir. Bu çalışmada veri setinin alt kümesi olan Processed Windows 10 veri kümesi kullanılmıştır.

Oluşturulan veri seti kümesi, performans izleme aracı kullanılarak sahip olduğu ile disk, process, işlemci, memory gibi bilgiler çıkartılarak csv formatında, yapay zekâ eğitim ve testlerine daha uygun bir hale getirmiştir. Windows 10 veri setinde 125 features ve 2 class label içermektedir. Çizelge 4.3'te bu özelliklerin tanımları verilmiştir [47]. Çizelge 4.4'te verildiği gibi, Windows 10 veri seti için toplanan 35975 kayıt bulunmaktadır.

Çizelge 4.3. TON_IOT verisetideki özelliklerin tanımları

Id	Özellik	Açıklama
1	ts	Ağ verileri için yakalanan bağlantının zaman damgası
2	Processor_DPC_Rate	Ertelenmiş prosedür çağrılarını (DPC'ler) almak ve hizmet vermek için harcanan tek bir işlemcinin zaman oranı
3	Processor_pct_ Idle_Time	Herhangi bir program tarafından kullanılmayan işlemcinin boşta kalma süresi
4	Processor_pct_ C3_Time	Saat üretecinin ve işlemcinin önbelleğini uyumlu tutmasının gerekmediği işlemci derin uyku durumunun zaman oranı
5	Processor_pct_ Interrupt_Time	İşlemcinin örnekleme aralıkları sırasında donanım kesintilerini almak ve bunlara hizmet vermek için harcadığı zaman oranı
6	Processor_pct_ C2_Time	Çekirdek ve veri yolu saatlerinin kapalı olduğu ve işlemcinin yazılım tarafından görülebilen tüm durumu koruduğu ancak uyanmasının daha uzun sürebileceği işlemci durdurma saati durumunun zaman oranı
7	Processor_pct_ User_Time	İşlemcinin kullanıcı modunda geçirdiği sürenin yüzdesi. Kullanıcı modu uygulamalar, ortam alt sistemleri ve tümleşik alt sistemler için tasarlanmış sınırlı bir işleme modudur. Bu sayaç ortalama meşgul süresini örnek zamanın yüzdesi olarak görüntüler.
8	Processor_pct_ C1_Time	İşlemcinin C1 düşük güçte boşta durumunda harcadığı sürenin yüzdesi. % C1 Süresi, toplam işlemci boşta kalma süresinin bir alt kümesidir.
9	Processor_pct_ Processor_Time	İşlemcinin Boşta olmayan bir iş parçacığını yürütmek için harcadığı geçen sürenin yüzdesi. İşlemcinin boştaki iş parçacığını yürütmek için harcadığı sürenin yüzdesi ölçülerek ve ardından bu değer %100'den çıkarılarak hesaplanır. Bu sayaç, işlemci etkinliğinin birincil göstergesidir ve örnek aralık sırasında gözlemlenen meşgul sürenin ortalama yüzdesini görüntüler.
10	Processor_C1_ransitions_sec	CPU'nun C1 düşük güç boşta durumuna girme hızı. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
11	Processor_pct_ DPC_Time	Örnek aralık sırasında işlemcinin ertelenmiş prosedür çağrılarını (DPC'ler) almak ve bunlara hizmet vermek için harcadığı sürenin yüzdesi. DPC'ler standart kesintilerden daha düşük öncelikte çalışan kesintilerdir. Bu sayaç ortalama meşgul süresini örnek zamanın yüzdesi olarak görüntüler.
12	Processor_C2_ransitions_sec	CPU'nun C2 düşük güç boşta durumuna girme hızı. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

13	Processor_pct_ Privileged_Time	İşlem iş parçacıklarının ayrıcalıklı modda kod yürütmek için harcadığı geçen sürenin yüzdesi.
14	Processor_C3_ransitions_sec	CPU'nun C3 düşük güç boşta durumuna girme hızı. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
15	Processor_DPCs_Queued_sec	Ertelenmiş prosedür çağrılarının (DPC'ler) işlemcinin DPC kuyruğuna eklendiği saniye başına olay cinsinden ortalama oran. DPC'ler standart kesintilerden daha düşük öncelikte çalışan kesintilerdir. Bu sayaç, kuyruktaki DPC sayısını değil, DPC'lerin kuyruğa eklenme hızını ölçer. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
16	Processor_Interrupts_sec	İşlemcinin aldığı ve bakımı yapılan donanımın kesintiye uğradığı saniye başına ortalama olay oranı. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
17	Process_Pool_Paged Bytes	Disk belleği havuzunun bayt cinsinden boyutu; sistem sanal belleğinde, kullanılmadıklarında diske yazılabilen nesneler için kullanılan alan. Bellek/Havuzda Sayfalanan Bayt Sayısı, İşlem/Havuzda Sayfalanan Bayt sayısından farklı şekilde hesaplanır; dolayısıyla eşit olmayabilir. İşlem(_Toplam)/Havuzda Sayfalanan Bayt Sayısı. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
18	Process_IO Read_Operations_sec	Bu sayaç, dosya, ağ ve cihaz I/O'lerini içerecek şekilde işlem tarafından oluşturulan tüm I/O etkinliklerini sayar.
19	Process_Working_Set_ Private	Yalnızca bu işlem için kullanılan ve diğer işlemler tarafından paylaşılmayan veya paylaşılamayan çalışma kümesinin bayt cinsinden boyutu.
20	Process_Working_Set_Peak	Herhangi bir zamanda bu işlemin Çalışma Kümesinin bayt cinsinden maksimum boyutu. Çalışma Seti, süreçteki iş parçacıkları tarafından yakın zamanda dokunulan hafıza sayfaları kümesidir.
21	Process_IO_Write Operations_sec	Süreç, yazma I/O işlemlerini gerçekleştirmedir. Bu sayaç, dosya, ağ ve cihaz I/O'lerini içerecek şekilde işlem tarafından oluşturulan tüm I/O etkinliklerini sayar.
22	Process_Page_File Bytes	Bu işlemin disk belleği dosyasında/dosyalarında kullanılmak üzere ayırdığı bayt cinsinden geçerli sanal bellek miktarı. Disk belleği dosyaları, işlem tarafından kullanılan ve diğer dosyalarda bulunmayan bellek sayfalarını depolamak için kullanılır.
23	Process_pct_ User_Time	İşlem iş parçacıklarının kullanıcı modunda kod yürütmek için harcadığı geçen sürenin yüzdesi. Uygulamalar, ortam alt sistemleri ve entegre alt sistemler kullanıcı modunda yürütülür.
24	Process_Virtual_Bytes Peak	İşlemin herhangi bir zamanda kullandığı sanal adres alanının bayt cinsinden maksimum boyutu. Sanal adres alanının kullanılması, mutlaka diskin veya ana bellek sayfalarının karşılık gelen kullanımı anlamına gelmez. Ancak sanal alan sınırlıdır ve süreç, kütüphanelerin yükleme yeteneğini sınırlayabilir.
25	Process_Page_File Bytes Peak	Bu işlemin disk belleği dosyasında/dosyalarında kullanılmak üzere ayırdığı bayt cinsinden maksimum sanal bellek miktarı. Disk belleği dosyaları, işlem tarafından kullanılan ve diğer dosyalarda bulunmayan bellek sayfalarını depolamak için kullanılır.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

26	Process_IO_Other_Bytes_sec	İşlem, kontrol işlemleri gibi verileri içermeyen I/O işlemlerine bayt vermedir. Bu sayaç, dosya, ağ ve cihaz I/O'lerini içerecek şekilde işlem tarafından oluşturulan tüm I/O etkinliklerini sayar.
27	Process_Private_Bytes	Bu işlemin ayırdığı ve diğer işlemlerle paylaşılamayan belleğin bayt cinsinden geçerli boyutu.
28	Process_IO_Write_Bytes_sec	İşlem baytların I/O işlemlerine yazılmasıdır. Bu sayaç, dosya, ağ ve cihaz I/O'lerini içerecek şekilde işlem tarafından oluşturulan tüm I/O etkinliklerini sayar.
29	Process_Elapsed_Time	Bu işlemin yürütüldüğü saniye cinsinden toplam geçen süre.
30	Process_Virtual_Bytes	İşlemin kullandığı sanal adres alanının bayt cinsinden geçerli boyutu. Sanal adres alanının kullanılması, mutlaka diskin veya ana bellek sayfalarının karşılık gelen kullanımı anlamına gelmez.
31	Process_pct_ Processor_Time	Tüm işlem iş parçacıklarının talimatları yürütmek için işlemciyi kullandığı geçen sürenin yüzdesi. Talimat, bilgisayardaki temel yürütme birimidir, iş parçacığı talimatları yürüten nesnedir ve süreç bir program çalıştırıldığında oluşturulan nesnedir.
32	Process_Creating Process ID	Süreci oluşturan sürecin İşlem Kimliği. Oluşturma işlemi sonlandırılmış olabilir, bu nedenle bu değer artık çalışan bir işlemi tanımlamayabilir.
33	Process_Pool Nonpaged Bytes	Disk belleği olmayan havuzun bayt cinsinden boyutu; sistem sanal belleğinin, diske yazılamayan ancak ayrıldıkları sürece fiziksel bellekte kalması gereken nesneler için kullanılan alanı. Bellek/Disk Belleği Olmayan Havuz Bayt Sayısı, İşlem/Disk Belleği Olmayan Havuz Bayt Sayısından farklı şekilde hesaplanır, bu nedenle İşlem(_Toplam)/Havuz Disk Belleğe Alınmayan Bayt Sayısına eşit olmayabilir. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değildir.
34	Process_Working Set	Bu işlemin Çalışma Kümesinin bayt cinsinden geçerli boyutu. Çalışma Seti, süreçteki iş parçacıkları tarafından yakın zamanda dokunulan hafıza sayfaları kümesidir.
35	Process_Page Faults_sec	Bu süreçte yürütülen iş parçacıklarının sayfa hatalarının oluşma oranı. Bir iş parçacığı, ana bellekteki çalışma kümesinde olmayan bir sanal bellek sayfasına başvurduğunda sayfa hatası oluşur.
36	Process_ID Process	Bu işlemin benzersiz tanımlayıcısı. Kimlik İşlem numaraları yeniden kullanıldığından, yalnızca bir işlemin ömrü boyunca bir işlemi tanımlarlar.
37	Process_IO Other Operations_sec	İşlemin, okuma veya yazma işlemi olmayan G/Ç işlemlerini (örneğin bir kontrol işlevi) yayınlama hızı. Bu sayaç, dosya, ağ ve cihaz G/Ç'lerini içerecek şekilde işlem tarafından oluşturulan tüm G/Ç etkinliklerini sayar.
38	Process_IO Data Operations_sec	İşlemin okuma ve yazma G/Ç işlemlerini gerçekleştirme hızı. Bu sayaç, dosya, ağ ve cihaz G/Ç'lerini içerecek şekilde işlem tarafından oluşturulan tüm G/Ç etkinliklerini sayar.
39	Process_Thread Count	Bu süreçte şu anda etkin olan iş parçacığı sayısı. Talimat, bir işlemcideki temel yürütme birimidir ve iş parçacığı, talimatları yürüten nesnedir. Çalışan her işlemin en az bir iş parçacığı vardır.
40	Process_pct_ Privileged_Time	İşlem iş parçacıklarının ayrıcalıklı modda kod yürütmek için harcadığı geçen sürenin yüzdesi. Bir Windows sistem hizmeti çağrıldığında, hizmet, sistemin özel verilerine erişim sağlamak için genellikle ayrıcalıklı modda çalışır. Bu tür veriler, kullanıcı modunda yürütülen iş parçacıkları tarafından erişime karşı korunur.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

41	Process_IO Data Bytes_sec	G/Ç işlemlerinde işlemin bayt okuma ve yazma hızı. Bu sayaç, dosyayı içerecek şekilde işlem tarafından oluşturulan tüm G/Ç etkinliklerini sayar.
42	Process_IO Read Bytes_sec	İşlemin G/Ç işlemlerinden bayt okuma hızı. Bu sayaç, dosya, ağ ve cihaz G/Ç'lerini içerecek şekilde işlem tarafından oluşturulan tüm G/Ç etkinliklerini sayar.
43	Process_Priority Base	Bu sürecin mevcut temel önceliği. Bir süreç içindeki iş parçacıkları, sürecin temel önceliğine göre kendi temel önceliklerini yükseltebilir veya düşürebilir.
44	Process_Handle Count	Bu işlem tarafından şu anda açık olan toplam tanıtıcı sayısı. Bu sayı, bu işlemdeki her iş parçacığının o anda açık olan tanıtıcıların toplamına eşittir.
45	Network_I(Intel[R]_82574L_GNC)TCP_APS	Bu ağ arayüzündeki tüm TCP bağlantılarında alınan paketlerin bayt cinsinden ortalama boyutu.
46	Network_I(Intel[R]_82574L_GNC)/Packets Received Unknown	Arayüz aracılığıyla alınan ve bilinmeyen veya desteklenmeyen bir protokol nedeniyle atılan paketlerin sayısı.
47	Network_I(Intel[R]_82574L_GNC)/Bytes Received/sec	Çerçeveleme karakterleri de dahil olmak üzere her ağ bağdaştırıcısı üzerinden baytların alınma hızı. Ağ Arayüzü\Alınan Bayt/sn, Ağ Arayüzü\Toplam Bayt/sn'nin bir alt kümesidir.
48	Network_I(Intel[R]_82574L_GNC)/Bytes Sent/sec	Çerçeveleme karakterleri de dahil olmak üzere her ağ bağdaştırıcısı üzerinden baytların gönderilme hızı. Ağ Arayüzü\Gönderilen Bayt/sn, Ağ Arayüzü\Toplam Bayt/sn'nin bir alt kümesidir.
49	Network_I(Intel[R]_82574L_GNC)/Packets Outbound Errors	Hatalar nedeniyle iletilemeyen giden paketlerin sayısı.
50	Network_I(Intel[R]_82574L_GNC)/Packets Received Discarded	Daha yüksek katman protokolüne teslim edilmelerini önlemek için hiçbir hata tespit edilmemiş olmasına rağmen atılmak üzere seçilen gelen paketlerin sayısı. Paketleri atmanın olası bir nedeni arabellek alanını boşaltmak olabilir.
51	Network_I(Intel[R]_82574L_GNC)/Bytes Total/sec	Çerçeveleme karakterleri de dahil olmak üzere her ağ bağdaştırıcısı üzerinden baytların gönderilme ve alınma hızı.
52	Network_I(Intel[R]_82574L_GNC)/Packets Outbound Discarded	İletimi önlemek için hiçbir hata tespit edilmemiş olmasına rağmen atılmak üzere seçilen giden paketlerin sayısı. Paketleri atmanın olası bir nedeni arabellek alanını boşaltmak olabilir.
53	Network_I(Intel[R]_82574L_GNC)/TCP RSC Exceptions/sec	Bu ağ arayüzündeki tüm TCP bağlantılarında paket alma için RSC istisna oranı.
54	Network_I(Intel[R]_82574L_GNC)/Packets Sent Unicast/sec	Paketlerin üst düzey protokoller tarafından alt ağ unicat yayın adreslerine iletilmesinin talep edilme hızı. Oran, atılan veya gönderilmeyen paketleri içerir.
55	Network_I(Intel[R]_82574L_GNC)/Output Queue Length	Çıkış paketi kuyruğunun uzunluğu (paketler halinde). Eğer bu ikiden uzun olursa gecikmeler olur ve mümkünse darboğaz bulunup ortadan kaldırılmalıdır. Bu uygulamada istekler Network Driver Interface Specification (NDIS) tarafından kuyruğa alındığı için bu her zaman 0 olacaktır.
56	Network_I(Intel[R]_82574L_GNC)/Packets Received/sec	Ağ arayüzünde paketlerin alınma hızı.
57	Network_I(Intel[R]_82574L_GNC)/Current Bandwidth	Ağ arayüzünün saniye başına bit (BPS) cinsinden mevcut bant genişliği. Bant genişliği değişmeyen veya doğru tahminin yapılamadığı arayüzler için bu değer nominal bant genişliğidir.
58	Network_I(Intel[R]_82574L_GNC)/Packets/sec	Ağ arayüzünde paketlerin gönderilme ve alınma hızı.
59	Network_I(Intel[R]_82574L_GNC)/TCP Active RSC Connections	Bu ağ arabirimindeki RSC özellikli ağ bağdaştırıcısından şu anda büyük paketler alan TCP bağlantılarının (hem IPv4 hem de IPv6 üzerinden) sayısı.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

60	Network_I(Intel[R]_82574L_GNC)/Packets Sent/sec	Ağ arayüzünde paketlerin gönderilme hızı.
61	Network_I(Intel[R]_82574L_GNC)/Packets Received Unicast/sec	Alt ağlardaki unicast yayın paketlerinin daha yüksek katman protokolüne iletilme hızı.
62	Network_I(Intel[R]_82574L_GNC)/Packets Sent Non-Unicast/sec	Paketlerin unicast olmayan adreslere, üst düzey protokoller tarafından iletilmesinin talep edilme hızıdır. Bu orana reddedilen veya gönderilmeyen paketler dahildir.
63	Network_I(Intel[R]_82574L_GNC)/Packets Received Non-Unicast/sec	Unicast olmayan paketlerin daha yüksek katman protokolüne iletilme hızı.
64	Network_I(Intel[R]_82574L_GNC)/TCP RSC Coalesced Packets/sec	Bu ağ arayüzündeki tüm TCP bağlantılarında büyük paket alma hızı.
65	Network_I(Intel[R]_82574L_GNC)/Offloaded Connections	Şu anda TCP kanalı boşaltma özellikli ağ bağdaştırıcısı tarafından işlenen TCP bağlantılarının (hem IPv4 hem de IPv6 üzerinden) sayısı.
66	Network_I(Intel[R]_82574L_GNC)/Packets Received Errors	Daha yüksek katman protokolüne teslim edilmelerini engelleyen hatalar içeren gelen paketlerin sayısı.
67	Memory/Pool Paged Bytes	Disk belleği havuzunun bayt cinsinden boyutu; sistem sanal belleğinde, kullanılmadıklarında diske yazılabilen nesneler için kullanılan alan. Bellek/Havuzda Sayfalanan Bayt Sayısı hesaplanır. İşlem/Havuzda Sayfalanan Baytlardan farklı olduğundan, İşlem(_Toplam)/Havuzda Sayfalanan Bayt sayısına eşit olmayabilir. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
68	Memory/Free & Zero Page List Bytes	Boş ve sıfır sayfa listelerine atanan bayt cinsinden fiziksel bellek miktarı. Bu bellek önbelleğe alınmış veri içermiyor. Bir sürece tahsis edilmek veya sistem kullanımı için hemen kullanılabilir.
69	Memory/Cache Bytes Peak	Sistemin son yeniden başlatılmasından bu yana sistem dosya önbelleği tarafından kullanılan maksimum bayt sayısı. Bu, önbelleğin geçerli boyutundan daha büyük olabilir. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
70	Memory/System Code Resident Bytes	Şu anda fiziksel bellekte yerleşik ve etkin olan sayfalanabilir işletim sistemi kodunun bayt cinsinden boyutu. Bu değer Bellek/Sistem Kodu Toplam Baytının bir bileşenidir. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
71	Memory/Available Bytes	Bir işleme veya sistem kullanımına tahsis edilmek üzere anında kullanılabilen, bayt cinsinden fiziksel bellek miktarı. Bekleme (önbelleğe alınmış), boş ve sıfır sayfa listelerine atanan belleğin toplamına eşittir.
72	Memory/Commit Limit	Disk belleği dosyasını/dosyalarını genişletmeye gerek kalmadan kaydedilebilecek sanal bellek miktarı. Bayt cinsinden ölçülür. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
73	Memory/Transition Pages RePurposed/sec	Geçiş önbelleği sayfalarının sayısının farklı bir amaç için yeniden kullanılma hızı. Bu sayfaların özel veya paylaşılabilir bellek içerebileceği unutulmamalıdır.
74	Memory/Pages Output/sec	Fiziksel bellekte yer açmak için sayfaların diske yazılma hızı. Yüksek oranda sayfa çıkışı, bellek yetersizliğine işaret edebilir. Bu sayaç sayfa sayısını gösterir ve dönüşüm olmadan diğer sayfa sayılarıyla karşılaştırılabilir.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

75	Memory/Page Reads/sec	Sabit sayfa hatalarını çözmek için diskin okunma hızı. Her işlemde alınan sayfa sayısına bakılmaksızın okuma işlemlerinin sayısını gösterir. Bu sayaç, sistem genelinde gecikmelere neden olan hata türlerinin birincil göstergesidir. Her işlem sırasında okunan ortalama sayfa sayısını belirlemek için Bellek/Sayfa Okuma/sn değerini Bellek/Sayfa Girişi/sn değeriyle karşılaştırın.
76	Memory/Demand Zero Faults/sec	Arızanın giderilmesi için sıfırlanmış sayfanın gerekli olduğu hız. Sıfırlanmış sayfalar, yani önceden depolanan verilerden arındırılmış ve sıfırlarla doldurulmuş sayfalar, Windows'un, işlemlerin bellek alanını kullanan önceki işlemler tarafından depolanan verileri görmesini engelleyen bir güvenlik özelliğidir. Bu sayaç, arızanın giderilmesi için alınan sayfa sayısına bakılmaksızın arıza sayısını gösterir. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
77	Memory/Available KBytes	Bir işleme tahsis edilmek veya sistem kullanımı için anında kullanılabilen, Kilobayt cinsinden fiziksel bellek miktarı. Bekleme (önbelleğe alınmış), boş ve sıfır sayfa listelerine atanan belleğin toplamına eşittir.
78	Memory/Pages/sec	Sabit sayfa hatalarını çözmek için sayfaların diskten okunma veya diske yazılma hızı. Bellek/Sayfa Girişi/sn ve Bellek/Sayfa Çıkışı/sn'nin toplamıdır. Sayfa sayısı olarak sayılır, böylece dönüştürme olmadan Bellek/Sayfa Hatası/sn gibi diğer sayfa sayımlarıyla karşılaştırılabilir. Dosya sistemi önbelleğindeki (genellikle uygulamalar tarafından talep edilen) önbelleğe alınmamış eşlenmiş bellek dosyalarındaki hataları gidermek için alınan sayfaları içerir.
79	Memory/Cache Bytes	Sistem dosya önbelleğinin fiziksel bellekte o anda yerleşik ve etkin olan kısmının bayt cinsinden boyutu. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
80	Memory/Pool Nonpaged Bytes	Disk belleği olmayan havuzun bayt cinsinden boyutu; sistem sanal belleğinin, diske yazılamayan ancak ayrıldıkları sürece fiziksel bellekte kalması gereken nesneler için kullanılan alanı. Bellek/Havuzda Diske Alınmayan Bayt, İşlem/Havuzda Diske Alınmayan Bayttan farklı şekilde hesaplanır. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
81	Memory/Page Faults/sec	Saniye başına hatalı sayfaların ortalama sayısı. Her hatalı işlemde yalnızca bir sayfa hatalı olduğundan, saniyede hatalı sayfa sayısıyla ölçülür, dolayısıyla bu aynı zamanda sayfa hatalı işlem sayısına da eşittir. Bu sayaç hem donanım hatalarını (disk erişimi gerektirenler) hem de yazılım hatalarını (hatalı sayfanın fiziksel belleğin başka bir yerinde bulunduğu durum) içerir.
82	Memory/Transition Faults/sec	Sayfayı paylaşan başka bir işlem tarafından kullanılan, değiştirilmiş sayfa listesinde veya bekleme listesinde bulunan veya sayfa hatası sırasında diske yazılan sayfaların kurtarılmasıyla sayfa hatalarının çözülme hızı. Geçiş hataları, hata sayısıyla sayılır; Her işlemde yalnızca bir sayfa hatalı olduğundan, hatalı sayfa sayısına da eşittir.
83	Memory/System Cache Resident Bytes	Sistem dosya önbelleğinin fiziksel bellekte o anda yerleşik ve etkin olan kısmının bayt cinsinden boyutu. Sistem Önbelleği Yerleşik Baytları ve Bellek/Önbellek Baytları sayaçları eşdeğerdir. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
84	Memory/Long-Term Average Standby Cache Lifetime (s)	Bekleme önbelleğindeki verilerin uzun bir aralıktaki ortalama ömrü ölçülür.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

85	Memory/Standby Cache Reserve Bytes	Yedek yedek önbellek sayfa listelerine atanan bayt cinsinden fiziksel bellek miktarı. Bir sürece tahsis edilmek veya sistem kullanımı için hemen kullanılabilir.
86	Memory/Page Writes/sec	Fiziksel bellekte yer açmak için sayfaların diske yazılma hızı. Bu sayaç, her işlemde yazılan sayfa sayısına bakılmaksızın yazma işlemlerini gösterir. Bu sayaç, son iki örnekte gözlemlenen değerler arasındaki farkı, örnek aralığının süresine bölünerek görüntüler.
87	Memory/System Code Total Bytes	Şu anda sistem sanal adres alanına eşlenen sayfalanabilir işletim sistemi kodunun bayt cinsinden boyutu. Bu değer, Ntoskrnl.exe, Hal.dll, önyükleme sürücüleri ve Ntldr/osloader tarafından yüklenen dosya sistemlerindeki baytların toplanmasıyla hesaplanır. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
88	Memory/Standby Cache Core Bytes	Çekirdek yedek önbellek sayfa listelerine atanan bayt cinsinden fiziksel bellek miktarı. Bu bellek, işlemler, sistem ve sistem önbelleği tarafından aktif olarak kullanılmayan önbelleğe alınmış verileri ve kodları içerir.
89	Memory/System Driver Resident Bytes	Aygıt sürücüleri tarafından kullanılan sayfalanabilir fiziksel belleğin bayt cinsinden boyutu. Sürücülerin çalışma kümesidir (fiziksel hafıza alanı). Bu değer, diske yazılan sürücü belleğini de içeren bellek/Sistem Sürücüsü Toplam Baytının bir bileşenidir.
90	Memory/Standby Cache Normal Priority Bytes	Normal öncelikli bekleme önbellek sayfa listelerine atanan bayt cinsinden fiziksel bellek miktarı. Bu bellek, işlemler, sistem ve sistem önbelleği tarafından aktif olarak kullanılmayan önbelleğe alınmış verileri ve kodları içerir. Bir sürece tahsis edilmek veya sistem kullanımı için hemen kullanılabilir.
91	Memory/Pool Paged Allocs	Disk belleği havuzunda yer ayırmak için yapılan çağrıların sayısı. Her çağrıda tahsis edilen alan miktarına bakılmaksızın, alan tahsis edilecek çağrı sayısıyla ölçülür. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
92	Memory/Pool Nonpaged Allocs	Disk belleği olmayan havuzda yer ayırmak için yapılan çağrıların sayısı. Her çağrıda tahsis edilen alan miktarına bakılmaksızın, alan tahsis edilecek çağrı sayısıyla ölçülür. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
93	Memory/pct_ Committed Bytes In Use	Bellek/Kaydedilen Baytların Bellek/Kaydet Sınırına oranı. Bu sayaç yalnızca geçerli yüzde değerini görüntüler; bu bir ortalama değil.
94	Memory/Free System Page Table Entries	Şu anda sistem tarafından kullanılmayan sayfa tablosu girişlerinin sayısı. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
95	Memory/Available MBytes	Bir işleme veya sistem kullanımına tahsis edilmeye hemen hazır olan, Megabayt cinsinden fiziksel bellek miktarı. Bekleme (önbelleğe alınmış), boş ve sıfır sayfa listelerine atanan belleğin toplamına eşittir.
96	Memory/Modified Page List Bytes	Değiştirilen sayfa listesine atanan bayt cinsinden fiziksel bellek miktarı. Bu bellek, işlemler, sistem ve sistem önbelleği tarafından aktif olarak kullanılmayan önbelleğe alınmış verileri ve kodları içerir.
97	Memory/Cache Faults/sec	Dosya sistemi önbelleğinde aranan bir sayfa bulunamadığında ve belleğin başka bir yerinden (yazılım hatası) veya diskten (donanım hatası) alınması gerektiğinde hataların oluşma hızı. Bu sayaç, her işlemde hatalı sayfa sayısına bakılmaksızın hata sayısını gösterir.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

98	Memory/Committed Bytes	Bayt cinsinden taahhüt edilen sanal bellek miktarı. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
99	Memory/System Driver Total Bytes	Aygıt sürücüleri tarafından kullanılmakta olan sayfalanabilir sanal belleğin bayt cinsinden boyutu. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
100	Memory/Pages Input/sec	Sabit sayfa hatalarını çözmek için sayfaların diskten okunma hızı. Sabit sayfa hataları, bir işlemin sanal bellekteki, çalışma kümesinde veya fiziksel belleğin başka bir yerinde olmayan ve diskten alınması gereken bir sayfaya başvurması durumunda ortaya çıkar.
101	Memory/Pool Paged Resident Bytes	Disk belleği havuzunun şu anda fiziksel bellekte yerleşik ve etkin olan kısmının bayt cinsinden boyutu. Disk belleği havuzu, kullanılmadıklarında diske yazılabilen nesneler için kullanılan sistem sanal belleğinin bir alanıdır. Bu sayaç yalnızca son gözlemlenen değeri görüntüler; bu bir ortalama değil.
102	Memory/Write Copies/sec	Sayfa hatalarının, fiziksel belleğin başka bir yerinden sayfanın kopyalanmasıyla karşılanan yazma girişimlerinden kaynaklanma oranı. Bu sayaç, her işlemde kopyalanan sayfa sayısına bakılmaksızın kopya sayısını gösterir.
103	LogicalDisk(_Total)/Avg. Disk Bytes/Write	Yazma işlemleri sırasında diske aktarılan ortalama bayt sayısı.
104	LogicalDisk(_Total)/pct_ Idle Time	Örnek aralık sırasında diskin boşta kaldığı sürenin yüzdesi.
105	LogicalDisk(_Total)/Disk Reads/sec	Diskteki okuma işlemlerinin hızı.
106	LogicalDisk(_Total)/pct_ Free Space	Seçilen mantıksal disk sürücüsündeki boş olan toplam kullanılabilir alanın yüzdesi.
107	LogicalDisk(_Total)/Disk Read Bytes/sec	Okuma işlemleri sırasında baytların diskten aktarılma hızı.
108	LogicalDisk(_Total)/Avg. Disk sec/Read	Diskten veri okumanın saniye cinsinden ortalama süresi.
109	LogicalDisk(_Total)/Disk Writes/sec	Diskteki yazma işlemlerinin hızı.
110	LogicalDisk(_Total)/Current Disk Queue Length	Performans verilerinin toplandığı sırada diskte bekleyen isteklerin sayısı. Ayrıca toplama sırasında hizmette olan talepleri de içerir. Bu anlık bir anlık görüntüdür, zaman aralığı boyunca bir ortalama değildir. İyi performans için bu farkın ortalama ikiden az olması gerekir.
111	LogicalDisk(_Total)/Split IO/Sec	Diske giden G/Ç'lerin birden fazla G/Ç'ye bölünme hızı. Bölünmüş bir G/Ç, tek bir G/Ç'ye sığmayacak kadar büyük boyuttaki verilerin talep edilmesinden veya diskin parçalanmış olmasından kaynaklanabilir.
112	LogicalDisk(_Total)/Free Megabytes	Disk sürücüsünde megabayt cinsinden ayrılmamış alan. Bir megabayt 1.048.576 bayta eşittir.
113	LogicalDisk(_Total)/Avg. Disk sec/Write	Verilerin diske yazılmasının saniye cinsinden ortalama süresi.
114	LogicalDisk(_Total)/Disk Bytes/sec	Hız baytları, yazma veya okuma işlemleri sırasında diske veya diskten aktarılır.
115	LogicalDisk(_Total)/Avg. Disk Read Queue Length	Örnek aralık sırasında seçilen disk için sıraya alınan okuma isteklerinin ortalama sayısı.
116	LogicalDisk(_Total)/pct_ Disk Time	Seçilen disk sürücüsünün okuma veya yazma isteklerine hizmet vermekle meşgul olduğu geçen sürenin yüzdesi.

Çizelge 4.3. (devam) TON_IOT verisetideki özelliklerin tanımları

117	LogicalDisk(_Total)/Avg. Disk Bytes/Read	Okuma işlemleri sırasında diskten aktarılan ortalama bayt sayısı.
118	LogicalDisk(_Total)/Avg. Disk Write Queue Length	Örnek aralık sırasında seçilen disk için sıraya alınan yazma isteklerinin ortalama sayısı.
119	LogicalDisk(_Total)/Avg. Disk Queue Length	Örnek aralık sırasında seçilen disk için kuyruğa alınan hem okuma hem de yazma isteklerinin ortalama sayısı.
120	LogicalDisk(_Total)/pct_ Disk Read Time	Seçilen disk sürücüsünün okuma isteklerine hizmet vermekle meşgul olduğu geçen sürenin yüzdesi.
121	LogicalDisk(_Total)/Disk Write Bytes/sec	Yazma işlemleri sırasında baytların diske aktarılma hızı.
122	LogicalDisk(_Total)/Disk Transfers/sec	Diskteki okuma ve yazma işlemlerinin hızı.
123	LogicalDisk(_Total)/Avg. Disk Bytes/Transfer	Yazma veya okuma işlemleri sırasında diske veya diskten aktarılan ortalama bayt sayısı.
124	LogicalDisk(_Total)/pct_ Disk Write Time	Seçilen disk sürücüsünün yazma isteklerine hizmet vermekle meşgul olduğu geçen sürenin yüzdesi.
125	LogicalDisk(_Total)/Avg. Disk sec/Transfer	Ortalama disk aktarımının saniye cinsinden süresi.
126	label	Normal ve saldırı kayıtlarını etiketleridir. Burada 0 normali, 1 ise saldırıları belirtir.
127	type	DoS, DDoS ve Backdoor gibi saldırı kategorilerini ve normal kayıtların etiketidir.

Çizelge 4.4. TON_IOT- Processed Windows10 verisetindeki atakların miktarı

Etiket	Miktar
Normal	24871
Backdoor	-
DDoS	4608
Ransomware	-
Injection	612
XSS	1268
Password	3628
Scanning	447
DoS	525
MITM	15
Total	35974

5. KULLANILAN YAPAY ZEKÂ ALGORİTMALARI

Makine öğrenme ve derin öğrenme algoritmalarında temel mantık, veri setinden cıkarılan özellikleri eğitim sırasında öğrenerek model oluşturulmasıdır. Daha sonra oluşturulan bu model bilinmeyen veriler için tahminleme yapması sağlanır. Yapay zekâ tabanlı anamoli tespit sistemleri bu yapıyı kullanarak geliştirilen modelin ağdaki normal olmayan durumları tespit etmesi sağlanır. Bu bölümde çalışmada kullanılan makine öğrenme ve derin öğrenme algoritmalarını açıklanmaktadır.

5.1. Convolutional Neural Network (CNN)

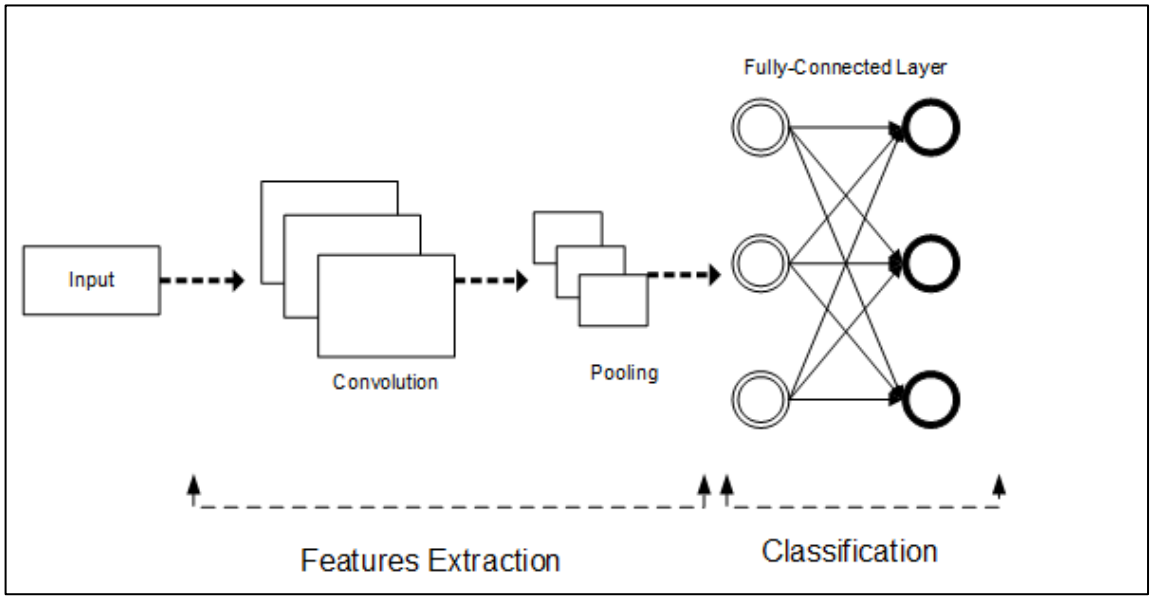
Convolutional Neural Network (CNN) algortiması, yapay sinir ağını temel alan derin öğrenme algoritmalarından biridir. Convolution, pooling, flatten ve fully connected katmanlarından oluşmaktadır. Şekil 5.1'de CNN algoritmasının temel yapısı yer almaktadır. Convolution katmanı, CNN'nin temel taşıdır. Burada giriş değerinin özelliklerini çıkartacak filtre ile konvolüsyon işlemi yapılır. Bu katman ile özellik haritası çıkartılır. Denklem (1) ile konvolüsyon işlemi sonrası çıkış hacminin boyutu hesaplanabilir. Burada P stride değeri, Wi giriş hacminin boyutu, S kernel size değeri ve M ise zero padding miktarıdır.

$$\mathrm{Wo} = \frac{\mathrm{Wi} - \mathrm{S} + 2\mathrm{M}}{\mathrm{P}} + 1 \tag{5.1}$$

Konvolüsyon işlemi sonrasında hesaplanacak verinin parametre sayısı düşürmek ve eğitimin daha kolay olması için pooling işlemi yapılarak giriş yapılan verinin boyutu düşürülmüş olur. Pooling katmanı, seçilen boyuttaki alan içerisindeki değerlerin en büyüğü (max pooling) veya değerlerin ortalaması (average pooling) olarak iki farklı şekilde seçilebilir. Oluşturulan algoritmada birden fazla Convolution/Pooling katmanı olabilir. Bu aşama özellik çıkartma olarak bilinmektedir. Özelliği çıkartılan veri artık hesaplama için kullanılabilir duruma gelmektedir. CNN algoritması bir boyutlu, iki boyutlu veya üç boyutlu olabilir. Modelde 1D CNN kullanılmıştır. 1D, 2D ve 3D CNN'nin farkları şu şekildedir:

- 1D CNN'de filtre bir boyutta hareket etmektedir. Giriş ve çıkış datası iki boyutlu olmalıdır. Zaman serisi tipinde datalarda kullanılabilir.
- 2D CNN'de filtre iki boyutta hareket etmektedir. Giriş ve çıkış datası üç boyutlu olmalıdır. Girdi olarak resim kullanılan algoritmalarda kullanılabilir.
- 3D CNN'de filtre üç boyutta hareket etmektedir. Giriş ve çıkış datası dört boyutlu olmalıdır. Girdi olarak video kullanılan algoritmalarda kullanılabilir.

Sınıflandırma bölgesi ise flatten ve fully-connected olarak bilinen katmanlardan oluşur. Convolution aşamasından sonra gelen veri fully-connected aşamasında kullanılabilmesi için düzleştirme yapılması gerekir. Bu aşama, flatten katmanında yapılır. Fully-connected katmanı ise klasik yapay sinir ağı modelini kullanır. Burada ağırlık değerleri hesaplanarak sınıflandırma sonucu belli olur. Görüntü olmayan verilere CNN uygulanabilmesi için veri boyutlarının dönüştürülmesi gerekmektedir. Böylece CNN tek boyutlu evrişimli katmanlarla kullanılabilir [48].

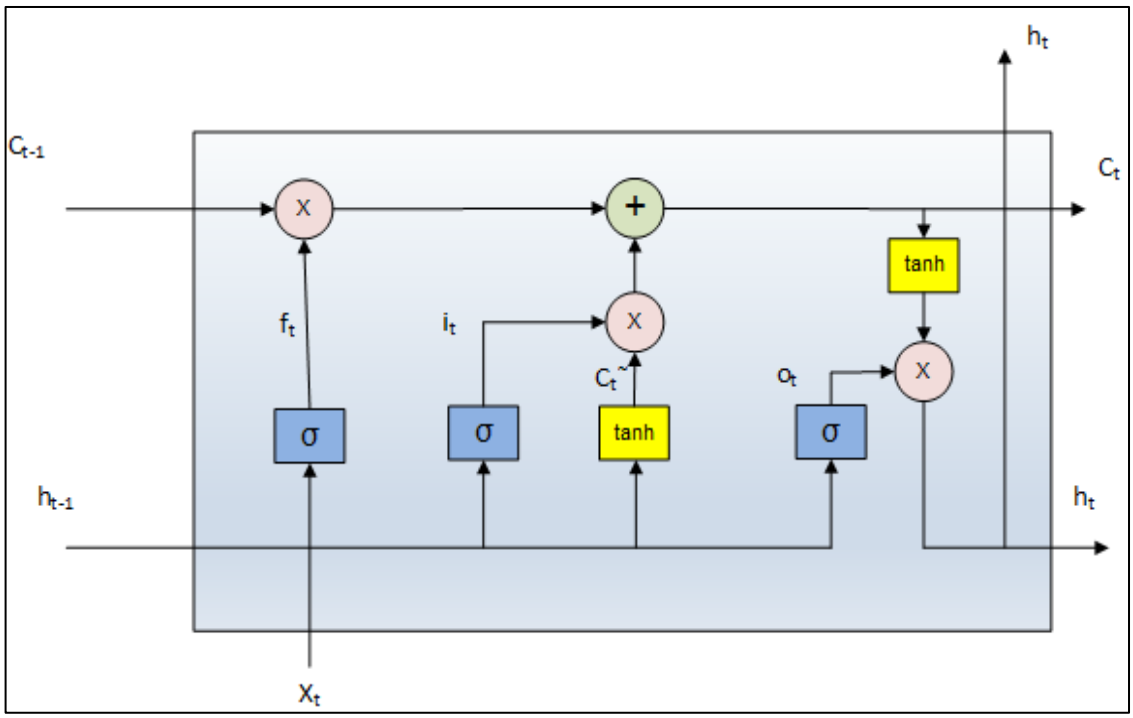


Şekil 5.1. CNN temel algoritması

5.2. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) algoritması, uzun süreli bağımlılıkları öğrenebilen ve sıralı verileri hafızasında tutabilen bir Tekrarlayan Sinir Ağı (RNN) türüdür. Hesaplama sırasında gradyan ters çevirme işlemlerinde kademeli azalmanın neden olduğu kaybolan gradyan problemini çözer. LSTM, zaman serileriyle ilgili konularda kullanılması uygun bir algoritmadır [49]. Bu özellikleri sayesinde dil işleme, video işleme ve konuşma tanıma gibi

algoritmalarda kullanılabilmektedir. LSTM algoritması hücre adı verilen bellek bloklarından oluşur ve bunlar algoritmanın ana bileşenleridir. Şekil 5.2'de LSTM algoritmasının temel yapısı verilmiştir. Şekilden de görülebileceği üzere LSTM algoritması forget gate (ft), input gate (it) ve output gate (ot) olmak üzere üç bölümden oluşur. Input ve Output kapıları, t anında verilerin giriş ve çıkışını temsil eder. Forget gate ise, anlık veri girişlerinin bir önceki veri durumu ile karşılaştırma yaparak verinin unutulup unutulmayacağına karar verir [3].



Şekil 5.2. LSTM temel algoritması

Bir LSTM hücresindeki kapılar (ft, it, ot) arasındaki ilişkiyi açıklayan matematiksel denklem aşağıdaki gibidir:

$$i_t = \sigma(w_i \cdot [h_{t-1}, X_t]) \quad (5.2)$$

$$f_t = \sigma(w_f \cdot [h_{t-1}, X_t]) \quad (5.3)$$

$$o_t = \sigma(w_o \cdot [h_{t-1}, X_t]) \quad (5.4)$$

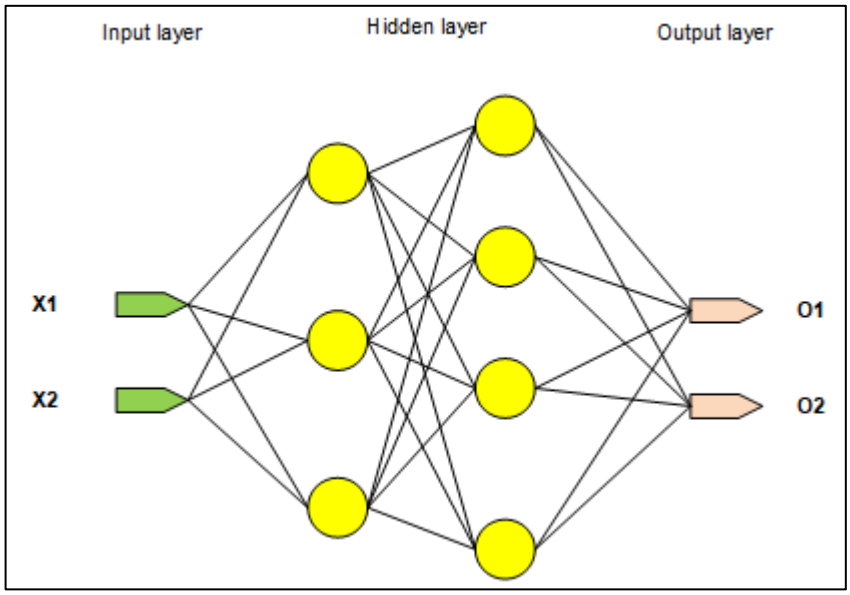
$$C_t\tilde{} = \tanh(w_t \cdot [h_{t-1}, x_t]) \quad (5.5)$$

$$C_t = f_t \times C_{t-1} + i_t \times C_t\tilde{} \quad (5.6)$$

$$h_t = o_t \times \tanh(C_t) \quad (5.7)$$

5.3. Multilayer Perceptron (MLP)

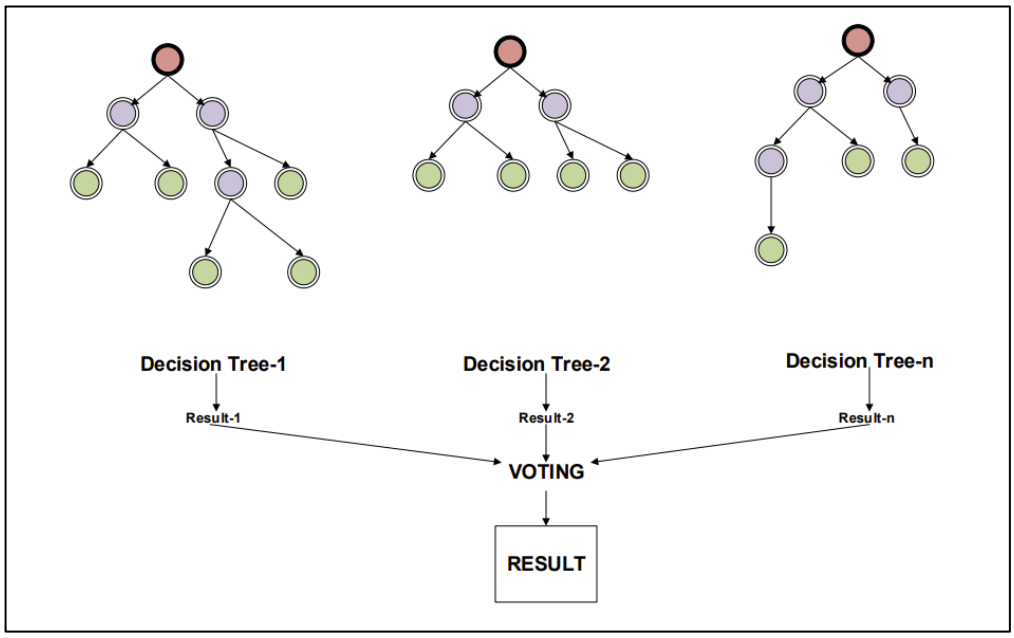
MLP algoritması, çok katmanlı yapay sinir ağlarının temelini oluşturur. MLP karmaşık problemleri çözmek için birden fazla algılayıcıdan oluşan sınıflandırma algoritması olarak kullanılabilir. Bu algoritma giriş katmanı, gizli katman ve çıkış katmanı olmak üzere en az üç katmandan oluşur (Şekil 5.3). Gizli katmanın sayısı arttırılabilir ve her katmanda birden fazla algılayıcı kullanılabilir. Girişe gelen bir veri, giriş katmanında bulunan sensörlerin ağırlık değerlerine ve aktivasyon fonksiyonuna göre değerlendirilerek bir sonraki katmana aktarılır. Her katmanın çıkışı, diğer katmanın girişi olmaktadır ve her algılayıcı diğer katmanda bulunan algılayıcıların hepsine bağlıdır. Katmanlar boyunca hesaplamalar yapılarak çıkış katmanına ulaşılır.



Şekil 5.3. MLP temel algoritması

5.4. Random Forest (RF)

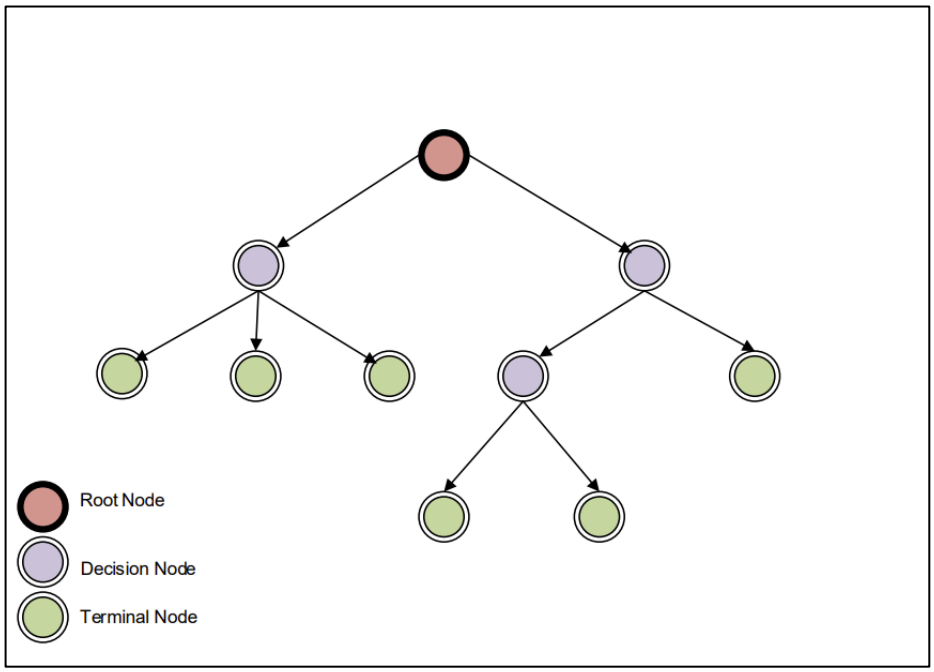
Rastgele bir orman algoritması, veri kümesinden özelliklerin rastgele çıkarılmasına dayalı olarak oluşturulmuş yüzlerce karar ağacının (Şekil 5.4) bir kombinasyonudur. Karar ağacındaki aşırı öğrenme eğilim durumunu çözmek için kullanılır [50]. Rastgele orman algoritmasında karar, her ağaçtan tahminleme alındıktan sonra bu karara göre çıkan oylamaya göre verilir. Herhangi bir ağaçtan hatalı tahmin gelse bile ortak karara bakıldığı için hatalara karşı dirençli olur. Ağaç sayısını (derinlik) artırmak doğruluğu arttırmaktır.



Şekil 5.4. Random Forest temel algoritması

5.5. Decision Tree (DT)

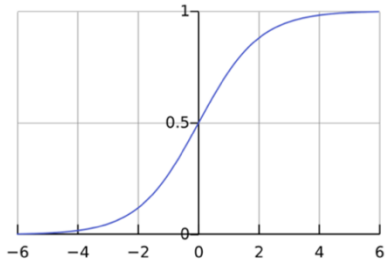
Karar ağacı algoritmasının temel yapısı Şekil 5.5'te verilmiştir. Buradaki root node'dan başlayan dallanmalar, sınıflandırmaları ve kararları temsil eden bir ağaç yapısını oluşturur. Root node, karar ağacındaki en üst düğüm olarak bilinir. Burada alınan kararlara göre bölünmeler başlar. Burada her dahili düğüm bir öznitelik üzerindeki bir testi belirtir, her dal, testin bir sonucunu temsil eder ve her yaprak (terminal) düğüm bir sınıf etiketine sahiptir [51]. Bir değerin ağaç sınıflandırması, kök düğümden başlayıp yaprakta biten bir ağaç üzerinden geçirilerek gerçekleştirilir.



Şekil 5.5. Decision Tree temel algoritması

5.6. Logistic Regression (LR)

Denetimli makine öğrenimi algoritmalarından biri olan Lojistic Regresyon, bağımlı değişkenin tahmini değerlerini olasılık olarak hesaplayarak sınıflandırma imkânı veren bir yöntemdir. Bu yöntem olasılık değerlerini hesaplamak için sigmoid fonksiyonunu (Şekil 5.6) kullanır. Bu fonksiyonun sonucu 0 ile 1 arasında olduğu için tahminleme değerleri olasılıkla eşleştirerek sınıflandırma yapabilmektedir.



Şekil 5.6. Sigmoid fonksiyonu

5.7. K-Nearest Neighbour (KNN)

KNN algoritmasının çalışma prensibi, tahmin edilecek değerin önceden sınıflandırılan verilere olan vektörel uzaklığını ölçerek çalışmasıdır. Yeni veri, vektörel ölçüme göre en yakın olan sınıfa ait olacak şekilde sınıflandırılır. KNN sınıflandırılması aşağıdaki steplere göre yapılır [2]:

Step 1: Choose the number of K neighbors.
Step 2: Calculate the distance between neighboring samples using the (8).
Distance = |a1 − b1|+|a2 − b2|+...+|an − bn| (5.8)
Step 3: Select the K nearest neighbors based on the distance estimated in Step 2.
Step 4: Summarize the total number of data points in each class of K neighbors.
Step 5: Allocate the new data point to a class based on the maximum number of neighbors.

5.8. Naive Bayes (NB)

Naive Bayes makine öğrenme algoritması, Bayes olasılık teoreme dayanır. Bayes teoremi, problemdeki bir değişkenin varlığının başka bir değişkenin varlığı üzerinde hiçbir etkisinin olmadığını ve her değişkenin bağımsız, sonuca eşit oranda katkı sağladığını söyler. Bayes teoreminin matematiksel ifadesi şöyle ifade edilir (9);

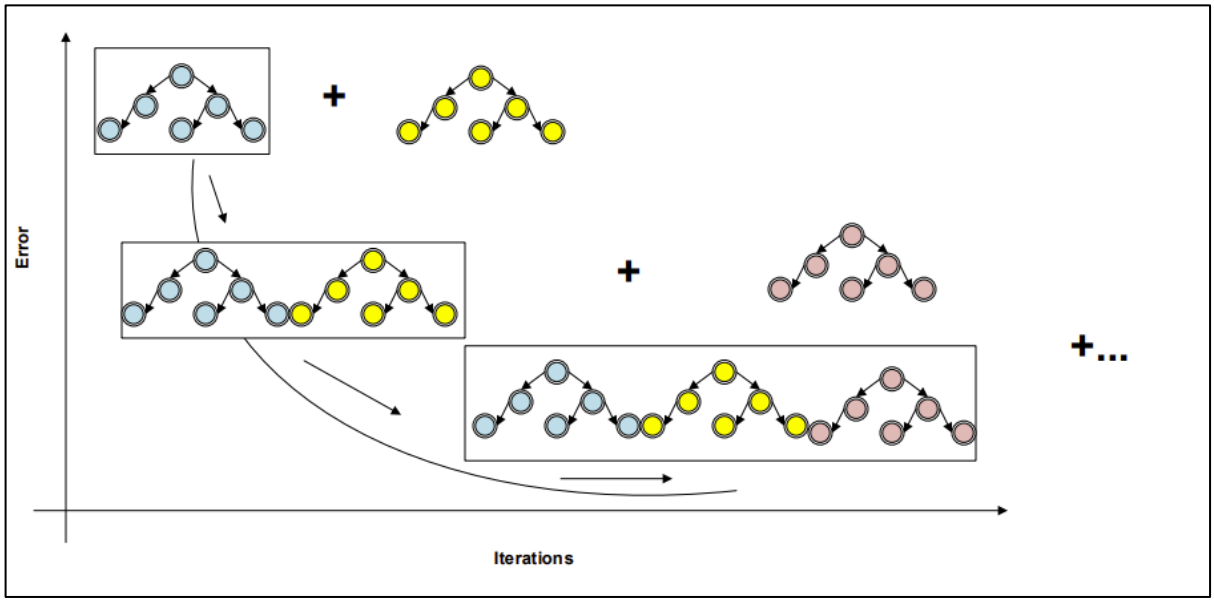
P(A|B) = P(B|A) P(A) / P(B) (5.9)

- P (A\B) = B olayı gerçekleştiğinde A olayının olma olasılığını ifade eder.
- P (B\A) = A olayı gerçekleştiğinde B olayının olma olasılığını ifade eder.
- P(A) = A olayının olma olasılığını ifade eder,
- P(B) = B olayının gerçekleşme olasılığını ifade eder.

5.9. Gradient Boost (GB)

GB, sınıflandırma ve regresyon yapabilmek için topluluk yöntemini kullanan bir makine öğrenmesi algoritmasıdır [52]. Sıralı bir şekilde çalışır. Bunun anlamı ise ilk olarak initial leaf oluşturarak tahminleme yapılır. Daha sonra ise ilk yaprağın tahmin hataları göz önünde bulundurularak yeni ağaçlar oluşturularak olası tahminleme hataları gradyan

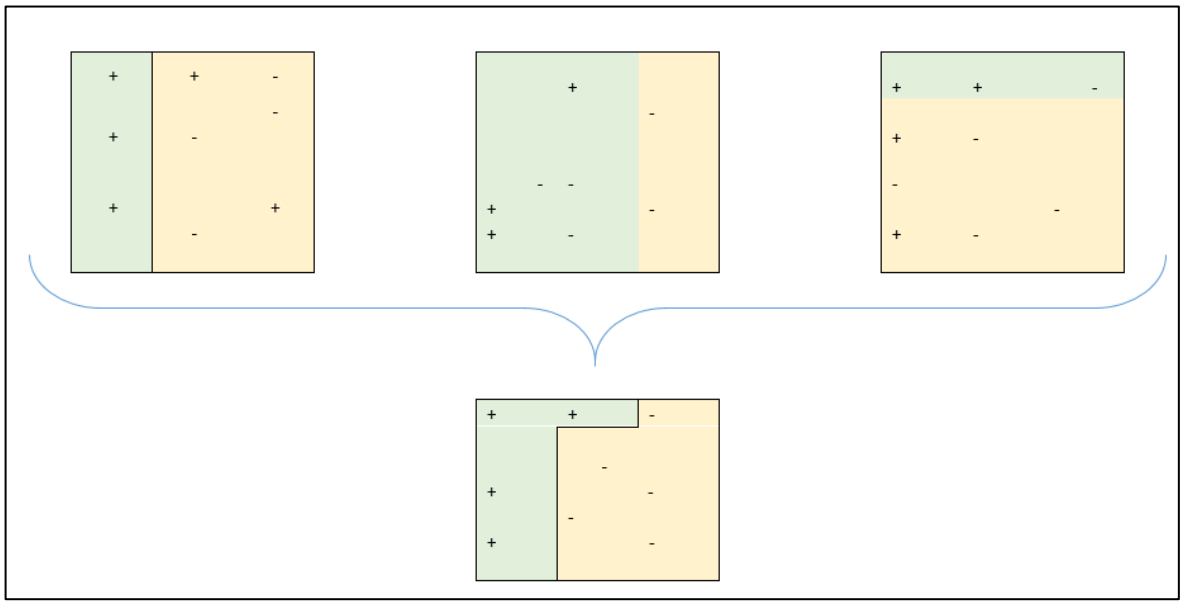
azalma yöntemiyle en aza indirilmeye çalışılır. Sıralı oluşturma ise bir önceki ağaçtan daha iyi bir kayıp fonksiyonu elde edilmeyinceye kadar devam eder. Şekil 5.7'de bu algoritmanın çalışma mantığı çizilmiştir.



Şekil 5.7. GB temel algoritması

5.10. Ada Boost (Ada)

Adaptive Boosting (Ada) algoritması, topluluk yöntemini kullanan bir başka makine öğrenimi algoritmasıdır. AdaBoost'un en yaygın kullandığı tahmin edici, tek seviyeli karar ağaçlarıdır ve bunların derinliği ise birdir. Bu algoritmanın başlangış noktası, oluşturulan modelde tüm veri noktalarına eşit ağırlıklar vermektir. Daha sonra yanlış sınıflandırılan noktaların ağırlıklarını değiştirerek çalışır. Kayıp fonksiyonu en düşük hale getirilinceye kadar iterasyon devam eder. Şekil 5.8'de bu algoritmanın çalışma prensibi gösterilmiştir.



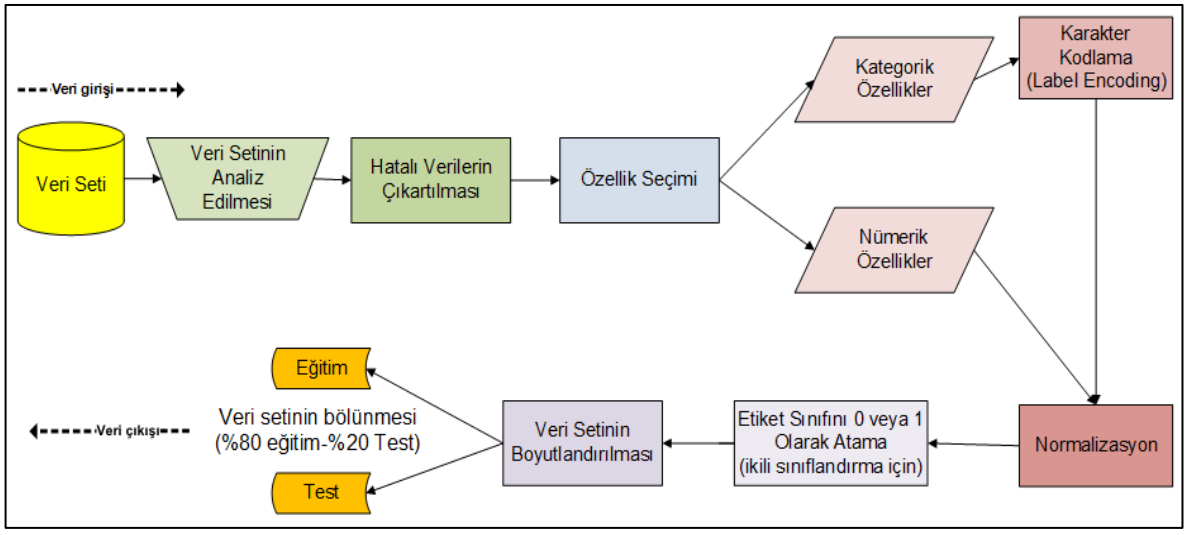
Şekil 5.8. ADA temel algoritması

6. GELİŞTİRİLEN YÖNTEM

Bu bölümde öncelikle çalışmada kullanılan CICIoT2023 ve TON_IOT verisetleri hakkında bilgi verilmiştir. Sonrasında veri setinin ön işleme adımları anlatılmıştır. Daha sonra ise çalışmada kullanılan çeşitli makine öğrenmesi ve derin öğrenme algoritmalarının tanımlamaları yapılmıştır.

6.1. Ön İşleme Adımları

Veri setlerinin ön işlemden geçirilmeden derin öğrenme algoritmalarında veri olarak kullanılması uygun değildir. Gerçek ortamlardan toplanan verilerin içinde birçok hata vardır, düzensizdirler ve bunların elemine edilmesi gerekmektedir. Örneğin, veri setinin içerisinde string değer varsa ve bu değerler sayısallaştırmadan derin öğrenme eğitimlerinde kullanılamazlar. Ön işleme yapılarak algoritmaya daha düzgün veriler sağlanarak, modelin verimliliğinin artması da hedeflenir. Şekil 6.1'de veri işleme aşamalarının akış diyagramı verilmiştir.



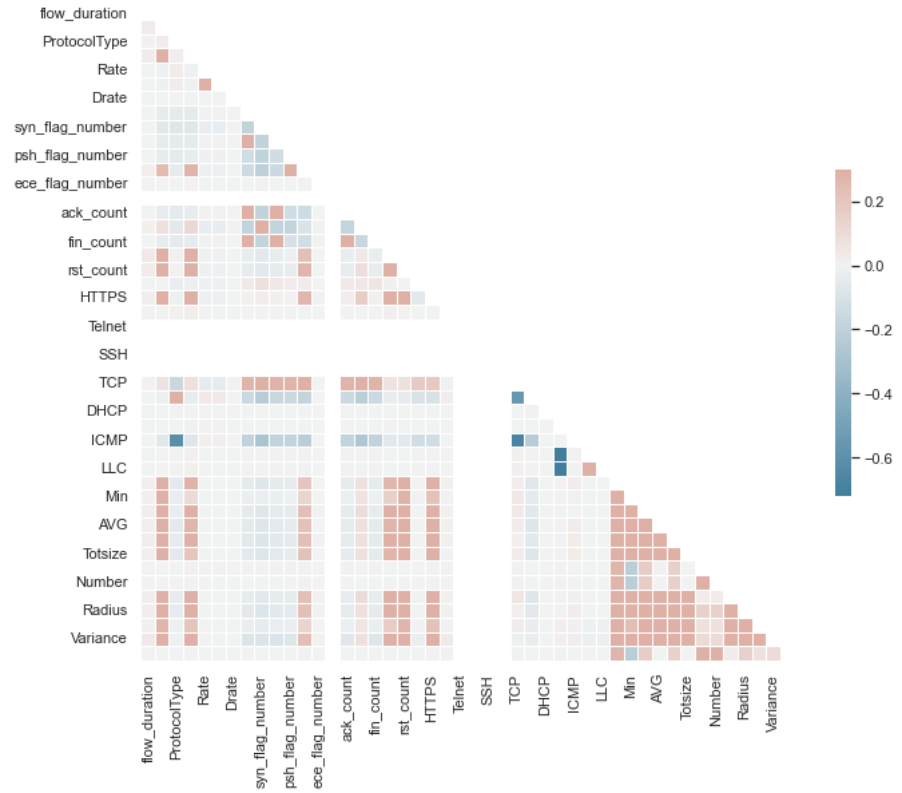
Şekil 6.1. Önişleme akış diyagramı

Veri setine yapılan ilk işlem, içinde veri bulunmayan hücrelerin silinmesi ve boşluk değerlerin kaldırılmasıdır. Boş veri içeren satırlar kaldırılarak modelin olumsuz etkilenmesinin önüne geçilmiştir.

Büyük verisetlerinde bulunan tüm özelliklerin eğitime dâhil edilmesi her zaman fayda sağlamaz. Veri setindeki özellikler birbirleriyle korele olup sonuca fayda sağlamayabilir. Ayrıca cok fazla değerin bulunması eğitimin maaliyetini de arttırmaktadır. Bunu görebilmek için veri setindeki özelliklerin korelasyon matrisi cıkarılarak, korelasyon değeri yüksek özelliklerin veri setinden cıkarılması sağlanır. Bu calısmada da Pearson Korelasyon Katsayısı (PCC) yöntemi kullanılarak özellik seçimi yapılmıstır. PCC formülü (10) aşağıda verilmiştir.

$$\mathrm{PCC}(X, Y) = \frac{\Sigma[(\mathrm{Xi} - \mu_x)(\mathrm{Yi} - \mu_y)]}{\sigma x.\, \sigma y} \tag{6.10}$$

Burada, μ değişkenin ortalaması ve σ ise standart sapma değeridir. Korelasyon değeri -1 ile 1 arasında değer alır. PCC'nin pozitif 1'e yaklaşması, iki değişken arasında pozitif bir korelasyona işaret ettiği anlamına gelir. Bu, bir değişken azaldığında veya arttığında pozitif ilişkili diğer değişkenin de aynı yönde hareket ettiği anlamına gelir. Benzer şekilde negatif korelasyona sahip iki değişken ters yönde davranır. Korelasyon değeri belirli bir eşiğin üzerinde olan özellikler kaldırılabilir. CICIoT2023 veri setine ait korrelasyon matrisi Şekil 6.2'de verilmiştir. Korelasyon matrisi verilen veri seti için bu değer 0,99 olarak belirlenmiştir. Bunun sonucunda CICIoT2023 veri setinde 40 özellik, Processed Windows 10 of TON_IOT veri setinde ise 85 özelliğin seçilmesi sağlanmıştır.



Şekil 6.2. CICIoT2023 veriseti özelliklerinin korelasyon matrisi

Veri seti üzerinde yapılan diğer işlem ise Label Encoding işlemi sayısal olmayan özelliklerin sayısal değerlere dönüştürülmesidir. Veri seti içerisindeki categorical tipindeki özelliklere label encoding uygulanmıştır. Veri seti içerisindeki numeric özelliklere bu işlemin uygulanmasına gerek yoktur. String değerlerin sayılaştırılmasının ardından normalizasyon işlemi (11) yapılmıştır.

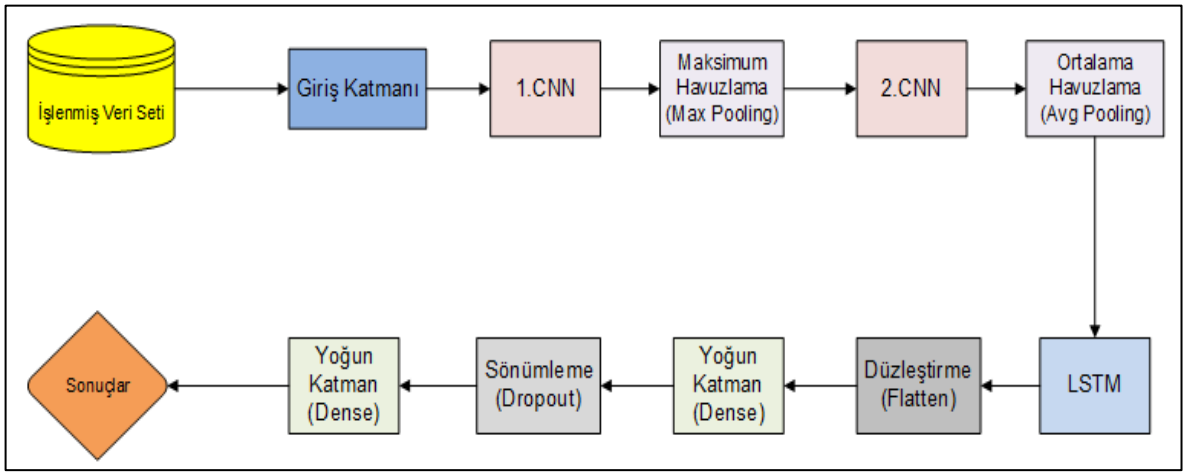
$$x' = x - \mu\ (1)\ \sigma \tag{6.11}$$

x orijinal değerdir, x' normalleştirilmiş değer μ ve σ sırasıyla ortalama ve standart sapma değerleridir. Normalizasyon işlemi sayesinde sayısal olarak büyük olan özelliklerin derin öğrenme modelinin sonucunu performansını olumsuz etkilemesi engellenmiştir [3]. Veri setinin label diye adlandırılan, trafiğin normal mi yoksa herhangi bir saldırı türüne mi ait olduğu bilgisi ise binary değerlendirme yapılacaksa normal değer 0, saldırı değerleri ise 1 olarak etiketlendirilir. Multi sınıflandırma yapılacaksa bu işlem yapılmaz. Sonraki işlem ise eldeki veri şekline yeni boyut eklenerek, CNN katmanı için uyumlu hale getirilmesi sağlanır. Veri işlemenin son aşaması ise, eğitime hazır hale gelen veri setinin, eğitim ve test olarak ikiye ayrılmasıdır. Bu oran 0,8 olarak seçilmiştir.

Şekil-11'de gösterilen adımlar sırasıyla tamamlandıktan sonra veri setinde bulunan atak sınıflarının seçimi yapılmıştır. CICIoT2023 veri setinde DDoS ana başlığı altında 12 farklı atak türü bulunmaktadır. Bunlardan en az sayıda bulunan UDP Fragmentation, ACK Fragmentation, HTTP_Flood ve SlowLoris DDoS atakları cıkartılarak 8 tane DDoS atağının seçilmesi sağlanmıştır. TON_IoT veri setinde bulunan DDoS saldırısının alt türü bulunmadığı için burada sadece DDoS-Binary sınıfları alınarak sadece binary değerlendirme yapılmıstır.

6.2. Modelin Tanımlanması

DDoS ataklarının tespit edilmesinde, bir boyutlu (1D) CNN ile LSTM algoritmaları kullanılarak hibrit bir derin öğrenme modeli geliştirilmiştir. Model, katmanlardan oluşan sıralı bir algoritmaya sahiptir. Şekil 6.3'te modelin akışı gösterilmiştir.



Şekil 6.3. Önerilen modelin akış diyagramı

Geliştirilen sıralı model, giriş katmanıyla başlamaktadır. Bu katman, kullanılan veri seti boyutunu deklare etmektedir. Şekil 6.3'te görüldüğü gibi 2 tane CNN algoritması uygulanmıştır. Kullanılan CNN'lerin parametre ayarı en iyi sonucu verecek şekilde ayarlanmıştır. Çizelge 6.1'de bu parametreler tablo halinde sunulmuştur. Konvolüsyon işlemi sonrasında Pooling katmanı uygulanmıştır. Pooling katmanının ana görevi, çıkarılan özellik matrisinin boyut indirgemesi yapmaktır. Hesaplama yükü havuzlama katmanında düşürülürken önemli bilgiler korunmaktadır [53]. 1. ve 2. CNN arasında max pooling olarak uygulanmış, average pooling ise 2.CNN'den sonra uygulanmıştır. CNN sonrasında

ardışık olacak şekilde LSTM algoritması kullnılmıştır. Kullanılan LSTM algoritmasının parametleri units=140, dropout=0.2 ve recurrent_dropout=0.4 olacak şekilde ayarlanmıştır. LSTM algoritmasından sonra ise Flatten katmanında boyutlar dense katmanı için uygun hale getirilmiştir. Flatten katmanını ise fully-connected olarak bilinen Dense katmanları takip etmektedir. %30 oranında Dropout fonksiyonu kullanılmıştır. Dropout katmanının kullanılmasının sebebi ise algoritmanın aşırı öğrenmesini önlemek amacıyladır. Bu katmanın işlevi, rastgele bir şekilde bazı düğümleri yok saymaktır. Bu kısmen, nöronların diğer nöronların hatalarını düzeltme şeklini değiştirebilecekleri bir duruma işaret eder [54]. İki CNN algoritmasında da aktivasyon fonksiyonu olarak Rectified Linear Unit (ReLu) fonksiyonu (12) kullanılmıştır. ReLU fonksiyonu, hesaplama basitliği sağladığı ve negatif değerleri ortadan kaldırdığından dolayı kullanılmıştır.

$$f(x) = \max(0,x) \tag{5.12}$$

Modelin çıktı katmanı, bir Softmax aktivasyon fonksiyonu ile sonlanır. Bu fonksiyon, çok sınıflı bir olasılık probleminde değerlendirilen örneğin olasılık açısından en doğru etikete sahip olması için maksimum değerin olasılığını döndürür [26]. Softmax fonksiyonu ile sonuç, bir olasılık dağılımı olarak alınır.

Çizelge 6.1. Kullanılan CNN parametreleri

	Filtre Sayısı	Kernel Büyüklüğü	Aktivasyon fonksiyonu
Conv1D_1	128	4	Relu
Conv1D_2	128	2	Relu

Algoritma da LSTM kullanılmasının nedeni, derin öğrenme algoritması olan LSTM'lerin döngü boyunca akış dinamiklerini yakalamada ve bilgiyi sürdürmede iyi olmasıdır. LSTM algoritması uzun süreli bağımlılıkları öğrenebilir ve sıralı verileri hafızasında tutabilir. LSTM algoritmasında bulunan forget gate, bir önceki verinin unutulup unutulmayacağına karar vermektedir. DDoS gibi yoğun olarak yaşanan saldırılarda ardışık verileri de kullanarak yapılan hesaplamalar sayesinde bu yapının kullanılması uygun görülmüştür. Sınıflandırma aşamasında kullanılan bir diğer algoritma ise CNN'dir. CNN algoritması; görüntü sınıflandırması, ses sınıflandırması, video sınıflandırmalarında başarılı verici sonuçlar vermektedir, ağın derinliğini ve genişliğini değiştirerek sınıflandırma problemler

ile başa çıkmak için güçlü yeteneklere sahiptir. CNN, konvolüsyon işlemi sayesinde benzer sayıda katmana sahip standart ileri beslemeli sinir ağlarına kıyasla daha az bağlantı ve parametre ile zamana duyarlı saldırı durumlarını tespit edebilir. Gelen veriden birçok özellik çıkarıldığı için DDoS gibi benzersiz özellikleri sahip saldırı türünü tespit etmede etkilidir [24].

7. DENEYSEL SONUÇLAR

Bu bölümde, geliştirilen hibrid algoritmanın eğitildiği ve test edildiği ortam, değerlendirme parametreleri ve test sonuçları verilmiştir. Yapılan çalışmada Apache Spark üzerinde python programlama dilinde yazma imkânı sağlayan PySpark, Google Colab platformu aracılığıyla kullanılmıştır. Derin öğrenme algoritmalarının oluşturulmasında Scikit-learn ve Keras kütüphaneleri kullanılmıştır.

Modelin eğitim ve testleri, aşağıdaki konfigürasyona sahip bilgisayar üzerinde yapılmıştır:

- MacOS v12.6 işletim sistemi
- M1 Apple Silicon (2020)
- 13.3” Ekran
- 8 çekirdekli CPU
- 8 çekirdekli GPU
- 8 GB RAM
- 256 GB SSD

CICIoT2023 veri seti birden fazla veri dosyasından oluştuğu için bunların birleştirilmesiyle çok büyük veri ortaya çıkmaktadır. İncelenen calısmalar da veri setinden örneklem alınarak yapılan çalışmalar mevcuttur [29] [16]. Bu durum hem eğitim maaliyetini azaltmakta hem de sonuca ciddi bir etki yapmamaktadır. Aynı zamanda tüm veri setini kullanmak, kullanılan bilgisayarın kapasitelerini doldurmakta ve işlem yapılamaz hale getirmektedir. Bu sayede kullanımı pahalı ve ulaşılması zor olan yüksek kapasiteli bilgisayar ve sunucuların kullanım gereksinimini ortadan kaldırmaktadır. Verisetlerinin tamamı yerine %20 oranına azaltılarak veri setinin alt uzay kümesi kullanılmıştır. Alt uzay kümesinin atak sınıfları oranı orjinal haliyle aynı olup, atak sayılarının direkt %20 oranında alınmasıyla oluşmuştur. Böylece eğitim ve test masrafından ve süreden tasarruf sağlanmıştır. Önerilen algoritma testlerinin yanında, 10 farklı makine öğrenmesi ve derin öğrenme algoritması ile de test edilmesi sağlanarak sonuçlar karşılaştırılmıştır.

Kullanılan yapay zekâ ve makine algoritmaların parametre değerleri aşağıdaki verilmiştir:

- Random Forest: max_depth=4, n_estimators=100
- Decision Tree: max_depth=5, random_state=0
- Gradient Boost: n_estimators=10, max_depth=3, learning_rate=0.1
- Ada Boost: n_estimators=10, learning_rate=0.1, random_state=0
- Naive Bayes: default
- Logistic Regression: default
- K-Nearest Neighbour: n_neighbors=3, leaf_size=50
- MLP: hidden_layer_sizes= (5,10,5), max_iter=5,
- CNN: Filters=64, kernel_size=2, activation='relu'
- LSTM: Units=100, dropout=0.2 and recurrent_dropout=0.2

Çalışmada elde edilen sonuçlar farklı açıdan değerlendirilmesi yapılmıştır. İlk olarak CICIoT2023 veri setinde bulunan DDoS atak sınıflarının çoklu sınıf ve binary değerlendirilmesi yapılmıştır. Geliştirilen algoritmanın, 10 farklı makine ve derin öğrenme algoritmalarıyla karşılaştırılması sağlanmıştır. Daha sonra ise kullanılan ikinci veri seti olan TON_IOT-Windows10 veri setinin binary değerlendirlendirilmesidir. Yapılan değerlendirmeler doğruluk, kesinlik, geri çağırma ve F1-Skor parametleriyle yapılmıştır. Ayrıca ROC Curves ve confusion matrix grafikleri de oluşturularak çalışmaya dâhil edilmiştir.

7.1. CICIoT2023 Veri Setini Analizleri

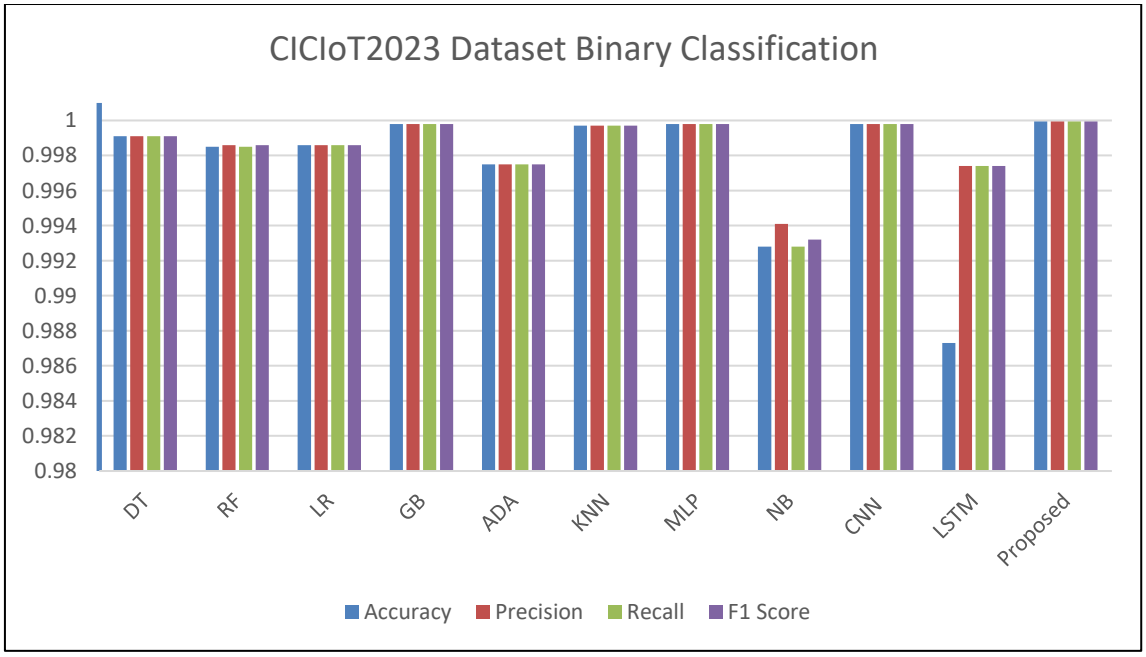
7.1.1. Binary değerlendirme

Ön işlemesi tamamlanan CICIoT2023 veri setinin, geliştirilen hibrit algoritma ile öncelikle binary olarak test edilmiştir. Makine öğrenimi ve derin öğrenme algoritmalarından Random Forest, Decision Tree, Gradient Boost, Ada Boost, Naive Bayes, Logistic Regression, K-Nearest Neighbour, CNN, MLP ve LSTM ile test edilerek geliştirilen algoritma ile karşılaştırılmıştır. Çizelge 7.1'de algoritmaların, CICIoT2023 veri setine ait

binary değerlendirme sonuçlarına yer verilmiştir. Şekil 7.1'de ise sonuçların grafiğe dökülmüş hali verilmiştir.

Çizelge 7.1. CICIoT2023 veri seti binary sınıflandırma sonuçları

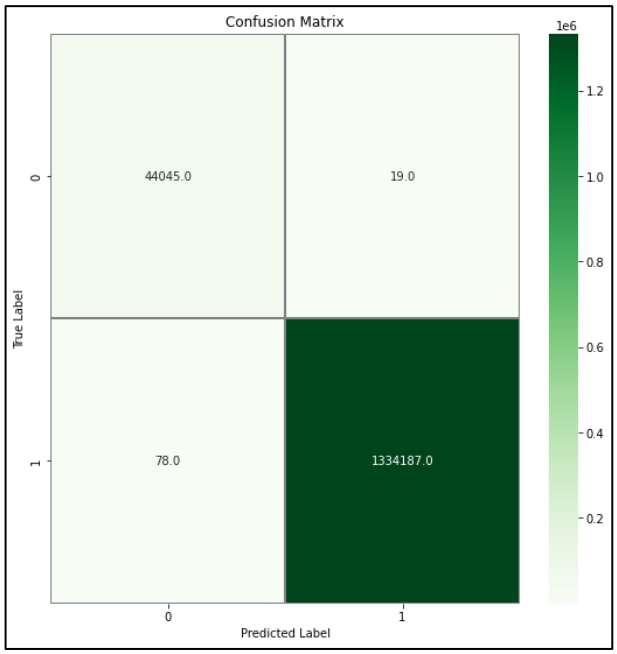
	Doğruluk	Kesinlik	Geri Çağırma	F1 Skor
DT	99,91	99,91	99,91	99,91
RF	99,85	99,86	99,85	99,86
LR	99,86	99,86	99,86	99,86
GB	99,98	99,98	99,98	99,98
ADA	99,75	99,75	99,75	99,75
KNN	99,97	99,97	99,97	99,97
MLP	99,98	99,98	99,98	99,98
NB	99,28	99,41	99,28	99,32
CNN	99,98	99,98	99,98	99,98
LSTM	98,73	99,74	99,74	99,74
Proposed	99,995	99,995	99,995	99,995



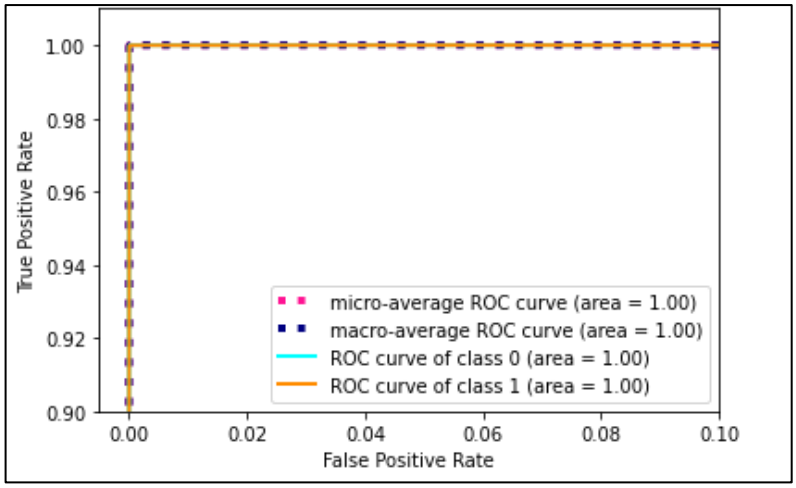
Şekil 7.1. CICIoT2023 veri seti binary sınıflandırma sonuç grafiği

Çizelge 7.1 ve Şekil 7.1'de görüleceği üzere en yüksek binary sınıflandırma doğruluk değerine önerilen algoritma ulaşmıştır. Bunu GB, MLP ve CNN algoritmaları takip

etmiştir. En düşük sonucun ise NB algoritması olduğu görülebilir. Geliştirilen hibrit algortimanın binary olarak test edilmesine ait confusion matrisi Şekil 7.2'de verilmiştir. Oluşturulan ROC Curve ait diyagram ise Şekil 7.3'te sunulmuştur.



Şekil 7.2. CICIoT2023 veriseti binary sınıflandırmaya ait confusion matrisi



Şekil 7.3. CICIoT2023 veriseti binary sınıflandırmaya ait ROC eğrisi

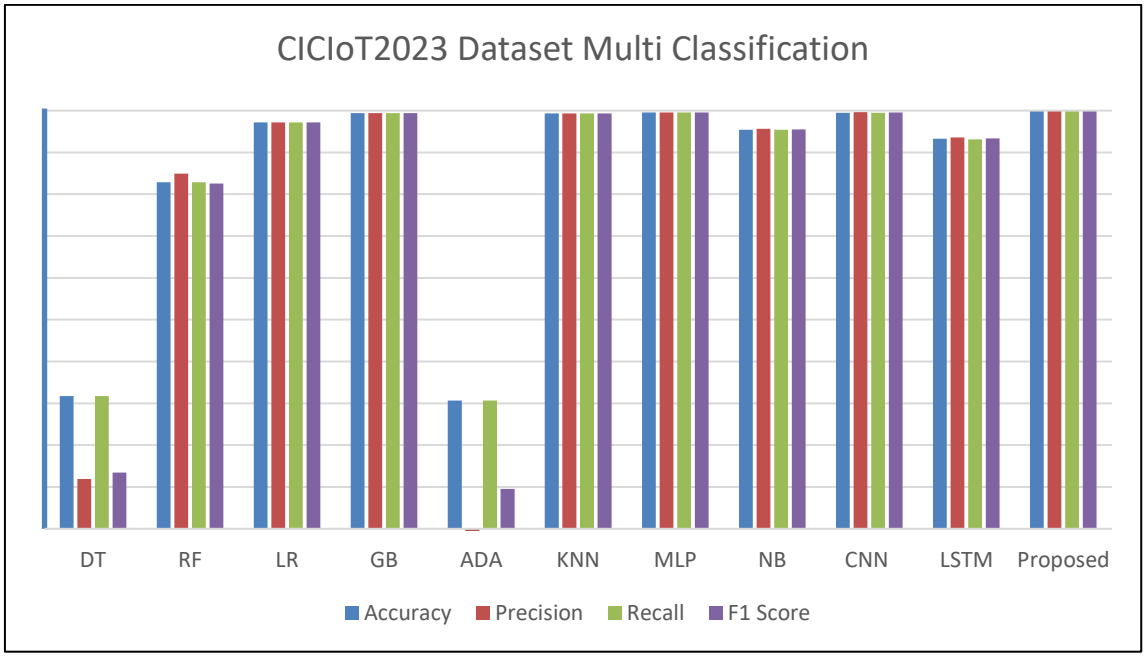
Şekil 7.2'de bulunan Confusion matrix'e göre yanlış pozitif oranının (FPR) neredeyse ihmal edilebilir seviyede, sadece yüze yakın kayıt yanlış sınıflandırılmış, olduğu görülebilir. Doğru pozitif (TPR) kayıtları ise oldukça yüksek oranda olmuştur. Şekil 7.3'te bulunan ROC grafiğine göre ise AUC-ROC değeri 0,99 üzerinde çıkmıştır.

7.1.2. Çoklu sınıf değerlendirme

Algoritmaların çoklu sınıf değerlendirilmesi de yapılmıştır. Geliştirilen algoritma ve 10 farklı makine öğrenimi ve derin öğrenme algoritmaları test edilmiştir. Çizelge 7.2'de algoritmaların, CICIoT2023 veri setine ait çoklu sınıf değerlendirme sonuçlarına yer verilmiştir. Şekil 7.4'te ise sonuçların grafiğe dökülmüş hali sunulmuştur.

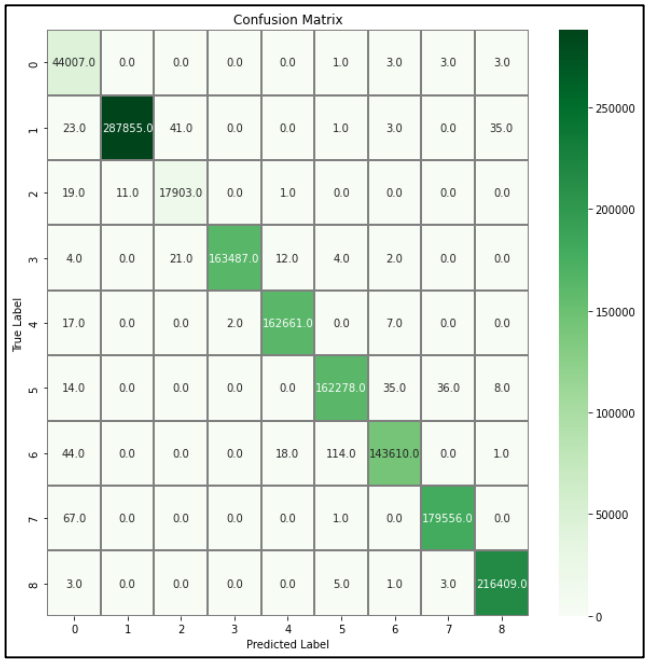
Çizelge 7.2. CICIoT2023 veri seti çoklu sınıflandırma sonuçları

	Doğruluk	Kesinlik	Geri Çağırma	F1 Skor
DT	86,34	82,39	86,34	82,69
RF	96,58	96,98	96,58	96,51
LR	99,43	99,44	99,43	99,43
GB	99,88	99,88	99,88	99,88
ADA	86,14	79,44	86,14	81,91
KNN	99,86	99,86	99,86	99,86
MLP	99,91	99,91	99,91	99,91
NB	99,09	99,13	99,09	99,10
CNN	99,90	99,93	99,90	99,91
LSTM	98,66	98,71	98,63	98,67
Proposed	99,96	99,96	99,96	99,96

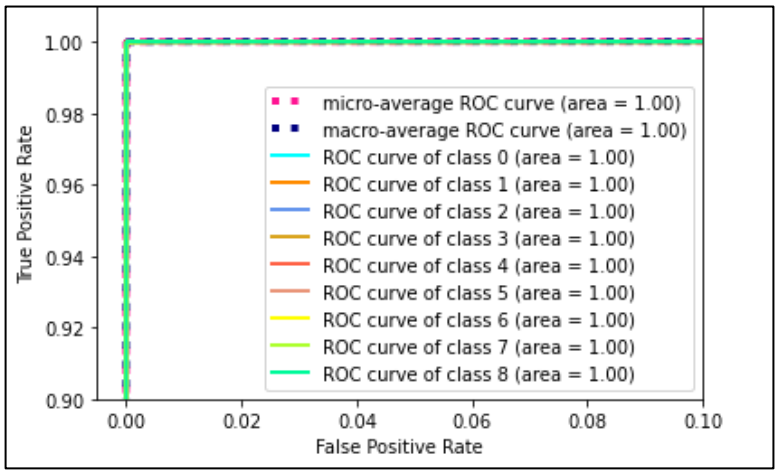


Şekil 7.4. CICIoT2023 veri seti çoklu sınıflandırma sonuç grafiği

Çizelge 7.2'de ve Şekil-7.4'te görüleceği üzere en yüksek çoklu sınıflandırma doğruluk değerine önerilen algoritma ulaşmıştır. Bunu MLP, CNN ve GB algoritmaları takip etmiştir. En düşük sonuç ile ADA ve DT algoritmalarına aittir. Geliştirilen hibrit algortimanın çoklu sınıf olarak test edilmesine ait confusion matrisi Şekil 7.5'te verilmiştir. Oluşturulan ROC Curve ait diyagram ise Şekil 7.6'da sunulmuştur.



Şekil 7.5. CICIoT2023 veriseti çoklu sınıflandırmaya ait confusion matrisi



Şekil 7.6. CICIoT2023 veriseti çoklu sınıflandırmaya ait ROC eğrisi

Şekil 7.5'teki Confusion matrix'e göre yanlış pozitif oranı (FPR) tüm kombinasyonlarda maksimum 41 kayıt değerine ulaşarak oldukça düşük seviyelerde kalmıştır. Doğru pozitif (TPR) kayıtları ise sınıfların birebir eşlemesindeki performansının yüksek değerlere ulaştığı görülmüştür. Şekil-7.6'daki çoklu sınıflandırmaya ait ROC grafiğine göre ise tüm atak sınıflandırmalarında AUC-ROC değeri 0,99 değerine yakın cıkmısıtr. Çizelge 7.3'te CICIoT2023 veri seti kullanılarak daha önce yapılan çalışmalarla, bu çalışmada sunulan modelin karşılaştırılması verilmiştir.

Çizelge 7.3. CICIoT2023 veri setinde yapılan çalışmaların karşılaştırılması

Çalışma	Model	Kullanılan Veri Seti	Doğruluk
Neto ve diğerleri (2023) [41]	RF, DNN, MLP, LR, AdaBoost	CICIoT2023	%99,68 (b), %99,43(m) (8 sınıf)
Proposed	Hybrid Deep Learning	CICIoT2023	%99,995 (b), %99,96 (m) (9 sınıf)

Yukarıdaki yapılan değerlendirme sonuçlarına göre, binary sınıflandırma ile çoklu sınıflandırma arasında ciddi bir fark oluşmaktadır. Veri seti içerisindeki atak sınıfı attıkça algoritmanın doğruluk değeri düşmektedir. CICIoT2023 veri setinde binary sınıflandırma da ise test edilen makine öğrenimi ve derin öğrenme algoritmaları yaklaşık olarak birbirine yakın sonuçlar vermiştir. Çoklu sınıflandırma da ise DT ve ADA Boost algoritmalarında ciddi bir düşüş yaşanmıştır. Geliştirilen hibrit algoritmada ise ciddi bir düşüş söz konusu

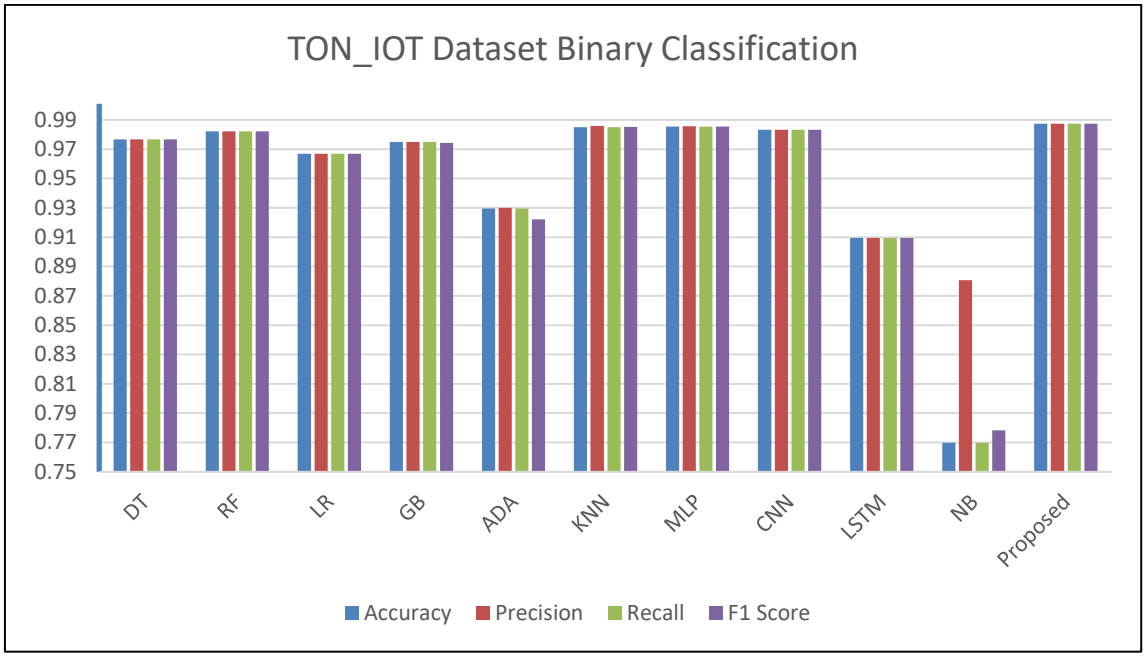
değildir. CICIoT2023 veri seti kullanılarak yapılan çalışmalarda en iyi sonuca, saldırı tespitinde %99,995 ve saldırı türü belirleme de ise %99,96 oranları ile geliştirilen hibrit algoritma ile ulaşılmıştır.

7.2. TON_IoT Veri Setini Analizleri

Sunulan hibrit algoritma, TON_IOT veri seti kullanılarak da değerlendirilmiştir. Veri setinin içerdiği ProcessedWindowsDataset-Windows10 veri kümesine ait saldırı tespit doğruluk oranı Çizelge 7.4'ta verilmiştir. Şekil 7.7'de ise sonuçların grafiğe dökülmüş hali verilmiştir.

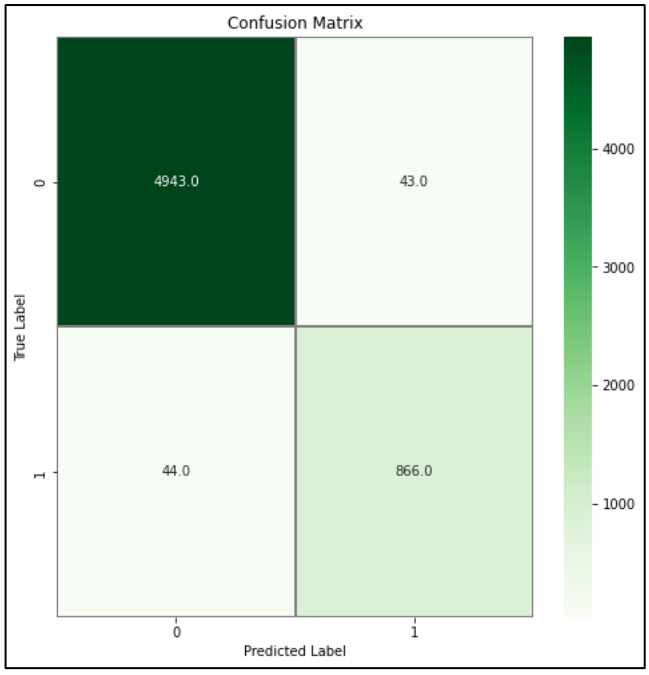
Çizelge 7.4. TON_IOT veri setine ait binary sınıflandırma sonuçları

	Doğruluk	Kesinlik	Geri Çağırma	F1 Skor
DT	97,67	97,67	97,67	97,67
RF	98,23	98,23	98,23	98,23
LR	96,69	96,69	96,69	96,69
GB	97,50	97,49	97,50	97,44
ADA	92,96	93,01	92,96	92,22
KNN	98,50	98,58	98,50	98,52
MLP	98,54	98,57	98,54	98,55
CNN	98,32	98,32	98,32	98,32
LSTM	90,94	90,94	90,94	90,94
NB	76,98	88,07	76,98	77,83
Proposed	98,75	98,75	98,75	98,75

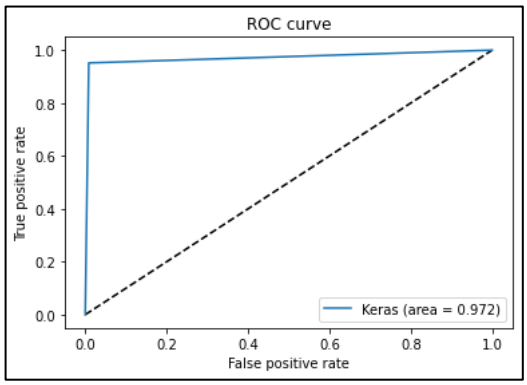


Şekil 7.7. TON_IOT veri seti binary sınıflandırma sonuç grafiği

Çizelge 7.4'te ve Şekil 7.7'de görüleceği üzere TON_IOT veri setindeki binary sınıflandırmadaki en yüksek doğruluk değerine önerilen algoritma ulaşmıştır. Bunu MLP, KNN ve CNN algoritmaları takip etmiştir. En düşük sonuç NB algoritmasına aittir. Geliştirilen hibrit algortimanın TON_IOT veri setinde binary olarak test edilmesine ait confusion matrisi Şekil 7.8'de verilmiştir. Oluşturulan ROC Curve ait diyagram ise Şekil 7.9'da sunulmuştur.



Şekil 7.8. TON_IOT veriseti binary sınıflandırmaya ait confusion matrisi



Şekil 7.9. TON_IOT veriseti binary sınıflandırmaya ait ROC eğrisi

Şekil 7.8'deki Confusion matrix'e göre yanlış pozitif oranı (FPR), toplam 87 kayıt ile oldukça düşük değerde kalmıştır. Şekil 7.9'daki çoklu sınıflandırmaya ait ROC grafiğine göre ise AUC-ROC değeri 0,99 değerine yakın cıkmısıtr. Çizelge 7.5'te ise TON_IOT veri seti kullanılarak daha önce yapılan çalışmalar listelenmiştir. İki tabloda da görüleceği üzerine geliştirilen ve burada sunulan hibrit algoritma iki veri setinde yapılan calısmalardan daha iyi sonucu almıştır.

Çizelge 7.5. TON-IoT veri setinde yapılan çalışmaların karşılaştırılması

Çalışmalar	Kullanılan Veri Seti	Doğruluk
Rehab ve diğerleri (2022) [55]	Processed Windows dataset	%98,39
Dobrojevic ve diğerleri (2023) [56]	Processed Windows dataset	%96,65
Önerilen	Processed Windows dataset	%98,75

Diğer veri setinde en düşük sonucu veren DT ve ADA Boost algoritmalarının doğruluk değerleri TON_IOT veri setinde yükselmiştir. Aynı sekilde ilk veri setinde yüksek sonuç veren NB algoritmasının doğruluk değeri ise ikinci veri setinde %76'a kadar düşerek en düşük sonucu alan algoritma olmuştur. Geliştirilen algoritma farklı bir veri seti olan TON_IOT veri setinde de test edilerek, algoritmanın güvenilirliği gösterilmiştir. TON_IOT veri setinde ise %98,75 oranında yüksek bir saldırı tespit başarısına ulaşılmıştır.

8. TARTIŞMA, SONUÇ VE ÖNERİLER

Modern dünyamızda internet kullanımı, cihazların birbiriyle haberleşmesi gibi ihtiyaçlar kaçınılmazdır. Bu ihtiyaçların bize sağladığı yararların yanında bunların kötüye kullanılma durumları da vardır. Ağ ve haberleşme kanallarının kötü amaçlı kullanım türlerinden biri kötücül saldırılardır. Bunun en yaygın olarak kullanılanı, hedef sistemlerin kullanımını sınırlandırmak veya tamamen erişilemez hale getirmeyi amaçlayan DDoS saldırılarıdır. DDoS saldırılarının tespit edilmesi, bunlara karşı konulabilmesi için oldukça önemlidir. Bu çalışmada, DDoS saldırlarının tespit edilmesi için CNN ve LSTM derin öğrenme modellerinin kullanıldığı hibrit bir derin öğrenme algoritması geliştirilmiştir. Bu algoritmanın eğitiminde ve test edilmesinde güncel veri setlerinden olan CICIoT2023 ve TON_IOT verisetleri kullanılmıştır. Öncelikle CICIoT2023 veri seti binary olarak test edilmiş, daha sonra ise çoklu sınıf olarak test edilmesi sağlanmıştır. Algoritma değerlendirilmesi doğruluk, kesinlik, geri çağırma, F1-Skor ve ROC Curve verileri hesaplanarak yapılmıştır. Bu eğitimlerin ve testlerin sonucunda %99,995 saldırı tespit ve %99,96 oranında ise saldırı türü belirleme oranına ulaşılmıştır. Bu yüksek doğruluk oranına ulaşılarak, literatüre katkı sağlanarak, gelecek çalışmalar için bir referans noktası oluşturulmuştur. TON_IOT veri setin kullanılarak yapılan değerlendirme de ise %98,75'lik saldırı tespit oranına ulaşılmıştır. Bu çalışmada geliştirilen hibrit derin öğrenm algoritmasının en yüksek doğruluk değerine ulaşması amaçlanmıştır.

Algoritmada LSTM kullanılmasının nedeni, derin öğrenme algoritmaları olan LSTM'lerin akış dinamiklerini yakalamada ve döngü boyunca bilgiyi muhafaza etmede etkili olmasıdır. LSTM algoritması uzun vadeli bağımlılıkları öğrenebilir ve sıralı verileri bellekte tutabilir. LSTM algoritması, önceki verinin unutulup unutulmayacağına karar verir. Sıralı veriler kullanılarak yapılan hesaplamalar sayesinde bu yapının DDoS gibi yoğun saldırılarda kullanılması uygun görülmüştür. Sınıflandırma aşamasında kullanılan bir diğer algoritma ise CNN’dir. CNN algoritması, görüntü sınıflandırma, ses sınıflandırma ve video sınıflandırmada başarılı sonuçlar sağlamakta ve ağın derinliğini ve genişliğini değiştirerek sınıflandırma problemlerinin üstesinden gelebilecek güçlü yeteneklere sahiptir. Evrişim süreci sayesinde CNN, benzer sayıda katmana sahip standart ileri beslemeli sinir ağlarına kıyasla daha az bağlantı ve parametreyle zamana duyarlı saldırı durumlarını tespit edebilmektedir. LSTM ve CNN'in tamamlayıcı özelliklerinden yararlanılarak hibrit

kullanımının, yapılan testlerde bireysel kullanımlarına göre daha iyi sonuçlar verdiği gözlemlenmiştir.

Ayrıca veri setinin ön işleme adımlarında analiz edilmesi ve eksik verilerin giderilmesi, verilerin kullanılabilir hale getirilmesini sağlamıştır. Veri kümesinden en uygun özelliklerin seçilmesiyle algoritmanın hesaplama yükü azaltılır, bu da eğitim ve test maliyetlerinin azalmasına neden olur. Algoritmamızda kullanılan ön işleme adımları, hem eğitim hem de test verilerinin karmaşık olmayan bilgilerle işlenmesini sağlar.

Kullanılan algoritmalar aynı olmasına rağmen sınıflandırma doğruluğu farklı veri setlerinde farklı sonuçlar verebilir. Bu durum çalışmada ikinci veri seti olarak kullanılan TON_IOT veri setinin değerlendirme sonuçlarından da görülebilmektedir. CICIoT2023 ile karşılaştırıldığında TON_IOT veri setinde ADA ve DT algoritmalarının doğruluk değerleri artmıştır. NB algoritması çoklu sınıf değerlendirmesinde yüksek sonuçlar vermesine rağmen her iki ikili değerlendirmede de en düşük doğruluk sonucunda kalmıştır. NB algoritması, doğruluğunu etkileyen özellikler arasındaki karşılıklı bağımlılığı dikkate almamaktadır [57]. NB algoritmasının ikili değerlerdeki düşük sonuçları, ikili sınıf değerlendirmesinde yoğun bağımlılıkların olması ve bunun da doğruluk değerini etkilemesi olarak düşünülebilir.

Yüksek doğruluk değerine ulaşılması için büyük hacimde veri kullanılması gerekmektedir. Kullanılan yüksek hacimli veriler de eğitim ve test sürelerini arttırmaktadır. Gelecek çalışma olarak bu eğitim sürelerinin optimize edilerek hem yüksek doğruluk oranlarına hem de maaliyeti düşük saldırı tespit sistemlerinin geliştirilmesi literatür için yüksek bir katkı olacaktır.

KAYNAKLAR

1. Dener, M., Al, S., Orman, A. (2022). STLGBM-DDS: An Efficient Data Balanced DoS Detection System for Wireless Sensor Networks on Big Data Environment. *IEEE Access*, 10, 92931–92945.

2. Batchu, R. K., Seetha, H. (2021). A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning. *Computer Networks*, 200, 108498.

3. Al, S., Dener, M. (2021). STL-HDL: A new hybrid network intrusion detection system for imbalanced dataset on big data environment. *Computers & Security*, 110, 102435

4. Cil, A. E., Yildiz, K., Buldu, A. (2021). Detection of DDoS attacks with feed forward based deep neural network model. *Expert Systems with Applications*, 169, 114520.

5. Giusto, D., Iera, A., Morabito, G., Atzori, L. (2010). *The Internet of Things*. 20th Tyrrhenian Workshop on Digital Communications, 442p, Springer.

6. Taş, O., Kiani, F. (2021). Nesnelerin interneti (IoT) ve kablosuz algılayıcı ağların güvenliğine yapılan saldırıların tespit edilmesi ve önlenmesi. *Politeknik Dergisi*, 24(1): 219-235.

7. Gündüz, M. Z., Daş, R. (2018). Nesnelerin interneti: Gelişimi, bileşenleri ve uygulama alanları. *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, 24(2), 327-335.

8. Granjal J., Monteiro E., Silva J.S. (2015). Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues. *IEEE Communication Surveys & Tutorials*, 7(3), 1294-1312.

9. Kalaycı T.E. (2009). *Kablosuz Sensör Ağlar ve Uygulamaları*. XI. Akademik Bilişim Konferansı Harran Üniversitesi, Şanlıurfa.

10. Salminen, J., Kaartemo, V. (2014). *Big Data: Definitions, Business Logics, and Best Practices to Apply in Your Business*. New York: Amazon

11. Özköse, H. (2020). Büyük Veri Kavramı ile İlgili Akademik Yayınların Metin Madenciliği Yöntemi ile Analizi. *Veri Bilimi*, 3(1), 11-20.

12. Aktan, E. (2018). Büyük Veri: Uygulama Alanları, Analitiği ve Güvenlik Boyutu. *Bilgi Yönetimi Dergisi*, 3-4.

13. Gandomi, A., Haider, M. (2015). Beyond the Hype: Big Data Concepts, Methods, and Analytics. *International Journal of Information Management*. 137-144.

14. Atan, S. (2016). Veri, Büyük Veri ve İşletmecilik. *Balıkesir Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*, 19(35), 137-154. https://doi.org/10.31795/baunsobed.645312

15. Gahi, Y., Guennoun, M., Mouftah, H. T. (2016). *Big Data Analytics: Security and Privacy Challenges*. IEEE Symposium on Computers and Communication (ISCC), Messina, Italy, 952-957.

16. Patil, N. V., Krishna, C. R., Kumar, K. (2022). SSK-DDoS: distributed stream processing framework based classification system for DDoS attacks. *Cluster Computing*, 25(2), 1355-1372.

17. Cebeloglu, F.S., Karakose, M. (2019). *A cyber security analysis used for unmanned aerial vehicles in the smart city.* 1st International Informatics and Software Engineering Conference (UBMYK), Ankara, Turkey, pp. 1–6.

18. Sreeram, I., Vuppala, V.P.K. (2019). HTTP flood attack detection in application layer using machine learning metrics and bio inspired bat algorithm. *Applied Computing and Informatics*, 15, 59–66.

19. Chen, E.Y. (2005). *Detecting TCP-based DDoS attacks by linear regression analysis.* The Fifth IEEE International Symposium on Signal Processing and Information Technology, Athens, Greece, pp. 381–386.

20. Raptis, G.E., Katsini, C., Alexakos, C. (2021, 26–28 July). *Towards Automated Matching of Cyber Threat Intelligence Reports based on Cluster Analysis in an Internet-of-Vehicles Environment.* The 2021 IEEE International Conference on Cyber Security and Resilience (CSR), Rhodes, Greece, pp. 366–371.

21. Kumari, P., Jain, A.K. (2023). A Comprehensive Study of DDoS Attacks over IoT Network and Their Countermeasures. *Computers & Security*, 127, 103096.

22. Güven, E. N. (2007). *Zeki Saldırı Tespit Sistemlerinin Incelenmesi, Tasarımı ve Gerçekleştirilmesi*. Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

23. Almaraz-Rivera, J. G., Perez-Diaz, J. A., Cantoral-Ceballos, J. A. (2022). Transport and Application Layer DDoS Attacks Detection to IoT Devices by Using Machine Learning and Deep Learning Models. *Sensors*, 22(9), 3367.

24. Jia, Y., Zhong, F., Alrawais, A., Gong, B., Cheng, X. (2020). Flowguard: an intelligent edge defense mechanism against IoT DDoS attacks. IEEE Internet of Things *Journal*, 7(10), 9552-9562.

25. Alghazzawi, D., Bamasag, O., Ullah, H., Asghar, M. Z. (2021). Efficient detection of DDoS attacks using a hybrid deep learning model with improved feature selection. *Applied Sciences*, 11(24), 11634.

26. Chartuni, A., Márquez, J. (2021). Multi-Classifier of DDoS Attacks in Computer Networks Built on Neural Networks. *Applied Sciences*, 11(22), 10609.

27. Ferrag, M. A., Shu, L., Djallel, H., Choo, K. K. R. (2021). Deep learning-based intrusion detection for distributed denial of service attack in Agriculture 4.0. *Electronics*, 10(11), 1257.

28. Bhati, A., Bouras, A., Qidwai, U. A., Belhi, A. (2020, July). *Deep learning based identification of DDoS attacks in industrial application.* The Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), 190-196, IEEE.

29. Wei, Y., Jang-Jaccard, J., Sabrina, F., Singh, A., Xu, W., & Camtepe, S. (2021). Ae-mlp: A hybrid deep learning approach for ddos detection and classification. *IEEE Access*, 9, 146810-146821.

30. Ur Rehman, S., Khaliq, M., Imtiaz, S. I., Rasool, A., Shafiq, M., Javed, A. R., Bashir, A. K. (2021). Diddos: An approach for detection and identification of distributed denial of service (ddos) cyberattacks using gated recurrent units (gru). *Future Generation Computer Systems*, 118, 453-466.

31. Susilo, B., Sari, R. F. (2020). Intrusion detection in IoT networks using deep learning algorithm. *Information*, 11(5), 279.

32. Kumar, P., Bagga, H., Netam, B. S., Uduthalapally, V. (2022). SAD-IoT: Security analysis of ddos attacks in iot networks. *Wireless Personal Communications*, 122(1), 87-108.

33. Alzahrani, R.J., Alzahrani, A. (2021). Security Analysis of DDoS Attacks Using Machine Learning Algorithms in Networks Traffic. *Electronics*, 10, 2919.

34. Haq, Mohd Anul, Khan, Mohd. (2021). Development of PCCNN-Based Network Intrusion Detection System for EDGE Computing. *Computers, Materials and Continua*. 71. 10.32604/cmc.2022.018708.

35. Iwendi, C., Rehman, S., Javed, A.R., Khan, S., Srivastava, G. (2021). Sustainable Security for the Internet of Things Using Artificial Intelligence Architectures. *ACM Transactions on Internet Technology*. 21. 1-22.

36. Gamal, M., Abbas, H. M., Moustafa, N., Sitnikova, E., Sadek, R. A. (2021). Few-Shot Learning for Discovering Anomalous Behaviors in Edge Networks. *Computers, Materials and Continua*, 69(2), 1823-1837.

37. Gad, A.R., Nashat, A. A., Barkat, T. M. (2021). Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset. *IEEE Access*, vol. 9, pp. 142206-142217.

38. Disha, R.A., Waheed, S. (2022). Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique. *Cybersecurity* 5,1.

39. Kaur, J., Agrawal, A., Khan, R.A. (2023). P2ADF: a privacy-preserving attack detection framework in fog-IoT environment. *International Journal of Information Security*, 22, 749–762

40. Verma, R., Chandra, S. (2023). RepuTE: A soft voting ensemble learning framework for reputation-based attack detection in fog-IoT milieu. *Engineering Applications of Artificial Intelligence*, 118, 105670.

41. Neto, E.C.P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., Ghorbani, A.A. (2023). CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors*, 23, 5941.

42. İnternet: Google Colaboratory. URL: https://colab.research.google.com, Son Erişim Tarihi: 02.11.2023.

43. Gülburun, S. (2022). *Oy Birliği ve Özelleşmiş Sınıflandırıcılar ile Zararlı Yazılım Tespiti*. Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

44. Taylan, K. (2022). *Android Kötücül Yazılım Analizinde Derin Öğrenme Modellerinin Performansının Karşılaştırılması*. Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

45. Tülay, A. (2020). *Android Zararlı Yazılım Tespit Sistemi*. Yüksek Lisans Tezi, Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir .

46. İnternet: Ton iot dataset. URL: https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-ton-iot-Datasets/, Son Erişim Tarihi: 12.11.2023.

47. İnternet: Description of Windows 10 Features. URL: https://cloudstor.aarnet.edu.au/plus/s/ds5zW91vdgjEj9i?path=%2FDescription_stats_datasets%2FDescription_stats_Windows_dataset#pdfviewer, Son Erişim Tarihi: 12.11.2023.

48. Tsimenidis, S., Lagkas, T., Rantos, K. (2022). Deep Learning in IoT Intrusion Detection. *Journal of Network System Management*, 30, 8.

49. Lin, S., Tian, H. (2020). *Short-Term Metro Passenger Flow Prediction Based on Random Forest and LSTM*. IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, pp. 2520-2526.

50. Sayed, M. S. E., Le-Khac, N. A., Azer, M. A., Jurcut, A. D. (2022). A Flow-Based Anomaly Detection Approach With Feature Selection Method Against DDoS Attacks in SDNs. *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 4, pp. 1862-1880.

51. Han, J., Pei, J., Kamber, M. (2011). Data Mining: Concepts and Techniques. *Elsevier: Amsterdam*, The Netherlands.

52. Alsirhani, A., Sampalli, S., Bodorik, P. (2018). *DDoS detection system: utilizing gradient boosting algorithm and apache spark*. The IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1-6, IEEE.

53. Khattak, A., Asghar, M.Z., Ali, M., Batool, U. (2021). An efficient deep learning technique for facial emotion recognition. *Multimedia Tools and Appl*ications, 81, 1649–1683

54. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.

55. Mohamed, R., Mosa, F., Sadek, R. (2022). Efficient Intrusion Detection System for IoT Environment. *International Journal of Advanced Computer Science and Applications*. 13. 10.14569/IJACSA.2022.0130467.

56. Dobrojevic M, Zivkovic M, Chhabra A, Sani NS, Bacanin N, Mohd Amin M. (2023). Addressing Internet of Things security by enhanced sine cosine metaheuristics tuned hybrid machine learning model and results interpretation based on SHAP approach. *PeerJ Computer Science*, 30;9:e1405.

57. Asharf, J., Moustafa, N., Khurshid, H., Debie, E., Haider, W., Wahab, A. (2020). A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions. *Electronics*, 9, 1177.

Gazili olmak ayrıcalıktır