



**T.C.
KIRIKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

**GWO VE MFO ALGORİTMALARININ HİBRİTLENMESİYLE
SÜRÜ ZEKASI TABANLI OPTİMİZASYON ALGORİTMASI
GELİŞTİRİLMESİ VE YAPAY SİNİR AĞI YAKLAŞIMIYLA AĞ
SALDIRILARININ TESPİTİNDE KULLANILMASI**

**HASAN DALMAZ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI**

DOKTORA TEZİ

**DANIŞMAN
Dr. Öğr. Üyesi Erdal ERDAL**

KIRIKKALE-2023

Hasan DALMAZ tarafından hazırlanan “GWO VE MFO ALGORİTMALARININ HİBRİTLENMESİYLE SÜRÜ ZEKASI TABANLI OPTİMİZASYON ALGORİTMASI GELİŞTİRİLMESİ VE YAPAY SİNİR AĞI YAKLAŞIMIYLA AĞ SALDIRILARININ TESPİTİNDE KULLANILMASI” adlı tez çalışması, aşağıdaki jüri tarafından OY BİRLİĞİ ile Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalında DOKTORA TEZİ olarak kabul edilmiştir.

Danışman: Dr. Öğr. Üyesi Erdal ERDAL

İmza.....

Bilgisayar Yazılımı Anabilim Dalı, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum/onaylamıyorum

Başkan: Doç. Dr. Halil Murat ÜNVER

İmza.....

Bilgisayar Donanımı Anabilim Dalı, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum/onaylamıyorum

Üye: Dr. Öğr. Üyesi Mahmut ÜNVER

İmza.....

Bilgisayar Programcılığı Anabilim Dalı, Kırıkkale Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum/onaylamıyorum

Üye: Dr. Öğr. Üyesi Abdül Kadir GÖRÜR

İmza.....

Yazılım Mühendisliği Anabilim Dalı, Çankaya Üniversitesi

Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum/onaylamıyorum

Üye: Dr. Öğr. Üyesi Murat SARAN

İmza.....

Bilgisayar Mühendisliği Anabilim Dalı, Çankaya Üniversitesi

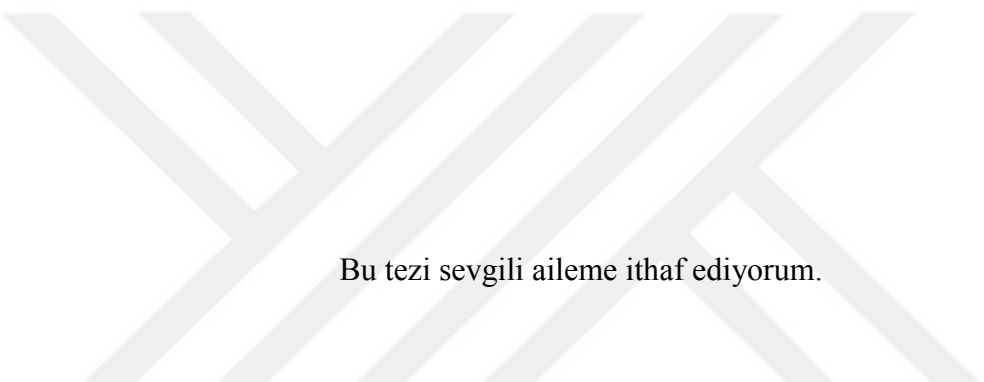
Bu tezin, kapsam ve kalite olarak Doktora Tezi olduğunu onaylıyorum.

Tez Savunma Tarihi: 21/02/2023

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

.....

Prof. Dr. Recep ÇALIN
Fen Bilimleri Enstitüsü Müdürü



Bu tezi sevgili aileme ithaf ediyorum.

ETİK BEYANI

Kırıkkale Üniversitesi Fen Bilimleri Enstitüsü Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmada;

- 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 çalışmada 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.

Hasan DALMAZ

21.02.2023

ÖZET

GWO VE MFO ALGORİTMALARININ HİBRİTLENMESİYLE SÜRÜ ZEKASI TABANLI OPTİMİZASYON ALGORİTMASI GELİŞTİRİLMESİ VE YAPAY SİNİR AĞI YAKLAŞIMIYLA AĞ SALDIRILARININ TESPİTİNDE KULLANILMASI

Kırıkkale Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı, Doktora Tezi

Danışman: Dr. Öğr. Üyesi Erdal ERDAL

Şubat 2023, 96 sayfa

Bu tez çalışması, son yıllarda önemli ölçüde artan ağ saldırılarının yüksek doğrulukla tespit edilmesi ve bu saldırıların olumsuz etkilerinden kaçınılması konusundaki ihtiyaca yönelik hazırlanmıştır. Saldırı tespit sistemleri, saldırı modellerine ve yaklaşımlarına sorunsuz bir şekilde yanıt vermelidir. Saldırı tespitinde metasezgisel algoritmaların kullanılması, düşük hesaplama maliyetleriyle optimuma yakın çözümler üretebilir. Bu algoritmalarından daha iyi performans elde etmek ve sonuçları daha da iyileştirebilmek için algoritmalar hibritleştirilebilir ve bu da daha başarılı sonuçların elde edilmesini sağlayabilmektedir. Günümüzde bu konuda pek çok çalışma yapılmaktadır. Bu çalışmada, Güve Alevi Optimizasyon (MFO) ve Gri Kurt Optimizasyon (GWO) algoritmalarını kullanan yeni bir hibrit yaklaşım geliştirilmiş ve yaygın olarak kullanılan NSL-KDD, UNSW-NB15 ve CIC IDS 2017 gibi veri setlerine de uygulanmıştır. GWO algoritmasının hibritleştirme kolaylığı, basitliği, global optimal arama yapabilme yeteneği ve MFO algoritmasının en iyi çözümü elde etmedeki başarısı, bu iki algoritmanın birleştirilmesiyle daha iyi bir çözüm elde edilebileceğini düşündürmüştür. Bu sebeplerden dolayı geliştirilen hibrit algoritma ile hem GWO algoritmasının hem de MFO algoritmasının iyi yanları kullanılarak daha iyi sonuçların elde edilmesi amaçlanmaktadır. Sonuçlar incelendiğinde geliştirilen hibrit algoritma ile optimizasyon algoritmalarının test edilmesinde kullanılan Benchmark fonksiyonlarında yüksek düzeyde başarı elde edildiği görülmüştür. Karşılaştırılan 13 Benchmark fonksiyonunun 12'sinde daha iyi sonuçlar elde edilmiştir. Ayrıca farklı veri setlerinde değerlendirme kriterlerine göre elde edilen başarı oranları da dikkat çekicidir. NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri setlerinde elde edilen sırasıyla %97,4, %98,3 ve %99,2 sınıflandırma doğruluk sonuçları literatürdeki çalışmalarla karşılaştırıldığında oldukça başarılı görünmektedir.

Anahtar kelimeler: Ağ saldırısı, saldırı tespit, makine öğrenmesi, GWO, MFO

ABSTRACT

DEVELOPMENT OF A SWARM INTELLIGENCE-BASED OPTIMIZATION ALGORITHM BY HYBRIDING GWO AND MFO ALGORITHMS AND USING IT IN DETECTION OF NETWORK ATTACKS WITH AN ARTIFICIAL NEURAL NETWORK APPROACH

Kırıkkale University

Graduate School of Natural and Applied Sciences

Department of Computer Engineering, Doctorate's Thesis

Supervisor: Dr. Erdal ERDAL

February 2023, 96 pages

This thesis has been prepared to address the need to detect network attacks, which have increased significantly in recent years, with high accuracy and to avoid the negative effects of these attacks. Intrusion detection systems must respond seamlessly to attack patterns and approaches. The use of metaheuristic algorithms in intrusion detection can produce near-optimal solutions with low computational costs. To get better performance from these algorithms and further improve the results, the hybridization of the algorithms can be used, which can lead to more successful results. Today, many studies are carried out on this subject. In this study, a new hybrid approach using Moth-Flame Optimization (MFO) and Gray Wolf Optimizer (GWO) algorithms was developed and applied to widely used datasets such as NSL-KDD, UNSW-NB15 and CIC IDS 2017. The ease of hybridization, simplicity, global optimal search capability of the GWO algorithm and the success of the MFO algorithm in obtaining the best solution suggest that an effective solution can be obtained by combining these two algorithms. For these reasons, the developed hybrid algorithm aims to achieve better results by using the good aspects of both the GWO algorithm and the MFO algorithm. When the results were examined, it was seen that the developed hybrid algorithm achieved a high level of success in Benchmark functions used in testing the optimization algorithms. Better results were obtained in 12 of the 13 benchmark functions compared. In addition, the success rates obtained according to the evaluation criteria in different data sets are remarkable. The classification accuracy results of 97.4%, 98.3% and 99.2% respectively, obtained in the NSL-KDD, UNSW-NB15 and CIC IDS 2017 data sets seem to be quite successful when compared with the studies in the literature.

Key Words: Network attack, attack detection, machine learning, GWO, MFO

TEŞEKKÜR

Tezimin hazırlanması esnasında desteğini benden esirgemeyen; bilimsel deney imkânlarını, bilgi ve deneyimini sonuna kadar bizlerin hizmetine sunan tez yöneticim çok değerli hocam Sayın Dr. Öğr. Üyesi Erdal ERDAL'a, tezimin birçok aşamasında bana destek olan değerli hocalarım Sayın Doç. Dr. H. Murat ÜNVER ve Sayın Dr. Öğr. Üyesi Mahmut ÜNVER'e, bu süreçte bana destek olan çok kıymetli eşim Esra Kevser DALMAZ ve canım oğlum Bera DALMAZ'a teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET.....	iv
ABSTRACT	v
TEŞEKKÜR	vi
İÇİNDEKİLER DİZİNİ	vii
ÇİZELGELER DİZİNİ	x
ŞEKİLLER DİZİNİ	xi
SİMGELER VE KISALTMALAR	xii
1. GİRİŞ.....	1
1.1. Makine Öğrenmesi	3
1.2. Makine Öğrenmesi Türleri	4
1.2.1. Denetimli Öğrenme	4
1.2.2. Denetimsiz Öğrenme	4
1.2.3. Yarı Denetimli Öğrenme	5
1.2.4. Takviyeli Öğrenme	5
1.3. Yapay Sinir Ağları (YSA)	5
1.3.1. Yapay Sinir Ağlarının Sınıflandırılması	8
1.3.1.1. Öğrenme Metoduna Göre YSA	8
1.3.1.2. Yapılarına Göre YSA	8
1.3.1.3. Katman Sayılarına Göre YSA	10
1.3.1.4. Kullanım Amacına Göre YSA	11
1.4. Meta Sezgisel Algoritmalar	14
1.4.1. Meta Sezgisel Algoritmaların Özellikleri	14
1.4.2. Meta Sezgisel Algoritmaların Sınıflandırılması	15
2. SİBER SALDIRILAR ve TESPİT SİSTEMLERİ	17
2.1. Siber Saldırıların Evrimi	17
2.2. Saldırı Tespit Sistemleri	18
2.3. Saldırı Tespit Sistemi (STS) Modelleri	19
2.3.1. İmza Tabanlı Saldırı Tespit Sistemleri	19
2.3.2. Anomali Tabanlı Saldırı Tespit Sistemleri	20

2.4. Saldırı Tespit Sistemi Tipleri	22
2.4.1. Ağ Tabanlı Saldırı Tespit Sistemleri	23
2.4.2. Sunucu Tabanlı Saldırı Tespit Sistemleri	23
2.5. Saldırı Tespit Sistemi İçin Kullanılan Araçlar	24
2.5.1. Haystack	24
2.5.2. W&S	25
2.5.3. Midas	25
2.5.4. Ides	25
2.5.5. Nadir	25
2.5.6. NSM	26
2.5.7. Hyperview	26
2.5.8. Ustat	26
2.5.9. Dids	26
2.5.10. Idiot	26
2.5.11. Janus	27
2.5.12. Emerald	27
2.5.13. Ripper	27
2.5.14. Bro	27
2.5.15. Snort	27
2.5.16. Selks	28
2.5.17. Ossec	28
2.5.18. Firestorm	28
2.5.19. Kfsensor	28
2.5.20. Suricata	29
2.5.21. Phpids	29
2.5.22. Net Ids	29
2.5.23. Prelude	29
2.5.24. Aide	29
2.5.25. Samhain	30
3. LİTERATÜR İNCELEMESİ.....	31
3.1. Çalışma Kapsamında İncelenen Algoritmalar	36
3.1.1. Diferansiyel Gelişim Algoritması (DE)	36
3.1.2. Parçacık Sürü Optimizasyon Algoritması (PSO)	38
3.1.3. Çoklu Evren Optimizasyon Algoritması (MVO)	39
3.1.4. JAYA Algoritması	41

3.1.5. Salp Sürü Algoritması (SSA)	42
3.1.6. Sinüs Cosinüs Algoritması (SCA)	44
4. MATERYAL ve YÖNTEM.....	45
4.1. YSA Eğitiminde Kullanılan Algoritmalar.....	45
4.1.1. Gri Kurt Optimizasyon Algoritması (GWO)	45
4.1.2. Güve Alevi Optimizasyon Algoritması (MFO)	48
4.2. Benchmark Fonksiyonları	50
4.3. Veri Setleri	56
4.3.1. NSL-KDD	56
4.3.2. UNSW-NB15	58
4.3.3. CIC IDS 2017	60
4.4. Veri Ön İşleme	62
4.5. Hibrit Sınıflandırma-GWOMFO	63
4.6. Eğitim	66
4.7. Test.....	66
4.8. Değerlendirme.....	66
4.9. Simülasyon Ortamı	67
5. ARAŞTIRMA BULGULARI	70
5.1. Ağ Saldırı Tespit Sonuçları	70
6. SONUÇ	76
KAYNAKLAR.....	78
EKLER.....	90
EK.1 NSL-KDD Veri Seti Örneği.....	90
EK.2 UNSW-NB15 Veri Seti Örneği.....	92
EK.3 CIC IDS 2017 Veri Seti Örneği	94
ÖZGEÇMİŞ	96

ÇİZELGELER DİZİNİ

Sayfa

1.1. YSA ile biyolojik sinir ağı arasındaki benzerlikler.....	7
1.2. Kullanım amaçlarına göre YSA topolojileri.....	12
2.1. STS modelleri avantaj ve dezavantajları.....	22
2.2. Saldırı tespit sistemi tiplerine göre avantaj ve dezavantaj karşılaştırması.....	23
3.1. İlgili çalışmaların özeti	35
3.2. DE algoritması kaba kodu.....	36
3.3. PSO algoritması kaba kodu.....	38
3.4. MVO algoritması kaba kodu.....	40
3.5. JAYA algoritması kaba kodu.....	42
3.6. SSA algoritması kaba kodu.....	43
3.7. SCA algoritması kaba kodu	44
4.1. GWO algoritması kaba kodu	47
4.2. MFO algoritması kaba kodu	49
4.3. Kullanılan benchmark fonksiyonları ve karakteristikleri.....	51
4.4. Geliştirilen algoritma için benchmark sonuçları ve karşılaştırılması	52
4.5. NSL-KDD veri seti dağılımı	57
4.6. UNSW-NB15 veri seti dağılımı.....	59
4.7. CIC IDS 2017 veri seti dağılımı	61
4.8. GWOMFO algoritması kaba kodu	65
5.1. GWOMFO algoritması performansı	70
5.2. GWOMFO, GWO ve MFO algoritmalarının 3 farklı veri setinde karşılaştırılması	71
5.3. Çeşitli sürü zekası tabanlı algoritmalar ile GWOMFO algoritmasının 3 farklı veri seti üzerinde karşılaştırılması.....	72
5.4. GWOMFO algoritmasının NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri setleri kullanılarak yapılan çalışmalarla karşılaştırılması	75

ŞEKİLLER DİZİNİ

ŞEKİL

Sayfa

1.1. Nöron yapısının şematik gösterimi	6
1.2. Yapay sinir ağı mimarisi	7
1.3. Geri beslemeli ağların genel yapısı	9
1.4. İleri beslemeli ağların genel yapısı	10
1.5. Çok katmanlı YSA yapısı	11
1.6. Tek katmanlı YSA yapısı	11
1.7. Meta sezgisel algoritmaların kategorilendirilmesi	15
2.1. Creeper programı	17
2.2. Anomali tabanlı STS'lerin kullandığı temel metotların sınıflandırılması	21
4.1. GWOMFO ile optimize edilmiş benchmark fonksiyonları için ortalama yakınsama eğrileri	53
4.2. NSL-KDD veri seti özellikleri	58
4.3. UNSW-NB15 veri seti özellikleri	60
4.4. CIC IDS 2017 veri seti özellikleri	62
4.5. GWOMFO ile eğitilen MLP yapısı	69
5.1. GWO, MFO ve GWOMFO algoritmalarının üç farklı veri setinde doğruluk sonuçlarının karşılaştırılması	70
5.2. Çeşitli sürü zekası tabanlı algoritmalar ile geliştirilen algoritmanın (GWOMFO) doğruluk sonuçlarının 3 farklı veri kümesi üzerinde karşılaştırılması	71
5.3. NSL-KDD veri setinde TPR sonuçlarının karşılaştırılması	73
5.4. UNSW-NB15 veri setinde TPR sonuçlarının karşılaştırılması	73
5.5. CIC IDS 2017 veri setinde TPR sonuçlarının karşılaştırılması	74

SİMGELER DİZİNİ

b	Bias
Cr	Çaprazlama oranı
D	Boyut
I	İterasyon
lb	Alt limit
NP	Popülasyon büyüklüğü
P	Popülasyon
ub	Üst limit
w	YSA ağırlık değeri
y	YSA çıkış değeri

KISALTMALAR DİZİNİ

ABC	Yapay arı kolonisi algoritması
DE	Diferansiyel gelişim algoritması
DoS	Hizmet reddi saldırısı
FN	Yanlış negatif
FP	Yanlış pozitif
FPR	Yanlış pozitif oranı
GA	Genetik algoritma
GWO	Gri kurt optimizasyon algoritması
IoT	Nesnelerin interneti
JAYA	JAYA algoritması
KNN	K-en yakın komşular algoritması
LAN	Yerel alan ağı

MFO	Güve alevi optimizasyon algoritması
MLP	Çok katmanlı algılayıcı
MVO	Çoklu evren optimizasyon algoritması
OSVM	Optimize edilmiş destek vektör makineleri
PSO	Parçacık sürü optimizasyon algoritması
RBF	Çapsal tabanlı fonksiyon
SC	Sistem çağrısı
SCA	Sinüs cosinüs algoritması
SSA	Salp sürü algoritması
STS	Saldırı tespit sistemleri
SVM	Destek vektör makineleri
TN	Doğru negatif
TP	Doğru pozitif
TPR	Doğru pozitif oranı
YSA	Yapay sinir ağ

1. GİRİŞ

Teknolojinin evrimi, genel olarak tüm alanlarda ağ güvenliği konusunun önemini artırmıştır. Ağ trafiğinin ve ağ bağlantılı sistemlerin sayısının sürekli artması uygulama anlamında kolaylıklar sağlayabilirken ağ güvenliği için alınan önlemlerin yetersiz kalmasına ve olası ağ saldırılarına karşı sistemlerde zafiyet oluşmasına sebep olabilmektedir. Günlük ve çalışma hayatımızın vazgeçilmez bir parçası hâline gelen internet, internet teknolojilerindeki gelişmeler, nesnelerin interneti (IoT) teknolojisinin birçok alanda kullanımının artması ve internet tabanlı sosyal ağlar, ağ saldırılarında yaşanan artışın başlıca sebepleri arasında sayılabilir [1]. Ağ güvenliği konusunda yayımlanmış çeşitli raporlarda, ağ saldırılarının özellikle son yıllarda oldukça rahatsız edici karmaşıklık ve ölçek düzeyine ulaştığı belirtilmektedir. Ağ güvenliği konusunda güvenlik önlemleri için başlangıç seviyesinde veri şifreleme, kullanıcı güvenliği ve güvenlik duvarlarının kullanımı gibi geleneksel yöntemler kullanılmaktadır. Ancak kullanılan parolaların zayıflığı, parola güvenlik ihlalleri, yetkisiz kullanımı engellemez ve kullanıcı kimlik doğrulamasının başarısız olmasına sebep olur. Güvenlik duvarlarının yapılandırmalarında hatalar varsa bu da güvenlik duvarlarının etkisiz olmasına sebep olabilmektedir [2]. Özellikle askeri, ticari ve kamu ağ sistemleri gibi hedeflere yönelik oluşan tehditler, siber güvenlik ve bu alandaki farkındalığın artırılmasını gerekli kılmaktadır [3].

Saldırı Tespit Sistemleri (STS), içeriden veya dışarıdan ağa yapılan saldırıların tespit edilmesini ve önlenmesini amaçlamaktadır. Bu sistemler, imza tabanlı ve anomali tabanlı olmak üzere 2 ana başlık altında incelenmektedir. İmza tabanlı sistemler, bilinen ve daha önceden görülen saldırı türlerini veri tabanında depolarken anomali tabanlı sistemler, gerçek zamanlı olarak paketleri anormallik derecelerine göre değerlendirmektedir. Bu değerlendirme için de çeşitli makine öğrenmesi teknikleri kullanılmaktadır [4]. Bunlara ek olarak STS'nin bazı zayıf yönleri de bulunmaktadır. Bunların en başında ise tehdit oluşturmayan olayların saldırı olarak (yanlış pozitif), sisteme tehdit oluşturan olayların ise saldırı değil olarak (yanlış negatif) sayılabilmektedir. Bu sebepler nedeniyle saldırılarla ilgili elde edilen bilgilerin yorumlanması büyük önem arz etmektedir.

İnternet kullanımının yaygınlaşması, sistemlere yönelik gerçekleştirilecek saldırıların daha kolay yapılabilmesine yol açmıştır. Kritik ve önemli verileri korumak

için STS'ye olan ihtiyaç her geçen gün artmaktadır [5]. Ağ saldırıları ve yöntemleri geliştikçe bu saldırıları önlemek için yeni yaklaşımlar da ortaya çıkmaktadır. Özellikle bir optimizasyon problemi olarak değerlendirilen ağ saldırı tespitinde en uygun çözümün bulunabilmesi için kullanılacak algoritma büyük önem taşımaktadır. Optimal çözümü bulabilmek için kullanılan algoritmalar deterministik ve stokastik olmak üzere 2 sınıfa ayrılmaktadır. Deterministik algoritmalarda giriş değeri değişmedikçe elde edilen çözüm değişmez. Ancak çözümün bulunmasında yapısal zorluklar ortaya çıkabilmektedir ve dolayısıyla istenilen çözüm elde edilemeyebilir. Bu nedenlerle, doğadan ilham alan bir stokastik algoritma türü olan metasezgisel algoritmalar kullanılmaktadır. Bu algoritmalar; oluşturulmaları kolay ve metasezgisel algoritmalarla kolay hibritlenebilir, farklı problemlere kolayca uygulanabilir ve yerel optimal değerlerden kaçınırlar [6,7]. Metasezgisel algoritmalar, sürü ve popülasyon tabanlı algoritmalarlardır. Bu algoritmalara örnek olarak Salp Sürü Algoritması (SSA), Genetik Algoritma (GA), Güve Alevi Optimizasyon Algoritması (MFO) ve Gri Kurt Optimizasyon Algoritması (GWO) verilebilir. Bu algoritmaların avantajları olduğu gibi yerel arama, tekrarlanamayan kesin çözümler ve yakınsama belirsizlikleri gibi bazı dezavantajları da bulunmaktadır. Daha iyi bir çözüm elde etmek için birden fazla algoritma bir arada kullanılarak başarılı özelliklerin birleştirildiği hibrit yöntemler geliştirilebilmektedir [8]. Geliştirilen hibrit algoritmalarda, mevcut algoritmaların zayıf yönleri giderilmeye ve daha başarılı sonuçlar elde edilmeye çalışılmaktadır.

Son yıllarda ağ saldırılarının tespiti ve önlenmesi konuları oldukça popüler hâle gelmiştir. Ancak literatürde saldırıların sınıflandırılmasında algoritmaların nasıl daha verimli hâle getirilebileceğine dair az sayıda çalışma bulunmaktadır. Farklı algoritmalar kullanan bir hibrit algoritma geliştirirken Benchmark fonksiyonları kullanılarak sunulan çözümün ne tür problemlerde daha etkili çalışabileceği hakkında daha fazla bilgi edinmek gerekmektedir. Ağ güvenliği ve ağ saldırılarının tespiti ve önlenmesi sürekli iyileştirme gerektiren bir alan olduğundan bu alandaki yeni çalışmalar büyük önem kazanmaya devam etmektedir. Yapılan yeni çalışmalar, önceki çalışmaların eksikliklerini giderebilmekte ve gelecekte yapılacak çalışmalara ışık tutabilmektedir. Bu nedenle bu çalışma, sadece ağ güvenliği için değil aynı zamanda çeşitli optimizasyon problemlerinin çözülmesi için de geçerli olması sebebiyle oldukça önemlidir.

Bu çalışmada yeni bir hibrit algoritma geliştirilmiş ve literatürde sıklıkla kullanılan NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri setleri kullanılarak test edilmiştir.

Literatürde yer alan birçok çalışmanın aksine, bu çalışma kapsamında geliştirilen hibrit GWOMFO algoritması Benchmark fonksiyonları üzerinde de test edilmiş sonrasında ise ağ saldırılarının tespit edilmesinde kullanılmıştır. GWOMFO, GWO ve MFO'nun en iyi özelliklerini içerir. GWOMFO algoritmasının performansı, en iyiye yakın küresel optimal çözümü bulmak ve 13 Benchmark fonksiyonu kullanarak hızlı bir yakınsama oranı elde etmektir.

Bu çalışma; yeni bir hibrit algoritma geliştirmeyi, mevcut çalışmalardan daha başarılı sonuçlar elde ederek etkinliğini göstermeyi, yeni bir hibrit algoritmayı bilim camiasında çeşitli problemlerde kullanılabilecek şekilde literatüre kazandırmayı amaçlamakta ve çalışma bu bakımdan özgün değer taşımaktadır.

1.1. Makine Öğrenmesi

Makine Öğrenmesi, bir problemi o problemten elde edilen veriler dâhilinde modelleyen bilgisayar algoritmaları için kullanılan bir tekniktir. Kullanılan algoritma ve veri seti ile elde edilen model en iyi performansı sağlayacak şekilde tasarlanır [9]. Girdi olarak sağlanan veriler üzerindeki ilişkilerin modellenmesi amaçlanır ve yeniden bilgi şeması yapılandırılır. Elde edilen öğrenme sonucu ise tahmin, kestirim ve sınıflandırma için kullanılabilir [10]. Tahminler, karar alma veya eğitim için kullanılan verilere dayalı bir model oluşturulur [11]. Buradaki genel amaç ise verinin örüntüsünün tanınmasıdır. Makine öğrenmesinin sağladığı en önemli avantajlardan biri çok miktarda veriyi inceleyerek gözden kaçma ihtimali olan örüntüleri ortaya çıkarabilmesidir [12].

Günümüzde, teknolojinin getirdiği olanaklarla birlikte daha hızlı bilgi işleme gücü ve daha uygun maliyetlerin yanı sıra veri seti olarak kullanılabilecek depolanmış verinin oldukça büyük boyutlara ulaşması, kullanılan verilerdeki örüntü ya da eğilimin kolaylıkla bulunabilmesi için bu alanda algoritmaların hızlıca geliştirilmesi ve kullanılmasını sağlamıştır. Bu da makine öğrenmesi alanının yaygınlaşmasına ve hemen hemen her türlü problem için başvuru bir teknik olarak öne çıkmasını sağlamıştır. Temel olarak veri, hesaplama ve istatistik teorisine dayanmaktadır [13].

Girdi olarak kullanılan verilerle model oluřturma sreci, eēitim ve bu srete kullanılan veriler, eēitim verisi olarak adlandırılmaktadır. Bu sre sonucunda eēitilen model, girdinin ıktı ile iliřkisine ynelik yeni bilgiler verebilir ve farklı veriler kullanıldığında da eřitli tahminler yapmak iin kullanılabilir [14]. Makine ērenmesi, doēal dil iřleme, dokman ve metin sınıflandırma, aē saldırı tespiti ve medikal teřhis gibi birok konuda makine ērenmesi teknikleri kullanılarak zmler retilenmektedir [15].

1.2. Makine ērenmesi Trleri

Makine ērenmesi, ērenme yntemlerine gre drt bařlıkta ele alınmaktadır. Bunlar; takviyeli ērenme, yarı denetimli ērenme, denetimsiz ērenme ve denetimli ērenme olarak adlandırılmaktadır.

1.2.1. Denetimli ērenme

Sistemin girdi verilerini kullanarak ıkıřta beklenen sonucun elde edilebilmesi iin eřitli grevlerin iřletilmesidir. Bu ērenme trnde veriler etiketli olarak sunulmaktadır. Her rnek iin girdi ve ıkıř bilgileri biliniyor olmalıdır. Sistem alıřmasında gzlem sonucu elde edilen deēerler eēitim ařaması iin aēa giriř olarak sunulur. Bu deēerler ile girdi ve ıkıř arasında iliřki kurulması amalanır. Oluřan bu model, karřılařılacak sonraki girdilerle bu girdilere karřılık gelen ıkıřların tahmin edilebilmesini saēlamaktadır.

1.2.2. Denetimsiz ērenme

Bu ērenme tr sadece giriř veri kmesine odaklıdır. Ynlendirilmemiř ya da tanımlayıcı sınıflandırma olarak da bilinmektedir. Verilerdeki gizli kalıpların ya da gruplamaların keřfedilmesi iin olduka kullanıřlıdır [16]. Etiketli verinin olmadıēı denetimsiz ērenme trnde ama gzlemlenen verideki karakteristiēi ortaya ıkarmak ve ele alınan problemin yapısı ile ilgili fikir sahibi olunmasını saēlamaktır [17]. Bařka bir ifadeyle denetimsiz ērenme, ele alınan rneklerin kmelenmesi ve blmlenmesi iin yapılan keřifsel analizdir [18].

1.2.3. Yarı Denetimli Öğrenme

Denetimli ve denetimsiz öğrenmeden farklı olarak etiketlenmemiş veriler bulunmakta fakat algoritma için bazı ek denetim bilgileri sunulmaktadır. Sunulan bu bilgiler genellikle ele alınan örneklerle ilgili hedefler olmaktadır. Manifold ve küme varsayımı olmak üzere iki temel varsayım üzerinde incelenen bu öğrenmede, manifold varsayımında benzer girdilerin benzer çıktıları olmalı varsayımı bulunmaktadır. Küme varsayımında ise benzer girdileri olan verilerin aynı sınıf etiketinin olduğu varsayılmaktadır. Verilerin etiketlenme şekline göre bu öğrenme türü; kendi kendine eğitim, ortak eğitim ve aktif öğrenme olarak da kategorilere ayrılabilir [11].

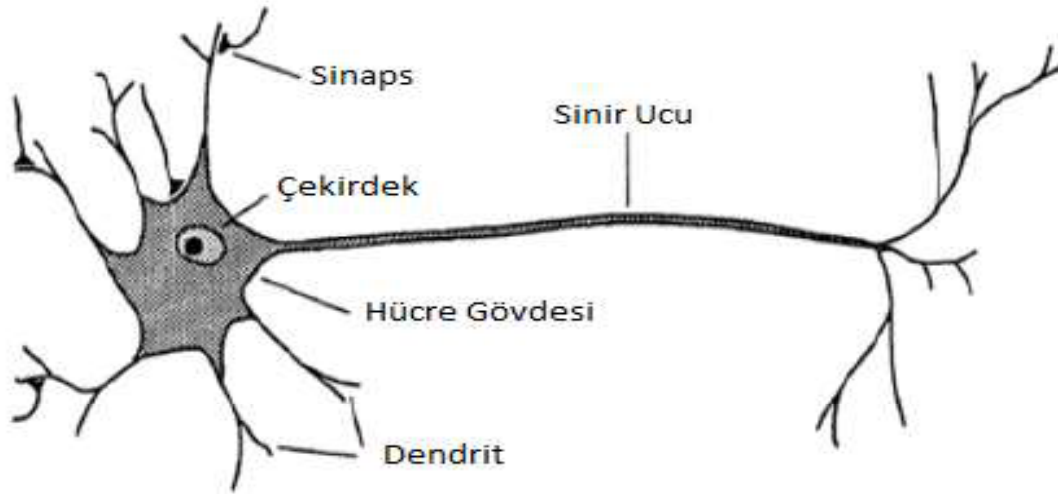
1.2.4. Takviyeli Öğrenme

Ödül ve ceza kavramının olduğu bu öğrenme türünde sisteme çok detay verilmez. Veriyle ilgili bir karar verildiğinde bu kararın doğruluğuna göre ödüllendirme ya da cezalandırma yapılır. Dolayısıyla sisteme, hedeflenen sonuca ulaşabilmek için nasıl doğru kararlar almayı öğrenebileceğini gösterir. Birçok probleme uygulanabilir olmakla birlikte temel ya da doğrudan sorunların çözülmesi için kullanılması uygun görülmez [19]. İstenen örneklerin yer almadığı fakat çeşitli kriterlere göre örneklerin puanlamasının yapılabilirdiği durumlarda tercih edilmektedir.

1.3. Yapay Sinir Ağları (YSA)

YSA, insan beynini model olarak alarak öğrenme süreçlerinin matematiksel modellenmesi ile ortaya çıkmış olan ve kesin olmayan ya da karmaşık veriler içerisinde bir anlam çıkarmaya, yönelimleri tespit etmeye yarayan bilgisayar sistemleridir [20]. Bu sistemler, elde ettikleri bilgiler doğrultusunda daha önce görülmemiş ya da benzer bilgiler için çıkarım yapabilme yeteneğine sahiptir. Problemlerin çözümünde daha önce ayarlanmış çeşitli komut dizilerini izlerler. Geleneksel bilgisayarlar problemlerin tanımlanması ve çözülmesiyle sınırlıdır fakat yapay sinir ağları varlıklarıyla ilgili bilgiler içerisindeki daha önce bilinmeyen örüntülerin tespit edilmesini sağlar [21]. YSA'da bilgilerin saklanması ağ üzerinde olup klasik programlama metodlarından farklı olarak işlenmektedir. YSA eğitimi başarılı bir şekilde gerçekleştirilir ve test edilirse daha iyi çıkarımlar yapılabilirmekte fakat eksik bilgiler ile çalışma gerçekleştirilirse verimli bir istenilen sonuca ulaşamayabilir [22].

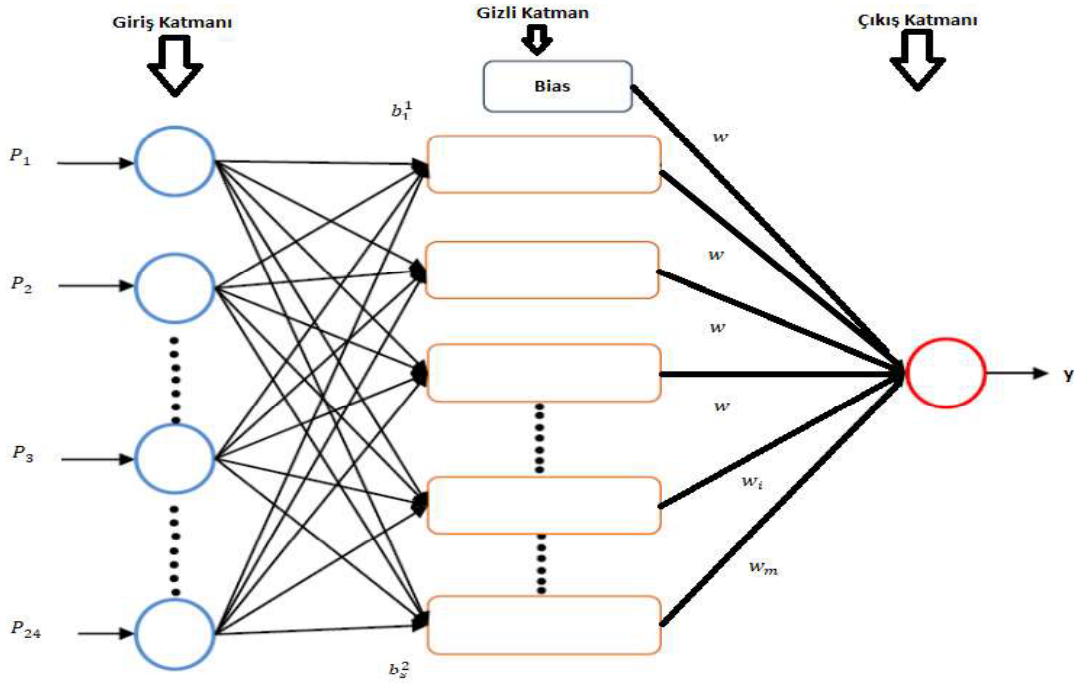
YSA için yapılan mühendislik alanındaki ilk çalışma McCulloch ve Pitts [23] tarafından 1943 yılında, mantıksal olarak sinir aktivitelerinin hesaplanması üzerine yapılmış olup ilk YSA hücresi oluşturulmuştur. 1980’li yıllara kadar YSA ile ilgili çalışmalarda bir durağanlık yaşanmış olsa da daha sonra birçok çalışma yapılmaya başlanmıştır [24]. YSA ile ilgili çalışmalar her geçen gün artmış ve günümüzde yalnızca laboratuvar ortamında kalmayarak hayatın hemen her noktasında kendini gösteren bir yapıya bürünmüştür. Bankacılık, askeri, tıp, mühendislik vb. birçok alanda örüntü tanıma, sınıflandırma, kümeleme gibi birçok işlemde kullanılmaktadır [25]. YSA’nın çalışma mantığının anlaşılabilmesi için biyolojik sinir ağı sistemi yapısının ve biyolojik nöron yapısının incelenmesi gerekmektedir. Bir nöron, elektrik sinyalleri aracılığıyla bilgileri işleyip ileten ve uyarılabilen bir hücredir. Nöronlar bir araya gelerek birbirleri arasında bağlantı sağlayarak sinir ağlarını oluşturur. Nöronlar; Sinaps, Akson, Soma ve Dendrit olmak üzere dört ana bölümden oluşmaktadır. Dendritler, hücre merkezine elektrik sinyallerinin iletilmesinde iletişim kanalı görevi üstlenmektedir. Sinapslar, aksonların diğer dendritler ile olan bağlantısını sağlamaktadır. Bir nöronun şematik çizimi Şekil 1.1’de yer almaktadır.



Şekil 1.1. Nöron yapısının şematik gösterimi [26]

Bir sinir ağının yapısı da bu nöronların bir araya gelmesiyle oluşmaktadır. Yetişkin bir insan beyni ortalama yüz milyar nöron içermektedir ve her nöron yaklaşık olarak on bin nöron ile bağlantı sağlar [27]. YSA’lar nöronların modellendiği yapay nöron ağlarıdır. Temel olarak YSA’ların makine öğrenmesi alanında bu kadar güçlü olması bu yapıya dayanmaktadır. YSA’lar üç katmandan meydana gelmektedir.

Bunlar; giriş katmanı, gizli katman ve çıktı katmanlarıdır. Şekil 1.2.'de YSA'nın mimarisi gösterilmektedir.



Şekil 1.2. Yapay sinir ağı mimarisi [28]

YSA ile biyolojik sinir ağı arasındaki benzerlik Çizelge 1.1'de gösterilmiştir.

Çizelge 1.1. YSA ile biyolojik sinir ağı arasındaki benzerlikler

Biyolojik Sinir Ağı	YSA
Dendrit	Giriş
Hücre Gövdesi	Nöron
Sinaps	Ağırlık
Sinir Ucu	Çıktı

YSA'nın sahip olduğu çeşitli avantaj ve dezavantajlar bulunmaktadır. Kullanım kolaylığı, çözümü zor problemlerde kullanılabilirliği, bilgilerin ağ üzerinde tutulması, yeni durumlara kolay adapte olması ve doğrusal olmayan süreçlerde kullanılabilirliği başlıca avantajları olarak ifade edilebilmektedir. Bununla birlikte donanım bağımlı olması, eğitim ihtiyacının olması, öğrenme sürecinin yavaşlığı ve ağ yapısının deneme yanılma yoluyla belirlenmesi YSA'nın başlıca dezavantajları olarak görülmektedir.

1.3.1. Yapay Sinir Ağlarının Sınıflandırılması

YSA ile ilgili olarak literatür incelendiğinde birçok çalışmanın olduğu görülmektedir. Bu durum ise YSA'nın istenilen amaçlar doğrultusunda çeşitli şekillerde sınıflandırılmasına sebep olmuştur. Bu bölümde YSA sınıflandırılması hakkında detaylı bilgiler verilmiştir.

1.3.1.1.Öğrenme Metoduna Göre YSA

YSA, öğrenme aşamasında başvurulacak öğrenme metoduna göre üç ana bölümde sınıflandırılmaktadır.

- Takviyeli Öğrenme metodunda, ağa girdiler verilir fakat çıktılar sunulmaz. Elde edilen çıktı değerinin iyi ya da kötü olduğuna yönelik bir yardımcı tarafından yönlendirmeler yapılır. Sistem ceza/ödül yapısına göre çalışmaktadır. Yardımcının sistem çıktısına yönelik verdiği iyi cevaplar ne kadar fazla ise sistem bunu ödül olarak nitelendirmekte ve cezadan uzaklaşmayı öğrenmeye çalışmaktadır [29]. Bu ağlara örnek olarak eğitimde Genetik Algoritma ve Boltzman Makinesi gösterilebilir.
- Danışmansız Öğrenme metodunda, ağa sadece girdiler verilmektedir. Ağ, öğrenmeyi herhangi bir geri bildirim olmadan gerçekleştirmektedir. Özellikle boyut indirgeme, kümeleme gibi alanlarda kullanılan bu öğrenmeye SOM (Self Organizing Map) ve ART (Adaptive Resonance Theory) ağları örnek olarak gösterilebilir.
- Danışmanlı Öğrenme metodunda istenilen çıktı ve girdi verileri eşleştirilmiş bir şekilde ağa verilmektedir. Bu metotta amaç istenilen çıktı ile tahmin edilen çıktı arasında hata oranını minimize ederek eğitim ve öğrenme aşamaları gerçekleştikten sonra gelebilecek yeni girdilere karşı istenilen çıktı üretebilen bir sistemin tasarlanmasıdır [30].

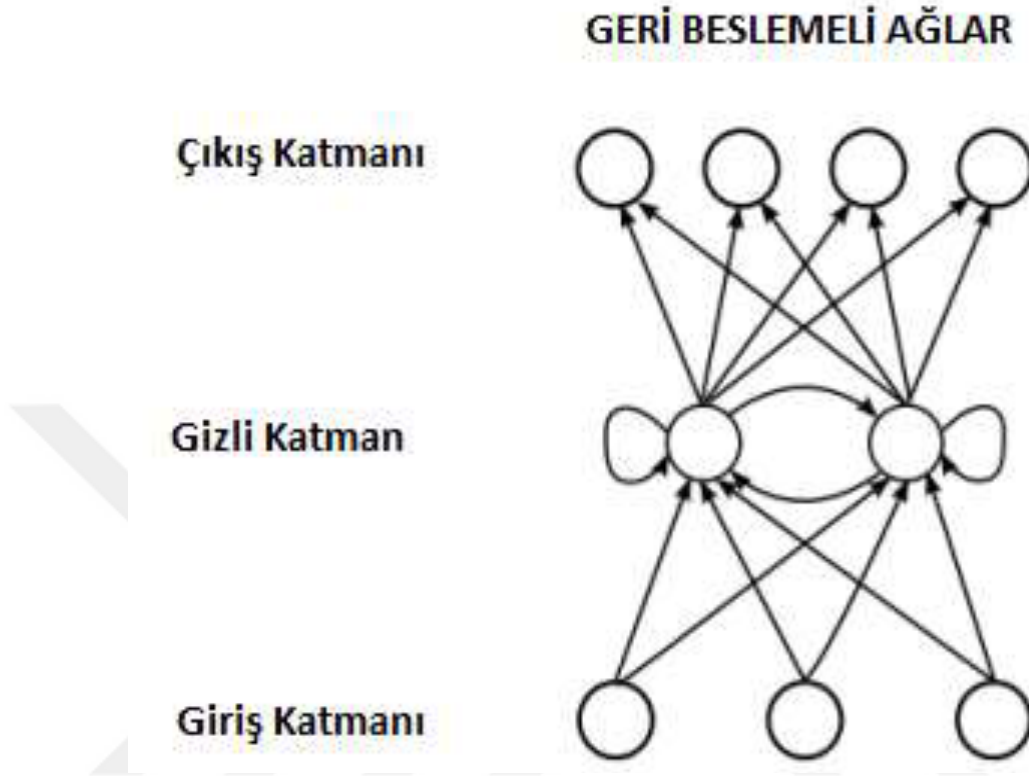
1.3.1.2.Yapılarına Göre YSA

YSA'lar yapılarına göre Geri Beslemeli (Feedback) ve İleri Beslemeli (FeedForward) olmak üzere iki şekilde sınıflandırılmaktadır.

- **Geri Beslemeli Ağlar**

Bu ağlarda, çıktı ya da ara katmanda bulunan nöronlar çıktılarını önceki ara katmanda yer alan nörona ya da girişe girdi olarak tekrar iletmektedirler.

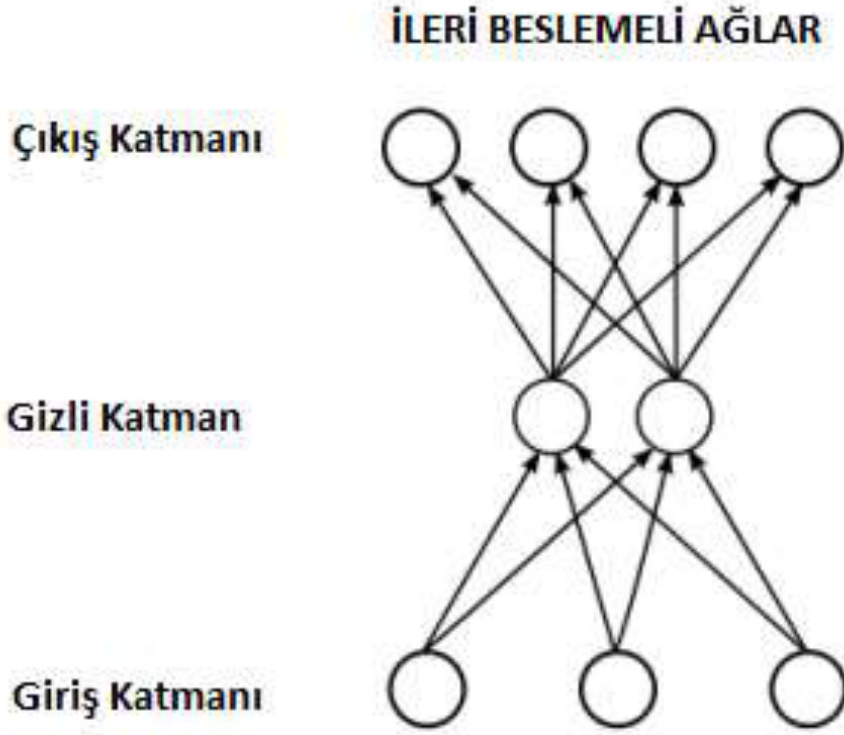
Bu şekilde bilginin iki yönlü aktarımı sağlanır. Dinamik bir hafızaya sahip bu ağlarda, elde edilen çıktı hem önceki girdileri hem de o anki girdileri göstermektedir [31]. Geri beslemeli ağların genel yapısı Şekil 1.3'te gösterilmektedir.



Şekil 1.3. Geri beslemeli ağların genel yapısı [31]

- **İleri Beslemeli Ağlar**

Bu ağlarda nöronlar arası iletişim tek yönlü olup giriş katmanından çıkış katmanına doğru gerçekleştirilir. Aynı katman nöronları arası bağlantı bulunmaz. Bununla birlikte nöronlar bir katmandan diğer katmana doğru bağlantı sağlamaktadır. Bu sebeple nöronlar arası bağlantılarda döngü oluşmamakta ve girdi verilerine yönelik hızlı bir şekilde çıktı sonucu üretilebilmektedir. Bu ağlara örnek olarak Learning Vector Quantization (LVQ) ve Multi Layer Perceptron (MLP) verilebilir. İleri beslemeli ağların genel yapısı Şekil 1.4'te yer almaktadır.



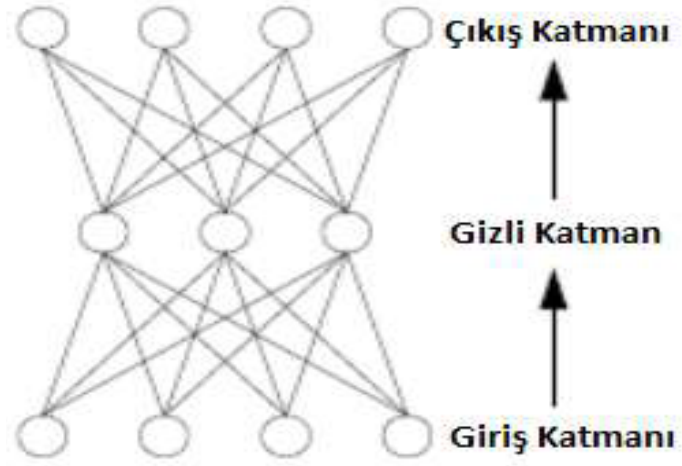
Şekil 1.4. İleri beslemeli ağların genel yapısı [31]

1.3.1.3.Katman Sayılarına Göre YSA

YSA, katman sayılarına göre çok katmanlı ve tek katmanlı olmak üzere iki sınıfta incelenmektedir.

- **Çok Katmanlı YSA**

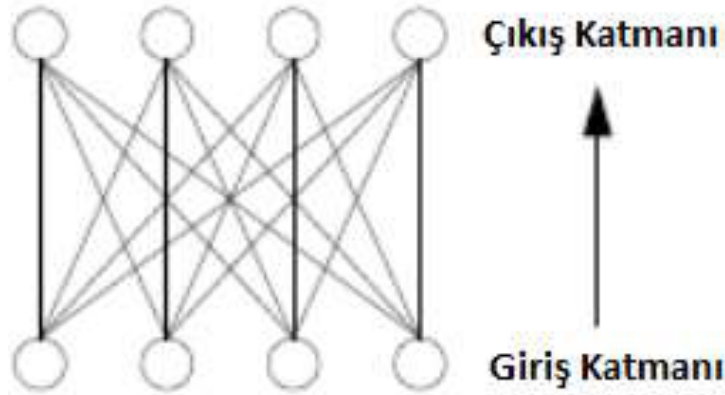
YSA yapıları genel olarak girdi, gizli ve çıktı katmanları olmak üzere en az üç katmandan oluşmaktadır. Bu katmanlardan girdi katmanı, dışardan elde edilen verinin bu katman üzerinde yer alan nöronlara iletilerek bias veya ağırlıkların da dâhil edilip sonraki katmana iletimini sağlamaktadır. Girdi katmanından gelen veriler, birden fazla da olabilen gizli katmandaki nöronlara iletilip bias veya ağırlıklar da dâhil edilerek sonraki gizli katmana ya da çıktı katmanına iletilir. Katmanlarda farklı sayılarda nöronlar olabilmektedir. Nöron sayısının artışı ya da gizli katmanın sayısı, analiz için gerekli süre artışına neden olabilmektedir. Bununla birlikte bu durum çözümü zor problemlerin çözümünde kullanılabilmesine imkân tanımaktadır. Çıktı katmanında ise önceki katmanların iletmış olduğu veriler işlenir ve çıktı olarak dışarıya verilir. YSA yapısında son katman olarak çıktı katmanı yer almaktadır [22]. Şekil 1.5'te çok katmanlı YSA genel yapısı gösterilmektedir.



Şekil 1.5. Çok katmanlı YSA yapısı [32]

- **Tek Katmanlı YSA**

Bu ağlarda yapı yalnızca girdi ve çıktı katmanlarından oluşur. Veriler arada başka bir katman bulunmadan girdi katmanından çıktı katmanına iletilir [22]. Şekil 1.6’da tek katmanlı YSA genel yapısı gösterilmektedir.



Şekil 1.6. Tek katmanlı YSA yapısı [32]

1.3.1.4. Kullanım Amacına Göre YSA

Araştırmacılar YSA’yı kullanım amacına göre Çizelge 4.x’de gösterildiği şekilde sınıflandırmışlardır. Kullanım alanları ve amaca göre hangi topolojilerin seçildiği 8 başlık altında incelenmiştir. Çizelge 1.2’de genel kullanım amaçlarına göre YSA gösterilmiştir.

Çizelge 1.2. Kullanım amaçlarına göre YSA topolojileri [33]

Fonksiyon Yaklaştırma	RBF Geri Yayılım
Veri İlişkilendirme	Boltzmann Makinesi Hopfield Ağları Hamming Ağları Çift Yönlü İlişkili Hafıza ART Kohonen's SOM Uzaysal-Mekansal Örüntü Tanıma
Öngörü-Tahmin	Elman Ağı Geri Yayılım Jordan Ağı Geri Yayılım İçinde SOM Yönlendirilmiş Rastsal Tarama Radyal Tabanlı Fonksiyon
Veri Filtreleme	Yeniden Dolaşım
Kontrol	LVQ Geri Yayılım RBF
Optimizasyon	LVQ RBF Geri Yayılım
Örüntü Sınıflandırma	ART Boltzmann Makinesi Olasılık Tabanlı Sinir Ağları LVQ Tek veya Çok Katmanlı Algılayıcı RBF
Kümeleme/Kavramlaştırma	SOM LVQ

Çizelge 4.2’de kullanım amaçlarına göre literatürde en çok kullanılan ağ topolojilerini göstermektedir. Bu ağlardan bazılarının farklı birçok problemin çözümünde kullanılabildiği görülmektedir.

- **Fonksiyon Yaklaştırma**

Bu işlem eldeki veri çiftlerinin işlenip bilinmeyen fonksiyonun tahmin edilmesini amaç edinen bir tekniktir. Mühendisliğin birçok alanında problemlerin modellenmesi ve karşılaşılan fonksiyonların tahmin edilmesinde kullanılmaktadır.

- **Veri İlişkilendirme**

Bu amaçla kullanılan YSA'da öğrenilen bilgiler ile eksik bilgilerin tamamlanması sağlanır. Farklı açılardan fotoğrafı çekilmiş bir suçlunun teşhis edilmesi ya da eksik olan bir resmin tamamlanması gibi çalışmalar örnek olarak verilebilir.

- **Öngörü-Tahmin**

Bu amaçla kullanılan YSA'da girdi değerlerinden çıktı değerinin tahmin edilmesi hedeflenir. Döviz kurunun tahmin edilmesi bu yapıya örnek olarak verilebilir.

- **Veri Filtreleme**

Bu amaç doğrultusunda eğitilen YSA'da birçok veri içerisinde uygun olanların belirlenmesi hedeflenir. İlk YSA yapılarından birisi Madaline da bu kategoriye girmektedir.

- **Kontrol**

Dinamik kontrolün ihtiyaç duyulduğu uygulamalar, kontrol verisi ile çıktı arasında var olan ilişkinin eş zamanlı incelenerek gerekli tepkilerin anında verilebilmesi amaç edinerek tasarlanmaktadır. Referans alınan modele uygun olarak amaçlanan çıktının üretilmesi için kontrol verilerinin değiştirilmesi esasına dayalı olarak çalışırlar. Bir makine tarafından üretilen ürünler için kalitelerin anlık gözlemlenmesini sağlayarak gerekli makine ayarlarının yapıldığı sistemler örnektir.

- **Optimizasyon**

Mühendislik, matematik, tıp vb. birçok alanda optimizasyon problemleriyle karşılaşilmektedir. Belirli şartlar altında belirli bir fonksiyonun minimum veya maksimum noktalarının tespiti ya da optimal çözümlerin elde edilmesi amaçlanmaktadır.

- **Örüntü Sınıflandırma**

YSA'da yaygın olarak kullanılan alanlardan biri de örüntü sınıflandırma. Kamera görüntüsü, resim, el yazısı, ses dalgaları vb. desen içeren girdi değerlerinin daha önce belirlenmiş desenlerle tanımlanması amaç edinir. Tıbbi görüntüleme cihazlarının üretmiş olduğu örüntülerin incelenerek hastalıkların teşhis edilebilmesi örnektir.

- **Kümeleme/Kavramlaştırma**

Veri setindeki benzer örüntüler incelenir ve benzerlik oranına göre veri setinin en optimal şekilde kümelenebilir. Eğitim için sınıflandırılmış ya da etiketlenmiş veri setine gereksinim duyulmaz. Veri sıkıştırma, veri madenciliği gibi konularda sıklıkla kullanılmaktadır.

1.4. Meta Sezgisel Algoritmalar

İlk olarak Glover tarafından 1986 yılında ortaya atılan meta sezgisel terimi, Yunanca “üst seviye, daha ilerisi” anlamına gelen “meta” kelimesiyle İngilizce sezgisel anlamına gelen “heuristic” kelimelerinin birleşiminden oluşmaktadır. Meta sezgisel yaklaşımlar, arama uzayında stokastik fakat bilinçli arama yapan metotlardan oluşmaktadır [34]. Bu metotlar her bir iterasyonda çözüm kümesi üzerinden yola çıkar ve yeni çözümlerin üretilmesini sağlar. Bu şekilde yerel en iyi sonuçlardan kaçınılıp çözüme en yakın olan sonuçlar elde edilir.

Meta sezgisel algoritmalar incelendiğinde genel olarak doğadan ilham alınarak geliştirildiği görülmektedir. Karmaşık problemler için kısa süre içerisinde uygun ve makul bir çözüme ulaşabilmek için oldukça önemli bir yöntemdir. Küresel optimuma kısa süre içerisinde ulaşma ve geniş arama uzayı en önemli avantajları olarak gösterilebilir [35]. Çözüm kümesini içeren arama uzayının araştırılması için bu algoritmaların keşif ve sömürü yeteneklerinden faydalanılır. Keşif, arama algoritmalarının yerel optimuma takılmaması amacıyla alternatif çözümlerin arama uzayında keşfedilmeye çalışıldığı aşamadır. Keşif sürecinden elde edilen uygun çözümlerin detaylı olarak incelenerek en iyi çözümün elde edilme süreci ise sömürü sürecini oluşturur [36].

1.4.1. Meta Sezgisel Algoritmaların Özellikleri

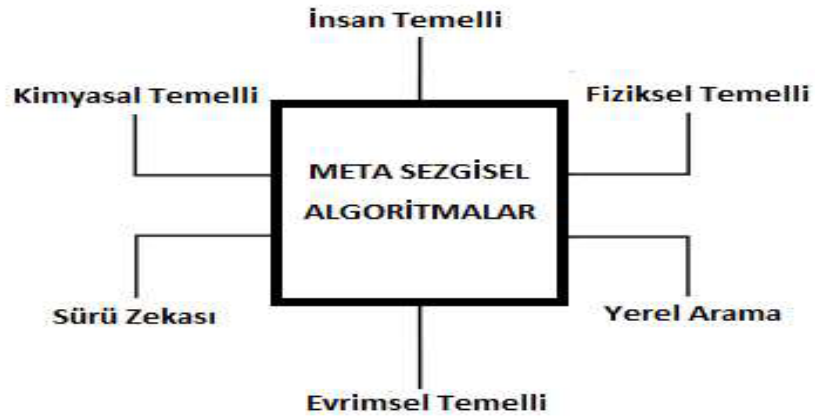
Literatür incelendiğinde yapılan araştırmalarda meta sezgisel algoritmalar için temel özellikler aşağıdaki gibi belirtilmiştir.

- Çözüm uzayında en iyi çözüme hızlıca ulaşır.
- Yerel iyi çözümlere takılmazlar.
- Problem türüne bağılılığı bulunmamaktadır.
- Amaç fonksiyonu kullanılmaktadır.

- Temel amaç, en yakın ya da en uygun çözümün bulunması yerine arama uzayındaki en iyi çözümün verimli ve etkili bir şekilde bulunmasıdır.
- Çözümler, arama süreci boyunca hafızada tutulur.
- Arama için kullanılacak çözüm sayısı başlangıçta ayarlanır.
- Arama uzayında yer alan yerel tuzaklardan kaçınmak için çeşitli teknikler kullanılır.
- Çözüm, arama süreci içerisinde elde edilen bilgileri rehber olarak kullanmaktadır [37].

1.4.2. Meta Sezgisel Algoritmaların Sınıflandırılması

Meta sezgisel algoritmalar temel olarak altı kategori içerisinde incelenmektedir. Bu kategoriler aşağıda yer alan Şekil 1.7’de gösterilmektedir.



Şekil 1.7. Meta sezgisel algoritmaların kategorilendirilmesi [37]

- İnsan temelli algoritmalarda; insanların sosyal davranış, zeka ve aktiviteleri ilham alınmıştır. İnsan temelli olarak değerlendirilen algoritmalara FGIA, FWA ve HSA algoritmaları örnek olarak verilebilir.
- Fiziksel temelli algoritmalarda; fizik yasalarına göre gelişen olaylardan ilham alınmıştır. Bu algoritmalar için CFO, MVO ve GSA algoritmaları örnek olarak verilebilir.
- Yerel arama algoritmaları; başlangıç aşamasında bir çözüm ile başlanan ve en iyi çözüm ya da belirtilen iterasyon değeri elde edilinceye kadar bulunan çözümün komşu çözümlere taşınıp farklı çözümler elde edilmesini sağlayan algoritmalar. Bu algoritmalara VNS, SA ve ILS algoritmaları örnek olarak

verilebilir.

- Evrimsel temelli algoritmalar; genel olarak popülasyon temelli olan ve doğadaki değişimin algoritmalara uyarlanması şeklinde ifade edilen algoritmalar. Başlangıçta rastgele bir popülasyon oluşturulur ve her iterasyonda en iyi bireylerin bir araya getirilmesi amaçlanır. Bu tür algoritmalar DE, GA ve ES algoritmaları örnek olarak verilebilir.
- Sürü zekasına dayalı algoritmalar; genel olarak biyolojik olaylardan ilham alınan, belirli bir grup canlı türünün yaşam mücadelesi içerisinde hedeflerini elde etmek için gerçekleştirdikleri davranışların matematiksel olarak modellenip gerçek yaşam problemlerinin çözülmesi için geliştirilmiş algoritmalar. Bu tür algoritmalar; GWO, WOA ve SSA algoritmaları örnek olarak verilebilir.
- Kimyasal temelli algoritmalar; genel olarak kimya biliminden esinlenen ve maddelerin kimyasal tepkimelere girmesi ve sonucunda oluşan yeni maddenin oluşması sürecini temel alan algoritmalar. Bu tür algoritmalar CRO, ACP ve GBMO algoritmaları örnek olarak verilebilir.

2. SİBER SALDIRILAR ve TESPİT SİSTEMLERİ

2.1. Siber Saldırıların Evrimi

Siber saldırılar temel olarak 1970’li yıllarda başlamıştır. Bilgisayar ve bilgisayar sistemleri henüz yaygınlaşmamışken Bob Thomas tarafından 1972 yılında “Creep” adında bir bilgisayar programı geliştirilmiştir. Bu program kendini klonlayabilmesi sebebiyle tarihteki ilk bilgisayar solucanı olarak bilinmektedir. Geliştirilen bu uygulamanın tespit edilmesi ve var olan programın silinebilmesi için ise Ray Tomlinson tarafından “Reaper” adında bir program geliştirilmiştir.. Şekil 2.1’ de Creep programının ekran görüntüsü gösterilmiştir.

```
BBN-TENEX 1.25, BBN EXEC 1.30
@FULL
@LOGIN RT
JOB 3 ON TTY12 08-APR-72
YOU HAVE A MESSAGE
@SYSTAT
UP 85:33:19 3 JOBS
LOAD AV 3.87 2.95 2.14
JOB TTY USER SUBSYS
1 DET SYSTEM NETSER
2 DET SYSTEM TIPSER
3 12 RT EXEC
@
I'M THE CREEPER : CATCH ME IF YOU CAN
```

Şekil 2.1. Creep programı

1980’li yıllarda ise ARPANET’ten internete geçiş süreciyle birlikte saldırılar da yeni bir boyut kazanarak daha karmaşık bir yapıya bürünmüştür. 1988 yılında Robert Morris tarafından geliştirilen “Morris Worm” internet üzerinden yayılan en eski bilgisayar soluncanlarından biri olmaktadır. Bu solucan; tüm ülkenin askeri, kamu, üniversite gibi kurumlarındaki bilgisayarlara bulaşp bilgisayarların kullanılamaz hale gelmesine sebep olmuştur [38].

Devam eden süreçte ise 1990-2000 yılları arasında internetin yaygınlaşması ve depolanan veri miktarının artması saldırganların daha çok saldırı gerçekleştirerek sistemlere zarar vermelerine sebep olmuştur. Artan bu saldırılar siber güvenlik konusuna daha fazla önem verilmesini sağlamıştır.

Bir NASA araştırmacısı tarafından ilk güvenlik duvarı geliştirilmiş ve takip eden dönemler içerisinde çeşitli antivirüs programları da geliştirilmiştir. Bununla birlikte yeni saldırı teknik ve yöntemleri de geliştirilmiştir. SophosLabs tarafından kötü amaçlı yazılım ve spam analizi konusunda hazırlanan teknik raporda 20 yıllık yakın tarih 3 ana döneme ayrılıp incelenmiştir [39].

- Worm Dönemi (2000-2004): Bu dönemde gerçekleştirilen saldırılar genel olarak kişisel merak sebebiyle ortaya çıkan ve sistemlerde bozulmaya sebep olabilen saldırılar olmaktadır. Bu saldırılara CodeRed, Welchia, Blaster, ILOVEYOU, MyDoom, Nimda, Sasser, Netsky, Slammer, Sobig ve Sober saldırıları örnek olarak verilebilir.
- Malware Dönemi (2005-2012) : Bu dönem içerisinde gerçekleştirilen saldırılar genel olarak kâr elde etme amacıyla yapılan saldırılardan oluşmaktadır. Bu saldırılara Rx Spam, Stuxnet, Blackhole exploit kit, Zeus, Storm, Conficker gibi saldırılar örnek olarak verilebilir.
- Ransomware Dönemi (2013-Günümüz): Bu dönemde solucanlar, truva atları, kötü amaçlı reklamlar ve spamlar gibi çeşitli güvenlik olayları meydana gelmiştir. Ransomware saldırıları bu dönem içerisinde daha fazla öne çıkmış ve sistemlere verdikleri zararlar diğer saldırılara göre daha fazla olmuştur. Bu tür saldırılarla sistemlere çok ciddi zararlar verilmiştir. Bu saldırılara Point-of-sale(POS) malware, Cryptolocker, Extortion ransomware, NotPetya, Mirai, Snowden leaks, WannaCry, Magecart attacks gibi saldırılar örnek olarak verilebilir.

Birçok güvenlik raporunda da siber saldırılar ve bu saldırılara karşı geliştirilecek STS'lerin önemine vurgu yapılmıştır.

2.2. Saldırı Tespit Sistemleri

Saldırı, bir bilgi sistemine zarar veren her türlü yetkisiz faaliyet olarak tanımlanabilir. Bu; bilgilerin gizliliği, bütünlüğü veya kullanılabilirliği için olası bir

tehdit oluşturabilecek herhangi bir saldırının izinsiz giriş olarak değerlendirileceği anlamına gelir. Örneğin, bilgisayar hizmetlerini meşru kullanıcılara yanıt veremeyecek hâle getirecek faaliyetler izinsiz giriş olarak kabul edilir. STS ise, bilgisayar ya da ağ sistemlerine yönelik şüpheli veya istenmeyen davranışların tespit edilmesini sağlayan sistemlerdir [40].

Bir STS'nin amacı, bir güvenlik duvarı tarafından tespit edilemeyen çeşitli türlerde kötü niyetli ağ trafiğinin ve bilgisayar kullanımının belirlenmesidir. STS; bilgisayar sistemlerinin gizliliğini, bütünlüğünü veya kullanılabilirliğini tehlikeye atan davranışlara yönelik yüksek savunma sağlaması sebebiyle hayati önem taşımaktadır. Özellikle günümüzde bilgisayar ve ağ sistemlerini oluşabilecek ihlallere karşı güvende tutmak, bu saldırıların önüne geçebilmek oldukça zor hâle gelmiştir. STS'ler, sistemleri otomatik olarak izleyip tehdit olarak değerlendirilen durumların tespit edilmesini sağlarlar. Ağ kaynaklarından ya da elde edilen verilerden olası tehditler için analizler yaparak tehditler ile ilgili bilgiler sunmakta ve ağda gerçekleştirilen tüm olayların depolanmasını sağlamaktadır. Sunulan bilgiler doğrultusunda sistemlerin güvenliğini artırabilecek çeşitli önlemlerin de alınabilmesine imkân tanımaktadır [41].

2.3. Saldırı Tespit Sistemi (STS) Modelleri

STS'ler tespit yaklaşımlarına göre iki sınıfta incelenmektedir. Bu sınıflar, imza ve anomali tabanlı yaklaşım modellerinden oluşmaktadır.

2.3.1. İmza Tabanlı Saldırı Tespit Sistemleri

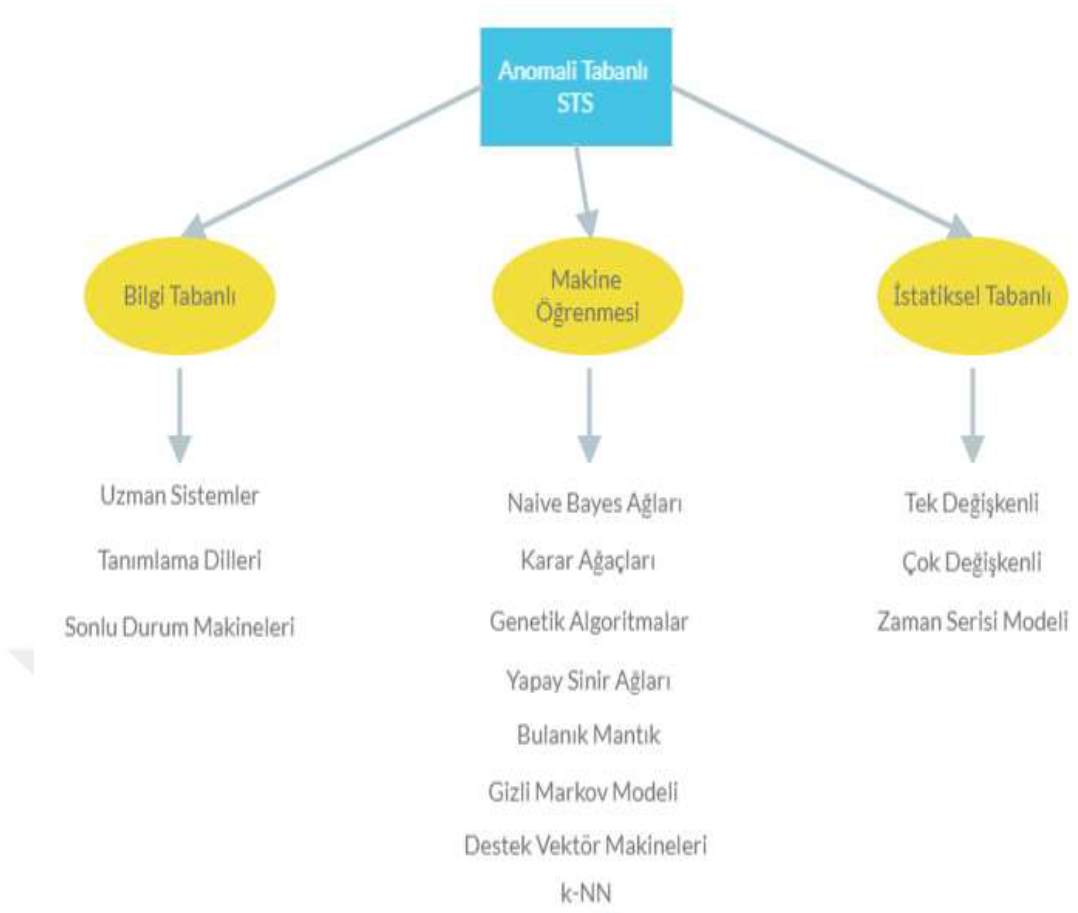
İmza tabanlı STS'ler, bilinen bir saldırıyı bulmak için çoğunlukla desen eşleştirme tekniklerine göre çalışır. Bu teknikler yanlış kullanım tespiti veya bilgiye dayalı tespit olarak adlandırılabilir. Bu yaklaşımda, tehdit olarak nitelendirilebilecek davranışların geçmişteki faaliyetlerine göre öğrenme gerçekleşmektedir. Öğrenilen bu model, gelecekte karşılaşılabilecek benzer saldırıların tahmin ve tespit edilmesinde kullanılmaktadır. Ağda yer alan bir davranışın imzası, saldırı imzasının yer aldığı veri tabanındaki imza ile örtüştüğünde sistem bunu tehdit olarak görüp uyarı vermektedir. Yapılan herhangi bir saldırı durumunda bir saldırı tespiti gerçekleşmişse mevcut imzaların yer aldığı veri tabanı bu yeni imza ile güncellenmektedir [42]. Bu sistemlerin performanslarının yeterliliği veri tabanında yer alan imzaların tam ve yeterli olmasıyla ilişkilidir. Bu yaklaşımda, daha önceden sisteme tanımlı bir saldırının basit bir

değişiklikle STS tarafından algılanmamasına sebep olabilmektedir. Bu durum önemli bir sorun olarak karşımıza çıkmaktadır. Bu eksikliğin önüne geçebilmek ve algılama doğruluğunun artırılabilmesi için imza veri tabanlarının düzenli olarak güncellenmesi büyük önem taşımaktadır [43]. Bunun yanında bu sistemin bazı önemli avantajları da bulunmaktadır.

Bu sistemlerin saldırı tespit etmede yanlış alarm oranı oldukça düşüktür. Bilinen bir saldırı türü için güvenilir ve hızlı bir şekilde tespit gerçekleştirilebilmektedir.

2.3.2. Anomali Tabanlı Saldırı Tespit Sistemleri

Anomali tabanlı saldırı tespit sistemleri, imza tabanlı sistemlerin sahip olduğu eksikliklerin üstesinden gelme kapasitesi nedeniyle birçok bilim insanının ilgisini çekmiştir. Anormallik tespiti modelinde, sistemdeki ihlal ya da tehdit olarak değerlendirilebilecek davranışların tespit edilmesi hedeflenmektedir. Bu sistemlerde, tehditlerin normal davranışlardan farklı olduğu üzerine bir yaklaşım bulunmaktadır. Normal davranışlar için geçmiş verilerden faydalanılarak bir profil oluşturulup ne zaman bu davranışlardan sapma gerçekleştiği belirlenmektedir [44]. Gözlemlenen model ve davranış arasında oluşan önemli bir sapma, izinsiz giriş olarak nitelendirilebilecek bir anormallik olarak kabul edilmektedir. Bilgisayar sistemleri için davranış modelleri ise makine öğrenimi, istatistiksel veya bilgi tabanlı yöntemler kullanılarak oluşturulur. Anomali tabanlı saldırı tespit sistemlerinin geliştirilmesi iki aşamadan oluşur: eğitim aşaması ve test aşaması. Eğitim aşamasında, normal davranış modelini öğrenmek için normal trafik profili kullanılır ve ardından test aşamasında, sistemin daha önce görülmemiş izinsiz girişlere genelleme kapasitesini oluşturmak için yeni bir veri seti kullanılır. Anomali tabanlı saldırı tespit sistemleri, eğitim için kullanılan yöntemle dayalı olarak istatistiksel temelli, bilgiye dayalı ve makine öğrenimi tabanlı olmak üzere bir dizi kategoride sınıflandırılabilir [45]. Şekil 2.2’de anomali tabanlı STS’lerin kullandığı temel metotların sınıflandırılması gösterilmektedir.



Şekil 2.2. Anomali tabanlı STS’lerin kullandığı temel metotların sınıflandırılması

Bu sistemlerin imza tabanlı sistemlerden en büyük avantajı yeni ve daha önce gerçekleşmemiş bir saldırıyı tespit edebilmesidir. Anomali tabanlı saldırı tespit sisteminin ana avantajı ise, anormal kullanıcı etkinliğinin tanınmasının bir imza veri tabanına dayanmadığı gerçeğinden dolayı sıfır gün saldırılarını tanımlama yeteneğidir [46]. Anomali tabanlı STS, incelenen davranış olağan davranıştan farklı olduğunda bir tehlike sinyalini tetikler. Ayrıca, Anomali tabanlı STS’nin farklı avantajları bulunmaktadır. İlk olarak, dahili kötü niyetli etkinlikleri keşfetme yeteneğine sahiptirler. Bir davetsiz misafir, tipik kullanıcı etkinliğinde tanımlanamayan çalıntı bir hesapta işlem yapmaya başlarsa sistem bir alarm oluşturur. İkincisi, saldırganlar, sistem özelleştirilmiş profillerden oluşturulduğu için herhangi bir uyarı üretmeden normal bir kullanıcı davranışının ne olduğunu anlayabilir. Bunların yanında tehdit olmayan bir davranışın normal davranış profilinden sapması nedeniyle tehdit olarak görülmesi ya da normal profilde olup tehdit olabilecek davranışların normal olarak değerlendirilmesi gibi hatalı sonuçlar da üretebilmektedir.

Bu gibi sorunları engellemek için ağ trafiğinin hangi özellikler kullanarak izleneceğinin belirlenmesi çok önemlidir [43]. Çizelge 2.1’de saldırı tespit sistemi modellerinin karşılaştırmaları gösterilmektedir.

Çizelge 2.1. STS modelleri avantaj ve dezavantajları

Model	Avantaj	Dezavantaj
İmza Tabanlı STS	Saldırıları, düşük yanlış alarm oranıyla tespit etmede etkilidir.	Yeni saldırı imzaları için sıklıkla güncellenmesine ihtiyaç vardır.
	İzinsiz girişleri hemen tespit eder.	Bilinen saldırıları tespit etmek için tasarlandığı için önceki saldırı tiplerinde olası bir değişiklik hâlinde saldırı tespiti mümkün değildir.
	Bilinen saldırıları tespit etmede oldukça başarılıdır.	Sıfır-gün saldırılarını tespit edemez.
	Tasarımı basittir.	Çok adımlı saldırıları tespit etmeye uygun değildir.
Anomali Tabanlı STS	Yeni saldırı türlerini tespit edebilir.	Şifrelenmiş paketleri işleyemez bu sebeple saldırı tespit edilmeden kalabilir.
	Saldırı imzası oluşturmak için kullanılabilir.	Yanlış pozitif alarm oranı yüksektir.
		Başlangıç eğitimi gerektirir.
		Çok dinamik bilgisayar sistemleri için normal bir profil oluşturmak zordur.

2.4.Saldırı Tespit Sistemi Tipleri

STS’ler izlenen sisteme göre iki farklı tipte incelenmektedir. Bu tipler, ağ ve sunucu tabanlı sistemlerdir. Bu sistemlerden ağ tabanlı sistemler, belirli bir ağ sistemine yerleştirilir ve ağ trafik akışı takip edilir. Sunucu tabanlı saldırı tespit sisteminde ise sistemlerin girdisi, uygulama kayıtları ve sistem günlükleri gibi doğrudan sistem üzerinden elde edilebilen faaliyetler izlenir [47].

2.4.1. Ağ Tabanlı Saldırı Tespit Sistemleri

Bu sistemlerde ağ trafiğindeki şüpheli, kötü niyetli ya da belirli bir örüntü aranmaktadır. Bu sistemler, ağ trafiğinin izlenmesi amacıyla ağda önemi yüksek lokasyonlara yerleştirilir. Bu sistemlerde özel bir donanım gereksinimi bulunmakta ve ağda yer alan paketler kontrol edilmektedir. Bu paketler aynı zamanda veri kaynağı olarak da kullanılmaktadır. Bu sistemlerde saldırı tespiti için çeşitli teknikler kullanılmaktadır [48]:

- Eşik geçişi ya da frekans
- İfade, desen yada bayt kodu eşlemesi
- Protokol kontrolü

Bu sistemlerin en önemli avantajları; işletim sisteminden bağımsız olması, bakım masrafının düşük olması ve tüm ağın izlenebiliyor olmasıdır.

2.4.2. Sunucu Tabanlı Saldırı Tespit Sistemleri

Bu sistemler izlenilen sistemde meydana gelen olaylar hakkında bilgi toplanarak çalışmaktadır. Geliştirilmiş olan ilk saldırı tespit sistemi olan bu türler, ana bilgisayarlara lokal olarak kurulurlar. Saldırıların tespit edilmesi için sıklıkla olay ve günlükler izlenmekte ve olası bir değişiklikte daha önceden kayıt altına alınmış saldırı imzalarıyla karşılaştırma yapılmaktadır. Herhangi bir eşleşme olduğunda sistem yöneticilerine uyarı göndererek tepki vermektedir [49]. Bu saldırı tespit sistemlerinin avantajları olarak, sistem programlarının ve uygulamalarının nasıl kullanıldığı ile ilgili tutarsızlıkları tespit edebilme ve LAN dışındayken sistemlerin korunabilir olması söylenebilir [50]. Bununla birlikte ağ trafiğinin izlenememesi ve pahalı bir sistem olması bu sistemlerin dezavantajı olarak söylenebilmektedir. Çizelge 2.2’de saldırı tespit sistemi tiplerinin karşılaştırmaları gösterilmektedir.

Çizelge 2.2. Saldırı tespit sistemi tiplerine göre avantaj ve dezavantaj karşılaştırması

STS Tipi	Avantaj	Dezavantaj
Ağ Tabanlı STS	Ağ paketlerini kontrol ederek saldırıları tespit eder.	Şifreli trafikten gelen saldırıları belirlemek zordur.
	Her bir ana bilgisayara yüklenmesi gerekmez.	Özel donanım gerekir.
	Aynı zaman periyodunda çeşitli ana bilgisayarları kontrol edebilir.	Yalnızca ağ saldırılarının tanımlanmasını destekler.
	En geniş ağ protokolleri aralığını algılayabilir	Yüksek hızlı ağ analiz etmek zordur.
Sunucu Tabanlı STS	Uçtan uca şifreli iletişim davranışını kontrol edebilir.	İçeriden gelen saldırılar ciddi tehdit oluşturur.
	Ekstra donanım gerekmez.	Saldırıları bildirmede gecikir.
	Ana bilgisayar dosya sistemini, sistem çağrılarını veya ağ olaylarını kontrol ederek saldırıları tespit eder.	Ana bilgisayar kaynaklarını tüketir.
	Her paket yeniden birleştirilir.	Her ana bilgisayara yüklenmesi gerekir.
		Yalnızca kurulu olduğu makinedeki saldırıları izleyebilir.

2.5.Saldırı Tespit Sistemi İçin Kullanılan Araçlar

2.5.1.Haystack

ABD Hava Kuvvetlerinde 1988 yılında kullanılan bu STS, Unisys 1100/60 bilgisayarları için geliştirilmiştir. Bu STS’de 6 farklı saldırı türünün tespiti amaçlanmıştır [51,52].

Bu saldırı türleri:

- Sızma saldırılar
- Hizmet engeli (DoS) saldırıları
- Kötüye kullanım saldırıları
- Kıрма girişimi yapan saldırılar

- Güvenlik kontrol sistemlerine sızma girişimi yapan saldırılar
- Maskelenmiş saldırılar

Temel olarak kullanıcı davranışları, anormal olaylar ve güvenlik olaylarının kısa özetlerini ele alan bu STS, sistem güvenliği sorumlusunun ve yetkili kullanıcıların içeriden müdahalelerinin tespit edilebilmesi için geliştirilmiştir. Kullanıcıların davranışlarını önceden tanımlanmış profile göre analiz eder ve günlük raporlar tutar.

2.5.2.W&S

Anomali tabanlı olarak 1989 yılında Los Alamos Ulusal Laboratuvarlarında geliştirilmiş olan bu STS, temel olarak geçmiş denetleme verilerinin analiz edilmesi ile normal davranış profillerinin oluşturulması ve bu normal davranış profillerinin uzman sistemlere entegre edilerek anormal etkinliklerin tespit edilmesi üzerine tasarlanmıştır [51,53].

2.5.3.Midas

1988 yılında Stanfor Araştırma Enstitüsü (SRI) ve Ulusal Bilgisayar Güvenlik Merkezi (NCSC) Bilgisayar Bilimleri Laboratuvarı iş birliğiyle geliştirilmiştir. NCSC ana bilgisayarı olan Honeywell DPS-8/70 için saldırıların tespit edilmesi amaçlanmıştır [51,53]. Sunucu tabanlı bir saldırı tespit sistemi modelidir.

2.5.4.Ides

Neuman ve Denning'in 1985 yılında yapmış oldukları çalışmalar neticesinde ortaya çıkan ve kullanıcıların geçmiş etkinliklerini denetleyip profil oluşturarak anormal ve normal davranışların tespit edilmesini amaçlamaktadır. Kullanıcı hareketleri süreç içerisinde değişime uğrayacağından profiller belirli aralıklarla güncellenmektedir. Başlangıç olarak tek bir sunucu için tasarlanmasına rağmen sonraki süreçte tekrardan tasarlanarak çoklu sistemler için kullanılabilir bir yapıya dönüşmüştür. Çeşitli versiyonları bulunan Ides son yıllarda Nides olarak tanımlanmaktadır.

2.5.5.Nadir

Los Alamos Ulusal Laboratuvarlarında 1991-1993 yılları arasında geliştirilmiştir. Temel olarak istatistiklere dayanan sunucu tabanlı bir saldırı tespit sistemidir. Çalışılan sistemde kullanıcıların haftalık davranışlarının istatistiklerini

tutarak profil oluşturur. Oluşturulan bu profiller doğrultusunda analizler gerçekleştirilerek raporlar tutar ve kurallara uymayan durumları saldırı olarak nitelendirmektedir [51].

2.5.6. NSM

1990-1994 yıllarında geliştirilmiş olan ve ağ güvenlik görüntüleyicisi anlamına gelen NSM (Network Security Monitoring), ağı dinleyerek ağ kullanımı için bir profil oluşturur ve gerçek zamanlı olarak ağ kullanımını oluşturulan kurallarla karşılaştırır. Karşılaştırma sonucunda oluşturulan profilden sapma gerçekleşmişse davranış anormal olarak belirlenmektedir [51].

2.5.7. Hyperview

1992 yılında geliştirilen ve diğer saldırı tiplerine göre farklı bir yapısı bulunan bir saldırı tespit sistemidir. İki farklı bölümden oluşan yapısında; ilk bölüm davranışların izlenerek sınıflandırılmasını, ikinci bölüm ise birinci bölüm sonucunda elde edilen bilgiler ile eğitilen yapay sinir ağlarını içermektedir [53].

2.5.8. Ustat

Unix sistemlerde kullanmak için 1995 yılında geliştirilmiş bir STS aracıdır. Durum geçiş analizi yöntemine dayanan bir yöntemin kullanıldığı bu sistemde gerçekleşen bir hareket, saldırı için tanımlanmış durum geçişlerini yapmaktaysa saldırı olarak kabul edilmektedir [54].

2.5.9. Dids

Dağıtık mimari sistemler için tasarlanan ve 1992 yılında geliştirilen bu saldırı tespit sistemi, ağın farklı bölgelerinden toplanılan verilerin merkezde analiz edilerek saldırıların tespit edilmesi esasına dayanmaktadır [51,53].

2.5.10. Idiot

1996 yılında Kumar tarafından geliştirilmiş ve saldırı metotlarının eşleştirilmesi ve karakteristiklerinin çıkartılması üzerine tasarlanmış bir saldırı tespit sistemidir [51].

2.5.11. Janus

Wagner tarafından 1996 yılında Berkeley Üniversitesi'nde tez çalışması sonucu geliştirilmiş olan bir saldırı tespit sistemidir [51]. Janus, güvenilmeyen uygulamaları kısıtlı bir çalışma ortamında korumalı alana almak için kullanılan bir güvenlik aracıdır.

2.5.12. Emerald

Dağıtık mimari yapısında olan 1998 yılında geliştirilmiş bir saldırı tespit sistemidir. Ölçeklenebilir bir yapıya sahip bu sistem, saldırı tespit sistemi ve bilgi güvenliği konusunda önemli bir kaynak olarak kullanılmaktadır [53].

2.5.13. Ripper

Veri madenciliğinin kullanıldığı bu sistem 1999 yılında geliştirilmiş ve DARPA değerlendirmesine dâhil edilmiş özgün sistemlerden biri olmaktadır [53].

2.5.14. Bro

Ağ trafiğinin gerçek zamanlı olarak izlenmesi temeline dayanan ve 1998 yılında Vern Paxson tarafından geliştirilen bu sistemde ağ sorunları için destek sağlanmakta ve performans temelli analizler gerçekleştirilmektedir. Bir tür ağ trafik analiz aracı olarak da belirtilen bu sistem Olay Motoru ve Politika Komut Yorumlayıcısı olmak üzere iki ana bileşenden oluşmaktadır. FreeBSD, MacOS, Linux gibi Unix tabanlı işletim sistemleri üzerinde çalışabilmesinin yanında offline ya da gerçek zamanlı olarak analizler gerçekleştirebilmektedir [53].

2.5.15. Snort

Kural tabanlı ve açık kaynak kodlu olan gerçek zamanlı ağ analizi yaparak ağ paketlerini kayıt edebilen bu saldırı tespit sistemi, 1990'lı yıllarda Martin Roesch tarafından geliştirilmiştir. Çözümleme, ön işleme, tespit motoru ve uyarı olmak üzere dört bölümden oluşmaktadır. Anomali tabanlı ve imza tabanlı sistemlerin avantajlarını bünyesinde barındırmaktadır. Çeşitli içerik analizleri yaparak birçok sızma girişiminin ve saldırının tespit edilmesini sağlayarak oluşturduğu uyarılarla kullanıcıların uyarılmasını sağlamaktadır. Saldırı imzalarını belirli periyotlarda güncellediği için farklı tür ve sayıda saldırıların tespit edilebilmesini sağlayabilmektedir [51]. Snort'un üç farklı çalışma modu bulunmaktadır. Bu modlar ağ tabanlı saldırı tespit sistemi modu, ağ trafiği izleme modu ve paket kaydedici modlarıdır.

2.5.16. Selks

Stamus Networks tarafından 2014 yılında geliştirilmiş açık kaynak kodlu bir saldırı tespit sistemidir. Bu saldırı tespit sistemi, Debian işletim sistemi üzerinde Elasticsearch, Kibana, Logstash, Suricata ve Scirius bileşenleri kullanılarak oluşturulmuştur [51].

2.5.17. Ossec

Birçok farklı işletim sistemi üzerinde sıkıntısız olarak çalıştırılabilen bu saldırı tespit sistemi, açık kaynak kodlu ve bilgisayar üzerinde kurulu uygulamaların çalışmalarının kontrolünü gerçekleştiren bir saldırı tespit sistemidir. 2003 yılında Daniel Cid tarafından geliştirilen, sunucu tabanlı bir saldırı tespit sistemi olan Ossec, mevcut servisler, veri paketleri ve kullanılan kaynakları analiz ederek kullanıcıların risklere karşı uyarılmasını sağlamaktadır. Bunların yanında ek olarak Log izleme, rootkit tespiti ve dosya bütünlük kontrolü özelliklerine sahiptir [55].

2.5.18. Firestorm

Oldukça esnek ve yönetilebilir bir yapıya sahip olan Firestorm, ağ üzerindeki şüpheli paketleri tespit edebilmekte ve TCP bağlantılarını izleyebilmektedir. 2004 yılında geliştirilen bu STS, ağ trafiğini kontrol ederek yapılan analiz sonuçlarını kayıt altına alır. Uygulama katmanında çalışan tüm protokollerini çözebilme yeteneğine sahiptir. Bununla birlikte DoS saldırıların tespit edilmesi ve kullanıcının uyarılmasında etkinliği yüksektir [56].

2.5.19. Kfsensor

Kfsensor, 2003 yılında geliştirilen solucan, trojan ve hackerların tespit edilmesini amaçlayan windows tabanlı bir saldırı tespit sistemidir. Saldırıları tespit edebilmek için port taraması yapar ve sızmaya çalışan tehditlerin log kaydını tutmaktadır. Yapılandırma, yükleme işlemleri basit ve özel bir donanım gereksinimi bulunmaz. Kfsensör tarafından üretilen bilgiler güvenlik duvarlarının iyileştirilebilmesi ve imza tabanlı sistemler için yeni imzaların oluşturulmasında kullanılabilmektedir [56].

2.5.20. Suricata

Açık Bilgi Güvenliği Vakfı (OISF) tarafından 2009 yılında geliştirilmiş olan açık kaynak kodlu ve imza tabanlı bir STS aracıdır. Saldırıların tespit edilmesi özelliğinin yanında saldırıları engelleme özelliği de bulunmaktadır [57].

2.5.21. Phpids

Kullanımı kolay, hızlı, güvenilir ve açık kaynak kodlu olan bu saldırı tespit sistemi 2007 yılında geliştirilmiştir. PHP web uygulamaları için kullanılmaktadır. SQL Injection, DoS ve dizin atlama gibi saldırı türlerini tespit edebilmektedir [57].

2.5.22. .Net Ids

Web tabanlı uygulamalara yönelik saldırıların tespit edilmesi için 2007 yılında geliştirilmiş bir araçtır. Esas olarak .Net uygulamalarının korunmasını amaçlayan bu araç, saldırıların tespit edilmesi için çeşitli fonksiyon ve filtreleme kurallarına sahiptir. Phpids'nin .Net framework uygulamaları için geliştirilen bir türüdür [58].

2.5.23. Prelude

Prelude, 2003 yılında geliştirilmiş bir tür bilgi güvenliği olay ve tehdit yönetimi aracıdır. Ağda gerçek zamanlı olarak yapılan saldırıların tespit edilmesini ve tehditlerle ilgili uyarı verilmesini sağlar. Güvenlikle ilgili olarak risk yönetimini ve operatör işlerini basite indirgeyerek operasyonlara destek sağlayacak bütün araçları barındırmaktadır [59].

2.5.24. Aide

İzinsiz giriş algılama ortamı olarak bilinen bu araç aynı zamanda dosya ve dizin bütünlük denetleyicisi olarak uygulanmaktadır. 1999 yılında geliştirilen bu saldırı tespit sistemi, yapılandırma dosyalarında yer alan kurallardan bir veri tabanı oluşturur ve bununla da dosyaların bütünlüklerini denetlemektedir. Bütünlüklerin denetlenmesinde farklı tip özet algoritmalara sahiptir [60].

2.5.25. Samhain

Sunucu tabanlı bir saldırı tespit sistemi olan Samhain, rootkit algılama, dosya bütünlük denetleme, dünlük dosyalarının analizi ve bağlantı noktalarını izleme gibi çeşitli özelliklere sahiptir. Esas olarak tek bir sunucuda bağımsız bir uygulama gibi kullanılabilmesinin yanında merkezi olarak loglama özelliği de bulunmaktadır [61].



3. LİTERATÜR İNCELEMESİ

Çeşitli sınıflandırma yöntemleri kullanılarak ağ saldırı tespitinde veri madenciliği ve makine öğrenimi son yıllarda yaygın olarak kullanılmaktadır. Bunun en önemli nedeni ise saldırı ve türlerini tespit edebilmektir. Başarılı ve etkili bir saldırı tespiti için kullanılan sınıflandırma tekniği büyük önem arz etmektedir. Literatür incelendiğinde bu konuyla ilgili olarak günümüze kadar birçok farklı teknik önerilmiştir. Bu tekniklerden bazıları bu bölümde tartışılmaktadır.

Literatürdeki birçok çalışmada, ağ saldırılarını tespit edebilmek için NSL-KDD, KDD Cup'99 veri setleri ve çeşitli yöntemler kullanılmıştır. NSL-KDD veri setinin kullanıldığı bazı çalışmaları incelediğimizde, Saleh ve arkadaşları [62], çok sınıflı sınıflandırma problemlerini çözmek için bir yaklaşım önermişlerdir. Çalışmalarında aykırı değer reddi için Optimize edilmiş Destek Vektör Makineleri (OSVM), özellik seçimi için Naive Bayes ve sınıflandırma için ise K-En Yakın Komşular (KNN) algoritmalarını kullanmışlardır. Elde ettikleri sonuçlar incelendiğinde %95,77 doğruluk elde ettiklerini belirtmişlerdir. Bir diğer çalışmada A. J. Ghazali ve arkadaşları [63], Chi-merge ayrıklaştırması ile veri ön işleme ve normalleştirme kullanmışlardır. PART, Ridor, Naive Bayes, SimpleCart ve BFTree teknikleri kullanılmış ve veriler on bölüme ayrılmıştır. Bu bölümlerden 9'u eğitim, 1'i ise test için kullanılmıştır. En iyi sonucu %96,7 doğruluk ile PART sınıflandırıcısının elde ettiği belirtilmiştir. Yogita B. Bhavsar ve arkadaşları [64], çalışmalarında NSL-KDD veri setini ve Destek Vektör Makinelerini (SVM) kullanmışlardır. Çapsal Tabanlı Fonksiyon (RBF) kullanılarak eğitim süresini azaltmışlar ve SVM'in dezavantajı olan eğitim süresini daha iyi bir seviyeye getirdiklerini belirtmişlerdir. Singaravelan ve arkadaşları [65], adli teknikleri ve veri madenciliğini birleştiren bir Sistem Çağrısı (SC) seviyesi önermişlerdir. SC modellerine dayalı bu sistemde kullanıcı profilleri korunmuş ve bu şekilde saldırılara karşı korunma amaçlanmıştır. Çalışmada sunulan bu modelin Terim Frekansı - Ters Belge Frekansı (TF-IDF) modeline göre daha yüksek doğruluk ve daha iyi yanıt süresine sahip olduğu tespit edilmiştir. Başka bir çalışmada, S. Revathi ve arkadaşları [66], NSL-KDD veri seti için farklı algoritmaların sonuçlarını analiz etmişlerdir. Çalışılan sınıflandırma algoritmaları arasında, çalışmada karşılaştırılan diğer algoritmalara göre doğruluk açısından en iyi sonucu Random Forest algoritmasının elde ettiği tespit edilmiştir.

Başka bir çalışmada Ray [67], ağ saldırılarını tespit etmede bir yapay sinir ağı mimarisinin etkisini incelemiştir. Çok katmanlı ileri beslemeli bir ağ için oluşturduğu denklemi kullanarak ideal gizli nöron sayısını bulduğunu ve uzun süren hesaplamalardan kaçındığını belirtmiştir. Bu veri seti için incelenen bir diğer çalışmada Elhag ve arkadaşları [68], çoklu hedefleri olan evrimsel bir bulanık sisteme dayalı bir saldırı tespit sistemi önermişlerdir. Bu yöntemin C4.5 karar ağacı algoritmalarından daha iyi sonuçlar verdiğini tespit etmişlerdir.

Bu çalışmada da kullanılan bir hibrit algoritma geliştirmedeki ana fikir; daha iyi birleştirilmiş bir model oluşturup daha doğru, daha iyi ve daha güvenilir sonuçlar elde etmektir. Birden çok algoritmayı entegre ederek yeni bir model oluşturma fikri kapsamlı bir şekilde araştırılmıştır. Konuyla ilgili yapılan çalışmalar incelendiğinde Atefi ve arkadaşları [69], Genetik Algoritma ve SVM kullanarak hibrit bir yaklaşım sunmuşlardır. Oluşturulan algoritmanın, SVM'e göre %98,33 doğruluk oranı ile daha iyi sonuçlar verdiği belirtilmiştir. Kim ve arkadaşları [70], ağ saldırısı tespiti için çoklu 1-sınıf SVM ve C4.5 karar ağaçlarına dayalı hibrit bir çözüm araştırmışlardır. Bu çalışmalarında NSL-KDD veri seti kullanılarak, geleneksel yöntemlere kıyasla yanlış pozitif oranı ve zaman karmaşıklığının azaldığını ve eğitim süresinin %50, test süresinin %40 oranında azaldığını belirtmişlerdir. Gupta ve Shrivastava [71], yaptıkları çalışmada Yapay Arı Kolonisi Algoritması (ABC) ve SVM algoritmasını birleştirerek hibrit bir algoritma geliştirmişlerdir. Bu çalışma, geliştirilen hibrit yöntemin hem ABC hem de SVM'den daha iyi ortalama doğruluk elde ettiğini belirtmişlerdir.

Yapılan çalışmalarda başarı oranını arttırmanın bir diğer yöntemi de özellik seçimidir. Özellik seçimi daha iyi sonuçlar elde etmek için en uygun özellikleri bulmanıza yardımcı olabilmektedir. Konuyla ilgili olarak Pervez ve arkadaşları [72], çalışmalarında SVM kullanmışlardır. Saldırı sınıflandırması ve özellik seçimine odaklanan bu çalışmada, %90,99 ile %98,96 arasında doğruluk elde etmişlerdir. Başka bir çalışmada Dhanabal ve arkadaşları [73], sınıflandırma algoritmaları ve korelasyona dayalı bir özellik seçim algoritması kullanmışlardır. Sonuç olarak, uygulanan özelliğin seçim algoritmasının tanıma süresini azalttığı ve çalışmalar karşılaştırıldığında J48 sınıflandırma algoritmasının diğer algoritmalara göre daha iyi sonuçlar verdiği tespit edilmiştir. Bununla birlikte Derin Öğrenme gibi yöntemlerin kullanıldığı veya ağ saldırılarının saldırı türlerine göre özelleştiği çalışmalar da mevcuttur.

Yin ve arkadaşları [74], ağ saldırı tespiti için Derin Öğrenme yaklaşımını incelemişlerdir. Çalışmalarında, yinelemeli bir sinir ağı modeli kullanarak Yapay Sinir Ağı (YSA), SVM ve J48 gibi yaygın olarak kullanılan yöntemlerden daha yüksek doğruluk %83,28 ve daha iyi performans elde ettiklerini belirtmişlerdir. Başka bir çalışmada Alkasassbeh ve arkadaşları [75], Hizmet Reddi (DoS) saldırısını tespit etmek ve sınıflandırmak için bir saldırı tespit sistemi önermişlerdir. Veri madenciliği tekniklerinin kullanıldığı bu saldırı tespit sisteminde Random Forest, Multi-Layer Perceptron (MLP) ve Naive Bayes gibi sınıflandırma algoritmaları kullanılmış ve MLP'nin bu algoritmalarından daha iyi sonuçlar verdiği tespit edilmiştir.

Ağ saldırılarının tespit edilmesine yönelik KDD Cup'99 veri seti kullanılarak geliştirilen çeşitli çalışmalar da bu çalışma kapsamında incelenmiştir. Kushwaha ve arkadaşları [76], en iyi özellikleri belirlemek için karşılıklı bilgi yöntemini kullanmışlardır. Bu çalışmada kullanılan veriler saldırı ve normal olmak üzere 2 farklı sınıfta analiz edilmiştir. Eğitim SVM, Bagging, Naive Bayes, One Rule (OneR), Random Forest, Adaboost, SVM+Naive Bayes, KNN (k=5) ve KNN (k=10) algoritmaları kullanılarak gerçekleştirilmiştir. Sonuç olarak, çalışmada karşılaştırılan SVM algoritmasının performansının diğer algoritmalara göre daha iyi olduğunu belirtmişlerdir. Başka bir çalışmada Kim ve arkadaşları[77], kalıcı ve gelişmiş tehditleri incelemişlerdir. Başka bir çalışmada, Gautam ve arkadaşları [78], farklı özel saldırılar için Naive Bayes, entropi tabanlı algoritma ve J48 algoritmasını incelemişlerdir. Elde ettikleri sonuçlarda, entropi tabanlı algoritmanın J48 ve Naive Bayes algoritmalarından daha iyi performans gösterdiği belirtilmiştir.

Son yıllarda, ağ güvenliğiyle ilgili sorunlara çeşitli açılardan yaklaşarak çözüm sunan birçok çalışma da literatürde kendine yer edinmiştir. Mishra ve arkadaşları [79], DDoS saldırılarına karşı koruma sağlamak için düşük hesaplama maliyetine sahip entropi tabanlı bir mekanizma sunmuşlardır. Kullandıkları yöntemde üç eşik değeri bulunmaktadır. Bunlar entropi, paket akış hızı ve sayım eşliğidir. İlk önce paket akış hızı eşik kontrol edilir. Bu eşik aşıldığında anahtarlardan veri akış tablosunun istatistikleri alınır ve entropi hesaplanır. Entropi küçükse sayım değeri artırılır ve sayım eşik aşıldığında bir saldırı uyarısı verilir. Çalışma sonucunda saldırı algılamasının yüksek doğrulukla yapıldığı ve saldırı etkisinin azaltıldığı belirtilmiştir. Diğer bir çalışmada, Gupta ve arkadaşları [80], sağlık verilerinin gizliliğini sağlamak için verileri şifreleme ihtiyacını incelemiştir. Veri şifrelemenin veri arama üzerindeki

olumsuz etkisini azaltmak ve dayanıklılık için bir çözüm olarak sundukları bu çalışmada Blockchain teknolojisinden faydalanılmıştır. Bu sistemin sağladığı en önemli avantajlardan birinin kullanılan Blockchain ağındaki düğümler arası maliyetlerin paylaşılmasıyla elde edilen verimlilik olarak ifade etmişlerdir. Cil ve arkadaşları [81], ağ saldırılarını tespit etmek için Derin Sinir Ağı (DNN) modelini önermişlerdir. CICDDoS2019 veri setinden 2 farklı örnek oluşturarak yapmış oldukları çalışmalarında, birinci veri setine saldırı tespit senaryosu, ikinci veri setine ise saldırı sınıflandırma senaryosu uygulamışlardır. Sonuç olarak hem saldırı tespitinde hem de saldırı sınıflandırmasında oldukça başarılı sonuçlar elde ettiklerini belirtmişlerdir. Kumar ve arkadaşları [82], IoT ortamlarında kullanım için birleşik bir saldırı tespit sistemi önermişlerdir. Eşik güven faktörüne bağlı olarak farklı karar ağacı modellerinden oluşturulan kurallar seçilmiş ve sonrasında veri setinin analizi yapılmıştır. UNSW-NB15 veri setinin kullanıldığı bu çalışmada, saldırı tespit oranlarında karar ağacı modellerinden daha iyi performanslar elde edildiği belirtilmiştir. Mousavi ve arkadaşları [83], IoT uygulamalarında veri güvenliğini iyileştirmek için ABC, Secure Hash Algorithm 256 (SHA256) ve Elliptic Curve Cryptography (ECC) kullanan hibrit bir model önermişlerdir. ABC algoritması kullanılarak bir özel anahtar üretildiğinde şifreleme ve şifre çözme işlemlerinin verimliliğinin %50'den fazla artırılabilceği ve şifreleme için yürütme süresinin RSA-AES yöntemine göre %52,31 daha düşük olduğu ifade edilmiştir. Başka bir çalışmada Stergiou ve arkadaşları [84], sis ağlarında güvenlik konusunu ele almışlardır. Büyük veri ile çalışmak için bulut bilişim tabanlı bir çözüm sunmuşlardır. Sundukları bu çözümde, ağ güvenliğini iyileştirecek bir mimari oluşturmak için bulut sunucusu ile internet arasına bir güvenlik duvarı yerleştirilmiştir. Mousavi ve arkadaşları [85], IoT tabanlı sulama sistemleri için bir çalışma önermişlerdir. Bu sistemlerin güvenliğini artırmak için kriptografik algoritma sunulmuştur. Bu çalışmada, kriptografinin düğümler arasındaki gizliliği ve bütünlüğü korumak için en önemli çözümlerden biri olduğu vurgulanmıştır. Çalışmalarında Rivest Cipher (RS4), SHA-256 ve ECC kullanılmıştır. Konuyla ilgili yapılan bazı çalışmalar ve detayları Çizelge 3.1'de yer almaktadır.

Çizelge 3.1. İlgili çalışmaların özeti

Yazar	Yıl	Metot	Veriseti	Avantaj	Dezavantaj
Teng ve arkadaşları [86]	2017	2-Sınıf destek vektör makinesi ve karar ağacı	KDD99	Eğitim süresi destek vektör makinesine göre çok daha kısadır.	Yeni veya bilinmeyen saldırının doğruluğu çok düşüktür.
Guo ve arkadaşları [87]	2016	Kümeleme ve k-NN algoritması	KDD99 ve Kyoto	Yanlış kullanım tespiti ve anormallik tespitinden oluşan iki seviyeli hibrit çözüm sunmaktadır.	TPR ve doğruluk değerleri düşüktür.
Khraisat ve arkadaşları [88]	2020	Tek sınıf destek vektör makinesi ve C5.0 karar ağacı	NSL-KDD ve ADFA	Bu çalışma, imza tabanlı STS ve anomali tabanlı STS'nin güçlü yönlerinden yararlanan imza ve anormal tabanlı saldırı tespit sistemlerinin entegrasyonudur.	Saldırı türlerine göre detaylı bir analiz yapılmamaktadır.
Ahmim ve arkadaşları [89]	2018	HCPTCIDS	KDD99 ve NSL-KDD	Düşük yanlış alarm oranı	DoS ve Probe saldırıları için düşük tespit oranı
Al-Yaseen ve arkadaşları [90]	2017	Aşırı öğrenme makineleri, K-means kümeleme ve destek vektör makinesi algoritmaları	KDD99	Yüksek algılama doğruluğu	Çalışmada sadece KDD99 veri seti kullanılmıştır. Saldırı türlerine göre detaylı bir analiz yapılmamaktadır.
Kevric ve arkadaşları [91]	2017	Rastgele ağaç, C4.5 karar ağacı, NBTree algoritmaları	NSL-KDD	Bireysel sınıflandırıcılara göre sonuçlar iyileştirilmiştir.	Saldırı türlerine göre detaylı bir analiz yapılmamaktadır.
Aslahi-Shahri ve arkadaşları [92]	2016	Genetik algoritma ve destek vektör makinesi	KDD99	Yüksek algılama doğruluğu	Çalışmada sadece KDD99 veri seti kullanılmıştır. Saldırı türlerine göre detaylı bir analiz yapılmamaktadır.
Mahmod ve arkadaşları [93]	2015	Çok katmanlı algılayıcılar ve yapay arı kolonisi algoritmaları	NSL-KDD	Anlaşılması kolay ve basit	Düşük doğruluk oranı
Elbasiony ve arkadaşları [94]	2013	Ağırlıklı K-means ve rastgele orman algoritmalarını kullanan hibrit bir yaklaşım	KDD99	Sınıflandırıcı, yeni saldırı türlerini tespit edebilir.	Çalışmada sadece KDD99 veri seti kullanılmıştır.
Panda ve arkadaşları [95]	2012	Rastgele orman, karar ağacı ve destek vektör makinesi	NSL-KDD	Yüksek algılama oranı ve düşük yanlış alarm oranı	Sadece ikili sınıflandırma çalışılmıştır.
Tahir ve arkadaşları [96]	2015	Destek vektör makinesi ve K-Means kümeleme	NSL-KDD	Yüksek TPR ve düşük yanlış alarm oranı	Saldırı türlerine göre detaylı bir analiz yapılmamaktadır.
Gupta ve Shrivastava [97]	2015	Yapay arı kolonisi algoritması ve destek vektör makinesi	KDD99	Bireysel sınıflandırıcılara göre iyileştirilmiş sonuçlar	Düşük algılama oranı ve sadece KDD99 veri seti kullanılmıştır.
Chuan-long ve arkadaşları [98]	2017	Tekrarlayan sinir ağları (Recurrent neural networks)	NSL-KDD	Özellik çıkarımı ile veri setinin daha iyi bir temsili elde edilmiştir.	Eğitim, en iyi öğrenme oranını ve gizli düğüm sayısını elde etmek için birkaç kez gerçekleştirilir.
Almi'Ani ve arkadaşları [99]	2018	K-means kümeleme ve kendini düzenleyen haritalar (Self-Organizing Map)	NSL-KDD	Anlaşılması kolay ve basit	Zayıf hassasiyet ve zaman alıcı yöntem
Kamarudin ve arkadaşları [100]	2017	Wrapper ve filter yöntemlerini kullanan hibrit özellik seçimi	NSL-KDD ve UNSW-NB15	Bilinmeyen saldırıları tespit edebilir.	Yalnızca beş özellik ve sekiz veri kümesi sınıfı kullanılır.
Naoum ve arkadaşları [101]	2013	Esnek geri yayımlı yapay sinir ağı	NSL-KDD	Yüksek genelleme Hızlı yakınsama	Düşük algılama oranı
Lei [102]	2017	Destek vektör makinesi ve parçacık sürüşü optimizasyonu (PSO)	KDD99 ve DARPA	PSO ile geliştirilmiş destek vektör makinesi parametreleri	Sistem sadece 6 özellik kullanmaktadır.

3.1.Çalışma Kapsamında İncelenen Algoritmalar

3.1.1.Diferansiyel Gelişim Algoritması (DE)

1995 yılında Storn ve Price tarafından geliştirilmiş bir algoritmadır. Sürekli verilerle ilgili problemlerde özellikle oldukça başarılı sonuçlar ortaya koyabilen ve temel olarak genetik algoritmaya dayalı popülasyon tabanlı sezgisel bir algoritmadır [17]. Küresel optimizasyon için popülasyon tabanlı güçlü bir algoritmadır [103]. Kullanım kolaylığı, basitliği ve hızı sahip olduğu avantajlar arasında yer almaktadır. Bu algoritmada eş zamanlı olarak birçok farklı noktada araştırmalar yapılmaktadır. Klasik Genetik Algoritmadan farklı olarak bu algoritmada değişkenler gerçek değerleriyle yer almaktadır. Genetik Algoritmada faydalanılan mutasyon, çaprazlama ve seçim operatörleri bu algoritmada da kullanılmaktadır. Genetik Algoritmadan farklılaşan özelliği ise bu algoritmada operatörlerin sırayla tüm popülasyona uygulanmamasıdır. Kromozomlar birer birer ele alınır ve seçimi rastgele olan diğer üç kromozomda dâhil edilerek yeni birey meydana gelmektedir. Mevcut kromozom ile elde edilmiş yeni kromozomun karşılaştırılması yapılmakta ve daha uygun olan sonraki popülasyona yeni bir birey olarak aktarılmaktadır.

Bu algoritmada da Genetik algoritmada olduğu gibi minimize edilmesi hedeflenen amaç fonksiyonu $F(X)$ bulunmaktadır. Bu amaç fonksiyonu kullanılarak her iterasyonda uygunluk değeri hesaplanır. Bulunan yeni uygunluk değeri, durdurma şartı veya önceki değere göre kontrol edilir ve işlemler devam eder. Kromozom gen sayısı değişken sayısına göre belirlenir. DE algoritmasında birçok farklı parametre kullanılmaktadır. Bu parametrelerden NP popülasyon büyüklüğünü, CR çaprazlama sabitini, F ise ölçekleme vektörünü temsil etmektedir. DE algoritmasının kaba kodu Çizelge 3.2’de gösterilmektedir.

Çizelge 3.2. DE algoritması kaba kodu

No	İşlem Adımları
01:	Adım 1: Kontrol parametrelerinin değerlerini ata, ölçekleme vektörü F, çaprazlama oranı Cr, popülasyon büyüklüğü NP, boyut D.
02:	Adım 2: Rastgele popülasyon oluştur
03:	Adım 3:
04:	For each jenerasyon G do

05: **For** i=1 to NP **do**
06: **Adım 3.1** Mutasyon
07: **Adım 3.2** Çaprazlama
08: **Adım 3.3** Uygunluk Değeri Hesaplama
09: **Adım 3.4** Seçim
10: **End for**
11: **End for**
12: **Adım 4:** Durdurma kriteri sağlanmadıysa **Adım 3'e** git

Bu algorithmada başlangıç popülasyonu aşağıda yer alan denklem kullanılarak oluşturulur.

$$x_{j,i,G=0} = x_j^{(l)} + rand_j[0,1].(x_j^{(u)} - x_j^{(l)}) \quad (3.1)$$

Denklem (3.2)'de mutasyon operatörünün matematiksel ifadesi yer almaktadır.

$$n_{j,i,G+1} = x_{j,r_3,G} + F.(x_{j,r_1,G} - x_{j,r_2,G}) \quad (3.2)$$

Mutasyon işleminden sonra oluşan ağırlıklandırılmış fark kromozomu ile x_{iG} kromozomu birlikte yeni deneme kromozomu u_{iG+1} i oluşturur. Bu kromozomda her gen ağırlıklandırılmış fark kromozomundan CR olasılığında, deneme kromozomundan da 1-CR olasılığında seçilir. Amaç, girilen miktarda genlerin seçileceği kromozomun yönlendirilmesidir. Bu işlem için aşağıda yer alan denklem kullanılmaktadır.

$$\forall j \leq D: x_{j,u,G+1} = \begin{cases} x_{j,n,G+1}, & rand[0,1] \leq RC \text{ } v \text{ } j = j_{rand} \\ x_{j,i,G}, & diğ er \end{cases} \quad (3.3)$$

Yeni üretilen kromozomlar ile mevcut jenerasyon arasında seçim yapılarak yeni jenerasyonlar belirlenir. Kromozomların uygunluk değerleri temel alınarak seçim işlemleri gerçekleştirilir. Bu seçim işlemi için aşağıda yer alan Denklem (3.4) kullanılmaktadır.

$$\forall i \leq NP: x_{i,G+1} = \begin{cases} x_{u,G+1}, & f(x_{u,G+1}) \leq f(x_{i,G}) \\ x_{i,G}, & diğ er \end{cases} \quad (3.4)$$

3.1.2. Parçacık Sürü Optimizasyon Algoritması (PSO)

1995 yılında Kenedy ve Eberhart'ın geliştirmiş olduğu bir algoritmadır. Sürü zekası temeline dayanmaktadır [104]. Hayvanların, güvenlik ve yiyecek bulma gibi durumları için sergiledikleri hareketlerin amaçlarına ulaşmada kolaylık sağladığı tespit edilmiştir. Genetik algorithmada olduğu gibi aramalar popülasyon sayısınca yapılır. Her birey parçacık olarak adlandırılır. Bireyler pozisyonlarını, önceki tecrübelerinden faydalanarak güncellerler. Sürüde yer alan bireylerin en iyi pozisyona sahip bireye yaklaşmasına dayanmaktadır. Yaklaşım hızı rastgele olmakla birlikte genellikle önceki konuma göre daha iyi bir konuma ulaşırlar ve bu işlem amaca ulaşınca kadar sürmektedir. Bu algoritma; yerel optimumlardan kaçınma başarısı, az parametre kullanması ve hızlı sonuç vermesi gibi nedenlerle literatürde yer alan birçok algoritmaya göre üstünlük sağlamıştır.

PSO algoritmasında sürüde yer alan tüm parçacıklar rastgele değerler ile arama işlemine geçerler. Her parçacığın, hız(V) ve konum(X) olmak üzere iki vektörel bileşeni bulunmaktadır. Hız vektörü, parçacık için yön ve konum değiştirme miktarlarını tutarken parçacığın pozisyon bilgisi konum vektöründe tutulmaktadır. Her iterasyonda konum ve hız bileşenleri güncellenir. Yeni hız vektörünün hesaplanmasında aşağıda yer alan Denklem (3.5) kullanılmaktadır.

$$V_{ij}^{t+1} = wV_{ij}^t + c_1r_1(P_{ij} - x_{ij}) + c_2r_2(G_{geniyi} - x_{ij}) \quad (3.5)$$

Bu denklemde t iterasyon sayısını, c1 ve c2 hızlandırma katsayılarını, w atalet ağırlığını, r1 ve r2 [0-1] aralığında rastgele üretilen sayıları, P_{ij} parçacık için yerel en iyi değeri ve G_{gbest} sürünün en iyi değerini temsil etmektedir. Yeni konum vektörü için ise aşağıda yer alan Denklem (3.6) kullanılmaktadır.

$$x_{ij}^{t+1} = x_{ij}^t + V_{ij}^{t+1} \quad (3.6)$$

PSO algoritmasının kaba kodu Çizelge 3.3'de gösterilmektedir.

Çizelge 3.3. PSO algoritması kaba kodu

No	İşlem Adımları
01:	Pop tane parçacığa konum ve hız değerlerini ata
02:	Repeat


```

03:          For i=1 to pop
04:              Uygunluk değeri hesapla
05:               $P_{best}$  değeri güncelle
06:               $g_{best}$  değeri güncelle
07:              Pozisyon ve hız değeri güncelle
08:          End for
09:          Until Sonlandırma kriteri
10:          End

```

3.1.3. Çoklu Evren Optimizasyon Algoritması (MVO)

Fizikte çoklu evren teorisinden ilham alınarak Mirjalili tarafından 2016 yılında geliştirilmiştir [105]. Beyaz delik, kara delik ve solucan deliği olmak üzere 3 esasa dayanmaktadır. Keşif, sömürü ve yerel aramayı gerçekleştirmek için bu üç kavram geliştirilmiştir. Solucan delikleri sömürü işlemini, beyaz ve kara delikler ise keşif işlemlerini kontrol etmektedir. MVO matematiksel olarak aşağıdaki gibi modellenenebilir.

İlk adımda, evrenler şişme oranlarına göre sıralanır ve her denemede bir beyaz delik seçilir. Bu adım aşağıdaki denklemlerle modellenenebilir.

Varsayımlar.

$$U = \begin{bmatrix} X_1^1 & . & . & . & X_1^d \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ X_n^1 & . & . & . & X_n^d \end{bmatrix} \quad (3.7)$$

U, rastgele oluşturulmuş evren matrisidir. Bu matriste değişken sayısı d ile, evren sayısı ise n ile gösterilmektedir.

$$X_i^j = \begin{cases} X_k^j, & r1 < NI(Ui) \\ X_i^j, & r1 \geq NI(Ui) \end{cases} \quad (3.8)$$

Bu denklemde i. evrenin j. parametresi X_i^j , i. evrenin genişleme oranı NI, i. evren U_i , k. evrenin j. parametresi için X_k^j ile gösterilir. Denklemde r1 [0,1] aralığında rastgele bir sayıdır. Evrenin gelişme kapasitesini ve çeşitliliğini korumak için solucan deliklerinin en iyi evren yönünde hareket ettiği ve evrende rastgele taşındığı varsayılmaktadır. Bu süreçler aşağıdaki denklemlerde ifade edilir:

$$x_i^j = \begin{cases} X_j + TDR \cdot ((ub_j - lb_j) \cdot r_4 + lb_j) & r_3 < 0.5, r_2 < WEP \\ X_j - TDR \cdot ((ub_j - lb_j) \cdot r_4 + lb_j) & r_3 \geq 0.5, r_2 < WEP \\ x_i^j & r_2 \geq WEP \end{cases} \quad (3.9)$$

Bu denklemde, en iyi evrenin j. parametresi için X_j , hareket mesafesi oranını temsil etmek için TDR, solucan deliği bulunma olasılığını temsil etmek için WEP, üst sınır için ub, alt sınır için lb değişkenleri kullanılmaktadır. r_2 , r_3 , r_4 ise [0,1] aralığında rastgele sayılardır. TDR ve WEP hesaplamaları için aşağıdaki denklemler kullanılır. Bu hesaplamalarda T, maksimum iterasyon sayısını temsil eder ve n mevcut iterasyon sayısını temsil eder.

$$WEP = min + n \cdot \left(\frac{max-min}{L} \right) \quad (3.10)$$

$$TDR = 1 - \frac{\frac{1}{n^p}}{\frac{1}{T^p}} \quad (3.11)$$

Algoritmanın kaba kodu Çizelge 3.4'te gösterilmektedir.

Çizelge 3.4. MVO algoritması kaba kodu

Sıra	İşlem Adımları
01:	Başlangıç parametre değerlerini WEP, TDR, ... ata
02:	SU= Sıralanmış Evren
03:	While iterasyon $\leq$ Maks_iterasyon do
04:	Tüm evren için maliyet fonksiyonunu hesapla
05:	For each Evren(i) do
06:	For each nesne(j) do
07:	$\varphi_1 = rand(0,1)$
08:	If $\varphi_1 < NR(U_i)$ then

```

09:          Beyaz delik indeks=Rulet tekeri seçimi
10:          U(Siyah delik indeks,j)=SU(Beyaz delik indeks,j)
11:          End if
12:           $\varphi_2 = rand(0,1)$ 
13:          If  $\varphi_2 < WEP$  then
14:               $\varphi_3 = rand(0,1)$ 
15:               $\varphi_4 = rand(0,1)$ 
16:              If  $\varphi_3 < 0.5$  then
17:                   $U_i^j = En\ iyi\ evren(j) + TDR \times ((Ub_j - Lb_j) \times \varphi_4 + Lb_j)$ 
18:              Else
19:                   $U_i^j = En\ iyi\ evren(j) - TDR \times ((Ub_j - Lb_j) \times \varphi_4 + Lb_j)$ 
20:              End if
21:          End if
22:      End for
23:  End for
24:  End while

```

3.1.4. JAYA Algoritması

Rao tarafından geliştirilen bu algoritma, kontrol parametlerinin kullanımını minimuma indirgeyerek meydana getirilen bir optimizasyon algoritmasıdır [106]. Bu algorithmada maksimum iterasyon sayısı ve popülasyon büyüklüğü parametreleri kullanılır. Popülasyondaki en kötü ve en iyi bireylerin etkileşimleri sonucu yeni bireylerin oluşturulması temel alınmıştır. Numerik ve basit bir yapıya sahip olan bu algoritmayı kodlamak ve uygulamak da oldukça basit olmaktadır. Sahip olduğu basit yapısı algoritmanın çoğu algoritmaya göre daha hızlı çalışıp daha kısa sürede sonuç elde etmesine olanak sağlamaktadır. Popülasyondaki bireylerin en kötü bireyden uzaklaşması ve en iyi bireye yaklaşması amaçlanmıştır.

Bu algorithmada da yine diğer algoritmalarda olduğu gibi rastgele başlangıç popülasyonu oluşturularak çalışma başlatılmaktadır. En iyi ve en kötü bireyler problemin amaç fonksiyonu kullanılarak belirlenir. En kötü bireyden uzaklaşmak ve en iyi bireye yaklaşmak için yapılan etkileşim doğrultusunda bireyler yenilenmektedir.

Popülasyondaki tüm bireylerin yenilenip problemin durdurma kriteri gerçekleşene kadar iterasyonlar sürdürülür. Popülasyondaki bireyler aşağıda yer alan Denklem (3.12)'ye göre güncellenmektedir:

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}(X_{j,en\ iyi,i} - |X_{j,k,i}|) - r_{2,j,i}(X_{j,en\ kötü,i} - |X_{j,k,i}|) \quad (3.12)$$

Bu denklemde $X_{j,k,i}$, i. iterasyon ve k. çözümün j. tasarım değişkenini, $X_{j,en\ iyi,i}$ en iyi çözüm için j. tasarım değişkenini ve $X_{j,en\ kötü,i}$ en kötü çözüm için j. tasarım değişkenini belirtmektedir. r_1 , r_2 ise [0-1] aralığı içinde oluşturulan rastgele değerlerdir [107]. Jaya algoritmasının kaba kodu Çizelge 3.5'te gösterilmektedir.

Çizelge 3.5. JAYA algoritması kaba kodu

Sıra	İşlem Adımları
01:	Popülasyonu belirle
02:	While (Durdurma kriteri sağlanana kadar)
03:	En iyi ve en kötü aday çözümleri belirle:
04:	En iyi ve en kötü çözüme göre çözümü değiştir
05:	If Yeni çözüm < varolan çözüm
06:	Yeni çözümü varolan çözümle değiştir
07:	Else
08:	Varolan çözümü tut
09:	End
10:	End while
11:	return Optimum çözüm

3.1.5. Salp Sürü Algoritması (SSA)

Bu algoritma, Mirjalili ve arkadaşları tarafından geliştirilmiş sürü tabanlı bir algoritmadır [108]. Salpidae ailesinin bir üyesi olan bu salpler bir deniz canlısıdır. Genellikle okyanusun derinlerinde salp zinciri olarak tanımlanan bir sürü oluştururlar. Sergilenen bu sürü davranışında iki tür salp bulunmaktadır. Bu türlerden ilki lider salp, diğeri ise lideri takip eden salplerdir. Lider salp sürünün yönlendirilmesine öncülük eder ve salp zincirinin en başında yer almaktadır. Geri kalan salpler ise lideri takip ederler. Arama uzayında TF olarak belirtilen, sürünün hedefinde bulunan besin

kaynağı bulunmaktadır. Hedefe ulaşmak için lider salp konum güncelleme işlemi aşağıda yer alan denkleme göre gerçekleştirilmektedir.

$$x_j^l = \begin{cases} TF_j + c_1(c_2(ub_j - lb_j) + lb_j) & c_3 \geq 0 \\ TF_j - c_1(c_2(ub_j - lb_j) + lb_j) & c_3 < 0 \end{cases} \quad (3.13)$$

Bu denklemde x_j^l ile lider salpin j. boyuttaki konumu, TF_j ile j. boyuttaki hedef besin kaynağı ifade edilmektedir. c_1 , c_2 , c_3 rastgele sayılar olmakla birlikte ub_j ve lb_j üst ve alt sınırları ifade etmektedir. Takipçi salplerin konumlarının güncellenmesi için ise aşağıda yer alan denklem kullanılmaktadır.

$$x_j^i = \frac{1}{2}(x_j^i + x_j^{i-1}) \quad \forall i \geq 2 \quad (3.14)$$

Denklemde x_j^i , j. boyutta i. takipçi salp konumunu ifade eder. Salplerin başlangıç konumları, sürü zekâsına dayalı diğer algoritmalarda olduğu gibi rastgele oluşturulmaktadır. SSA algoritmasının kaba kodu Çizelge 3.6’da gösterilmektedir.

Çizelge 3.6. SSA algoritması kaba kodu

Sıra	İşlem Adımları
01:	Başlangıç popülasyonunu belirle
02:	While Durdurma kriteri sağlanana kadar
03:	En iyi uygunluk değerini bul
04:	En iyi salp F’yi ayarla
05:	c_1 değerini güncelle
06:	For each salp(x_i) do
07:	If (i==1)
08:	Lider salp konumunu güncelle
09:	Else
10:	Takipçi salplerin konumunu güncelle
11:	End if
12:	End for
13:	Alt ve üst sınırlara göre sınırları aşan salpleri ayarla

14: End while

15: return F

3.1.6. Sinüs Cosinus Algoritması (SCA)

Bu algoritma 2016 yılında Seyedalı Mirjalili tarafından geliştirilmiştir. Optimizasyon problemlerine uygulanan bu algoritma popülasyon tabanlı metasezgisel bir algoritmadır. Optimizasyon süreci popülasyon bireylerinin problem yaklaşımları temsil eden arama uzayı içindeki hareketinden oluşur. Bu algoritmada sinüs ve kosinüs fonksiyonları kullanılır [109]. Hesaplamanın her adımında çözümleri aşağıdaki denklemlere göre günceller:

$$X_{ij}^{(t+1)} = \begin{cases} X_{ij}^{(t)} + r_1 \cos(r_2) |r_3 P_{ij}^{(t)} - X_{ij}^{(t)}|, & r_4 \geq 0.5 \\ X_{ij}^{(t)} + r_1 \sin(r_2) |r_3 P_{ij}^{(t)} - X_{ij}^{(t)}|, & r_4 < 0.5 \end{cases} \quad (3.15)$$

Denklemden $X_{ij}^{(t)}$ t. iterasyon için mevcut i. bireyi, $P_{ij}^{(t)}$ t. iterasyonda en iyi bireyin konumunu ifade etmektedir. Denklemden r_1, r_2, r_3 ve r_4 ise rastgele sayılardır. SCA algoritmasının kaba kodu Çizelge 3.7’de gösterilmektedir.

Çizelge 3.7. SCA algoritması kaba kodu

Sıra	İşlem Adımları
01:	Başlangıçta arama ajanlarını ayarla
02:	Do
03:	Her bir arama ajanını değerlendir
04:	En iyi çözümü güncelle
05:	r_1, r_2, r_3 ve r_4 değerlerini güncelle
06:	Arama ajanlarının konumlarını güncelle
07:	While (t<maks_iterasyon)
08:	Return Elde edilen en iyi çözüm

4. MATERYAL VE YÖNTEM

4.1. YSA Eğitiminde Kullanılan Algoritmalar

Makine öğrenmesi kapsamında kümeleme ve sınıflandırma problemlerinde çeşitli birçok algoritma kullanılmaktadır. Bu bölümde, ağ saldırılarının tespit edilmesi konusunda sıklıkla kullanılan ve tez kapsamında faydalanan algoritmalar hakkında detaylı bilgiler sunulmuştur.

4.1.1. Gri Kurt Optimizasyon Algoritması (GWO)

Gri kurtların sosyal liderliğini ve avlanma stratejisini dikkate alan bu algoritma 2014 yılında Seyedal Mirjalili tarafından sunulmuştur [110]. Genellikle gruplar hâlinde yaşayan gri kurtlarda, gruplar 5-12 kişiden oluşur. Alfa kurt olarak adlandırılan kurtlar, lider kurtlar olarak tanımlanır. Dört ana grup vardır: Omega, Delta, Beta ve Alfa kurtları. Alfa kurtlar en iyi seviyede yer alan kurtlardır, omega kurtlar ise en düşük seviyede yer almaktadır. Alfa kurtlar, gruptaki diğer kurtların yönetiminde, avlanma alanlarında ve uyku zamanlarında karar vericidir. Gri kurtların avlanma stratejisinde öncelik, avı bulmaktır. Ardından av, alfa kurdun önderliğinde kuşatılır. Delta, beta ve alfa kurtlar, kurtların konumunu güncellemek için en iyi üç çözüm olarak kullanılır çünkü avın konumu hakkında daha iyi bilgi sağladıkları düşünülmektedir [111,112].

GWO algoritması, model olarak kurtların hiyerarşisine dayanmaktadır. Bulunan en iyi çözüm alfa, ikinci ve üçüncü çözümler beta ve deltadır. Kalan çözümlere omega denir. Bu algorithmada alfa, beta ve delta, omega'ya komutlar verir ve omega, bu komutlar için çözüm uzayını arar. Bu algorithmada avlanma sürecinin ilk adımı avı çevrelemektir. Matematiksel modelde kullanılan (4.1), (4.2), (4.3) ve (4.4) numaralı denklemler ve açıklamaları aşağıda yer almaktadır.

$$X(t + 1) = X_p(t) - A \cdot D \quad (4.1)$$

$$D = |C \cdot X_p(t) - X(t)| \quad (4.2)$$

$$A = 2\alpha \cdot r_1 - \alpha \quad (4.3)$$

$$C = 2r_2 \quad (4.4)$$

Denklem (4.1)'de X gri kurtların konumunu, X_p avın konumunu ve t yineleme sayısını temsil eder. D , avdan uzaklıktır. Denklemdeki D , Denklem (4.2) ile hesaplanır ve A ve C , sırasıyla Denklem (4.3) ve (4.4) ile hesaplanan katsayılardır. Denklem (4.1) ve (4.2)'yi kullanarak gri kurtlar avının etrafındaki konumunu rastgele güncelleyebilir. Denklem (4.3)'teki α değeri, 2'den 0'a lineer olarak indirgenmiş bir değerdir. Denklem (4.5) ile hesaplanır. MaksIter değeri toplam iterasyon sayısını gösterirken $r1$ ve $r2$, 0 ile 1 arasında rastgele atanan ve optimal çözüm için kullanılan değerlerdir. Avına saldırma aşamasında, α 'nın değeri ve dolayısıyla A 'nın değişim aralığı azalır. A , $[-1,1]$ aralığı içinde rastgele değerlere sahipse arama aracısının sonraki konumu, mevcut konum ve av konumu arası bir yerdedir. Gri kurtlar genellikle alfa, beta ve delta konumlarını arar. Avlarını aramak için birbirlerinden ayrılırlar ve avlarına saldırdıkları anda bir araya gelirler. Dağılımın matematiksel modellenmesi için, A parametresi 1'den büyük veya küçük rastgele değerlerle kullanılır. Bu ise aramayı önemli yapar ve GWO'nun global aramasını destekler. Delta, beta ve alfa kurtları potansiyel konumla ilgili iyi bilgilere sahiptir. Bununla birlikte alfa, beta ve delta çözümleri diğer kurtların konumlarını güncellemelerine yardımcı olur. Denklem (4.6), konum güncellemesi için kullanılan denklemi içerir.

$$\alpha = 2 - t(2/\text{MaksIter}) \quad (4.5)$$

$$X(t + 1) = (X_1 + X_2 + X_3)/3 \quad (4.6)$$

$$X_1 = |X_\alpha - A_1 \cdot D_\alpha| \quad (4.7)$$

$$X_2 = |X_\beta - A_2 \cdot D_\beta| \quad (4.8)$$

$$X_3 = |X_\delta - A_3 \cdot D_\delta| \quad (4.9)$$

t zamanında en iyi 3 çözüm X_1 , X_2 ve X_3 ile temsil edilir. Denklemlerindeki A_1 , A_2 ve A_3 değerleri Denklem (4.3)'e göre hesaplanır. Denklemler (4.10), (4.11) ve (4.12) ayrıca D_α , D_β ve D_δ 'yi hesaplar. Bu denklemlerdeki C_1 , C_2 ve C_3 değerleri Denklem (4.4)'e göre hesaplanır.

$$D_\alpha = |C_1 \cdot X_\alpha - X| \quad (4.10)$$

$$D_\beta = |C_2 \cdot X_\beta - X| \quad (4.11)$$

$$D_{\delta} = |C_3 \cdot X_{\delta} - X| \quad (4.12)$$

Gri kurtların delta, beta ve alfa türleri, avlarının mevcut pozisyonu ile ilgili oldukça fazla bilgiye sahiptir. Bu sebeple, en iyi üç çözüm olarak kaydedilir ve diğer kurtlar konumlarını en iyi arama araçlarının konumlarına göre güncelleyebilir. Denklemler (4.6)-(4.12) bu bağlamda konum güncelleme için kullanılabilir. GWO algoritmasının kaba kodu Çizelge 4.1’de yer almaktadır.

Çizelge 4.1. GWO algoritması kaba kodu

No	İşlem Adımları
01:	İterasyon I için maks değeri ayarla
02:	Popülasyonu oluştur $P_i (i = 1, 2, 3, \dots k)$
03:	Başlangıç değerlerini ata
04:	Gri kurtlar için uygunluk değerlerini hesapla
05:	X_{α} = en iyi ajan
06:	X_{β} = en iyi ikinci ajan
07:	X_{δ} = en iyi üçüncü ajan
08:	While (t<I) do
09:	For each ajan do
10:	Mevcut ajanı belirtilen denklemle güncelle $X_{t+1} = (X_{\alpha} + X_{\beta} + X_{\delta})/3$
11:	End For
12:	Parametreleri güncelle
13:	Uygunluk değerlerini hesapla
14:	X_{α}, X_{β} and X_{δ} ‘ yi güncelle
15:	T=t+1
16:	End While
17:	Return X_{α}

4.1.2. Güve Alevi Optimizasyon Algoritması (MFO)

Seyedali Mirjalili'nin geliştirdiği bu algoritma, güvelerin çapraz yöntelendirme olarak adlandırılan gezinme davranışlarından faydalanılarak geliştirilmiştir [113]. Güveler hareket ettiğinde aya doğru sabit bir açı belirler ve o açıda uçarlar. Bu hareket, güvelerin uzun mesafeler boyunca belirli bir açıyla uymasını sağlamaktadır. Ancak bazen güvelerin ışık etrafında spiral hareket ettikleri görülebilmektedir. Bu duruma yapay ışıklar sebep olur [114]. Işık kaynağına olan aşırı yakınlık ve ışık kaynağına göre sabit açıyı koruma girişimi bu spiral yörüngeyi ortaya çıkarmaktadır. Bu algoritmanın matematiksel modelinde, güve kümesi M matrisi ile temsil edilmektedir. Uygunluk değeri diziM adlı bir dizide tutulmaktadır. Algoritmanın başlangıcında, alt ve üst sınırlara dayalı olarak rastgele bir konumda bir güve popülasyonu atanır. Alev matrisi uygunluk değerleri hesaplanarak güncellenir. Bu matris alev matrisi olarak adlandırılır ve her güvenin ulaştığı en iyi pozisyonu içerir. Güve, mevcut çözümden daha iyi olan bir çözüme ulaştığında alevler de güncellenir. Her güve, alevlerden yalnızca biri ile pozisyonunu güncellemelidir. Her yinelemeden sonra konumlar aleve göre güncellenir. Güvelerin konumu Denklem (4.13) ile güncellenir. Denklem (4.13)'te verilen denklemde güvelerin arama uzayındaki konumları logaritmik spiral fonksiyonuna göre güncellenmektedir.

$$S(M_i, F_j) = D_i \cdot e^{bt} \cdot \cos(2\pi t) + F_j \quad (4.13)$$

Denklemde i'nci güve M_i , j'inci alev F_j ve her yinelemede güve ile alev arasındaki mesafe D_i ile temsil edilmektedir. Denklemde t, -1 ile 1 arasında rastgele oluşturulmuş bir sayıdır ve b ise bir sabittir. Güveler güncellendikten sonra uygunluk değerleri yeniden değerlendirilir. Bu işlemten sonra en iyi çözümler alev matrisinde güncellenir ve tutulur. Konum güncelleme işleminin daha etkili ve başarılı olması için her yineleme adımında alev sayısı azaltılır. Alev sayısındaki azalmanın değişimi Denklem (4.14)'te gösterilmiştir. Her yineleme için alev sayısı bu denklem kullanılarak hesaplanabilmektedir.

$$AlevSayısı(N, I, T) = \text{round}(N - I(N - I)/T) \quad (4.14)$$

Bu denklemde N , maksimum alev sayısını ifade etmektedir. Diğer ifadelerde ise l yineleme sayısını ve T maksimum yineleme sayısını temsil etmektedir. MFO algoritmasının kaba kodu Çizelge 4.2’de gösterilmektedir.

Çizelge 4.2. MFO algoritması kaba kodu

No	İşlem Adımları
01:	Başlangıç parametre değerlerini ata
02:	M_i popülasyonu için rastgele pozisyon oluştur
03:	For each $i = 1:k$ do
04:	Uygunluk değerlerini hesapla
05:	End For
06:	While ($\text{iterasyon} \leq \text{maks_iterasyon}$) do
07:	M_i için güve pozisyonunu güncelle
08:	$AlevSayısı = \text{round}(N - l(N-l)/T)$ denklemi kullanılarak alev sayısını hesapla
09:	Uygunluk değerlerini kontrol et
10:	If ($\text{iterasyon}=1$) then
11:	$F = \text{sirala}(\text{Moth } M)$
12:	$AF = \text{sirala}(\text{Moth Array dizi } M)$
13:	Else
14:	$F = \text{sirala}(M_{t-1}, M_t)$
15:	$AF = \text{sirala}(M_{t-1}, M_t)$
16:	End If
17:	For each $i=1:k$ do
18:	For each $j=1:m$ do
19:	Parametre değerlerini güncelle
20:	$D_i = M_i - F_j $ denklemini hesapla
21:	$M(i,j)$ yi $M_i = S(M_i, F_j)$ denklemini kullanarak güncelle
22:	End For
23:	End For
24:	End While
25:	En iyi sonucu yazdır

4.2. Benchmark Fonksiyonları

Benchmark fonksiyonları, yeni oluşturulan bir optimizasyon algoritmasını farklı özelliklerde test ederek doğrulamak için kullanılan kıyaslama fonksiyonlarıdır. Bu fonksiyonların en önemli özellikleri; ayrıştırılabilir, türevlenebilir, ölçeklenebilir, sürekli, süreksiz, tek modlu ve çok modlu olmalarıdır. Bu özellikler geliştirilen bir algoritmanın hangi tür problemlerde başarılı olacağını belirleyebilmektedir. Bu tez çalışmasında kullanılan hibrit algoritma öncelikle benchmark fonksiyonlarında test edilmiştir ve literatürdeki çeşitli optimizasyon algoritmalarıyla karşılaştırılmıştır. Tek ve çok modlu olmak üzere iki sınıfta incelenen benchmark fonksiyonlarında yalnızca bir global optimum içeren fonksiyonlar tek modlu, birden fazla yerel ve global optimuma sahip fonksiyonlar ise çok modlu benchmark fonksiyonları olarak nitelendirilmektedir.

Benchmark fonksiyonlarından F1-F7 arasındaki fonksiyonlar tek bir çözümü olan tek modlu fonksiyonlardır. Geliştirilen yeni algoritmaların yakınsama oranlarının test edilmesi için uygun fonksiyonlardır. F8-F13 arasındaki fonksiyonlar ise geliştirilen algoritmanın keşif performansının değerlendirilebilmesi için kullanılabilen fonksiyonlardır. Çok modlu kıyaslama fonksiyonlarında birden fazla optimal değer bulunmakta ve bunlardan biri global optimum iken diğer değerler yerel optimum olarak tanımlanmaktadır. Geliştirilen bir algoritmanın keşif süreci kötü tasarlanmışsa etkili bir geniş açılı arama gerçekleştiremez ve bu da algoritmanın yerel optimumda takılı kalmasına sebep olmaktadır. Bu nedenlerle bu fonksiyonlar birçok algoritma için çözülmesi zor problemler olarak görülmektedir.

Bu tez çalışmasında kullanılan yeni hibrit algoritmanın YSA eğitiminde kullanılmasının sebeplerinden biri yeni bir yaklaşım olması iken diğer bir neden ise bu benchmark fonksiyonlarının çözümünde oldukça başarılı olmasıdır. Test edilen benchmark fonksiyonları Çizelge 4.3'te yer almaktadır.

Çizelge 4.3. Kullanılan benchmark fonksiyonları ve karakteristikleri

Matematiksel Formül	Adı	B	Mod	Arahk	fmin
$F1(x) = \sum_{i=1}^n X_i^2$	Sphere	30	Tek	[-100,100]	0
$F2(x) = \sum_{i=1}^n x^i + \prod_{i=1}^n x^i $	Schwefel 2.22	30	Tek	[-10,10]	0
$F3(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	Schwefel 1.2	30	Tek	[-100,100]	0
$F4(x) = \max_i \{ x_i , 1 \leq i \leq n\}$	Schwefel 2.21	30	Tek	[-100,100]	0
$F5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	Rosenbrock	30	Tek	[-30,30]	0
$F6(x) = \sum_{i=1}^n ([x_i + 0.5])^2$	Step	30	Tek	[-100,100]	0
$F7(x) = \sum_{i=1}^n ix_i^4 + random[0,1]$	Quartic	30	Tek	[-1.28,1.28]	0
$F8(x) = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	Schwefel	30	Çok	[-500,500]	-418.9829xB
$F9(x) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	Rastrigin	30	Çok	[-5.12,5.12]	0
$F10(x) = 20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}) - \exp(\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i)) + 20 + e$	Ackley	30	Çok	[-32,32]	0
$F11(x) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}}) + 1$	Griewank	30	Çok	[-600,600]	0
$F12(x) = \frac{\pi}{n} (10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + \text{Penalized } y_n - 1)^2 + \sum_{i=1}^n u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$		30	Çok	[-50,50]	0
$F13(x) = 0.1 \{ \sin^2(3\pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] \} + \text{Penalized2} + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^n u(x_i, 5, 100, 4)$		30	Çok	[-50,50]	0

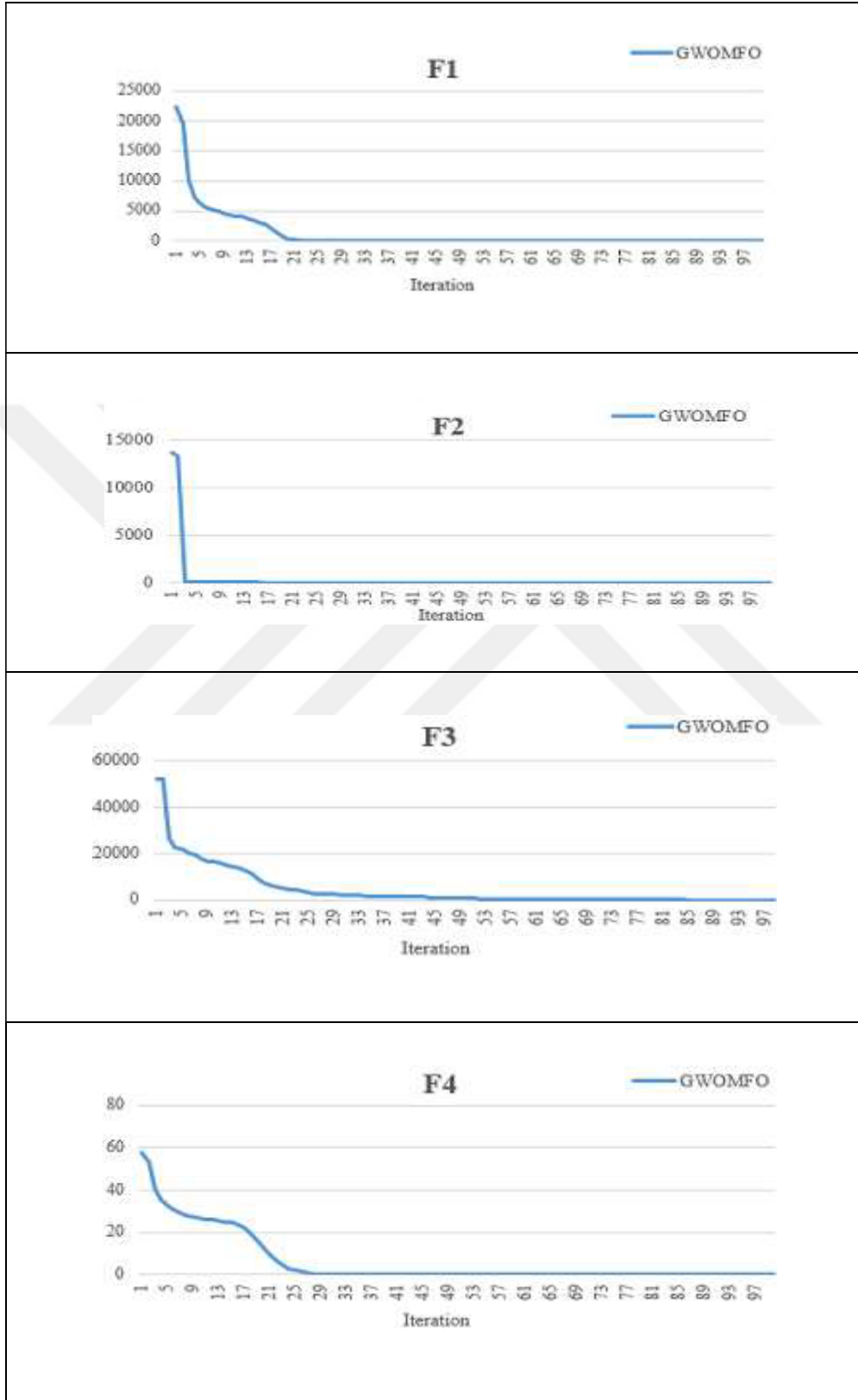
Her bir fonksiyon otuz kez çalıştırılmış ve elde edilen sonuçlara göre standart sapma ve ortalamalar hesaplanmıştır. Bu fonksiyonlar PSO, DE, JAYA, SCA, MVO, SCA ve SSA algoritmaları için de çalıştırılmıştır ve Çizelge 4.4'te karşılaştırma sonuçları yer almaktadır.

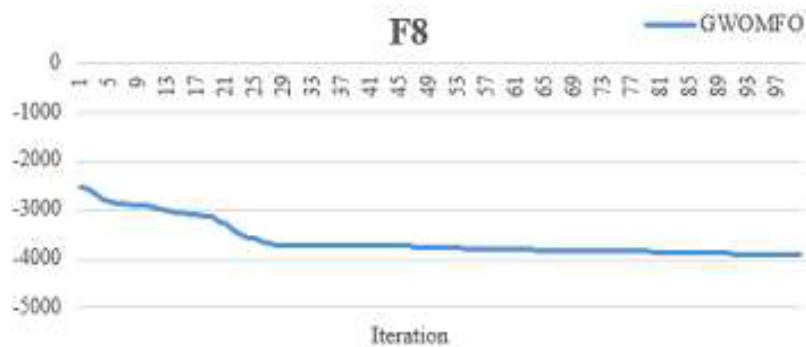
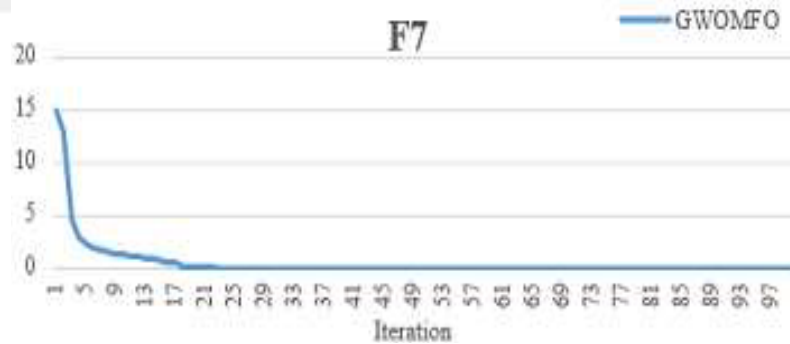
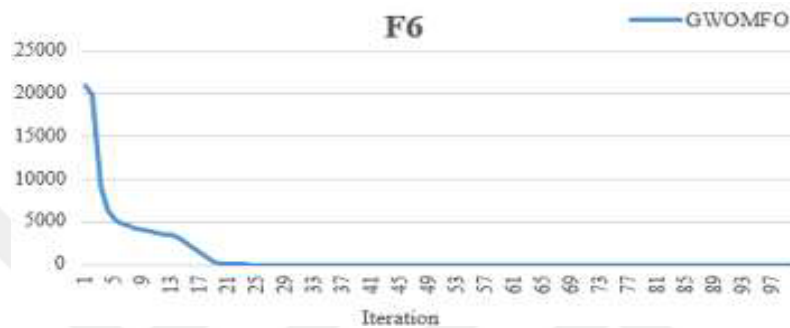
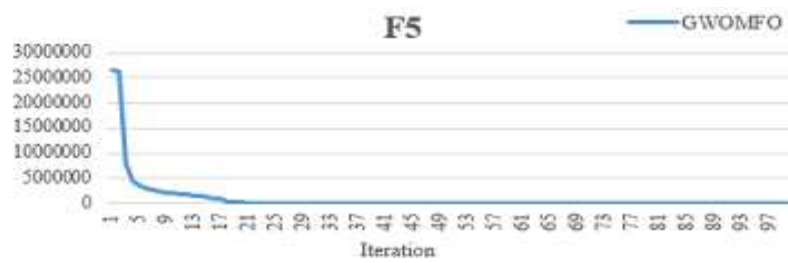
Çizelge 4.4. Geliştirilen algoritma için benchmark sonuçları ve karşılaştırılması

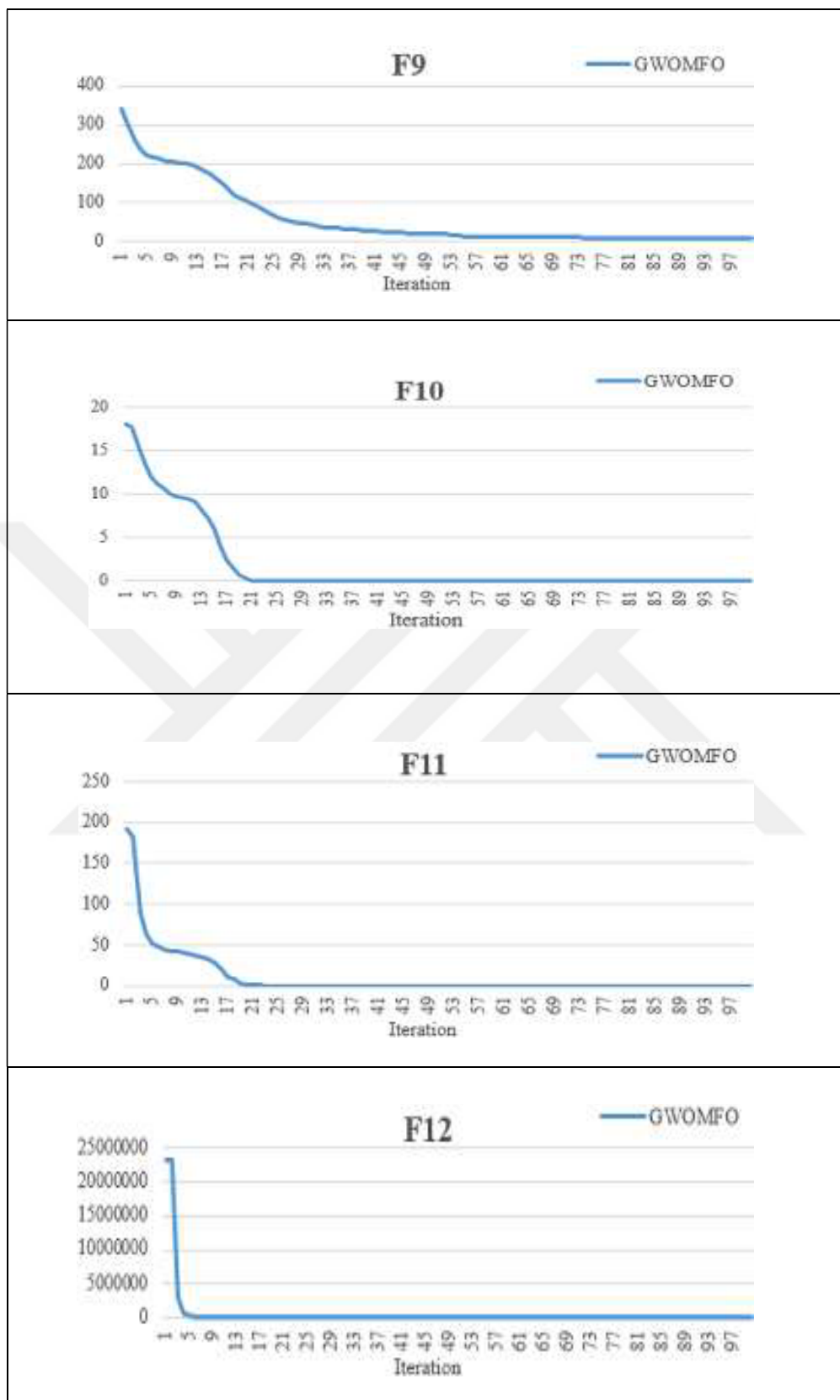
F	GWOMFO		DE		PSO		MVO		JAVA		SSA		SCA	
	ort	std	ort	std	ort	std	ort	std	ort	std	ort	std	ort	std
F1	0	0	1263.87	400.4283	6.88	2.27534	36.04	10.386851	786.1	517.815	305.25	60.019	4344.5	2731.652186
F2	0	0	28.97	6.490931	14.1	7.132155	75.63	61.952689	12.26	4.894991	8.24	0.9479	5.6	2.97684102
F3	9.72	53.23197	47715.65	8485.433	1236.11	485.9593	6671.85	1747.6725	36812	11799.82	2853.3	1056.6	31771	12076.90141
F4	0	0	75.67	6.841525	5.31	1.668242	22.46	8.2750305	55.13	9.855786	12.87	2.8782	64.94	10.07644517
F5	28.84	0.256805	603539.7	377497.8	5109.08	16475.11	2690.73	2216.8964	869149	749194.9	13073	6020.2	8E+06	8702090.465
F6	3.32	0.416849	1208.85	298.9181	6.17	3.0117	35.96	8.9201995	847.74	374.5994	319.89	70.395	3996	2521.325613
F7	0	0.00032	0.59	0.235218	5.3	4.603797	0.14	0.0410475	0.89	0.69831	0.15	0.056	5.97	5.758827531
F8	-3917.17	466.6463	-4597.06	341.6723	-3254.98	492.1964	-7187.4	627.55625	-4247	528.4435	-5825.1	318.35	-3283	292.6505876
F9	7.5	31.89674	246.02	19.02749	213.72	31.3239	164.88	36.019504	200.97	36.83116	125.4	22.725	158.98	53.80637744
F10	0	1.5E-31	10.06	2.134396	3.11	0.440712	4.82	4.154069	9.44	2.635121	5.49	0.3557	17.47	4.587843638
F11	0	0	11.71	3.326986	18.58	4.774702	1.34	0.1050404	8.32	4.738595	3.85	0.4845	36.31	22.94477716
F12	0.5	0.091182	115920.1	299382.4	1.11	1.199464	8.75	3.4550983	497755	930442.9	8.47	3.6403	4E+07	34016524.72
F13	2.8	0.193006	653490.9	587023.1	2.28	1.074185	24.91	17.691802	2E+06	3341609	20.53	7.3844	7407E	80031189.99

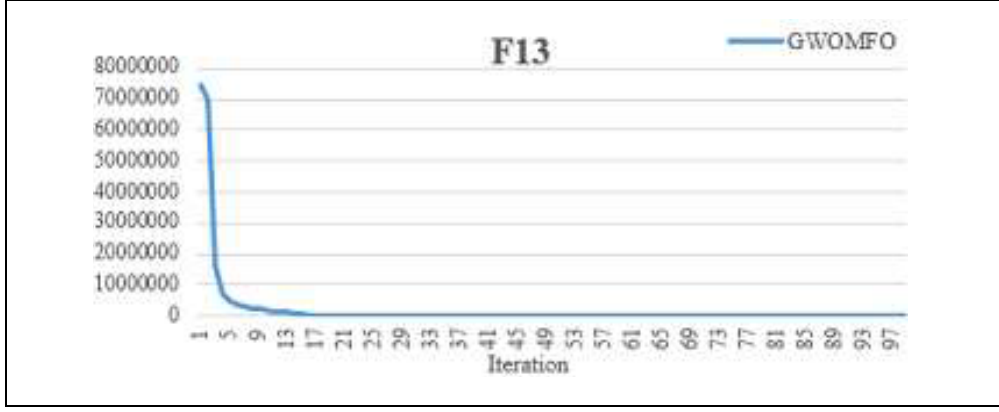
= En iyi sonuç .

Bu sonuçlara ek olarak, GWOMFO tarafından optimize edilen kıyaslama fonksiyonlarının ortalama yakınsama eğrileri Şekil 4.1'de gösterilmektedir.









Şekil 4.1. GWOMFO ile optimize edilmiş benchmark fonksiyonları için ortalama yakınsama eğrileri

4.3. Veri Setleri

Bu çalışma kapsamında 3 farklı veri seti incelenmiştir. Bu çalışmada kullanılan veri kümeleri literatürde ağ saldırı tespiti için yaygın olarak kullanılan veri kümeleri olarak bilinmektedir. Saldırı türleri ile özellik sayısı ve türü açısından farklılık gösteren bu veri setleri, geliştirilen algoritmanın genellenebilirliği ve sağlamlığı hakkında daha doğru sonuçlar elde etmek için çalışmada değerlendirilmiştir. Bu üç farklı veri seti kullanılarak bu çalışmada geliştirilen algoritmanın performans değişimleri ve tutarlılığı incelenecektir.

4.3.1. NSL-KDD

NSL-KDD, ağ saldırı tespiti için yaygın olarak kullanılan bir veri setidir ve bir model olarak da gösterilebilir. Bu veri seti literatürde birçok çalışmada kullanılmıştır [115-119]. KDD Cup'99 olarak bilinen veri setinin genişletilmiş versiyonudur. KDD Cup'99 veri seti ile ilgili bazı sorunlar NSL-KDD veri seti ile düzeltilmiştir [120]. Özellikle, gereksiz kayıtlar kaldırılarak yanlış sonuçlardan kaçınılmalıdır [37]. NSL-KDD veri seti, KDD Cup'99 veri setinin bir evriminin sonucudur ancak özellikler açısından KDD Cup'99 veri seti ile aynı özelliklere sahiptir. Gereksiz veri içermez ancak eğitmek ve test etmek için yeterli veri içerir. Saldırı olarak tanımlanan bu veri kümeleri dört ana sınıfa ayrılmaktadır. Bunlar Remote to Local (R2L), Denial of Service (DoS), Probe ve User to Root (U2R) saldırı türleridir. Bu saldırı türleri:

1. DoS: Bu saldırı türünde tehdit aktörü bir sunucuya çok yüksek sayıda kötü niyetli istek gönderir. Makinenin belleği ve bilgi işlem kaynakları, yasal

trafiğe hizmet veremeyecek kadar dolu veya meşgul olur ve böylece gerçek kullanıcılara hizmet verilemez.

2. U2R: Saldırganın ilk normal kullanıcı erişimiyle kök veya yönetici ayrıcalıkları kazanmaya çalıştığı bir saldırı türünü temsil eder.
3. R2L: Bu saldırı türünde erişim yetkisi olmayan saldırgan ağ üzerinden paket iletir. Bu paket ile birlikte sisteme erişimi sağlanır ve izinsiz olarak veriler kullanılır.
4. Probe: Bu saldırılar herhangi bir makinanın ya da sunucunun ip adresi, aktif port bilgisi vb. bilgilerini öğrenebilmek için gerçekleştirilen saldırılardır.

Kullanılan veri setinin dağılımı Çizelge 4.5'te gösterilmiştir.

Çizelge 4.5. NSL-KDD veri seti dağılımı

Saldırı Türü	Kayıt Sayısı
DoS	45.927
Remote to Local	995
Probe	11.656
User to Root	52
Normal	67.343
Toplam	125.973

125.973 kaydın yer aldığı NSL-KDD veri setinde verilerin 67.343'ü normal, 58.630'u saldırı olarak sınıflandırılmıştır. 41 özelliği (32 sayısal, altı ikili, üç nominal) içeren bu veri seti, normal olarak etiketlenmiş 24 farklı saldırı ve kaydı içermektedir. NSL-KDD veri seti özellikleri Şekil 4.2'de gösterilmiştir.

Özellik No	Özellik Adı	Özellik No	Özellik Adı	Özellik No	Özellik Adı
1.	Duration	2.	Protocol Type	3.	Service
4.	Flag	5.	Src Bytes	6.	Dst Bytes
7.	Land	8.	Wrong Fragment	9.	Urgent
10.	Hot	11.	Num Failed Logins	12.	Logged in
13.	Num compromised	14.	Root shell	15.	Su attempted
16.	Num root	17.	Num file creations	18.	Num shells
19.	Num access files	20.	Num outbound cmds	21.	Is host login
22.	Is guest login	23.	Count	24.	Srv count
25.	Serror rate	26.	Srv serror rate	27.	Rerror rate
28.	Srv serror rate	29.	Same srv rate	30.	Diff srv rate
31.	Srv diff host rate	32.	Dst host count	33.	Dst host srv count
34.	Dst host same srv rate	35.	Dst host diff srv rate	36.	Dst host same src port rate
37.	Dst host srv diff host rate	38.	Dst host serror rate	39.	Dst host srv serror rate
40.	Dst host rerror rate	41.	Dst host srv rerror rate	42.	Label

Şekil 4.2. NSL-KDD veri seti özellikleri

4.3.2. UNSW-NB15

Bu veri seti; modern, gerçekçi normal ağ etkinliğini ve ağ trafiğinden gelen saldırı davranışını birleştirmek için Avustralya Siber Güvenlik Merkezi (ACCS) Siber Menzil Laboratuvarında IXIA PerfectStorm aracı kullanılarak oluşturulmuştur [121]. Bu veri setinde saldırılar 9 kategoride incelenmektedir. Bu saldırı türleri:

1. DoS: Bu saldırı türünde tehdit aktörü bir sunucuya çok yüksek sayıda kötü niyetli istek gönderir. Makinenin belleği ve bilgi işlem kaynakları, yasal trafiğe hizmet veremeyecek kadar dolu veya meşgul olur ve böylece gerçek kullanıcılara hizmet verilemez.
2. Analysis: Bu saldırı türü, bağlantı noktası taraması, güvenlik açığı taraması, spam dosyaları gibi yöntemlere dayalı çeşitli saldırı biçimleri içerir.
3. Exploits: Exploit saldırıları genellikle işletim sistemlerinde var olan bilinen güvenlik açıklarını hedefleyen ve bunlardan çıkar sağlamak için gerçekleştirilen saldırı türleridir.

4. Generic: Çeşitli hash fonksiyonu kullanarak blok şifrenin yapılandırmasına bakılmaksızın bir çarpışmaya neden olan bir tür penetrasyon saldırısıdır.
5. Reconnaissance: Hedef ağ/sunucu hakkında bilgi toplayan basit tekniklerden oluşan saldırı türüdür.
6. Worm: Kötü amaçlı kendi kendini kopyalayan kod. Çok fazla sistem belleği ve ağ bant genişliği tüketir. Sistemlerin kullanılabilirliğini azaltır.
7. Shellcode: Kusurlu bir program tarafından enjekte edilen ve yürütülen talimatlar/ifadeler seti. Bu tür saldırılarla bir programın işlev ve kayıtları doğrudan yönetilebilmektedir.
8. Fuzzer: Yazılım buglarından faydalanan ve hedef programda çeşitli permütasyonlar denenerek bir güvenlik açığı ortaya çıkarmaya çalışılan saldırı türleridir.
9. Backdoor: Veri tabanları ve dosya sunucularına uzaktan erişim sağlamak amacıyla geliştirilen kötü amaçlı yazılım türüdür.

Bu veri seti literatürde birçok çalışmada kullanılmıştır [122-126]. Kullanılan veri setinin dağılımı Çizelge 4.6'da gösterilmiştir.

Çizelge 4.6. UNSW-NB15 veri seti dağılımı

Saldırı Türü	Kayıt Sayısı
DoS	12.264
Analytics	2.000
Exploits	33.393
Generic	40.000
Reconnaissance	10.491
Worm	130
Shellcode	1.133
Fuzzer	18.184
Backdoor	1.746
Normal	56.000
Toplam	175.341

Bu veri seti 175.341 kayıt ve 49 özellik içermektedir [127]. UNSW-NB15 veri setinin özellikleri Şekil 4.3'te gösterilmiştir.

Özellik No	Özellik Adı	Özellik No	Özellik Adı	Özellik No	Özellik Adı
1.	Srcip	2.	Sport	3.	Dstip
4.	Dsport	5.	Proto	6.	State
7.	Dur	8.	Sbytes	9.	Dbytes
10.	Sttl	11.	Dttl	12.	Sloss
13.	Dloss	14.	Service	15.	Sload
16.	Dload	17.	Spkts	18.	Dpkts
19.	Swin	20.	Dwin	21.	Stcpb
22.	Dtcpb	23.	Smeansz	24.	Dmeansz
25.	Trans depth	26.	Res bdy len	27.	Sjit
28.	Djit	29.	Stime	30.	Ltime
31.	Sintpkt	32.	Dintpkt	33.	Tcprtt
34.	Synack	35.	Ackdat	36.	Is sm ips ports
37.	Ct state ttl	38.	Ct flw http mthd	39.	Is ftp login
40.	Ct ftp cmd	41.	Ct srv src	42.	Ct srv dst
43.	Ct dst ltm	44.	Ct src ltm	45.	Ct src dport ltm
46.	Ct dst sport ltm	47.	Ct dst src ltm	48.	Attack cat
49.	Label				

Şekil 4.3. UNSW-NB15 veri seti özellikleri

4.3.3. CIC IDS 2017

Bu veri seti, New Brunswick Üniversitesi, Bilgisayar Bilimleri Okulu tarafından 2017 yılında geliştirilmiştir. ISCX 2012 veri setinin geliştirilmiş bir versiyonudur [128]. Bu veri seti literatürde birçok çalışmada tercih edilmiştir [128-132]. Gerçek ağ trafiğinin genelleştirilmesinden oluşur. Bu veri setindeki toplam kayıt sayısı 2.829.463'tür. Bu kayıtların 2.358.036'sı normal olup geri kalan kayıtlar saldırı sınıfındadır. Bu veri setinde saldırılar 7 kategoride incelenmektedir. Bu saldırı türleri:

1. DoS: Bu saldırı türünde tehdit aktörü bir sunucuya çok yüksek sayıda kötü niyetli istek gönderir. Makinenin belleği ve bilgi işlem kaynakları, yasal trafiğe hizmet veremeyecek kadar dolu veya meşgul olur ve böylece gerçek kullanıcılara hizmet verilemez.

2. DDoS: Çok sayıda farklı sistemden DOS saldırısının yapılması şeklinde oluşan saldırı türüdür. Saldırıları birden fazla bilgisayardan hedef bilgisayar ve ağlara yönelik eş zamanlı olarak gerçekleştirilir.
3. Botnet: Veri çalmak, kötü amaçlı yazılım veya spam iletmek, saldırganın cihaza ve bağlantısına erişmesine izin vererek saldırılar başlatmak gibi çeşitli kötü amaçlı görevleri gerçekleştirmek için kullanılmaktadır.
4. Portscan: Ağa bağlı bir bilgisayar ya da sistemlerin tüm bağlantı noktalarını otomatik olarak tarayarak olası açık bağlantı noktalarını ve hangilerinin güvenlik protokollerinin zayıf olabileceğini tespit etmeye çalışan saldırı türüdür.
5. Web Attack: Bir hedefin bilgisayarını araştırmak ve güvenlik açıklarından yararlanmak için kasıtlı olarak yazılan yazılım programları veya yürütülebilir kodlar olan ve bir saldırganın kullanması için hedefin sistemine bir yol sağlayabilen saldırı türüdür.
6. Infiltration: Ağa içeriden yapılan bir saldırı türüdür. Hedef bilgisayarda bir arka kapı kullanılarak kurbanın ağında tam port taraması, IP taraması vb. farklı saldırılar gerçekleştirilebilir.
7. Heartbleed: Web siteleri tarafından yaygın olarak kullanılan OpenSSL veri şifreleme yöntemindeki bir kusurdan faydalanılarak yapılan bir saldırı türüdür. Kullanılan veri setinin dağılımı Çizelge 4.7’de gösterilmiştir.

Çizelge 4.7. CIC IDS 2017 veri seti dağılımı

Saldırı Türü	Kayıt Sayısı
DoS	252.661
DDoS	41.835
Botnet	1966
Portscan	158.930
Infiltration	36
Web Attack	15.988
HeartBleed	11
Normal	2.358.036
Toplam	2.829.463

Bu veri setinde 80'den fazla özellik bulunmaktadır. Şekil 4.4'te CIC IDS 2017 veri setindeki özellikleri gösterilmektedir.

Özellik No	Özellik Adı	Özellik No	Özellik Adı	Özellik No	Özellik Adı
1.	Flow ID	2.	Source IP	3.	Source Port
4.	Destination IP	5.	Destination Port	6.	Protocol
7.	Time stamp	8.	Flow Duration	9.	Total Fwd Packets
10.	Total Backward Packets	11.	Total Length of Fwd Pck	12.	Total Length of Bwd Pck
13.	Fwd Packet Length Max	14.	Fwd Packet Length Min	15.	Fwd Pck Length Mean
16.	Fwd Packet Length Std	17.	Bwd Packet Length Max	18.	Bwd Packet Length Min
19.	Bwd Packet Length Mean	20.	Bwd Packet Length Std	21.	Flow Bytes/s
22.	Flow Packets/s	23.	Flow IAT Mean	24.	Flow IAT Std
25.	Flow IAT Max	26.	Flow IAT Min	27.	Flow IAT Total
28.	Fwd IAT Mean	29.	Fwd IAT Std	30.	Fwd IAT Max
31.	Fwd IAT Min	32.	Bwd IAT Total	33.	Bwd IAT Mean
34.	Bwd IAT Std	35.	Bwd IAT Max	36.	Bwd IAT Min
37.	Fwd PSII Flags	38.	Bwd PSII Flags	39.	Fwd URG Flags
40.	Bwd URG Flags	41.	Fwd Header Lenth	42.	Bwd Header Lenth
43.	Fwd Packets/s	44.	Bwd Packets/s	45.	Min Packet Length
46.	Max Packet Length	47.	Packet Length Mean	48.	Packet Length Std
49.	Packet Len. Variance	50.	FIN Flag Count	51.	SYN Flag Count
52.	RST Flag Count	53.	PSH Flag Count	54.	ACK Flag Count
55.	URG Flag Count	56.	CWE Flag Count	57.	ECE Flag Count
58.	Down/Up Ratio	59.	Average Packet Size	60.	Avg Fwd Segment Size
61.	Avg Bwd Segment Size	62.	Fwd Avg Bytes/Bulk	63.	Fwd Avg Packets/Bulk
64.	Fwd Avg Bulk Rate	65.	Bwd Avg Bytes/Bulk	66.	Bwd Avg Packets/Bulk
67.	Bwd Avg Bulk Rate	68.	Subflow Fwd Packets	69.	Subflow Fwd Bytes
70.	Subflow Bwd Packets	71.	Subflow Bwd Bytes	72.	Init Win bytes fwd
73.	Act data pkt fwd	74.	Min seg size fwd	75.	Active Mean
76.	Active Std	77.	Active Max	78.	Active Min
79.	Idle Mean	80.	Idle Packet	81.	Idle Std
82.	Idle Max	83.	Idle Min	84.	Label

Şekil 4.4. CIC IDS 2017 veri seti özellikleri

4.4. Veri Ön İşleme

Bu çalışmada, ağ saldırısı tespiti için yaygın olarak kullanılan NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri kümeleri kullanılmıştır. Genellikle veri kümeleri, insanların okuyup anlayabileceği sayısal ve kategorik veriler içerir. Ancak bazı makine öğrenimi ve derin öğrenme modelleri kategorik verileri işleyemez. Bu nedenle gereksiz verileri temizlemek ve performansı artırmak için normalleştirme ve dönüştürme işlemleri gerçekleştirilir. Sonuç olarak daha uygun bir veri seti elde edilir.

Dönüştürme işlemlerinde veri setindeki nominal değerler sayısal değerlere dönüştürülmüştür. Etiket kodlaması kullanılarak verilerin bire bir sayısallaştırılması ve her kategorik değere sayısal bir değer atanması gerçekleştirilir. Bu işlemlerin sonucu sayısal değerlerden oluşan bir veri setidir.

Normalizasyon işleminde Min.-Maks. normalizasyon kullanılarak veriler üzerinde lineer bir dönüşüm yapılmış ve veriler 0-1 arasında yayılmıştır. Bu işlem için kullanılan Denklem (4.15) aşağıda yer almaktadır.

$$X' = \frac{(X - \text{Min}(X))}{(\text{Max}(X) - \text{Min}(X))} \quad (4.15)$$

Min.-Maks. normalizasyon işleminde, alan değeri için minimum değerden ne kadar uzak olduğunu inceler ve bu farklılıkları sıralar [133].

4.5. Hibrit Sınıflandırma - GWOMFO

Ağ saldırı tespit sistemleri için literatürde yer alan çözümler, düşük tespit doğruluğu, dengesiz tespit oranları ve gerçek zamanlı ağlarda saldırı tespitinin zorluğu gibi birçok problemle karşı karşıyadır. Sonuç olarak, bu problemleri çözmek için hibrit teknikler geliştirilmiştir. Hibrit tekniklerde genel olarak her bileşenin ön işleme, sınıflandırma, kümeleme gibi farklı aşamalarda görevleri vardır ve bunlar aynı aşamada birlikte de kullanılabilirler. Bu çalışmada GWO ve MFO algoritmalarının birleştirilmesiyle elde edilen hibrit yapı, sınıflandırma aşamasında uygulanmış ve test edilmiştir. Amaç, yapay sinir ağı (YSA) eğitimi için yeni bir metasezgisel yaklaşım geliştirmek ve karşılaştırılan algoritmalar ile eğitilmiş modelleri kullanarak geliştirilen yöntemin ağ saldırı tespit sisteminde de başarılı olduğunu göstermektir. Hibrit algoritma popülasyonundaki değerler, YSA'nın eğitiminde kullanılan ağırlıkları ve ön yargıları oluşturur. Gözlemlenen ve elde edilen veriler, ağın iyi eğitildiğini ve diğer algoritmalarla eğitilen çalışmalara kıyasla saldırıları başarılı bir şekilde tespit ettiğini göstermektedir.

Bu bölümde, geliştirilen hibrit algoritma sunulmuştur. İki makine öğrenimi algoritmasını birleştiren bir hibrit STS geliştirmek için bir yöntem önerilmiştir. Bu hibrit modelde, GWO ve MFO birleşerek hibrit bir model oluşturulmuştur. Metasezgisel algoritmalar, global veya optima yakın çözümler bulmayı amaçlayan algoritmalar. GWO algoritmasının global bir optimum bulma yeteneği ve MFO algoritmasının en iyi sonucu elde etmedeki başarısı, bu iki algoritmayı birleştirerek daha iyi performans elde edilebileceğini düşündürmektedir.

GWO'da kurtlar, besin kaynağına göre konumlarını günceller. Bu güncelleme sırasında alfa kurtlar ve diğer kurtlar besin kaynağına en yakın olmaya çalışırlar. GWO hesaplamalarında kötü elde edilen pozisyonlar dikkate alınmaz. MFO algoritmasında, yeni çözümü elde etmek için en iyi çözümler hesaplamalara dâhil edilir. Bu şekilde besin kaynağına ulaşmak için en iyi çözümün bulunduğu varsayılır.

Bir önceki bölümde açıklanan veri ön işlemlerinden sonra elde edilen veri seti, önerilen sistemin eğitilmesi ve test edilmesi için kullanılmıştır. Bu algorithmada, GWO algoritmasının ajanlarının etkisi MFO algoritmasına dayalı olarak iyileştirilmiştir. Bu yöntem, varyantı sayısız nesiller boyunca iyileştirmeden çalıştırmak yerine küresel yakınsama, keşif ve yararlanma performansını iyileştirmeyi amaçlar.

Gri kurt avlama stratejisi için matematiksel modelde delta, beta ve alfa kurtların av konumu hakkında daha iyi bilgi sağladığı varsayılmaktadır. Bu nedenle, GWO algoritmasında kurtların konumlarını güncellemek için ilk üç en iyi çözüm kullanılır. Ayrıca bulunan çözümlerle MFO algoritmasından en iyi çözüm değerlendirilmiş ve yeni bir en iyi çözüm oluşturulmuştur. Denklem (4.16) ile verilen denklem, GWOMFO algoritmasındaki konumu iyileştirmek için kullanılır. Bu denklemde, GWO ajanları ile elde edilen pozisyon, MFO algoritmasından alınan pozisyon ile birlikte değerlendirilir.

$$X_{t+1} = (X_a + X_b + X_d + X_m)/4 \quad (4.16)$$

Bu algorithmada standart GWO algoritmasına 2 aşamada müdahale edilerek hibrit yapı sağlanmıştır. İlk olarak, avlanma performansını iyileştirmek için sömürü aşamasında bir koşul eklenmiştir. Daha sonra denklem (4.16) uygulandı. Çözümü geliştirmek için yeni bir mekanizma eklenmiştir. Bu, GWO'nun avlanma mekanizmasını geliştirmiştir. Her yinelemeden sonra çözüm geliştirilir. Ayrıca eklenen koşul, arama yeteneğini ve keşif aşamasını geliştirir ve çözümün kalitesini artırır.

GWOMFO, hem MFO hem de GWO arama araçlarının popülasyon boyutunun başlatılmasıyla çalıştırılır. Bu işlemten sonra uygunluk değeri hesaplanır. Ajanlar için değer atamaları yapıldıktan sonra, mevcut arama ajanlarının konum güncellemeleri, p parametresinin değerine göre döngüdeki maksimum iterasyon sayısına (L) kadar uygun denklemler kullanılarak gerçekleştirilir. Bu işlemlerden sonra uygunluk değeri

hesaplanır ve uygun bir çözüm bulunana kadar işleme devam edilir. p parametresi rastgele oluşturulmuş bir sayıdır. Rastgele değer 0,5'ten küçük veya eşitse, konum güncellemeleri alfa, beta ve delta konum denklemleri kullanılarak gerçekleştirilir. Hesaplama bu durumda Denklem (4.6) kullanılarak yapılır. Yeni pozisyon eskisinden daha iyiye pozisyonlar güncellenir. Eğer 0,5'ten büyükse Denklem (4.16) kullanılır ve eski pozisyonlarla karşılaştırılarak tekrar güncellenir. Sonuç olarak uygunluk değeri tekrar hesaplanır. En iyi uygunluk değeri döndürülür.

Metasezgisel yaklaşımlarda, algoritma arama yeteneği ile doğrudan ilgili iki ana bileşen bulunmaktadır. Bunlar keşif ve sömürü bileşenleridir. Keşif, bilinmeyen bölgelerin derinliklerine inerek umut veren çözümler bulmaya çalışır. Başka bir ifadeyle keşif, çözümlerde çeşitliliği artırmayı amaçlamaktadır. Buna karşılık sömürü, keşfedilebilir ve gelecek vaat eden çözümler çevresinde yerel olarak arama yaparak çözümlerin kalitesini iyileştirmeyi amaçlar. Bu bileşenler birbirleriyle çatışır ve birbirlerini etkilerler. Bu nedenle, sömürü ve keşif arasında doğru ve uygun bir denge ile bir optimizasyon algoritması tasarlanmalıdır. Bu hibrit algoritma ile keşif ve sömürü performansında iyileşme gözlemlenmiştir. Bu durum elde edilen sonuçlara da yansımıştır. GWOMFO algoritmasının sözde kodu Çizelge 4.8'de gösterilmiştir.

Çizelge 4.8. GWOMFO algoritması kaba kodu

Sıra	İşlem Adımları
01:	Başlangıç parametre değerlerini ata
02:	Popülasyonu oluştur $X_i (i = 1, 2, \dots, k)$
03:	Uygunluk değerlerini hespla
04:	X_a = en iyi ajan
05:	X_b = en iyi ikinci ajan
06:	X_d = en iyi üçüncü ajan
07:	X_m = en iyi Güve ajanı
08:	While ($t < L$) do
09:	For each agent do
10:	If ($p \leq 0,5$)
11:	Mevcut ajan konumunu Denklem (4.6) kullanarak güncelle

```
12:           Else
13:           Mevcut ajan konumunu Denklem (4.16) kullanarak
              güncelle
14:       End For
15:       Parametreleri güncelle
16:       Tüm ajanlar için uygunluk değerlerini hesapla
17:        $X_a, X_b, X_d$  ve  $X_m$  yi güncelle
18:       t++
19:       End While
20:       Return çözüm
```

4.6. Eğitim

Modelleme veya tahminden önce veri kümeleri eğitim ve test veri kümelerine ayrılır. Bu ayırmada hold-out yöntemi kullanılmış ve orijinal verilerin 2/3'ü eğitim ve 1/3'ü test veri seti olarak ayrılmıştır [134]. Bu şekilde elde edilen eğitim setleri kullanılarak algoritmanın eğitimi gerçekleştirilmiştir.

4.7. Test

Test aşamasında sekiz sürü zekâsı tabanlı algoritma (GWO, MFO, DE, PSO, MVO, JAYA, SSA ve SCA) ve bu çalışmada geliştirilen algoritma ayrı ayrı uygulanmış ve NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri setleri üzerinde test edilmiştir. Her bir algoritma tarafından elde edilen sonuçlar ayrı ayrı incelenmiştir.

4.8. Değerlendirme

Çalışmada gerçekleştirilen test süreci ile elde edilen sonuçları yorumlamak için çeşitli nicel metrikler kullanılır. Bu metrikler şunlardır:

Doğruluk: Doğru tahminlerin yüzdesini gösterir. Karışıklık Matrisine göre hesaplanır.

$$\text{Doğruluk} = \frac{TP + TN}{n}$$

Gerçek Pozitif Oranı (TPR): Toplam gerçek pozitif sayısı.

$$TPR = TP / (FN + TP)$$

Yanlış Pozitif Oranı (FPR): Bu oranın hesaplanmasında kullanılan formül aşağıdadır.

$$FPR = FP / (TN + FP)$$

Duyarlılık (Recall): Olumlu durumların ne kadar iyi tahmin edildiğini gösterir.

$$\text{Duyarlılık} = TP / (TP + FN)$$

Kesinlik: Olumlu tahminlerde başarıyı gösteren ölçüttür.

$$\text{Kesinlik} = TP / (TP + FP)$$

F-ölçümü: Hassasiyet ve kesinlik değerlerinin harmonik ortalaması.

Çalışmalar bu metrikler kullanılarak değerlendirildi ve karşılaştırıldı.

4.9. Simülasyon Ortamı

Çalışma ortamı ve bu çalışmada kullanılan programlardan kısaca bahsetmek gerekirse, önerilen yöntem için kullanılan bilgisayar konfigürasyonları aşağıdaki gibidir. Geliştirme ve dağıtım için iki bilgisayar kullanılmıştır. Geliştirme bilgisayarı, Intel Core i7-4500U CPU, 16 GB RAM ve 1 TB SATA HDD'ye sahiptir. Dağıtım bilgisayarında Intel Xeon E5-2620 v4 2.10 GHz işlemci (8 çekirdekli) sunucu, 64 GB RAM, 512 GB SSD ve 1 TB SATA HDD bulunur. Ayrıca programlama dili olarak Python 3.7.6 sürümü kullanılmış ve Spyder IDE ile programlanmıştır.

Tüm deneylerde, her algoritmada kullanılan toplam iterasyon sayısı ve popülasyon büyüklüğü gibi ortak parametrelerin değerleri aynı olacak şekilde seçilir. Tüm çalıştırma algoritmalarında eşit arama aracı, eşit çalıştırma ve eşit yineleme ayarlanmıştır. Tüm algoritmalarda kullanılan arama aracı 30, çalıştırma sayısı 30 ve iterasyon sayısı 100 olarak ayarlanmıştır. Geliştirilen algoritma önce benchmark fonksiyonlarında test edilmiş ve sonuçları incelenmiştir. Daha sonra literatürde sıklıkla kullanılan 3 farklı veri seti üzerinde test edilmiştir. Geliştirilen algoritma bu veri setlerinde aynı koşullar altında çalıştırılarak algoritmanın tutarlılığının ve genellenebilirliğinin sağlanması amaçlanmıştır.

Çalışmada YSA eğitimi, optimizasyon problem çözme, kümeleme işlemleri ve öznitelik seçimi için geliştirilen Evolopy Framework kullanılmıştır. İncelenen her

algoritma kendi parametrelerini, çeşitli sabit değerleri kullanabilir ve bunlar uygulanan problemlere göre değişiklik gösterebilir. Evolopy çerçevesindeki değerler, algoritmaların parametre değerlerinde kullanılır [135]. Bu çalışmada çok katmanlı YSA yapısı Multi Layer Perceptron (MLP) kullanılmıştır. MLP, girdi verisi kümelerini bir çıktı kümesine ileri beslemeli bir şekilde eşleyen bir ağıdır [136]. Genel yapısı, birden çok katmanda (giriş, gizli ve çıkış katmanları) birbirine bağlı düğümlerden oluşmaktadır. Her katmanın hem bir önceki hem de bir sonraki katmana tamamen bağlılığı bulunmaktadır [137]. Her düğümün çıktıları, doğrusal olarak ayrılamayan verileri ayırt etmek için doğrusal olmayan bir aktivasyon fonksiyonu tarafından takip edilen ağırlıklı birimlerdir [138]. Bu çalışmada gizli katmanda bulunan gizli nöron sayısının $2 \times N+1$ olduğu yöntem seçilmiştir; N, her veri kümesindeki özelliklerin sayısıdır. Her veri seti için, tüm girdi özellik değerleri normalizasyon tekniği ile [0,1] aralığında normalize edilmektedir.

Kullanılmış olan Evolopy Framework'de yapay sinir ağları kullanılarak eğitim gerçekleştirilir. YSA'da ağırlık, gelen bilginin hücredeki etkisini belirtmektedir. Girdi verisi üzerinden gelen değerlerin ağırlıklarla çarpılması sonucu bağlantı değeri hesaplanır. İlk aşamada rastgele bir sayı değeri bulunan ağırlık, ağ eğitimiyle birlikte güncellenmektedir. Bias değeri ise yerel optimum noktalarına takılmamak için ve öğrenme işleminin hızlanmasını sağlayan değer olarak ifade edilir.

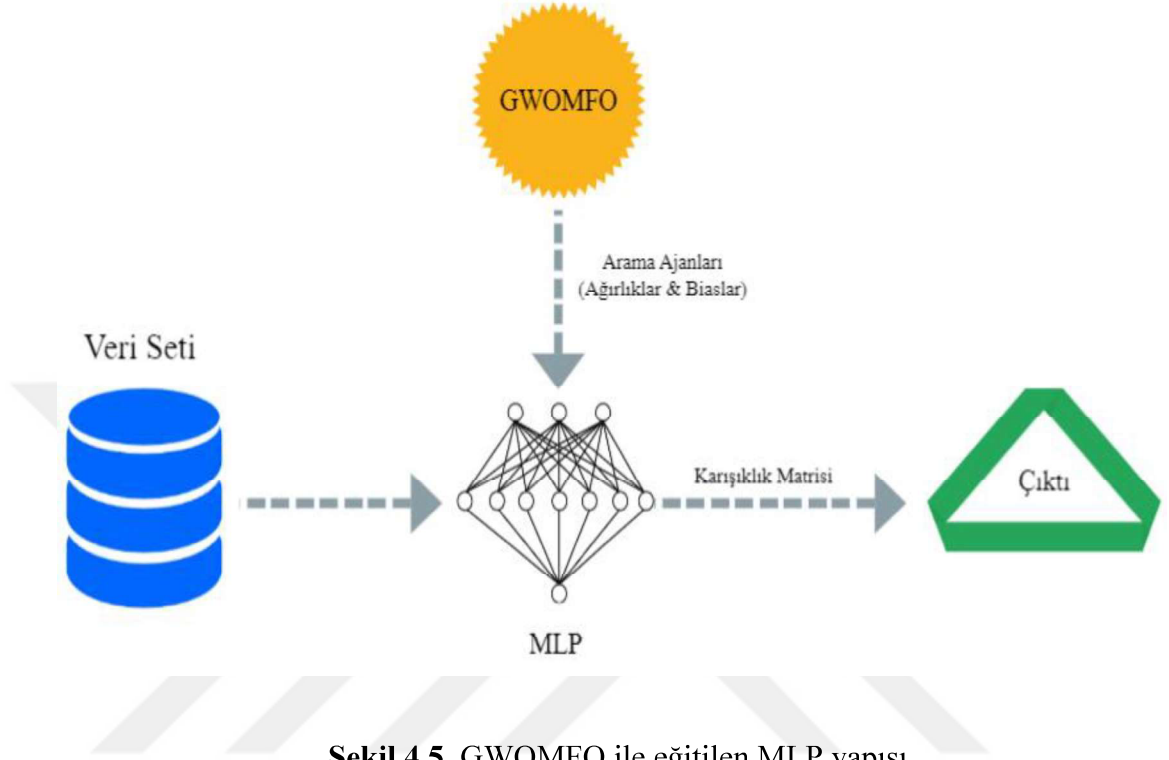
Bir hücre için net girdi hesaplanmasında toplama fonksiyonları kullanılmaktadır. Bu çalışma için kullanılan toplama fonksiyonunda ağırlıkların girdi verileri ile çarpılması sonucuna bias değeri eklenerek toplama işlemi gerçekleştirilir. Kullanılan toplama fonksiyonu Denklem (5.1)'de yer almaktadır.

$$Net = \sum_{i=1}^n (x_i w_i) + b \quad (5.1)$$

Net girdiye göre hücre için çıkış değeri ise aktivasyon fonksiyonu ile üretilir. Bu çalışmada tanjant-sigmoid fonksiyonu kullanılmış olup Denklem (5.2)'de gösterilmektedir. Aktivasyon fonksiyonu sonucunda çıktı sonucu elde edilir. Çalışmada elde edilen çıktı bu fonksiyon tarafından elde edilmektedir.

$$f(n) = \tansig(n) = \frac{2}{1+e^{-2n}} - 1 \quad (5.2)$$

Ağ eğitimi için kullanılan arama ajanları ise $[-1,1]$ aralığında değerler olup bias ve ağırlık değerlerinin oluşması için kullanılmaktadır. GWOMFO algoritması ile eğitilen MLP yapısı Şekil 4.5'te gösterilmektedir.



5. ARAŞTIRMA BULGULARI

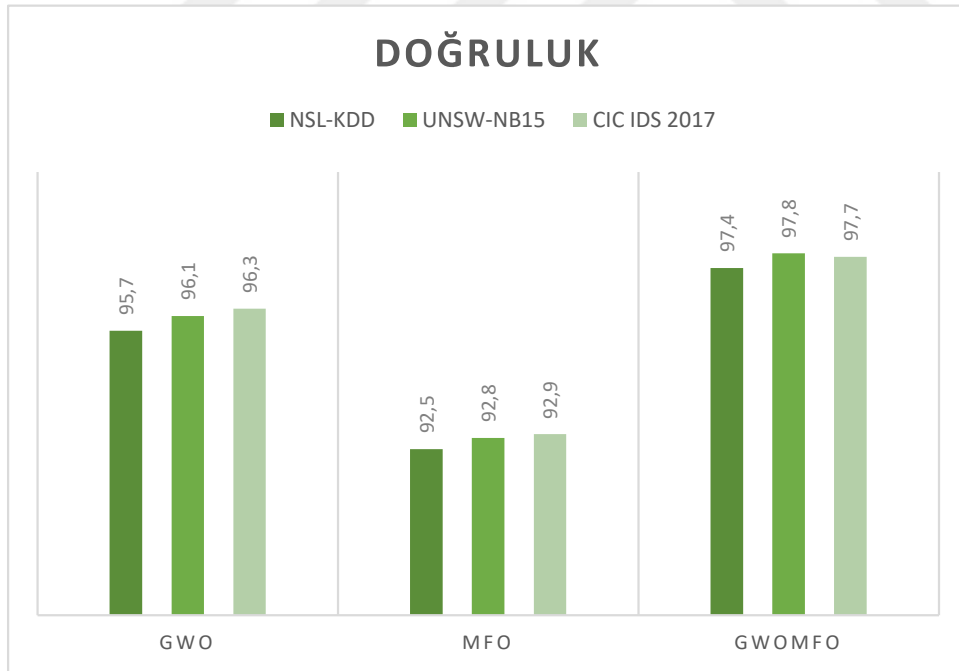
5.1. Ağ Saldırı Tespit Sonuçları

Bu bölümde ağ saldırılarının tespitinde geliştirilen hibrit algoritmanın sonuçları paylaşılmaktadır. Geliştirilen GWOMFO algoritması çalışma kapsamında incelenen veri setlerine uygulanmıştır. Önerilen tekniğin performansını değerlendirmek için karışıklık matrisi kullanılmıştır. Doğruluk, TPR ve FPR metrikleri kullanılarak 3 farklı veri seti ile elde edilen sonuçlar Çizelge 5.1' de gösterilmiştir. Popülasyon büyüklüğü 30, iterasyon sayısı 100 olarak seçilmiş ve uygulanmıştır.

Çizelge 5.1. GWOMFO Algoritması Performansı (%)

Veri Seti	Doğruluk	TPR	FPR
NSL-KDD	97.4	95.7	8.64
UNSW-NB15	98.3	97.2	5.71
CIC IDS 2017	99.2	98.3	4.11

Geliştirilen algoritma, temel aldığı algoritmalar ile karşılaştırılmıştır. Elde edilen doğruluk sonuçları Şekil 5.1'de gösterilmiştir.



Şekil 5.1. GWO, MFO ve GWOMFO algoritmalarının üç farklı veri setinde doğruluk sonuçlarının karşılaştırılması

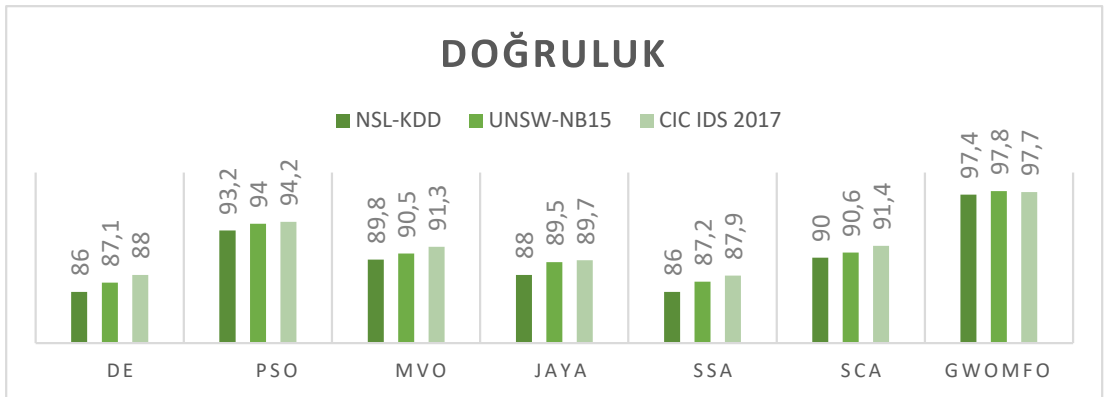
Önerilen algoritmanın doğruluk açısından en iyi sonucu tüm veri setlerinde elde ettiği tespit edilmiştir. Önerilen algoritma GWO algoritması ve MFO algoritmasından daha başarılı sonuçlar elde etmiştir. Başka bir deyişle, GWO ve MFO algoritmalarının birleştirilmesiyle daha iyi sonuçlar elde edilmiştir. Ancak çalışma kapsamındaki diğer kriterler incelenmiş ve sonuçlar Çizelge 5.2'de gösterildiği gibidir.

Çizelge 5.2. GWOMFO, GWO ve MFO algoritmalarının 3 farklı veri setinde karşılaştırılması

Sınıflandırıcı	Metrik	NSL-KDD	UNSW-NB15	CIC IDS 2017
GWO	Doğruluk	95.7	96.1	96.3
	Kesinlik	94.9	95.3	95.5
	Duyarlılık	94.7	94.8	95.1
	F-ölçümü	94.8	95.0	95.3
MFO	Doğruluk	92.5	92.8	92.9
	Kesinlik	92.0	92.4	93.1
	Duyarlılık	91.4	92.0	92.7
	F-ölçümü	91.7	92.2	92.9
GWOMFO	Doğruluk	97.4*	97.8*	97.7*
	Kesinlik	97.7*	97.4*	97.8*
	Duyarlılık	96.5*	97.2*	97.4*
	F-ölçümü	97.2*	97.3*	97.6*

*: En İyi Sonuç

Geliştirilen hibrit algoritmanın doğruluk, kesinlik, duyarlılık ve F-ölçümü açısından daha iyi sonuçlar elde ettiği tespit edilmiştir. Geliştirilen hibrit algoritmanın ağ saldırı tespitindeki başarısını göstermek için literatürde sınıflandırma işlemlerini gerçekleştiren çeşitli sürü zekâsı tabanlı algoritmaların başarıları karşılaştırılmıştır. Karşılaştırma için bu sürü zekâsı tabanlı algoritmalar ve önerilen algoritma, 3 veri kümesi üzerinde aynı parametrelerle çalıştırılmıştır. Doğruluk ölçüsünü dikkate alan karşılaştırma sonuçları Şekil 5.2'de yer alan grafikte gösterildiği gibidir.



Şekil 5.2. Çeşitli sürü zekâsı tabanlı algoritmalar ile geliştirilen algoritmanın (GWOMFO) doğruluk sonuçlarının 3 farklı veri kümesi üzerinde karşılaştırılması

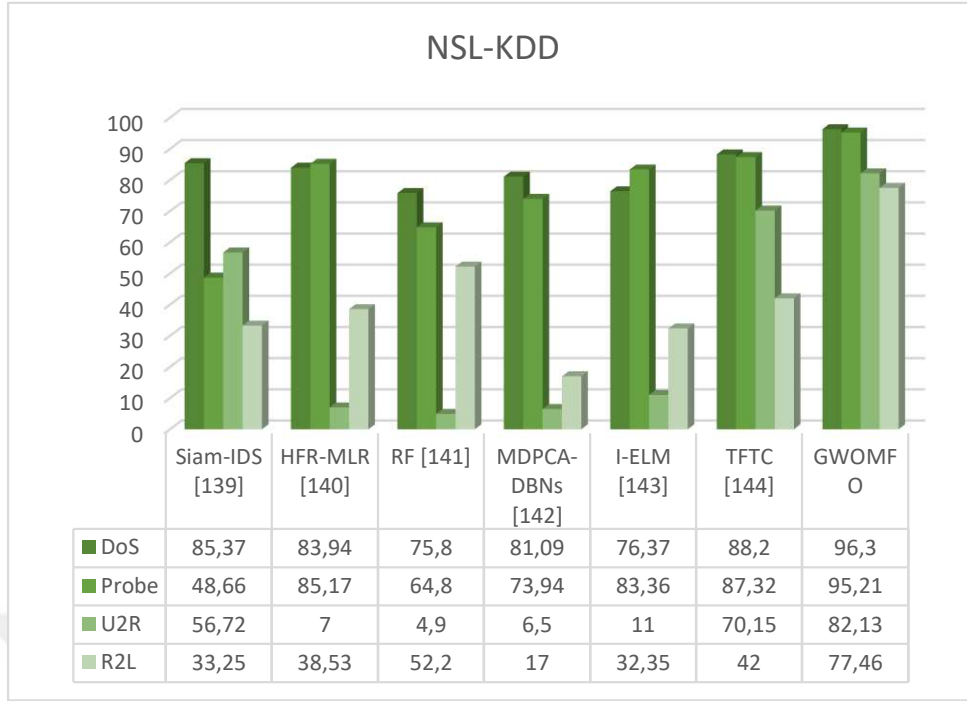
Doğruluk ölçümünde önerilen algoritmanın karşılaştırılan tüm algoritmalar arasında en başarılı algoritma olduğu gösterilmiştir. Diğer metrikler dikkate alınarak yapılan karşılaştırmanın sonuçları Çizelge 5.3'te gösterilmiştir.

Çizelge 5.3. Çeşitli sürü zekâsı tabanlı algoritmalar ile GWOMFO algoritmasının 3 farklı veri kümesi üzerinde karşılaştırılması

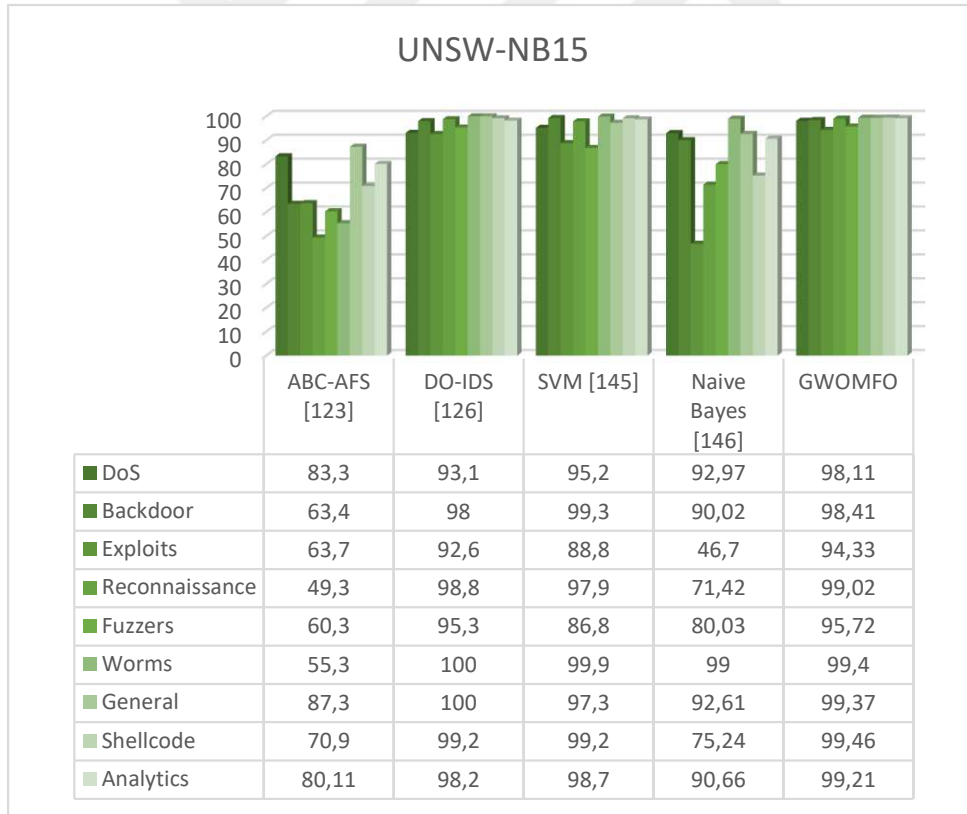
Veri Seti	Metrik	DE	PSO	MVO	JAYA	SSA	SCA	GWOMFO
NSL-KDD	Doğruluk	86.0	93.2	89.8	88.0	86.0	90.0	97.4*
	Kesinlik	88.8	94.1	88.0	90.8	87.9	90.8	97.7*
	Duyarlılık	88.4	93.7	87.5	92.3	86.7	92.0	96.5*
	F-ölçümü	88.6	93.9	88.0	91.5	87.3	91.4	97.2*
UNSW-NB15	Doğruluk	87.1	94.0	90.5	89.5	87.2	90.6	97.8*
	Kesinlik	89.3	94.3	90.2	89.7	88.0	91.5	97.4*
	Duyarlılık	89.1	94.6	91.3	90.3	88.5	91.6	97.2*
	F-ölçümü	89.2	94.4	90.7	90.0	88.2	91.5	97.3*
CIC IDS 2017	Doğruluk	88.0	94.2	91.3	89.7	87.9	91.4	97.7*
	Kesinlik	89.3	94.3	90.7	90.2	88.3	91.6	97.8*
	Duyarlılık	88.7	93.6	91.8	90.5	88.4	91.9	97.4*
	F-ölçümü	89.0	93.9	90.5	90.3	88.3	91.7	97.6*

*: En iyi Sonuç

Bu çalışmada, geliştirilen algoritmanın performansı saldırı tipine bağlı olarak incelenmiştir. Elde edilen sonuçlar yapılan diğer çalışmalarla karşılaştırılmıştır. Genel olarak, GWOMFO oldukça başarılı olduğunu kanıtlamış ve her tür saldırı için yüksek bir tespit oranı elde etmiştir. Sonuçlar Şekil 5.3, Şekil 5.4 ve Şekil 5.5'te sunulmuştur.



Şekil 5.3. NSL-KDD veri setinde TPR sonuçlarının karşılaştırılması



Şekil 5.4. UNSW-NB15 veri setinde TPR sonuçlarının karşılaştırılması



Şekil 5.5. CIC IDS 2017 veri setinde TPR sonuçlarının karşılaştırılması

Bu çalışmada geliştirilen hibrit algoritma, kamuya açık veri setleri NSL-KDD, UNSW-NB15 ve CIC IDS 2017 kullanılarak yapılan çalışmalarla karşılaştırılmış ve detaylı olarak incelenmiştir. Bu çalışmaların sonuçları doğruluk kriterleri kullanılarak karşılaştırılmıştır. Elde edilen sonuçlar Çizelge 5.4'te sunulmuştur.

Çizelge 5.4. GWOMFO algoritmasının NSL-KDD, UNSW-NB15 ve CIC IDS

2017 veri setleri kullanılarak yapılan çalışmalarla karşılaştırılması

Algoritma veya Model	Yazar	Veri Seti	Doğruluk
Quasi-optimal Algorithm	Nasr ve ark. [148]	NSL-KDD	0.933
Canonical correlation+ feature association impact scale	Jyothsna ve Prasad [149]	NSL-KDD	0.910
PCA and optimized SVM	Ikram ve ark. [150]	NSL-KDD	0.965
Deep Learning Approach	Niyaz ve ark. [151]	NSL-KDD	0.791
Random Forest	Aljawarneh ve ark. [152]	NSL-KDD	0.806
Naïve Bayes	Aljawarneh ve ark. [152]	NSL-KDD	0.765
Decision Tree	Aljawarneh ve ark. [152]	NSL-KDD	0.810
Information gain and RepTree	Belouch ve ark. [153]	UNSW-NB15	0.900
ABC and AFS algorithms	Hajisalem ve Babaie [123]	UNSW-NB15	0.980
Constrained-optimization-based extreme learning machines	Wang ve ark. [154]	UNSW-NB15	0.830
Skip-gram model classifier	Carrasco ve Sicilia [155]	UNSW-NB15	0.910
Hyper clique improved binary gravitational search algorithm as FS and SVM	Raman [156]	UNSW-NB15	0.960
Gradient boosted machine	Tama ve Rhee [157]	UNSW-NB15	0.930
Random forest as feature selection and k-nearest neighbors	Alrowaily ve ark. [158]	CIC IDS 2017	0.990
Negative selection algorithm and classifiers	Hosseini ve Seilani [159]	CIC IDS 2017	0.970
Random forest classifier	Bindra ve Sood [160]	CIC IDS 2017	0.960
LSTM, CNN AND FNN	Lee ve ark. [161]	CIC IDS 2017	0.980
Self-adaptive grasshopper optimization algorithm	Shukla [162]	CIC IDS 2017	0.990
LSTM	Kaur ve Singh [163]	CIC IDS 2017	0.990
BRS	Prasad ve ark. [128]	CIC IDS 2017	0.970
DBN	Elmasry ve ark. [164]	CIC IDS 2017	0.980
Önerilen Algoritma (GWOMFO)	Hasan ve ark. [165]	NSL-KDD	0.974*
		UNSW-NB15	0.983*
		CIC IDS 2017	0.992*

*: En iyi Sonuç

6. SONUÇ

Bu çalışmada ağ saldırıları ve tespiti konusu ele alınmış, bu problemin çözümünde yaygın olarak kullanılan veri setleri incelenmiş, literatürdeki çalışmalardan ve elde edilen sonuçlardan bahsedilmiştir. Çözüm için önerilen algoritmaların performansını artırmak için hibrit çözümlerin önemi vurgulanmıştır. Geliştirilen bir algoritmanın performansının ön değerlendirmesinde kıyaslama fonksiyonlarının kullanılmasının, algoritmanın performansı hakkında fikir verdiği gösterilmiştir.

Bu çalışmada; NSL-KDD, UNSW-NB15 ve CIC IDS 2017 veri setleri ile yeni bir hibrit algoritma geliştirilmiş ve test edilmiştir. Mevcut yöntemlerin çözüm kalitesini iyileştirmeye ve yeni yöntemler geliştirmeye gereksinim devam edecektir. Hibrit algoritmalar, algoritmanın çözüm kalitesini veya performansını iyileştirmek için kullanılabilir. Bu çalışmada geliştirilen algoritmada, GWO algoritmasının arama ajanlarının etkisi, MFO algoritması ile hibrit bir yapı kazandırılarak iyileştirilmiştir. GWO algoritması temel alınarak geliştirilen bu algoritmaya, GWO algoritmasının avlanma mekanizmasını geliştirmek için yeni bir mekanizma eklenmiştir. Her iterasyondan sonra arama yeteneğinin ve keşif aşamasının iyileştirildiği ve bunun da çözümü iyileştirdiği gözlemlenmiştir. Elde edilen sonuçlar, arama araçlarına bağlı olarak konum güncellemelerinin daha başarılı ve verimli olduğunu göstermektedir. Literatürdeki çalışmalardan farklı olarak bu çalışmada geliştirilen hibrit algoritma önce benchmark fonksiyonları ile test edilmiş, ardından ağ saldırıları için kullanılan veri setleri üzerinde test edilmiştir. Geliştirilen algoritma, 13 benchmark fonksiyonuna uygulanarak performans analizleri yapılmıştır. Bu çalışmada elde edilen sonuçlar birçok farklı çalışma ile karşılaştırılmış ve aşağıdaki sonuçlara varılmıştır:

- i. GWOMFO, diğer algoritmalarla kıyasla 13 kıyaslama işlevinin 12'sinde en iyi ortalama sonuçları elde etmiştir.
- ii. En iyi sınıflandırma doğruluğu %97,4 ile NSL-KDD veri setinde elde edilmiştir.
- iii. En iyi sınıflandırma doğruluğu %98,3 ile UNSW-NB15 veri setinde elde edilmiştir.

iv. En iyi sınıflandırma doğruluğu %99,2 ile CIC IDS 2017 veri setinde elde edilmiştir.

v. Genel olarak, GWOMFO iyi performans göstermiş ve tüm saldırı türleri için yüksek tespit oranları elde etmiştir.

Sonuçlar, literatürden farklı optimizasyon algoritmalarının sonuçları ile karşılaştırılmıştır. Geliştirilen hibrit modelin başarılı bir sonuç gösterdiği tespit edilmiştir. Elde edilen sonuçlar, GWOMFO algoritmasının karşılaştırılan algoritmalara oranla Benchmark fonksiyonlarında başarılı olduğunu kanıtlamıştır. Test edilen veri setlerinde elde edilen başarılı sonuçlar da algoritmanın başarısını desteklemektedir. Geliştirilen bu algoritmanın literatüre kazandırılmasıyla gerçek dünyaya uyarlanabilen ve çözülmesi zor problemlere yönelik tekniklerin yaygınlaştırılması hedeflenmiştir.

Ağ güvenliği her zaman önemli bir konu olmaya devam edecektir. Bunun nedeni, ağ savunmasının bir zorunluluk olmasıdır. Bu çalışmanın literatür incelemesi bölümünde yer alan [36], [37], [38], [39] gibi birçok yeni çalışma ile ağ güvenliği konusu farklı açılardan tartışılmaktadır. Bununla birlikte ağ güvenliği konusu geliştirilecek birçok yeni yöntemlerle ele alınmaya devam edecektir.. Bu çalışmaya ek olarak, gelecekte yapılabilecek çalışmaların içeriği aşağıda verilmiştir:

1. Bu çalışmada elde edilen başarı oranlarını daha da artırmak için çalışmalar yapılabilir. Bu amaçla literatürden geliştirilmiş çeşitli algoritmalar ile yeni bir hibrit yapı gerçekleştirilebilir. Ayrıca, çeşitli geliştirilmiş özellik çıkarma algoritmaları daha iyi performans sağlayabilir.
2. Bu hibrit yaklaşım, farklı veri setlerine uygulanabilir ve sonuçlar gözlemlenebilir. Özellik seçimi gibi çeşitli optimizasyon problemlerini de çözebilir. Ağ güvenliği altyapısını iyileştirmek için Derin Öğrenme; ağ tehditlerini anında ve daha yüksek doğrulukla tespit eden yeni nesil izinsiz giriş tespit sistemleri geliştirmek için kullanılabilir.
3. Ağ saldırılarının gerçek zamanlı tespit edilebilmesi için online saldırı tespit sistemi geliştirilmesine yönelik çalışmalar yapılabilir.

KAYNAKLAR

1. Resul, D. A. Ş., Bitikçi, B. (2020). Analysis of Different Types of Network Attacks on the GNS3 Platform. *Sakarya University Journal of Computer and Information Sciences*, 3(3), 210-230.
2. Summers, R. C. (1997). Secure computing: threats and safeguards. McGraw-Hill, Inc, p.140.
3. Baker, S., Filipiak, N., Timlin, K. (2011). In the dark: crucial industries confront cyberattacks McAfee annual critical infrastructure protection report (2. Ed.). Santa Clara, CA: The Center for Strategic and International Studies (CSIS), pp. 1-6.
4. Pehlivanoglu, M. K., Atay, R., Odabaş, D. E. (2019). İki Seviyeli Hibrit Makine Öğrenmesi Yöntemi ile Saldırı Tespiti. *Gazi Mühendislik Bilimleri Dergisi (GMBD)*, 5(3), 258-272.
5. Radoglou-Grammatikis, P. I., Sarigiannidis, P. G. (2019). Securing the smart grid: A comprehensive compilation of intrusion detection and prevention systems. *IEEE Access*, 7, 46595-46620.
6. Gupta, D., Gupta, V. (2016). Test suite prioritization using nature inspired meta-heuristic algorithms. In *Int. Conf. on Intelligent Systems Design and Applications ISDA 2016*, Springer, Cham, Porto, Portugal, pp. 216-226.
7. Mirjalili, S., Mirjalili, S. M. and Lewis, A. (2014). Grey wolf optimizer, *Advances in Engineering Software*, vol. 69, pp. 46-61, 2014.
8. Senel, F. A., Gökçe, F., Yüksel, A. S. and Yigit, T. (2019). A novel hybrid PSO–GWO algorithm for optimization problems. *Engineering with Computers*, vol. 35, pp. 1359-1373, 2019.
9. Raffaele, C., Marta, T., Giuseppina, P., Antonella, P., Fabio, F. Artificial Intelligence and Machine Learning Applications in Smart Production: Progress, Trends, and Directions, *MDPI Sustainability*, 2020.
10. Meng, T., Jing, X., Yan, Z. ve Pedrycz, W. (2020). A survey on machine learning for data fusion. *Information Fusion*, 57, 115-129.
11. Zhang, X. D. (2020). A matrix algebra approach to artificial intelligence (pp. 1-820). Springer.
12. Camacho, D. M., Collins, K. M., Powers, R. K., Costello, J. C. ve Collins, J. J. (2018). Next-generation machine learning for biological networks. *Cell*, 173(7), 1581-1592.
13. Bastani, H., Zhang, D. ve Zhang, H. (2020). Applied machine learning in operations management. Available at SSRN 3736466.
14. Baştanlar, Y. ve Özuysal, M. (2014). Introduction to machine learning. *miRNomics: microRNA biology and computational analysis. Methods in Molecular Biology (Methods and Protocols)*, 1107, 105-28.
15. Mohri, M., Rostamizadeh, A. ve Talwalkar, A. (2018). Foundations of machine learning. MIT press.
16. Paper D. (2020). Introduction to Scikit-Learn. In: *Hands-on Scikit-Learn for Machine Learning Applications*. Apress, Berkeley, CA. doi: https://doi.org/10.1007/978-1-4842-5373-1_1.

17. Mayer, D.G., Kinghorn, B.P., Archer, A.A., (2005), "Differential Evolution – An Easy and Efficient Evolutionary Algorithm for Model Optimisation", *Agricultural Systems*, 83, 315-328.
18. Gambella, C., Ghaddar, B. ve Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290(3), 807-828.
19. Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3), 1-21.
20. Fyfe, C., 2005, Artificial neural networks, In: *Do smart adaptive systems exist?*, Eds: Springer, p. 57-79.
21. Dong, C., Loy, C. C., He, K. ve Tang, X., 2014, Learning a deep convolutional network for image super-resolution, *European conference on computer vision*, 184-199.
22. Öztemel, E., 2006, Yapay sinir ağları, Papatya Yayıncılık, İstanbul, p. 13-113.
23. McCulloch, W. S. ve Pitts, W., 1943, A logical calculus of the ideas immanent in nervous activity, *The Bulletin of Mathematical Biophysics*, 5 (4), 115-133.
24. Keskenler, M. F. ve Keskenler, E. F., 2017, Geçmişten günümüze yapay sinir ağları ve tarihçesi, *Takvim-i Vekayi*, 5 (2), 8-18.
25. Dave, V. S. ve Dutta, K., 2014, Neural network based models for software effort estimation: A review, *Artificial Intelligence Review*, 42 (2), 295-307.
26. Hertz, J. A., 2018, *Introduction to the theory of neural computation*, CRC Press, p.
27. Herculano-Houzel, S., 2009, The human brain in numbers: a linearly scaled-up primate brain, *Frontiers in human neuroscience*, 3, 31.
28. Kayri, M., 2016, Predictive abilities of bayesian regularization and Levenberg-Marquardt algorithms in artificial neural networks: a comparative empirical study on social data, *Mathematical and Computational Applications*, 21 (2), 20.
29. Sutton, R. S. ve Barto, A. G., 2018, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, USA, p. 1-13.
30. Liu, Q. ve Wu, Y., 2012, Supervised learning, In: *Encyclopedia of the Sciences of Learning*, Eds: Seel, N. M., Springer, Boston, MA, p. 3243-3245.
31. Hughes, D. ve Correll, N., 2016, Distributed machine learning in materials that couple sensing, actuation, computation and communication [online], arXiv, <https://arxiv.org/abs/1606.03508v1> [Erişim Tarihi: 09 Kasım 2022].
32. Hossain, M. A., Rahman, M. M., Prodhan, U. K. ve Khan, M. F., 2013, Implementation of back-propagation neural network for isolated bangla speech recognition, *International Journal of Information Sciences and Techniques*, 3 (4), 1-9.
33. ANDERSON, Dave ve McNeill, George, "Artificial Neural Networks Technology", Rome Laboratory, A DACS State-of-the-Art Report, ELIN: A011, s:31,1992.

34. Blum C. and Roli A., (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Comput. Surv.*, 35, 268-308.
35. Hammouche, K., Diaf, M., Siarry, P., 2010, A comparative study of various metaheuristic techniques applied to the multilevel thresholding problem, *Engineering Applications of Artificial Intelligence*, 23 (5), 676-688.
36. El-Omari, N. K. T., 2020, Sea lion optimization algorithm for solving the maximum flow problem, *IJCSNS International Journal of Computer Science and Network Security*, 20 (8), 30-68.
37. Zitar, R. A., Al-Betar, M. A., Awadallah, M. A., Aba Doush, I. ve Assaleh, K., 2022, An intensive and comprehensive overview of jaya algorithm, its versions and applications, *Archives of Computational Methods in Engineering*, 29, 763-792.
38. Steven Vaughan-Nichols. (2018). The day computer security turned real: The Morris Worm turns 30. <https://www.zdnet.com/article/the-day-computer-security-turned-real-the-morris-worm-turns-30/>
39. Levy, J. (2020). Sophos 2021 Threat Report. 36. <https://www.sophos.com/en-us/medialibrary/pdfs/technical-papers/sophos-2021-threat-report.pdf>
40. Liao, H. J., Richard Lin, C. H., Lin, Y. C., & Tung, K. Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1), 16-24.
41. Hodo E., Bellekens X., Hamilton A., Tachtatzis C., Atkinson R., Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey, Cornell University, 1701.02145, 1-43, 2017.
42. Sen J., Mehtab S., Machine Learning Applications in Misuse and Anomaly Detection, Cornell University, 2009.06709, 1-21, 2020.
43. Herrero A, Corchado E., Mobile Hybrid Intrusion Detection, Springer, Berlin, Germany, 2011.
44. Bace R., Mell P., Intrusion Detection Systems, National Institute of Standards and Technology, 800-31, 1-51, 2001.
45. Butun I, Morgera SD, Sankar R (2014) A survey of intrusion detection systems in wireless sensor networks. *IEEE Communications Surveys & Tutorials* 16(1):266-282.
46. A. Alazab, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in 2012 international symposium on communications and information technologies (ISCIT), 2012, pp. 296-301.
47. Depren, O., Topallar, M., Anarim, E. ve Ciliz, M. K., 2005, An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, *Expert systems with Applications*, 29 (4), 713-722.
48. Prabhu, G. N., Jain, K., Lawande, N., Kumar, N., Zutshi, Y., Singh, R. ve Chinchole, J., 2014, Network intrusion detection system, *Int. J. Eng. Res. Appl*, 4 (4), 69-72.

49. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A. ve Rajarajan, M., 2013, A survey of intrusion detection techniques in cloud, *Journal of Network and Computer Applications*, 36 (1), 42-57.
50. Bace, R. ve Mell, P., 2001, NIST special publication on intrusion detection systems, BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.
51. Güven E. N. “Zeki Saldırı Tespit Sistemlerinin İncelenmesi, Tasarımı ve Gerçekleştirilmesi”, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, 2007.
52. S. E. Smaha, "Haystack: an intrusion detection system, " *Aerospace Computer Security Applications Conference*, Fourth, Orlando, FL, pp. 37-44, 1988.
53. Erol, M., “Saldırı Tespit Sistemlerinde İstatistiksel Anormallik Belirleme Kullanımı”. İTÜ Bilgisayar Mühendisliği Bölümü, İstanbul, 2005.
54. Ilgun K. “Ustat: A real-time intrusion detection system for Unix”, *Proceedings of the 1993 IEEE Symposium on Research in Security and Privacy*, Oakland, California, 16-28, (1993).
55. Joshua, S. Whitea, Thomas T. Fitzsimmons, Jeanna N. Matthewsc, “Quantitative Analysis of Intrusion Detection Systems: Snort and Suricata”, Wallace H. Coulter School of Engineering Department of Computer Science, 2011.
56. Leach J., Tedesco G., “Firestorm Network Intrusion Detection System”, Firestorm Documentation 2003.
57. Phpids. “PHP-Intrusion Detection System”. <https://github.com/PHPIDS/PHPIDS>(Erişim Tarihi: 12.10.2022).
58. NETIDS. “Intrusion detection system for .NET based on phpids”. <https://code.google.com/p/dotnetids/> (Erişim Tarihi: 14.10.2022).
59. Prelude.”Prelude Security Information & Event Management”. <http://www.prelude-siem.com/en> (Erişim Tarihi: 24.10.2022).
60. AIDE. “Advanced Intrusion Detection Environment”. <http://aide.sourceforge.net/> (Erişim Tarihi: 24.10.2022).
61. Samhain. “File integrity / host-based intrusion detection system”. <http://www.la-samhain.de/samhain/> (Erişim Tarihi: 20.10.2022).
62. Saleh, A. I., Talaat, F. M., Labib, L. M. (2019). A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. *Artificial Intelligence Review*, 51(3), 403-443.
63. Ghazali, A. J., Nuaimy, W., Al-Atabi, A., Jamaludin, I. (2015). Comparison of classification models for NSL-KDD data set for network anomaly detection. *Academic Journal of Science*, 4(1), 199-206.
64. Bhavsar, Y. B., Waghmare, K. C. (2013). Intrusion detection system using data mining technique: Support vector machine. *International Journal of Emerging Technology and Advanced Engineering*, 3(3), 581-586.
65. Singaravelan, S., Arun, R., Arunshunmugam, D., Joy, S. J. C., Murugan, D. (2020). Inner interruption discovery and defense system by using data mining. *Journal of King Saud University-Computer and Information Sciences*, 32(5), 592-598.

66. Revathi, S., Malathi, A. (2013). A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. *International Journal of Engineering Research & Technology (IJERT)*, 2(12), 1848-1853.
67. Ray, L. (2013). Determining the number of hidden neurons in a multi layer feed forward neural network. *Journal of Information Security Research*, 4(2), 63-70.
68. Elhag, S., Fernández, A., Alshomrani, S., Herrera, F. (2019). Evolutionary fuzzy systems: a case study for intrusion detection systems. In *Evolutionary and swarm intelligence algorithms* (pp. 169-190). Springer, Cham.
69. Atefi, K., Yahya, S., Dak, A. Y., Atefi, A. (2013). A hybrid intrusion detection system based on different machine learning algorithms. In *Proceedings of the 4th International Conference on Computing and Informatics, ICOCI* (pp. 312-320).
70. Kim, G., Lee, S., Kim, S. (2014). A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, 41(4), 1690-1700.
71. Gupta, M., Shrivastava, S. K. (2015). Intrusion detection system based on svm and bee colony. *International Journal of Computer Applications*, 111(10).
72. Pervez, M. S., Farid, D. M. (2014). Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs. In *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)* (pp. 1-6). IEEE.
73. Dhanabal, L., Shantharajah, S. P. (2015). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. *International journal of advanced research in computer and communication engineering*, 4(6), 446-452.
74. Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5, 21954-21961.
75. Alkasassbeh, M., Al-Naymat, G., Hassanat, A., Almseidin, M. (2016). Detecting distributed denial of service attacks using data mining techniques. *International Journal of Advanced Computer Science and Applications*, 7(1), 436-445.
76. Kushwaha, P., Buckchash, H. and Raman, B. (2017). Anomaly based intrusion detection using filter-based feature selection on KDD-CUP 99. In *IEEE Region 10 Conference TENCN*, Nov 2017, pp. 839-844.
77. Kim, J., Shin, N., Jo, S. Y. and Kim, S. H. (2017). Method of intrusion detection using deep neural network. In *Proc. IEEE Int. Conf. Big Data Smart Comput.*, Hong Kong, China, Feb. 2017, pp. 313-316.
78. Gautam, S. K. Om, H. (2017). Comparative Analysis of Classification Techniques in Network Based Intrusion Detection Systems. *Proceedings of the First International Conference on Intelligent Computing and Communication*, 2017, 591-601.
79. Mishra, A., Gupta, N., & Gupta, B. B. (2021). Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. *Telecommunication systems*, 77(1), 47-62.
80. Gupta, B. B., Li, K. C., Leung, V. C., Psannis, K. E., & Yamaguchi, S. (2021). Blockchain-assisted secure fine-grained searchable encryption for a cloud-based

81. healthcare cyber-physical system. *IEEE/CAA Journal of Automatica Sinica*, 8(12), 1877-1890.
82. 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.
83. Kumar, V., Das, A. K., & Sinha, D. (2021). UIDS: a unified intrusion detection system for IoT environment. *Evolutionary Intelligence*, 14(1), 47-59.
84. Mousavi, S. K., Ghaffari, A., Besharat, S., Afshari, H. (2021). Improving the security of internet of things using cryptographic algorithms: a case of smart irrigation systems. *Journal of Ambient Intelligence and Humanized Computing*, 12(2), 2033-2051.
85. Stergiou, C., Psannis, K. E., Gupta, B. B., & Ishibashi, Y. (2018). Security, privacy & efficiency of sustainable cloud computing for big data & IoT. *Sustainable Computing: Informatics and Systems*, 19, 174-184.
86. Mousavi, S. K., Ghaffari, A. (2021). Data cryptography in the Internet of Things using the artificial bee colony algorithm in a smart irrigation system. *Journal of Information Security and Applications*, 61, 102945.
87. Teng, S., Wu, N., Zhu, H., Teng, L., Zhang, W. (2017). SVM-DT-based adaptive and collaborative intrusion detection. *IEEE/CAA Journal of Automatica Sinica*, 5(1), 108-118.
88. Guo, C., Ping, Y., Liu, N., Luo, S. S. (2016). A two-level hybrid approach for intrusion detection. *Neurocomputing*, 214, 391-400.
89. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J., Alazab, A. (2020). Hybrid intrusion detection system based on the stacking ensemble of c5 decision tree classifier and one class support vector machine. *Electronics*, 9(1), 173.
90. Ahmim, A., Derdour, M., Ferrag, M. A. (2018). An intrusion detection system based on combining probability predictions of a tree of classifiers. *International Journal of Communication Systems*, 31(9), e3547
91. Al-Yaseen, W. L., Othman, Z. A., Nazri, M. Z. A. (2017). Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system. *Expert Systems with Applications*, 67, 296-303.
92. Kevric, J., Jukic, S., Subasi, A. (2017). An effective combining classifier approach using tree algorithms for network intrusion detection. *Neural Computing and Applications*, 28(1), 1051-1058.
93. Aslahi-Shahri, B. M., Rahmani, R., Chizari, M., Maralani, A., Eslami, M., Golkar, M. J., Ebrahimi, A. (2016). A hybrid method consisting of GA and SVM for intrusion detection system. *Neural computing and applications*, 27(6), 1669-1676.
94. Mahmood, M. S., Alnaish, Z. A. H., Al-Hadi, I. A. A. (2015). Hybrid intrusion detection system using artificial bee colony algorithm and multi-layer perceptron. *International Journal of Computer Science and Information Security*, 13(2), 1.
95. Elbasiony, R. M., Sallam, E. A., Eltobely, T. E., Fahmy, M. M. (2013). A hybrid network intrusion detection framework based on random forests and weighted k-means. *Ain Shams Engineering Journal*, 4(4), 753-762.

96. Panda, M., Abraham, A., Patra, M. R. (2012). A hybrid intelligent approach for network intrusion detection. *Procedia Engineering*, 30, 1-9.
97. Mohamad Tahir, H., Hasan, W., Md Said, A., Zakaria, N. H., Katuk, N., Kabir, N. F., ... & Yahya, N. I. (2015). Hybrid machine learning technique for intrusion detection system.
98. Gupta, M., Shrivastava, S. K. (2015). Intrusion detection system based on svm and bee colony. *International Journal of Computer Applications*, 111(10).
99. Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5, 21954-21961.
100. Almi'ani, M., Ghazleh, A. A., Al-Rahayfeh, A., Razaque, A. (2018). Intelligent intrusion detection system using clustered self organized map. In 2018 Fifth international conference on software defined systems (SDS) (pp. 138-144). IEEE.
101. Kamarudin, M. H., Maple, C., Watson, T., Safa, N. S. (2017). A logitboost-based algorithm for detecting known and unknown web attacks. *IEEE Access*, 5, 26190-26200.
102. Naoum, R. S., Abid, N. A., Al-Sultani, Z. N. (2012). An enhanced resilient backpropagation artificial neural network for intrusion detection system. *International Journal of Computer Science and Network Security (IJCSNS)*, 12(3), 11.
103. Lei, Y. (2017). Network anomaly traffic detection algorithm based on SVM. In 2017 International Conference on Robots & Intelligent System (ICRIS) (pp. 217-220). IEEE.
104. Price K. V.: Differential Evolution: a Fast and Simple Numerical Optimizer. In: Smith, M., Lee, M., Keller, J., Yen., J. (eds.): Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS. IEEE Press, New York (1996) 524-527.
105. Kennedy, J. and Eberhart, R., "Particle swarm optimization.", *Neural Networks Proceedings IEEE International Conference on*, Perth, 1942-1948 (1995).
106. Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications*, 27, 495-513.
107. R.V. Rao, Jaya: A Simple and New Optimization Algorithm For Solving Constrained and Unconstrained Optimization Problems, *International Journal of Industrial Engineering Computations*, 7 (2016) 19-34.
108. R.V. Rao, G. Waghmare, A new optimization algorithm for solving complex constrained design optimization problems, *Engineering Optimization* 49 (2017) 60-83.
109. Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M., Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, *Advances in Engineering Software*, 114, (2017), pp. 163-191.
110. S. Mirjalili, ``SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120-133, Mar. 2016.

- 111.Mirjalili, S., Mirjalili, S. M. and Lewis, A. (2014). Grey wolf optimizer, *Advances in Engineering Software*, vol. 69, pp. 46-61, 2014.
- 112.Mirjalili, S. (2015). The ant lion optimizer. *Advances in engineering software*, 83, 80-98.
- 113.Doğan, L., Yüzgeç, U. (2018). Robot Path Planning using Gray Wolf Optimizer. In *Proceedings-International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES'18)*, p. 70.
- 114.Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, 89, 228-249.
- 115.Li, Z., Zhou, Y., Zhang, S., Song, J. (2016). Lévy-flight moth-flame algorithm for function optimization and engineering design problems. *Mathematical Problems in Engineering*, 1-22, 2016.
- 116.Relan, N. G., Patil, D. R. (2015). Implementation of network intrusion detection system using variant of decision tree algorithm. In *2015 International Conference on Nascent Technologies in the Engineering Field (ICNTE)* (pp. 1-5). IEEE.
- 117.Chauhan, H., Kumar, V., Pundir, S., Pilli, E. S. (2013). A comparative study of classification techniques for intrusion detection. In *2013 International Symposium on Computational and Business Intelligence* (pp. 40-43). IEEE.
- 118.Bhattacharjee, P. S., Fujail, A. K. M., Begum, S. A. (2017). A comparison of intrusion detection by K-means and fuzzy C-means clustering algorithm over the NSL-KDD dataset. In *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICIC)* (pp. 1-6). IEEE.
- 119.Ullah, I., Mahmoud, Q. H. (2017). A filter-based feature selection model for anomaly-based intrusion detection systems. In *2017 IEEE International Conference on Big Data (Big Data)* (pp. 2151-2159). IEEE.
- 120.Saleh, A. I., Talaat, F. M., Labib, L. M. (2019). A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. *Artificial Intelligence Review*, 51(3), 403-443.
- 121.Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1-6). IEEE.
- 122.Moustafa, N., Slay, J. (2016). The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective*, 25(1-3), 18-31.
- 123.Khammassi, C., Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. *computers & security*, 70, 255-27

124. Hajisalem, V., Babaie, S. (2018). A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. *Computer Networks*, 136, 37-50.
125. Boulaiche, A., Adi, K. (2018). An auto-learning approach for network intrusion detection. *Telecommunication Systems*, 68(2), 277-294.
126. Ahmad, U., Asim, H., Hassan, M. T., Naseer, S. (2019). Analysis of classification techniques for intrusion detection. In 2019 International Conference on Innovative Computing (ICIC) (pp. 1-6). IEEE.
127. Ren, J., Guo, J., Qian, W., Yuan, H., Hao, X., Jingjing, H. (2019). Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms. *Security and communication networks*, 2019.
128. Nawir, M., Amir, A., Yaakob, N. and Lynn, O. B. (2018). Multi-classification of UNSW-NB15 dataset for. *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 15, pp. 5094-5104.
129. Prasad, M., Tripathi, S., and Dahal, K. (2020). An efficient feature selection based Bayesian and Rough set approach for intrusion detection. *Applied Soft Computing*, 87, 105980.
130. Panigrahi, R., Borah, S. (2018). A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. *International Journal of Engineering & Technology*, 7(3.24), 479-482.
131. D'hooge, L., Wauters, T., Volckaert, B., De Turck, F. (2019). Classification hardness for supervised learners on 20 years of intrusion detection data. *Ieee Access*, 7, 167455-167469.
132. Panwar, S. S., Negi, P. S., Panwar, L. S., Raiwani, Y. (2019). Implementation of machine learning algorithms on CICIDS-2017 dataset for intrusion detection using WEKA. *International Journal of Recent Technology and Engineering Regular Issue*, 8(3), 2195-2207.
133. D'hooge, L., Wauters, T., Volckaert, B., De Turck, F. (2019). In-depth comparative evaluation of supervised machine learning approaches for detection of cybersecurity threats. In 4th International Conference on Internet of Things, Big Data and Security (IoTBDS) (pp. 125-136).
134. Larose, D. T., Larose, C. D. (2014). *Discovering knowledge in data: an introduction to data mining* (Vol. 4). John Wiley & Sons, p. 411-412.
135. Omary, Z., Mtenzi, F. (2010). Machine learning approach to identifying the dataset threshold for the performance estimators in supervised learning. *International Journal for Infonomics (IJI)*, 3(3), 314-325.
136. H. Faris, R. Qaddoura, I. Aljarah, J.W. Bae, M. M. Fouad et al., "Evology, github," 2016. [Online]. Available: <https://github.com/7ossam81/Evolopy/blob/master/optimizers/>.
137. Atkinson, P.M., Tatnall, A.R.L., 1997. Introduction Neural networks in remote sensing. *Int. J. Remote Sens.* 18, 699–709. doi:10.1080/014311697218700
138. Del Frate, F., Pacifici, F., Schiavon, G., Solimini, C., 2007. Use of neural networks for automatic classification from high-resolution images. *IEEE Trans. Geosci. Remote Sens.* 45, 800–809. doi:10.1109/TGRS.2007.892009

139. Pacifici, F., Chini, M., Emery, W.J., 2009. A neural network approach using multi scale textural metrics from very high-resolution panchromatic imagery for urban land-use classification. *Remote Sens. Environ.* 113, 1276–1292. doi:10.1016/j.rse.2009.02.014.
140. Bedi, P., Gupta, N., Jindal, V. (2020). Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network. *Procedia Computer Science*, 171, 780–789. DOI 10.1016/j.procs.2020.04.085.
141. Kunang, Y. N., Nurmaini, S., Stiawan, D., Suprpto, B. Y. (2021). Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. *Journal of Information Security and Applications*, 58, 102804. DOI 10.1016/j.jisa.2021.102804.
142. Ma, C., Du, X., Cao, L. (2019). Analysis of multi-types of flow features based on hybrid neural network for improving network anomaly detection. *IEEE Access*, 7, 148363-148380. DOI 10.1109/Access.6287639.
143. Yang, Y., Zheng, K., Wu, C., Niu, X., Yang, Y. (2019). Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks. *Applied Sciences*, 9(2), 238. DOI 10.3390/app9020238.
144. Pajouh, H. H., Dastghaibfard, G., Hashemi, S. (2017). Two-tier network anomaly detection model: A machine learning approach. *Journal of Intelligent Information Systems*, 48(1), 61-74. DOI 10.1007/s10844-015-0388-x.
145. Pajouh, H. H., Javidan, R., Khayami, R., Dehghantanha, A., Choo, K. K. R. (2016). A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks. *IEEE Transactions on Emerging Topics in Computing*, 7(2), 314-323. DOI 10.1109/TETC.6245516.
146. Jing, D., Chen, H. B. (2019). SVM based network intrusion detection for the UNSW-NB15 dataset. 2019 IEEE 13th International Conference on ASIC (ASICON), pp. 1-4. Chongqing, China, IEEE.
147. Bagui, S., Kalaimannan, E., Bagui, S., Nandi, D., Pinto, A. (2019). Using machine learning techniques to identify rare cyber-attacks on the UNSW-NB15 dataset. *Security and Privacy*, 2(6), e91. DOI 10.1002/spy2.91.
148. Manimurugan, S., Manimegalai, P., Valsalan, P., Krishnadas, J., Narmatha, C. (2020). Intrusion detection in cloud environment using hybrid genetic algorithm and back propagation neural network. *International Journal of Communication Systems*.
149. Nasr, A. A., Ezz, M. M., Abdulmaged, M. Z. (2016). A learnable anomaly detection system using attributional rules. *International Journal of Computer Network and Information Security*, 8(11), 58. DOI 10.5815/ijcnis.
150. Jyothsna, V., Prasad, V. R. (2016). FCAAIS: Anomaly based network intrusion detection through feature correlation analysis and association impact scale. *ICT Express*, 2(3), 103-116. DOI 10.1016/j.icte.2016.08.003.
151. Ikram, S. T., Cherukuri, A. K. (2016). Improving accuracy of intrusion detection model using PCA and optimized SVM. *Journal of Computing and Information Technology*, 24(2), 133-148. DOI 10.20532/cit.

152. Javaid, A., Niyaz, Q., Sun, W., Alam, M. (2016). A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety*, 3(9), e2. DOI 10.4108/eai.24-5-2016.59124.
153. Aljawarneh, S., Yassein, M. B., Aljundi, M. (2019). An enhanced J48 classification algorithm for the anomaly intrusion detection systems. *Cluster Computing*, 22(5), 10549-10565. DOI 10.1007/s10586-017-1109-8.
154. Belouch, M., El Hadaj, S., Idhammad, M. (2017). A two-stage classifier approach using reptree algorithm for network intrusion detection. *International Journal of Advanced Computer Science and Applications*, 8(6), 389-394. DOI 10.14569/issn.2156-5570.
155. Wang, C. R., Xu, R. F., Lee, S. J., Lee, C. H. (2018). Network intrusion detection using equality constrained-optimization-based extreme learning machines. *Knowledge-Based Systems*, 147, 68-80. DOI 10.1016/j.knosys.2018.02.015.
156. Carrasco, R. S. M., Sicilia, M. A. (2018). Unsupervised intrusion detection through skip-gram models of network behavior. *Computers & Security*, 78, 187-197. DOI 10.1016/j.cose.2018.07.003.
157. Gauthama Raman, M. R., Somu, N., Jagarapu, S., Manghnani, T., Selvam, T. et al. (2020). An efficient intrusion detection technique based on support vector machine and improved binary gravitational search algorithm. *Artificial Intelligence Review*, 53(5), 3255–3286. DOI 10.1007/s10462-019-09762-z.
158. Tama, B. A., Rhee, K. H. (2019). An in-depth experimental study of anomaly detection using gradient boosted machine. *Neural Computing and Applications*, 31(4), 955-965. DOI 10.1007/s00521-017-3128-z.
159. Alrowaily, M., Alenezi, F., Lu, Z. (2019). Effectiveness of machine learning based intrusion detection systems. *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, pp. 277-288. Cham: Springer.
160. Hosseini, S., Seilani, H. (2021). Anomaly process detection using negative selection algorithm and classification techniques. *Evolving Systems*, 12(3), 769-778. DOI 10.1007/s12530-019-09317-1.
161. Bindra, N., Sood, M. (2019). Detecting DDoS attacks using machine learning techniques and contemporary intrusion detection dataset. *Automatic Control and Computer Sciences*, 53(5), 419-428. DOI 10.3103/S0146411619050043.
162. Lee, J., Kim, J., Kim, I., Han, K. (2019). Cyber threat detection based on artificial neural networks using event profiles. *IEEE Access*, 7, 165607-165626. DOI 10.1109/Access.6287639.
163. Shukla, A. K. (2021). Detection of anomaly intrusion utilizing self-adaptive grasshopper optimization algorithm. *Neural Computing and Applications*, 33(13), 7541-7561. DOI 10.1007/s00521-020-05500-7.
164. Kaur, S., Singh, M. (2020). Hybrid intrusion detection and signature generation using deep recurrent neural networks. *Neural Computing & Applications*, 32(12). DOI 10.1007/s00521-019-04187-9.

- 165.Elmasry, W., Akbulut, A., Zaim, A. H. (2020). Evolving deep learning architectures for network intrusion detection using a double PSO metaheuristic. *Computer Networks*, 168, 107042. DOI 10.1016/j.comnet.2019.107042.
- 166.Dalmaz, H., Erdal, E., Unver, H. M. (2023). A new hybrid approach using gwo and mfo algorithms to detect network attack. *CMES-Computer Modeling in Engineering & Sciences*, 136, pp. 1277-1314. DOI 10.32604/cmes.2023.025212.



EK-1 NSL-KDD Veri Seti Örneği

NSL-KDD	
Özellik Adı	Örnek Veri
Duration	0
Protocol Type	tcp
Service	ftp_data
Flag	SF
Src Bytes	491
Dst Bytes	0
Land	0
Wrong Fragment	0
Urgent	0
Hot	0
Num Failed Logins	0
Logged in	0
Num compromised	0
Root shell	0
Su attempted	0
Num root	0
Num file creations	0
Num shells	0
Num access files	0
Num outbound cmds	0
Is host login	0
Is guest login	0
Count	2
Srv count	2
Serror rate	0.00
Srv serror rate	0.00
Rerror rate	0.00
Srv rerror rate	0.00
Same srv rate	1.00
Diff srv rate	0.00
Srv diff host rate	0.00
Dst host count	150
Dst host srv count	25
Dst host same srv rate	0.17
Dst host diff srv rate	0.03
Dst host same src port rate	0.17
Dst host srv diff host rate	0.00
Dst host serror rate	0.00

Dst host srv serror rate	0.00
Dst host rerror rate	0.05
Dst host srv rerror rate	0.00
Label	normal



EK-2 UNSW-NB15 Veri Seti Örneği

UNSW-NB15	
Özellik Adı	Örnek Veri
Srcip	175.45.176.2
Sport	23357
Dstip	149.171.126.16
Dsport	80
Proto	tcp
State	FIN
Dur	0.240139
Sbytes	918
Dbytes	25552
Sttl	62
Dttl	252
Sloss	2
Dloss	10
Service	http
Sload	2.805.042.188
Dload	8.157.941.875
Spkts	12
Dpkts	24
Swin	255
Dwin	255
Stcpb	1708297952
Dtcpb	1939490744
Smeansz	77
Dmeansz	1065
Trans depth	1
Res bdy len	12026
Sjit	1.170.481.668
Djit	114.438.336
Stime	1421927416
Ltime	1421927416
Sintpkt	21.830.818
Dintpkt	9.570.304
Tcprtt	0.051475
Synack	0.006528
Ackdat	0.044947
Is sm ips ports	0
Ct state ttl	1
Ct flw http mthd	1
Is ftp login	0
Ct ftp cmd	0

Ct srv src	3
Ct srv dst	2
Ct dst ltm	2
Ct src ltm	1
Ct src dport ltm	1
Ct dst sport ltm	1
Ct dst src ltm	1
Attack cat	Exploits
Label	1



EK-3 CIC IDS 2017 Veri Seti Örneği

CIC IDS 2017	
Özellik Adı	Örnek Veri
Flow ID	192.168.10.14-209.48.71.168-49459-80-6
Source IP	192.168.10.14
Source Port	49459
Destination IP	209.48.71.168
Destination Port	80
Protocol	6
Time stamp	5/7/2017 8:42
Flow Duration	38308
Total Fwd Packets	1
Total Backward Packets	1
Total Length of Fwd Pck	6
Total Length of Bwd Pck	6
Fwd Packet Length Max	6
Fwd Packet Length Min	6
Fwd Pck Length Mean	6
Fwd Packet Length Std	0
Bwd Packet Length Max	6
Bwd Packet Length Min	6
Bwd Packet Length Mean	6
Bwd Packet Length Std	0
Flow Bytes/s	313.250.496
Flow Packets/s	52.208.416
Flow IAT Mean	38308
Flow IAT Std	0
Flow IAT Max	38308
Flow IAT Min	38308
Flow IAT Total	0
Fwd IAT Mean	0
Fwd IAT Std	0
Fwd IAT Max	0
Fwd IAT Min	0
Bwd IAT Total	0
Bwd IAT Mean	0
Bwd IAT Std	0
Bwd IAT Max	0
Bwd IAT Min	0
Fwd PSH Flags	0
Bwd PSH Flags	0
Fwd URG Flags	0
Bwd URG Flags	0

Fwd Header Lenth	20
Bwd Header Lenth	20
Fwd Packets/s	26.104.208
Bwd Packets/s	26.104.208
Min Packet Length	6
Max Packet Length	6
Packet Length Mean	6
Packet Length Std	0
Packet Len Variance	0
FIN Flag Count	0
SYN Flag Count	0
RST Flag Count	0
PSH Flag Count	0
ACK Flag Count	1
Urg Flag Count	1
CWE Flag Count	0
ECE Flag Count	0
Down/Up Ratio	1
Average Packet Size	9
Avg Fwd Segment Size	6
Avg Bwd Segment Size	6
Fwd Avg Bytes/Bulk	20
Fwd Avg Packets/Bulk	0
Fwd Avg Bulk Rate	0
Bwd Avg Bytes/Bulk	0
Bwd Avg Packets/Bulk	0
Bwd Avg Bulk Rate	0
Subflow Fwd Packets	0
Subflow Fwd Bytes	1
Subflow Bwd Packets	6
Subflow Bwd Bytes	1
Init Win bytes fwd	6
Act data pkt fwd	255
Min seg size fwd	946
Active Mean	0
Active Std	20
Active Max	0
Active Min	0
Idle Mean	0
Idle Packet	0
Idle Std	0
Idle Max	0
Idle Min	0
Label	0