T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAZILIM-TANIMLI AĞLAR İÇİN UYARLANABİLİR BİR SALDIRI TESPİT VE ÖNLEME SİSTEMİ TASARIMI

DOKTORA TEZİ

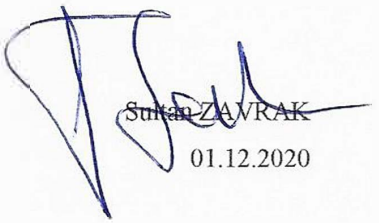
Sultan ZAVRAK

Enstitü Anabilim Dalı	**:**	**BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Murat İSKEFİYELİ**

Aralık 2020

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.



Sultan ZAVRAK

01.12.2020

TEŞEKKÜR

Doktora eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğretim Üyesi Murat İSKEFİYELİ'ye teşekkürlerimi sunarım.

Bugünlere gelmemi sağlayan ve her zaman yanımda olan aileme ve dostlarıma da destekleri için teşekkür ederim.

Tez çalışmasını, BİDEB 2211-C Öncelikli Alanlara Yönelik Yurt İçi Doktora Burs Programı kapsamında destekleyen TÜBİTAK'a teşekkür ederim.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

AE	: Otokodlayıcı
ANN	: Yapay Sinir Ağları
API	: Uygulama Programlama Arayüzü
AUC	: Alıcı İşletim Karakteristiği Altındaki Alan
BBNN	: Blok Tabanlı Sinir Ağı
DDoS	: Dağıtık Hizmet Engelleme
DNN	: Derin Sinir Ağı
DoS	: Hizmet Engelleme
DT	: Karar Ağacı
EncDecAD	: Kodlayıcı Kod Çözücü Anomali Algılayıcı
FN	: Yanlış Negatif
FP	: Yanlış Positif
FPR	: Yanlış Positiflik Oranı
FTP	: Dosya Transfer Protokolü
IDS	: Saldırı Tespit Sistemi
IPS	: Saldırı Önleme Sistemi
KNN	: K En Yakın Komşu
LSTM	: Uzun Kısa Süreli Bellek
MLP	: Çok Katmanlı Algılayıcı
OCSVM	: Tek Sınıflı Destek Vektör Makinesi
ONF	: Açık Ağ Vakfı
OSI	: Açık Sistemler Arabağlantısı
OVS	: Open vSwitch
PCA	: Temel Bileşen Analizi
RE	: Rekonstrüksiyon Hatası
ReLU	: Doğrultulmuş Doğrusal Birim

REST : Temsili Durum Transferi
RNN : Tekrarcı Sinir Ağı
ROC : Alıcı İşletim Karakteristiği
RP : Rekonstrüksiyon Olasılığı
SAnDet : Yazılım Tanımlı Ağlar için Anomali Tespit Sistemi
SDN : Yazılım Tanımlı Ağ
SOHO : Küçük Ofis Ev Ofis
SOM : Özdüzenlemeli AĞ
SQL : Yapısal Sorgu Dili
SSH : Güvenli Kabuk
SSL : Yarı Denetimli Öğrenme
SVM : Destek Vektör Makinesi
TN : Yanlış Negatif
TP : Gerçek Positif
TPR : Gerçek Pozitiflik Oranı
VAE : Varyasyonel Otokodlayıcı
XSS : Siteler Arası Betik Çalıştırma

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Akış anomalisi algılama, saldırı algılama, derin öğrenme, varyasyonel otokodlayıcı, yarı denetimli öğrenme, yazılım-tanımlı ağlar, zaman serisi anomali tespiti

Ağ trafiğindeki hızlı artış son zamanlarda az miktarda trafik verisini işleyen akış tabanlı saldırı algılama sistemlerinin önemine yol açmıştır Ayrıca bilinmeyen saldırıları tespit edebilen anomali tabanlı yöntemler de bu sistemlere entegre edilmiştir. Bu çalışmada, yarı denetimli öğrenme yaklaşımı ile denetimsiz derin öğrenme yöntemleri kullanılarak akış tabanlı verilerden anormal ağ trafiğinin (veya izinsiz girişlerin) saptanması üzerinde durulmaktadır. Daha açıkça belirtmek gerekirse, akış özellikleri kullanılarak bilinmeyen saldırıları tanımlamak için Otokodlayıcı ve Variational Otokodlayıcı yöntemleri kullanılmıştır. Yapılan deneylerde, tipik ve farklı saldırı türleri de dahil olmak üzere ağ trafiği verilerinden çıkarılan akış tabanlı özellikler kullanılmıştır. Bu yöntemlerin Alıcı İşletim Karakteristiği (ROC) ve ROC eğrisi altındaki alan ölçütlerine göre sonuçları hesaplandı ve Tek Sınıf Destek Vektör Makinesi ile karşılaştırıldı. ROC eğrileri çeşitli eşik değerlerinde yöntemlerin performansını analiz etmek için ayrıntılı olarak incelenmiştir. Deneysel sonuçlar, Varyasyonel Otokodlayıcı'nın çoğunlukla Otokodlayıcı ve Tek Sınıflı Destek Vektör Makinesi'nden daha iyi performans sergilediğini göstermektedir. Çalışmanın ikinci kısmında ise SDN mimarisinin sunuduğu imkanlardan faydalanarak bir anomali tabanlı saldırı tespiti yapabilen SAnDet mimarisi sunulmakta ve bunun Floodlight denenetleyici yazılımında uygulaması yapılmaktadır. İstatistik toplayıcı, anomali algılayıcı ve anomali önleme şeklide üç ana modülden oluşan bu sistemin ayrıntılı anlatımı yapılmaktadır. Daha açıkça belirtmek gerekirse, OpenFlow anahtarlardan toplanan akış özellikleri kullanılarak bilinmeyen saldırıları tanımlamak için Otokodlayıcının özel bir çeşidi olan RNN ve özellikle bir veri serisi girdi olarak verildiğinde başarılı sonuçlar üretebilen LSTM ağların özel bir çeşidi olan EncDecAD yöntemleri kullanılmıştır. Deneylerde, farklı saldırı türleri de dahil olmak üzere ağ trafik verilerinden çıkartılan akış tabanlı özellikler, zaman serisi olarak modellere girdi olarak verilir. ROC ve AUC ölçütlerine göre yöntemlerin sonuçları hesaplanır. ROC eğrileri çeşitli eşik değerlerinde yöntemlerin performansını analiz etmek için ayrıntılı olarak incelenir. Deneysel sonuçlar, EncDecAD'in RNN'den ve literatürde önerilen yöntemlerden daha iyi performans sergilediğini göstermektedir.

DESIGN OF AN ADAPTIVE INTRUSION DETECTION AND PREVENTION SYSTEM FOR SOFTWARE-DEFINED NETWORKS

SUMMARY

Keywords: Flow anomaly detection, intrusion detection, deep learning, variational autoencoder, semi-supervised learning, software-defined networks, time series anomaly detection

The rapid increase in network traffic has recently led to the importance of flow-based intrusion detection systems processing a small amount of traffic data Furthermore, anomaly-based methods, which can identify unknown attacks are also integrated into these systems. In this study, the focus is concentrated on the detection of anomalous network traffic (or intrusions) from flow-based data using unsupervised deep learning methods with semi-supervised learning approach. More specifically, Autoencoder and Variational Autoencoder methods were employed to identify unknown attacks using flow features. In the experiments carried out, the flow-based features extracted out of network traffic data, including typical and different types of attacks, were used. The Receiver Operating Characteristics (ROC) and the area under ROC curve, resulting from these methods were calculated and compared with One-Class Support Vector Machine. The ROC curves were examined in detail to analyze the performance of the methods in various threshold values. The experimental results show that Variational Autoencoder performs, for the most part, better than Autoencoder and One-Class Support Vector Machine. In the second part of the study, SAnDet architecture, which can do an anomaly-based intrusion detection by taking advantage of the capabilities offered by SDN architecture, is presented and implemented in Floodlight controller. A detailed description of this system which consists of three main modules which are statistics collector, anomaly detector and anomaly prevention is given. More specifically, RNN which is a special variant of the autoencoder, and EncDecAD methods which a special type of LSTM networks that can produce successful results especially in given data series, were used to identify unknown attacks using flow features collected from OpenFlow switches. In experiments, flow-based features extracted from network traffic data including different types of attacks, are given as input into models as time series. The results of the methods are calculated according to the ROC and AUC metrics. ROC curves have been examined in detail to analyze the performance of methods at various threshold values. Experimental results show that EncDecAD outperforms RNN and methods proposed in the literature.

BÖLÜM 1. GİRİŞ

Siber saldırılar (ağ saldırıları); bilgisayar ağlarında var olan bilgi ve hizmeti aksatan, engelleyen veya yok eden bir takım kötücül aktiviteler olarak tanımlanır. IDPS (Saldırı Tespit ve Önleme Sistemi – Intrusion Detection and Prevention System) sistemleri bu saldırıları ağ düzeyinde gözetleme yaparak tespit etme ve uygun müdahaleyi yapma işlevini üstlenen sistemlerdir. Saldırı tespitinin temel prensibi, izinsiz aktivitelerin belirgin derecede normal olanlardan farklı olması ve bu yüzden tespit edilebilir olması varsayımına dayalıdır. Literatürde birçok saldırı tespit yaklaşımı yer almasının yanı sıra bu günlerde araştırmacılar, bilinen saldırıların yanı sıra bilinmeyen saldırıları da tespit edebildiklerinden dolayı, daha çok anomali-tabanlı ağ saldırı tespiti üzerinde yoğunlaşmaktadır [1]. Geleneksel ağlarda, ağ çekirdeğindeki anomalileri tespit etmek için son birkaç yılda birçok anomali-tabanlı IDS (Saldırı Tespit Sistemi - Intrusion Detection System) önerilmiştir. Fakat bunların ağ çekirdeğine dağıtımında iki önemli problemle karşılaşılmaktadır [2]:

a. Düşük tespit oranları: Bazı sistemler yüksek tespit oranlarına sahip olsalar da çok fazla yanlış ihbar (false positive) ürettiklerinden pratik kullanımları uygun olamamaktadır.
b. Ağ çekirdeğinde IDS algoritmalarının hat hızında çalıştırılamaması: Paket ve akış örnekleme bu problemi aşmak için kullanılmakta fakat örnekleme yapmak önemli trafik özelliklerini bozarak anomali tespit doğruluğunu azaltmaktadır.

SDN (Software-Defined Networking), şu anda aktif olarak kullanılan geleneksel ağ altyapısının sınırlandırmalarını değiştirmeyi vaat eden yeni bir ağ paradigmasıdır [3]. Bu paradigma, yönlendirici (router) ve anahtarlar (switch) gibi trafik iletimi (forwarding) yapan cihazlardan ağın kontrol mantığını (control logic) alarak dikey bütünleşmeyi ayırarak kontrol mantığını mantıksal olarak merkezileştirilmiş

denetleyiciye (veya ağ işletim sistemine) yerleştirir [4]. Bu nedenle SDN, tüm ağ altyapısı boyunca veri akışlarının yönlendirilmesi için tek bir kontrol noktası belirler. Bu nedenle, SDN mimarisi, yüksek düzeyde reaktif (hassas) bir güvenlik izleme, analiz ve yanıt sistemi oluşturarak ağ güvenliğini artırmak için kullanılabilir [5], [6].

Ağ güvenliği açısından bakıldığında SDN, tüm ağ altyapısındaki veri akışlarının yönlendirilmesi için tek bir kontrol noktası tanımlar. Üst seviye reaktif (duyarlı) güvenlik gözetimi, analizi ve tepki sisteminin hazırlanmasıyla ağ güvenliğini artırmak için SDN mimarisinden istifade edilebilir [5], [6]. Merkezi denetleyici, bu sistemin temel bileşenidir. SDN paradigmasının değiştirilmiş bir versiyonu olan OpenFlow protokolü kullanılarak uygulanan ağ güvenliği uygulamaları ile, basitçe izin vermek veya engellemekten ziyade karmaşık mantığı işleyebilen akışları uygulamak da mümkündür [7]. Saldırı tespiti ve saldırı önleme veya azaltma algoritmaları OpenFlow güvenlik uygulamaları olarak uygulanabilir.

Ağ trafiği analizi veya anomali algılama yöntemleri, merkezi denetleyiciye iletilen güvenlikle ilgili verileri düzenli olarak üretir. Uygulamalar, tüm ağdan toplanan bu geri bildirimi analiz etmek ve ilişkilendirmek için denetleyici üzerinde çalıştırılabilir. Elde edilen analizin sonuçlarına bağlı olarak, yeni veya güncellenmiş güvenlik politikaları, akış kuralları şeklinde ağ bileşenlerine (örneğin; yönlendirici ve anahtar) dağıtılabilir. Bu birleşik yaklaşım, ağa yönelik güvenlik tehditlerinin kontrolünü etkili bir şekilde hızlandırabilir ve bunlardan korunabilir [5]. Burada önemli olan nokta, bu uygulamaların uygulanmasının genellikle daha kısa sürmesi ve geleneksel ağlara göre daha etkili olmasıdır.

Bu tez çalışması iki önemli konuda literatüre katkı sunmaktadır:

- Yarı denetimli öğrenme yaklaşımı ile denetimsiz derin öğrenme yöntemleri kullanılarak akış tabanlı verilerden anormal ağ trafiğinin (veya izinsiz girişlerin) saptanması üzerinde durulmaktadır. Daha açıkça belirtmek gerekirse, akış özellikleri kullanılarak bilinmeyen saldırıları tanımlamak için Otokodlayıcı ve Varyasyonel Otokodlayıcı yöntemleri kullanılmıştır. Yapılan

deneylerde, tipik ve farklı saldırı türleri de dahil olmak üzere ağ trafiği verilerinden çıkarılan akış tabanlı özellikler kullanılmıştır. Alıcı İşletim Karakteristiği (ROC) ve ROC eğrisi altındaki alan, bu yöntemlerden kaynaklanan hesaplandı ve Tek Sınıf Destek Vektör Makinesi ile karşılaştırıldı. ROC eğrileri çeşitli eşik değerlerinde yöntemlerin performansını analiz etmek için ayrıntılı olarak incelenmiştir. Deneysel sonuçlar, Varyasyonel Otokodlayıcı'nın çoğunlukla Otokodlayıcı ve Tek Sınıflı Destek Vektör Makinesi'nden daha iyi performans sergilediğini göstermektedir.

- SDN mimarisinin sunuduğu imkanlardan faydalanarak bir anomali tabanlı saldırı veya ihlal tespiti yapabilen SAnDet mimarisi sunulmakta ve bunun Floodlight denenetleyici yazılımında uygulaması yapılmaktadır. İstatistik toplayıcı, anomali algılayıcı ve anomali önleme şeklide üç ana modülden oluşan önerilen bu sistemin ayrıntılı anlatımı yapılmaktadır. Daha açıkça belirtmek gerekirse, OpenFlow anahtarlardan toplanan akış özellikleri kullanılarak bilinmeyen saldırıları tanımlamak için Otokodlayıcının özel bir çeşidi olan Tekrarcı Sinir Ağı (Replicator Neural Network - RNN) ve özellikle bir veri serisi girdi olarak verildiğinde başarılı sonuçlar üretebilen LSTM ağların özel bir çeşidi olan EncDecAD yöntemleri kullanılmıştır. Yapılan deneylerde, farklı saldırı türleri de dahil olmak üzere ağ trafiği verilerinden çıkarılan akış tabanlı özellikler bir zaman serisi olarak modellere girdi olarak verilmiştir. Bu yöntemlerin Alıcı İşletim Karakteristiği (ROC) ve AUC sonuçları hesaplandı. ROC eğrileri çeşitli eşik değerlerinde yöntemlerin performansını analiz etmek için ayrıntılı olarak incelenmiştir. Deneysel sonuçlar, EncDecAD'nin RNN'den daha iyi performans sergilediğini ve doğruluk ölçütüne göre literatürde önerilen diğer yöntemlerden daha iyi olduğunu göstermektedir.

Tezin organizasyonu şu şekilde yapılmıştır. İkinci bölümde, ağ akış özelliklerinden AE, VAE ve OCSVM yöntemleri kullanılarak anomali tabanlı saldırı tespiti yapılmakta ve ROC eğrileri ve AUC sonuçları üzerinde performans değerlendirilmesi yapılmaktadır. Üçüncü bölümde ise bir SDN denetleyici uygulaması olarak geliştirilen

SAnDet anomali tespit sisteminin tasarımıyla birlikte uygulanması anlatılmakta ve sistemde kullanılan ihlal tespit yöntemlerinin performans değerlendirilmesi yapılmaktadır. Sonuç tespitleri ise son bölümde açıklanmaktadır.

BÖLÜM 2. AĞ AKIŞ ÖZELLİKLERİNDEN VARYASYONEL OTOKODLAYICI KULLANILARAK ANOMALİ TABANLI İHLAL TESPİTİ

İnternetin hızla büyümesine paralel olarak ağlara ve bilgisayar sistemlerine yönelik siber saldırılar da süratle artmaktadır. Saldırı Tespit Sistemleri (Intrusion Detection Systems - IDS) bu saldırılara karşı bir önlem olarak ağ sistemlerine yerleştirilir. IDS'ler artan ağ saldırılarına karşı koymada hatırı sayılır derecede bir önem kazanmış olsalar da yük (payload) tabanlı IDS'ler, yüksek ağ hızı ve ağ trafiğindeki artış nedeniyle ölçeklenebilirlikten yoksundur. Sonuç olarak, akış tabanlı (flow-based) saldırı algılama yaklaşımları son zamanlarda IDS için potansiyel bir aday olmuştur. Akış tabanlı IDS'ler, iki nedenden ötürü derin paket incelemesine dayanan geleneksel IDS'lere tercih edilir. Bunlardan birincisi fark edilir derecede daha az miktarda veri işlemeleridir. İkincisi neden ise akış verilerinin standart protokolleri (Cisco NetFlow gibi) destekleyen ağ yönlendirme cihazlarından ilave bir yazılım yüklemeden toplanabilmesi ve bu işlemin ağdaki herhangi bir bilgisayardan kolayca yapılabilmesidir [8]. Saldırı tespit yaklaşımları temel olarak iki kategoriye ayrılır [8]: kötüye kullanım (misuse - suistimal) tabanlı ve anomali tabanlı. Kötüye kullanıma tabanlı yöntemler, evvelce bilinen saldırıların imzalarının karşılaştırılması şekliyle çalışır. Bilinen saldırılara karşı başarı oranı yüksek olan bu teknikler, bilinmeyen saldırıların tespitinde yetersiz kalır. Bunun aksine, anomali tabanlı saldırı tespit yöntemleri bilinmeyen veya sıfırıncı gün saldırılarını tanımlama yeteneğine sahiptir. Ağın normal davranışındaki aşırı değişiklik nedeniyle, normal davranışın tam tanımını elde etmek oldukça zor olduğundan anomali tabanlı tekniklerin yanlış alarmları diğerlerinden daha yüksektir.

Derin öğrenme (deep learning) yöntemlerinin metin sınıflandırma, nesne tanıma ve görüntü sınıflandırma gibi araştırma alanlarında karşılaşılan birçok problemi son

zamanlarda başarıyla çözebildiği görüldüğü için ağ saldırı tespit sistemlerinde de derin öğrenme yöntemleri uygulanmıştır. Bu aktif araştırma alanında, derin öğrenme yaklaşımlarını kullanan çalışmalar çoğunlukla boyut indirgeme (dimensionality reduction) [9][10][11][12][13] ve anomali tabanlı saldırı tespitine [14][15][16][17] odaklanılır. Paket tabanlı özellikleri de içeren KDDCUP99 veri seti [18], çalışmalardaki yöntemlerin performanslarını değerlendirmek için kullanılır [13][19]. KDDCUP99'un gözden geçirilmiş bir versiyonu olan NSL-KDD [20] veri seti de literatürde yer alan birçok çalışmada [10][11][13][14][15] önerilen yöntemlerin performanslarını değerlendirmek için kullanılır. Bu çalışmaların başlıca dezavantajları şu şekilde sıralanabilir: a) Bu çalışmalar, içerik tabanlı özelliklerden ağdaki ihlallerin veya saldırıların tespitine odaklanmaktadır. b) Kullanılan veri seti çok eski veya zaman aşımına uğramış olmasının yanı sıra gerçek ağ trafiğini yansıtmamaktadır [21].

Akış tabanlı veriler payload (yük) tabanlı verilere kıyasla ağ trafiği ile ilgili daha az bilgi içerdiğinden hem bilinen hem de bilinmeyen saldırıları tespit etmek çok daha zordur. Bu çalışmanın gayesi derin öğrenme yöntemlerinden faydalanılarak akışların istatistiksel özelliklerini de içeren akış tabanlı verilerden anormal ağ trafiğini (veya ihlalleri) tespit etmektir. Ayrıca, Otokodlayıcı (Autoencoder - AE) ve Varyasyonel Otokodlayıcı (Variational Autoencoder - VAE) yöntemlerinden yararlanılarak bilinmeyen saldırılar tespit edilir. Bilinmeyen saldırılardan maksat saldırı içeren ağ trafiğinden çıkarılan akış özellikleri (flow features) eğitim aşamasında kullanılmadığı da anlaşılabilir.

Başlıca katkıları şu şekilde özetlenlenebilir:

- Bu çalışma, anomali tabanlı yaklaşımla akış tabanlı özellikler kullanılarak ağ saldırılarının tespitine odaklanmaktadır.

- Denetimsiz (unsupervised) derin öğrenme yöntemleri olan AE ve VAE, OCSVM ile birlikte anomali detektörleri olarak kullanılır ve yarı denetimli (semi-supervised) öğrenme tarzında eğitilir. Buna ek olarak, bu çalışma akış

tabanlı verileri kullanarak izinsiz girişleri (ihlalleri) tespit etmek için derin öğrenme yöntemleri kullandığından yukarıda bahsedilen önceki çalışmalardan farklıdır.

- ROC ve AUC sonuçlarının ayrıntılı irdelenmesiyle VAE tabanlı anomali tespit sisteminin diğerlerine göre çok daha iyi performans gösterdiği gösterilir.

Bölümün organizasyonu şu şekilde yapılmıştır. Sonraki bölümde literatürde yer alan akış tabanlı saldırı tespiti üzerine yapılan çalışmalar özetlenmektedir. Üçüncü bölümde, çalışmada kullanılan yöntemler ve performans değerlendirme ölçütlerine ilişkin teorik bilgiler verilmektedir. Deneysel metodoloji ve sonuçlar dördüncü bölümde sunulmaktadır. Sonuç tespitleri bölümün sonunda açıklanmaktadır.

2.1. Literatür Taraması

Akış tabanlı saldırı tespiti yükselişte ve bu alanda yapılan araştırmalar gün geçtikçe hız kazanmaktadır. Son yıllarda, ihlalleri (izinsiz girişleri) saptamak için akış verilerini kullanan çok sayıda yöntem önerilmiştir. Bu bölümde, son trendlerin bir incelemesi ve akış tabanlı verilerden ihlalleri tespit edebilen en gelişkin ve yeni algoritmalar özetlenmektedir. Bunlar arasında Yapay Sinir Ağları (Artificial Neural Networks - ANN), Destek Vektör Makineleri (Support Vector Machines - SVM), K-En Yakın Komşu (K-Nearest Neighbor - KNN), Karar Ağaçları, kümeleme (clustering) ve istatistiksel teknikler bulunur. Akış tabanlı saldırı tespitinin daha kapsamlı ve ayrıntılı bir derlemesi [22] ve [23] çalışmalarında bulunabilir.

ANN'lerin amacı, birbirine bağlı küçük girdi birimleri olan nöronları taklit ederek insan beynini modellemektir. ANN'deki her bir nöron karar verme sürecine katılır ve bunların sonuçları birleştirilir. Anomalileri tespit etmenin bir yolunu bulmak için kullanıcı davranışları ANN'ler tarafından modellenir. Beghdad, anomali tabanlı IDS'lerde kullanılan çok sayıda ANN türünü çalışmasında ele alır [24]. Song vd. [25], geri yayılım sinir ağı sınıflandırıcısı ve istatistiksel özellik vektörleri kullanan bir anomali tespit sistemi önerir. Kaynak tüketme, DoS saldırıları, bant genişliği saldırısı

ve ağ akış kayıtlarını kullanan kaynak tüketme ve bant genişliği saldırısı DoS saldırılarının bir kombinasyonundan oluşan üç farklı senaryo üzerinde dururlar. Tran, Jiang ve Hu [26], tespit yöntemi olarak blok tabanlı sinir ağını (block-based neural network - BBNN) kullanan bir hibrit algılama motoru önerir. NetFlow verileriyle beslenen gerçek zamanlı bir IDS oluşturmak yüksek frekanslı bir FPGA kartı kullanır. Abuadlla vd. [8], akış tabanlı verilerdeki belirli ihlalleri (izinsiz girişleri) tespit edebilmek ve sınıflandırabilmek için iki aşamadan oluşan bir IDS önerir. İlk aşamada, olası saldırıların belirlenmesi için önemli değişiklikler gözetlenir. İkinci aşamada ise, bilinen bir saldırıyla karşılaşılmışsa, saldırıyı sınıflandırmak için çok katmanlı (multi-layer) ve radyal tabanlı işlev (radial basis function) ağları kullanılır. Jadidi, Muthukkumarasamy ve Sithirasenan [27], akış tabanlı verilerdeki anormal trafiği tespit etmek için Çok Katmanlı Algılayıcı (Multi-Layer Perceptron - MLP) tabanlı bir yöntem önerir. Cuckoo ve yerçekimsel arama algoritması (gravitational search algorithm) ile parçacık sürüsü optimizasyonu (particle swarm optimization) (PSOGSA) kullanılarak MLP'nin arabağlantı ağırlıkları optimize edilir. Mirsky vd. [28], yerel bir ağa yapılan saldırıları normal ve anormal trafik şablonlarını ayırt etmek AE olarak adlandırılan bir grup ANN kullanarak tanımlayan ve bunun yanında çevrimdışı anomali detektörleriyle karşılaştırılabilir bir performans sergileyebilen çevrimiçi bir anomali tespit sistemi olan Kitsune'u önerir. Marir vd. [29], büyük ölçekli ağlarda derin bir öznitelik çıkarımı ile birlikte bir grup çok katmanlı SVM'yi kullanarak anormal davranışı algılamak için yeni bir dağıtılmış yöntem sunar. Önerilen yaklaşımda, başlangıçta ağ trafik verileri üzerinde dağıtılmış derin inanç ağları (distributed deep belief networks) ile doğrusal olmayan bir boyut indirgeme (dimensionality reduction) gerçekleştirilir. Daha sonra çıkarılan özellikler, Spark tabanlı yinelemeli indirgeme paradigması (iterative reduce paradigm) sayesinde oluşturulan çok katmanlı SVM grubuna girdi olarak sağlanır. Vinayakumar vd. [30], "ölçek-hibrit-IDS-AlertNet" adlı kullanışlı ve esnek bir IDS oluşturmak ve öngörülemeyen ve önceden bilinmeyen ihlalleri tespit etmek için derin sinir ağını (Deep Neural Network - DNN) denetimli (supervised) öğrenme yaklaşımı aracılığıyla araştırır. Çeşitli hiperparametre ayarlarını uygulayarak DNN'ler için ağ topolojilerini ve en uygun ağ parametrelerini KDDCup99 veri setini kullanarak belirlerler. Ayrıca,

en iyi performans gösteren DNN modeli performans kıyaslamalarını gerçekleştirmek için diğer içerik- ve akış tabanlı genel veri setlerine de uygulanır.

SVM, n boyutlu girdi verilerini uzayda vektörler oluşturarak sınıflara dönüştüren bir sınıflandırma yöntemidir. Düşük yanlış pozitif oranları ve yüksek doğruluk (accuracy) sonuç sağladığı için SVM saldırı tespiti araştırma alanında sıkça tercih edilen yöntemdir [31]. Winter, Hermann ve Zeilinger [32], ağ akış verileri üzerinde çalışan ve OCSVM'leri analiz için kullanan tümevarımsal bir IDS önerir. Önerilen IDS, önceki yaklaşımlardan farklı olarak iyicil (benign) akışlar yerine kötücül (malicious) verilerle eğitilir. Wagner vd. [33] büyük hacimli Netflow kayıtlarını işleyerek SVM tabanlı çalışan bir anomali tespit yöntemi önerir. Netflow verilerinin nicel ve içeriksel (contextual) bilgilerinin dikkate alındığı yöntemde çekirdek (kernel) fonksiyonu Netflow kayıtlarıyla beslenmesiyle hesaplanan sonuçlar bir OCSVM'ye aktarılarak algılama işlemi gerçekleştirilir. Umer, Sher ve Bi [34] akış verileri üzerinde işlem yapan bir saldırı tespit modeli önerir. Geliştirilen iki aşamalı modelde, kötücül (malicious) akışları verimli bir biçimde tespit etmek için OCSVM yöntemini kullanır ve ardından kötü niyetli (malicious) trafiği algılama sürecinin ikinci aşamasına aktarır. Kötü niyetli akışların ayrıntılı analizi bu aşamada yapılır.

K-En Yakın Komşu (KNN), verilen örnekte sınıflandırmayı gerçekleştirmek için birbirine komşu noktaların bilgisini dikkate alır. Shubair vd. [35] , KNN yönteminin faydalarını bulanık mantık kombinasyonu ile etkili bir biçimde kullanan akış tabanlı bir IDS önerir. Çalışma, hatayı düşürmek için En Küçük Ortalama Kareler (Least Mean Square) yöntemini, en iyi eşleşen sınıfı bulmak için KNN'yi ve akış sınıfı etiketini seçmek için bulanık mantığı (fuzzy logic) kullanır. Costa vd. [36] , düğümlerin ağırlıklandırılması için olasılık yoğunluk fonksiyonundan istifade eden bir KNN grafı olan Optimum-yol orman kümelemesini (OPFC) kullanan bir saldırı tespit yöntemi önerir. Yazarlar, OPFC'nin optimizasyonunda k'nin değerine karar vermek için doğadan ilham alan gelişmiş tekniklerden (Yerçekimsel Arama, Yarasa Algoritması, Parçacık Sürüsü Optimizasyonu ve Harmoni arama) faydalanırlar.

Veri setindeki benzersiz ve faydalı kalıpları tespit etmek için kümeleme (clustering) yöntemleri kullanılır. Farklı setlerdeki benzer örneklerin bir araya getirilmesi, bu kalıplar kullanılarak gerçekleştirilir. Lakhina ve ark. [37], özellik dağılımlarını analiz ederek ağdaki akış anomalilerini tespit etmek için kümeleme yöntemlerini kullanır. Casas vd. [38], ağdaki paketleri toplayan ve bu paketleri akışlar olarak rastgele bir araya getirerek çalışan ve birden fazla denetimsiz kümeleme yöntemini kullanarak anomali tabanlı algılama yapan bir IDS önerir. Bu çalışmada, kötü niyetli (kötücül) akışları ayırt etmek için her bir alt uzayda veri alt kümelerini üreten alt uzay ve yoğunluğa dayalı (density-based) kümelemeyi kullanan bir değişiklik algılama algoritması (change detection algorithm) kullanır. Hosseinpour vd. [39] , Yapay Bağışıklık Sistemi (Artificial Immune System) ile kombine edilmiş denetimsiz kümelemeye dayalı dağıtık bir IDS önerir. Bu yaklaşımda trafik verilerini kötücül veya kötücül olmayan şeklinde etiketlemek için DBSCAN kümeleme yöntemi kullanılır ve ağlardaki ana bilgisayarlarının çevresinde yer alan bağışıklık tepki algılayıcılarını eğitmek için gerçek zamanlı ve çevrimiçi veriler kullanılabilir hale getirilir. Satoh, Nakamura ve Ikenaga [40] tarafından önerilen Ward Kümeleme yaklaşımı, SSH sözlüğünden basit ve kötü niyetli saldırıları tespit etmek için kullanılır. Algılama süreci, "bir bağlantı protokolünün varlığı" ve "bir kimlik doğrulama paketinin varış zamanı ve bir sonraki varış zamanı" esas alınarak akış özellikleri aracılığıyla "her bir alt protokolün geçiş noktası" belirlenerek gerçekleştirilir.

Karar Ağaçları (Decision Trees - DT), ağacın her bir düğümünün öznitelik değerine dayalı kurallar oluşturarak bir ağaç modeli üretir. Thaseen ve Kumar [41] saldırı tespiti için farklı DT tabanlı sınıflandırma algoritmalarının uygulanmasını tartışır. Zhao vd. [42], bot ağlarını sınıflandırabilen bir model oluşturmak için Azaltılmış Hata Budama (Reduced Error Pruning) yöntemiyle bir DT yaklaşımı kullanarak hem bilinen hem de yeni bot ağlarını tespit etmek için basit ve verimli bir akış tabanlı yaklaşım önerir. Haddadi vd. [43], botnetlerin davranışlarını genetik programlama ve DT yöntemlerini kullanarak tespit edebilmek için alternatif bir yaklaşım önerir. Önerilen yöntem, veri setlerinden çıkarılan ortak akış öznitelikleri ve TCF bayrak öznitelikleri olan iki farklı set kullanır. Stevanovic ve Pedersen [44] , denetimli makine öğrenimi yöntemlerinden oluşan bir koleksiyon oluşturarak 39 özellikli bir setin akış kayıtlarını kullanan etkili

bir botnet algılama yaklaşımı sunar. Sonuçlara göre, genel olarak en iyi performansın Rastgele Orman (Random Forest) yöntemi ile elde edildiği belirtilir.

Literatürde kullanılan istatistiksel yöntemler şu şekilde özetlenebilir. Kanda vd. [45], anormal ağ akışlarını tespit etmek için rastgele projeksiyona ve Temel Bileşen Analizine (Principal Component Analysis - PCA) dayalı ADMIRE olarak isimlendirilen bir anomali algılama yöntemi önerir. Değerlendirme, transpasifik bir bağlantıdan gelen trafik verilerini kullanılarak gerçekleştirilir. Haghighat ve Li [46], NetFlow trafik verilerini kullanarak anormal davranışları tespit etmek ve ağ ihlallerini (izinsiz girişlerini) azaltmak için Edmund olarak adlandırılan yeni bir entropi tabanlı yöntem önerir.

2.2. Teorik Bilgi

2.2.1. Otokodlayıcı

Otomatik kodlayıcı (Autoencoder - AE) [47], [48], girdi vektörlerini çıktı vektörleri olarak yeniden oluşturmak için denetimsiz yaklaşımla eğiten işletim mantığına sahip bir sinir ağı yöntemidir [49]. Mimarisi temelde bir kodlayıcı (encoder) ve bir kod çözücüden (decoder) oluşur. Bir AE katmanında sırasıyla Denklem 2.1 ve Denklem 2.2'de gösterildiği gibi bir kodlayıcı ve bir kod çözücü bulunur. Bu bağlamda, σ doğrusal olmayan dönüşüm fonksiyonudur ve b ve W sırasıyla sinir ağının biası ve ağırlığı olarak adlandırılır [50].

$$h = \sigma(W_{xh}x + b_{xh}) \tag{2.1}$$

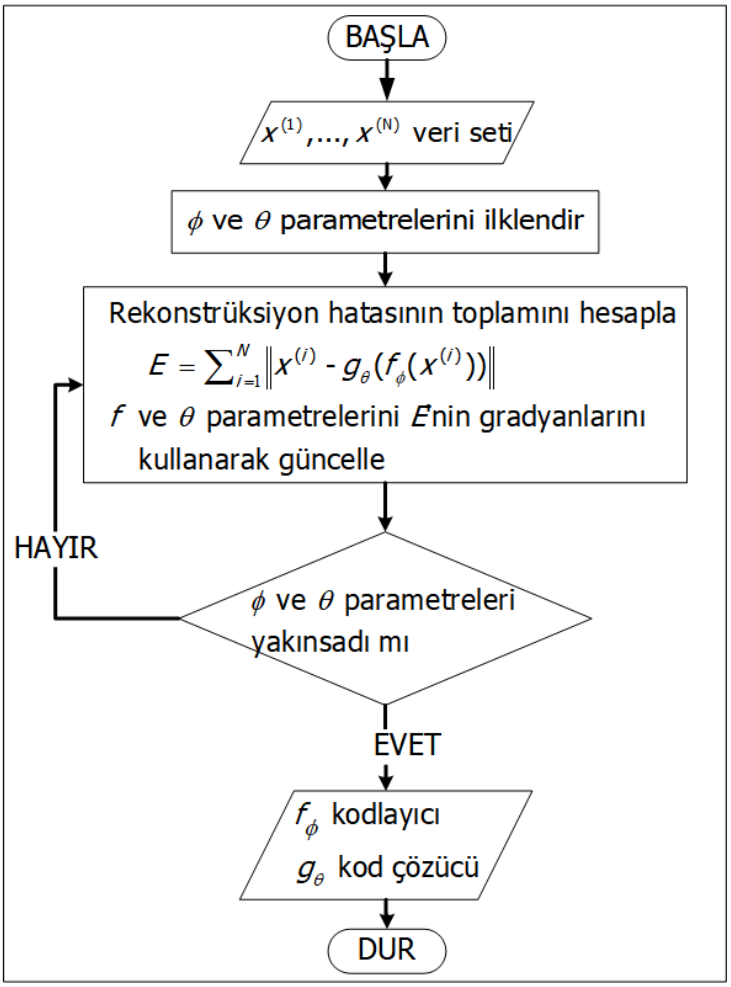
$$z = \sigma(W_{hx}h + b_{hx}) \tag{2.2}$$

$$r = \|x - z\| \tag{2.3}$$

Doğrusal olmama durumu ile sonuçlanan bir afin iz düşürme (mapping) kullanılarak x girdi vektörünün h gizli temsiline (hidden representation) dönüştürülmesi kodlayıcı kullanılarak gerçekleştirilir. Bir kod çözücü kullanarak ilk girdi uzayını rekonstrüksiyonu (reconstruct) için h gizli temsiline dönüştürme işlemi uygulanır. r

rekonstrüksiyon hatası, z yeniden oluşturulmuş vektörü ile x orijinal girdi vektörü arasındaki fark alınarak elde edilir. AE'de r rekonstrüksiyon hatasını en aza indirmek için denetimsiz eğitim prosedürü gerçekleştirilir. AE eğitim algoritmasının akış şeması Şekil 2.1.'deki gibi gösterilebilir.

AE tabanlı anomali tespiti, anomali skoru olarak rekonstrüksiyon hatası (reconstruction error - RE) kullanılarak gerçekleştirilir. Yüksek RE değerine sahip girdi verilerinde anormallikler olduğu varsayılır. AE eğitimi, sinir ağı girdisi yalnızca normal örneklerle beslenerek gerçekleştirilir. Eğitilmiş AE modeli, normal girdi verilerini çok düşük RE değeri üreten rekonstrüksiyonuna rağmen daha önce karşılaşmadığı anormal verilerle bunu yapması başarısızlıkla sonuçlanır. AE tabanlı anomali algılama algoritmasının akış şeması Şekil 2.2.'deki gibi gösterilebilir.



Şekil 2.1. AE eğitim algoritmasının akış şeması

2.2.2. Varyasyonel otokodlayıcı

Varyasyonel Otokodlayıcı (Variational Autoencoder - VAE) [51], yapay sinir ağının kendi sonsal olasılığına (posterior) yaklaştırımıyla (approximation) elde edilen yönlü (directed) olasılıksal bir grafik model olarak tanımlanır [50]. Üretici (generative) sürecin başladığı z gizli değişkeni (latent variable) grafik modelin VAE'deki en yüksek katmanıdır. x girdi verisine götüren karmaşık veri üretim prosedürü, yapay bir sinir ağının oluşumuyla modellenen $g(z)$ ile temsil edilir. Marjinal olasılık kolay elde edilemediğinden girdi verisinin marjinal olasılığının varyasyonel alt sınırı VAE'nin amaç fonksiyonudur (objective function). Marjinal olasılık Denklem 2.4'teki birbirinden farklı veri noktalarının marjinal olasılığının toplanmasıyla elde edilir. Farklı veri noktalarının marjinal olasılığı yeniden formüle edilirse Denklem 2.5 elde edilir [50].

$$log p_\theta\left(x^{(1)}, \dots, x^{(N)}\right) = \sum_{i=1}^{N} log p_\theta\left(x^{(i)}\right) \tag{2.4}$$

$$log\, p_\theta\left(x^{(i)}\right) \geq \mathcal{L}\left(\theta, \phi; x^{(i)}\right) \tag{2.5}$$

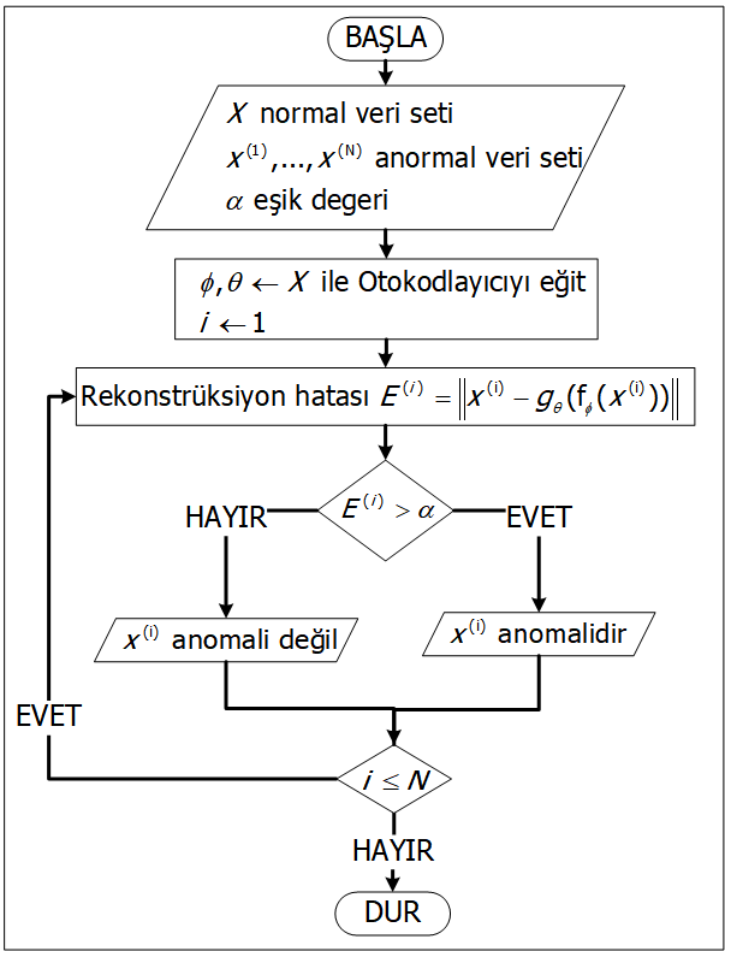
$$= E_{q_\phi\left(z|x^{(i)}\right)}\left[-log\, q_\phi\,(z|x) + log\, p_\theta\,(x|z)\right] \tag{2.6}$$

$$= -D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + E_{q_\phi(z|x^{(i)})}[log p_\theta(x|z)] \tag{2.7}$$

Denklem 2.7'deki D_{KL}, z gizli değişkeninin yaklaşık sonsal olasılığı (approximate posterior) ile önseli arasındaki Kullback-Leibler ıraksamasıdır (divergence). Verilen bir z gizli değişken için x girdi verilerinin olasılığı $p_\theta(x|z)$ ile temsil edilir.

VAE $q_\phi(z|x)$ yaklaşık sonsal olasılık parametrelerini bir sinir ağı kullanarak modeller. Şekil 2.3.'te gösterildiği gibi $p_\theta(x|z)$ yönlü olasılıksal grafik modeli bir kod çözücüdür ve $q_\phi(z|x)$ yaklaşık sonsal olasılığı ise bir kodlayıcıdır. Bu bağlamda, VAE'nin değerin kendisinden ziyade dağılım parametrelerini modellediğini vurgulanmalıdır. Yani, kodlayıcıdaki $f(x, \phi)$ yaklaşık sonsal olasılık $q_\phi(z|x)$ parametresini çıktı olarak üretir ve z gizli değişkenin gerçek değerini elde etmek için $q\left(z; f(x, \phi)\right)$ 'den örnekleme yapmaya ihtiyaç vardır. Dolayısıyla, VAE'nin kodlayıcıları ve kod çözücüleri olasılıksal kodlayıcılar ve kod çözücüler olarak

adlandırılabilir. Bir sinir ağı olan $f(x,\phi)$, x verisi ile z gizli değişkeni arasındaki karmaşık ilişkiyi temsil eder. Verilen z örneği için $\hat{x}$ rekonstrüksiyon (reconstruction) çıktısını üretmek için $p_\theta(x|z)$ parametresi $g(z,\theta)$ tarafından elde edilir. Burada, $\hat{x}$ rekonstrüksiyon (reconstruction) çıktısı $p_\theta\big(x;g(z,\theta)\big)$'den örneklenir. Özetlemek gerekirse, VAE'de modellenen değerin kendisinden ziyade dağılım parametreleridir. Dağılım seçimi yaparken parametrik dağılım türlerinden herhangi biri tercih edilebilir. z gizli değişkenin dağılımı olan $p_\theta(z)$ ve $q_\phi(z|x)$ için yaygın seçim izotropik normaldir. Bunun nedeni gizli değişken uzayındaki değişkenler arasındaki ikişkinin girdi veri uzayındakinden çok daha basit olduğunun varsayılmasıdır. $p_\theta(x|z)$ olasılık dağılımları verilerin doğasına bağlıdır. Daha spesifik olarak belirtmek gerekirse girdi verileri sürekli (continuous) formda olduğunda çok değişkenli Gauss (Multivariate Gauss) dağılımı uygulanır. İkili formda ise Bernoulli dağılımı kullanılır [50]. VAE eğitim algoritmasının akış şeması Şekil 2.4.'teki gibi gösterilebilir.



Şekil 2.2. AE tabanlı anomali tespit algoritmasının akış şeması

Geri yayılım (backpropagation) algoritması kullanılarak VAE eğitimi yapılır [50]. Denklem 2.7'deki ikinci terimin hesaplanmasında Monte Carlo tekniklerini kullanılarak hesaplandığından Monte Carlo gradyan tekniklerinin kullanılması zorunludur. Fakat varyasyonel alt sınırı optimize etmek için kullanılan geleneksel Monte Carlo gradyan tekniklerinin çok yüksek varyanstan muzdarip olduğu bilinmektedir ve bu yüzden kullanım için uygun değildir. VAE orijinal dağılımdaki ratstgele değişken yerine standart normal dağılımdan rastgele bir değişken kullanan bir yeniden yeniden parameterize etme (reparameterization) hilesi yoluyla bunun üstesinden gelir. $z \sim q_\phi(z|x)$ rastgele değişkeni $h_\phi(\epsilon, x)$ deterministik bir dönüşümü ile yeniden parameterize edilir. Buradaki ϵ standart normal dağılımdan gelir.

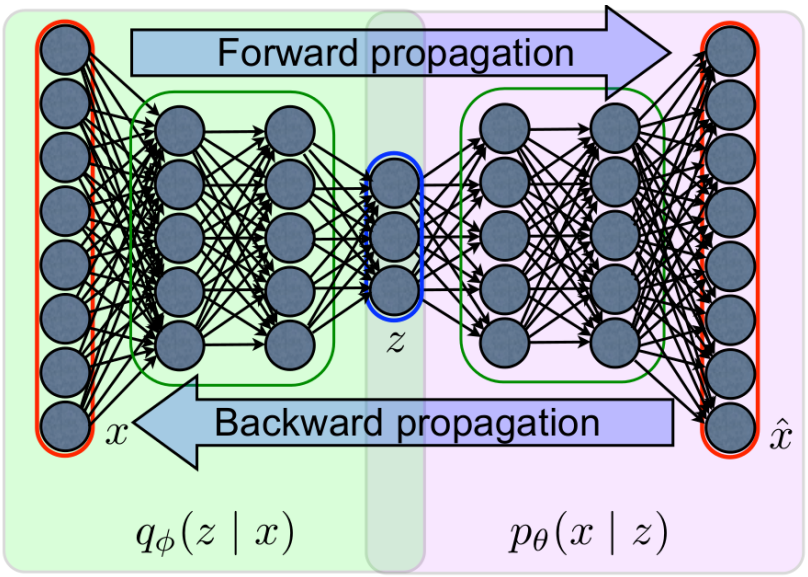
$$\tilde{z} = h_\phi(\epsilon, x) \quad \text{with} \quad \epsilon \sim \mathcal{N}(0,1) \tag{2.8}$$

Yeniden parameterize etme işlemi $\tilde{z}$'nin $q_\phi(z|x)$'nin dağılımını takip ettiğini garanti etmelidir. Bu işlemi yapmak doğrudan Monte Carlo gradyan yöntemini kullanmaktan çok daha kararlıdır [50].

Anomali tespit işlemi, VAE'yi eğitmek için yalnızca normal veri örnekleri kullanıldığı anlamına gelen yarı denetimli (semi-supervised) bir şekilde gerçekleştirilir [50]. f_ϕ olasılıksal kodlayıcı ve g_θ kod çözücünün her ikisi de sırasıyla gizli değişken uzayındaki ve orijinal girdi değişken uzayındaki izotropik bir normal dağılımı parametrelendirir. Eğitilen VAE modelinin olasılıksal kodlayıcısından birkaç örnek seçilerek test işlemi gerçekleştirilir. Dağılımdan üretilen orijinal verilerin olasılığı, olasılıksal kod çözücü veya kodlayıcıdan tarafından her bir örnek için üretilen ortalama ve varyans parametreleri kullanılarak hesaplanır. Rekonstrüksiyon olasılığı (reconstruction probability - RP) olarak da adlandırılan ortalama olasılık anomali skoru olarak kullanılır. Burada hesaplanan RP, Denklem 2.7'nin sağ tarafının ikinci terimi olan Denklem 2.9'un Monte Carlo tahminidir. RP'si yüksek olan veri noktaları anomaliler olarak sınıflandırılır.

$$E_{q_\phi(z|x)}[\log p_\theta(x|z)] \tag{2.9}$$

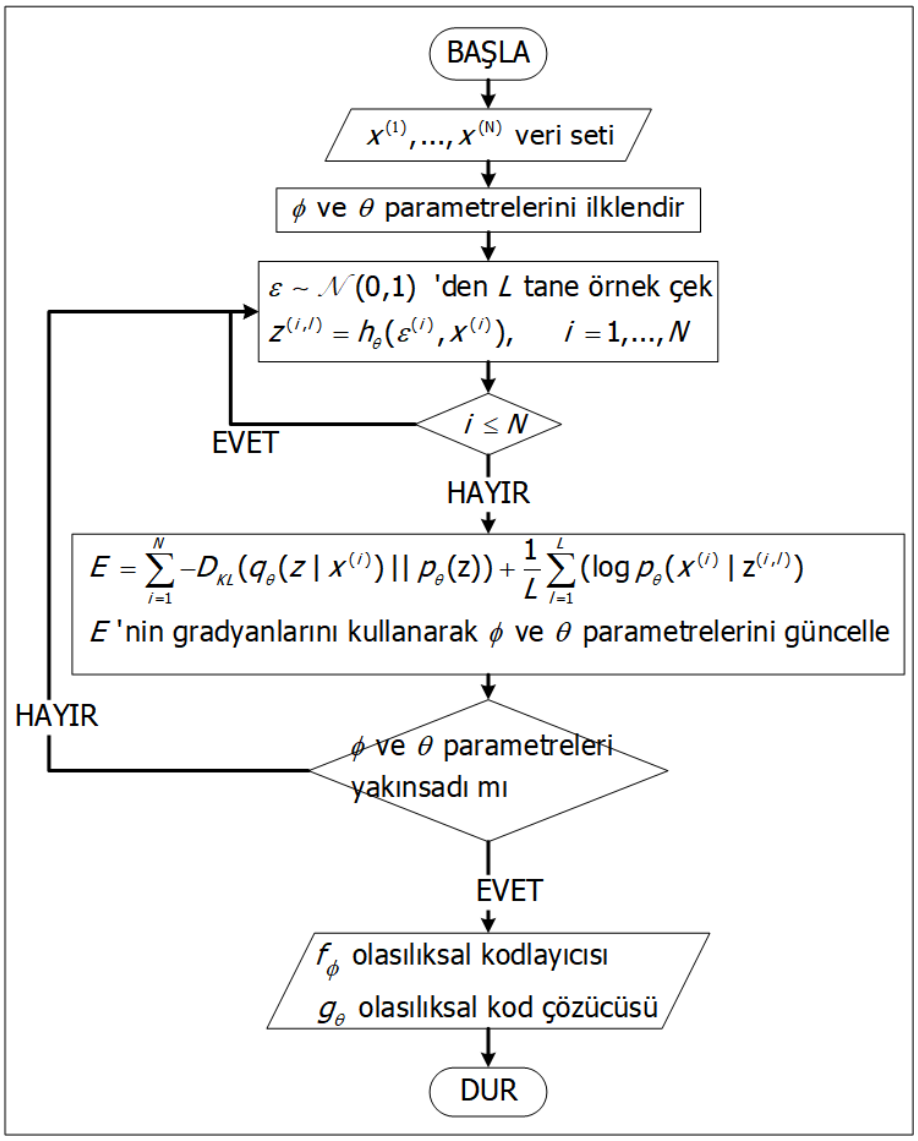
RP'yi hesaplamak için orijinal girdi değişken dağılımının parametrelerini üreten stokastik gizli değişkenler kullanılır. Burada yeniden oluşturulan değerler girdi değişkeninin kendisinden ziyade girdi değişken dağılımının parametreleridir. Bu, esasen yaklaşık sonsal olasılık dağılımından alınan belirli bir gizli değişkenden üretilen verilerin olasılığıdır.



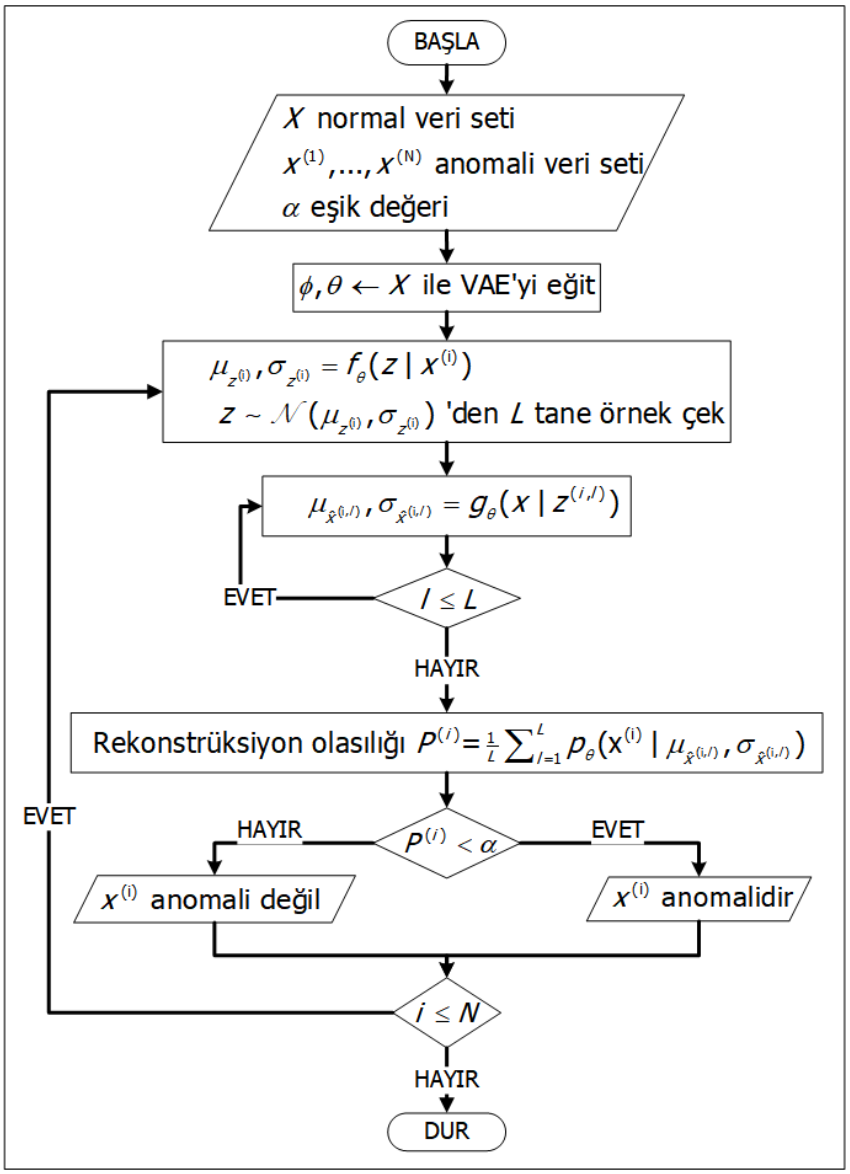
Şekil 2.3. VAE mimarisi [50]

VAE'de hesaplanan RP, AE'de hesaplanan RE'den birkaç yönden farklılık gösterir [50]. Öncelikle, gizli değişkenler AE'de deterministik izdüşürmeler olarak ifade edilirken, VAE'de stokastik değişkenler olarak tanımlanır. Gizli uzayın (latent space) değişkenliği, VAE'nin gizli değişkenin kendisi yerine gizli değişkenlerin dağılımını modellemek için olasılıksal kodlayıcı kullanması nedeniyle örnekleme sürecinde dikkate alınabilir. Bu durum normal veriler ve anomali verileri aynı ortalama değerini paylaşabilse bile değişkenlik farklı olabileceğinden VAE'nin ifade gücünü AE'ye kıyasla genişletir. Muhtemelen anormal veriler daha büyük varyansa sahip olacak ve daha düşük rekonstrüksiyon olasılığı gösterecektir. İkinci olarak, rekonstrüksiyonlar VAE'deki stokastik değişkenlerdir. RP sadece orijinal girdi ile rekonstrüksiyon arasındaki farkı hesaba katmakla kalmaz aynı zamanda varyans parametrelerini de kullanarak rekonstrüksiyon değişkenliğini de dikkate alır. AE'nin deterministik doğasından dolayı sahip olmadığı bu özellik, değişken varyansı doğrultusunda rekonstrüksiyona seçici hassasiyet sağlar. Üçüncüsü, olasılık ölçüleri

rekonstrüksiyonlara karşılık gelir. AE tabanlı anomali tespitinde anormali skorları RE'ler kullanılarak oluşturulur. Bu anlamda girdi değişkenleri heterojen olsaydı, veriye bağlı olarak değişen uygun ağırlıkları belirlemek için genel bir objektif tekniğin bulunmaması nedeniyle anomali skorlarının hesaplanması zor olurdu. Buna ek olarak, RE için uygun ve objektif bir eşiğin belirlenmesi sorunlu bir süreçtir. Öte yandan, her bir değişkenin olasılık dağılımı, bunların kendi bireysel değişkenliği ile bağımsız olarak hesaplanmasını sağladığından RP'nin hesaplanmasında heterojen verilerin RE'lerinin ağırlıklandırılmasına ihtiyaç duyulmaz. Bu nedenle, RP'nin eşik değerinin belirlenmesinin RE'ninkinden önemli ölçüde daha objektif ve kolayca anlaşılabilir bir şekilde gerçekleştirilebileceği sonucuna varılabilir.



Şekil 2.4. VAE eğitim algoritmasının akış şeması



Şekil 2.5. VAE tabanlı anormallik algılama algoritmasının akış şeması

2.2.3. Tek sınıflı destek vektör makinesi

Destek Vektör Makinesi (Support Vector Machine - SVM) ilk olarak Vapnik [52] tarafından önerilmiştir. SVM'nin birincil amacı belirli bir veri seti için en iyi makineyi bulmaktır. SVM, belirli bir eğitim veri seti için makinenin doğruluğunu en üst düzeye çıkararak bu hedefi gerçekleştirir. Buna ek olarak, gelecek test veri setlerini doğru bir şekilde sınıflandırmak için makinenin yeteneği de maksimize edilir. En iyi makine, matematiksel bir optimizasyon yöntemi [53] kullanılarak keşfedilir.

SVM yöntemi, [54]'da açıklandığı gibi Tek Sınıflı SVM'ye (One Class SVM - OCSVM) dönüştürülür. OCSVM'de algoritmaya girdi olarak verilen veri seti, pozitif ve negatif olmak üzere iki farklı sınıfa ait örneklerden oluşmaktadır. Veri setindeki pozitif örneklerdeki ve "anormallikler" veya "aykırı değerler" olarak da tanımlanan gürültüler, olumsuz örnekler olarak kullanılır. OCSVM'nin formülasyonu aşağıdaki gibi gerçekleştirilebilir [54]:

$$f(x) = \begin{cases} +1, if x \in S \\ -1, if x \in \overline{S} \end{cases} \quad (2.10)$$

Algoritmanın çalışma mantığı şu şekildedir. Girdi verileri önce uygun bir çekirdek fonksiyonu uygulanarak H özellik uzayına dönüştürülür. Daha sonra, önerilen algoritma, bu dönüştürülmüş özellik vektörlerini başlangıç noktasından maksimum marjla ayırmak için bir hiper düzlem bulmaya çalışır [55].

Verilen bir $(x_1, y_1), \dots, (x_N, y_N) \in R^n \times \{\pm 1\}$ veri seti için, $\phi: R^n \to H$ örnekleri H'ye izdüşüren bir çekirdek fonksiyonu olsun. Daha sonra, veri setini orijinden ayırmak için aşağıdaki ikinci dereceden programlama probleminin çözülmesi gerekir:

$$min(\frac{1}{2}\|\omega\|^2 + \frac{1}{\nu N}\sum_{i=1}^{l}\xi_i - \rho) \quad (2.11)$$

subject to

$$y_i(\omega \cdot \phi(x_i)) \geq \rho - \xi_i, \xi_i \geq 0, i = 1, \dots, N \quad (2.12)$$

$\nu \in (0,1)$ parametresi, eğitim veri setindeki "anormallikler" veya "aykırı değerler" oranını ifade eder. Alt düzlem tarafından oluşturulan bölgedeki verilerin çoğu dahil olmak üzere aradaki ödünleşimi (tradeoff) düzenler ve başlangıç noktasına olan mesafeyi en üst düzeye çıkarır. Bazı aykırı değerler, ξ_i gevşek değişkenleri aracılığıyla sınırın dışına yerleştirilir. $f(x) = sign\left(\left(\omega \cdot \phi(x)\right) - \rho\right)$ karar fonksiyonu, eğitim veri setindeki çoğu x_i örneği için pozitif olacaktır. Bazı veri setlerinin ayrılması doğrusal olarak gerçekleştirilemediğinden, doğrusal ayrılabilirliği sağlayabilmek için

girdileri yüksek boyutlu uzaya dönüştüren genellikle SVM'lerde genellikle radyal temel çekirdek (radial basis kernel) [56] ve polinom çekirdeği (polynomial kernel) gibi fonksiyonlar kullanılır.

2.2.4. Değerlendirme metrikleri

Önerilen yöntemin performans değerlendirilmesi uygun ölçütler (metrics) kullanılarak yapılmalıdır. İkili bir sınıflandırma için sonuçlar dört gruba ayrılabilir [57] [58]: 1) Gerçek Pozitif (TP): Doğru şekilde sınıflandırılan pozitif örnekler; 2) Yanlış Negatif (FN): Yanlış sınıflandırılmış pozitif bir örnek; 3) Yanlış Pozitif (FP): Yanlış sınıflandırılmış negatif bir örnek; 4) Doğru Negatif (TN): Doğru şekilde sınıflandırılmış negatif bir örnek. Ayrıca, sonraki metrikler öncekilerden hesaplanabilir [58]:

Gerçek Pozitif Oran (TPR): Bu metrik, tüm "doğru tanımlanmış örneklerin" tüm "tanımlanması gereken örneklere" oranına karşılık gelir.

$$TPR = TP/(TP + FN) \tag{2.13}$$

Yanlış Pozitif Oran (FPR): Bu metrik, “yanlış sınıflandırılan negatif örneklerin sayısının” “toplam negatif örnek sayısı”na oranını temsil eder.

$$FPR = FP/(FP + TN) \tag{2.14}$$

Alıcı İşletim Karakteristiği (ROC): ROC eğrisi [59][60], veri setinde karşılaşılan sınıf dengesizliği problemi durumunda sınıflandırıcıların değerlendirilmesinde standart bir kriter olarak kullanılır [61]. ROC eğrileri, sınıflandırıcıların çeşitli TPR'ler ve FPR'ler üzerindeki verimliliğinin özetini sağlayarak çarpık duyarsızlığın (skew insensitivity) üstesinden gelmeye çalışır. ROC eğrileri, eğitilmiş modelleri farklı hata değerlerinde değerlendirerek belirli bir FPR için hangi örneklerin yüzdesinin uygun şekilde sınıflandırılacağına karar verebilir. ROC eğrileri, bir sınıflandırıcının verimliliğine karar vermek için görsel bir yaklaşım sunar [61].

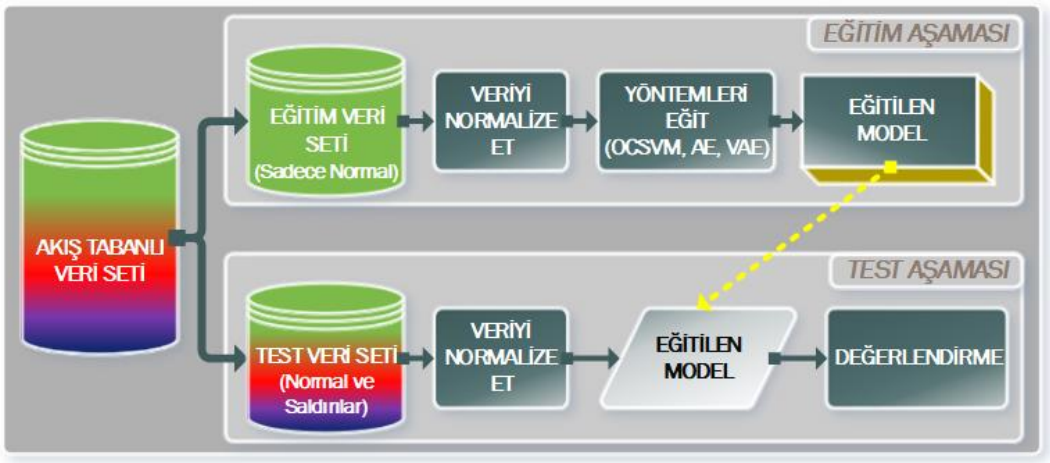
ROC eğrisinin altındaki alan (Area under ROC - AUC), sınıf dengesizliği sorunu ile karşı karşıya olan sınıflandırıcıların (classifier) verimliliğini (efficiency) ölçmek için fiili olarak kullanılan bir standarttır. Bunun arkasındaki sebebi AUC'nin önceki olasılıklardan ve seçilen eşikten (threshold) etkilenmemesidir. Ayrıca sınıflandırıcılar arasında karşılaştırma yapmak için tek bir nicelik sayı sağlar. Rastgele bir pozitif sınıf örneğinin sınıflandırma olasılıklarına göre sıralandıktan sonra negatif bir sınıf örneğinden ne sıklıkla daha üst sırada yer aldığını tahmin etmek için AUC dikkate alınabilir. AUC değerleri her zaman 0 ile 1 arasında sınırlandırılır. AUC değeri 0,5'ten küçükse, bu sınıflandırıcının gerçekçi olmadığı anlamına gelir [62]. Aşağıdaki gibi kaba bir sınıflandırma sistemi test doğruluğuna rehberlik edebilir [63]: i) Mükemmel (0,90-1), ii) İyi (0,80-0,90), iii) Orta (0,70-0,80), iv) Zayıf (0.60–0.70), v) Başarısız (0.50–0.60).

2.3. Deneysel Sonuçlar

Yarı Denetimli Öğrenme (Semi-Supervised Learning - SSL) paradigması, öğrenme stratejisi olarak kullanılır. SSL, birkaç farklı ayar kullanılarak gerçekleştirebilir [64]:

a. Yarı denetimli sınıflandırma (Semi-supervised classification): Etiketli ve etiketlenmemiş verilerle (veya kısmen etiketlenmiş verilerle) sınıflandırmadır. Bu denetimli sınıflandırma sorununun bir uzantısı olarak ortaya çıktı.
b. Kısıtlı kümeleme (Constrained clustering): Bu strateji, kümeler hakkındaki bir dizi "denetimli (supervised) bilgi"ye ilave olarak etiketlenmemiş örnekleri de içeren eğitim verilerini kullanarak denetimsiz kümelemeyi (unsupervised clustering) genişletir.

Daha az bilgi, zaman ve çaba gerektirmesi nedeniyle etiketlenmemiş verilerin ihlal (izinsiz giriş) tespit sistemi için elde edilmesi etiketli verilere göre daha kolay olduğu için kısıtlı kümeleme SSL stratejisi seçilmiştir. Ayrıca, bu strateji kullanılan tespit yöntemlerinin denetimsiz eğitim doğasına daha uygundur.



Şekil 2.6. Yarı denetimli öğrenme stratejisi

Şekil 2.6.'da görüldüğü gibi, eğitim ve test veri seti olmak üzere veri seti iki kısma ayrılır. SSL stratejisi uygulanarak, ağ trafiğinin normal profilini oluşturmak için sadece normal akış özelliklerini içeren etiketli veri seti eğitim aşamasında kullanılır. Test aşamasında ise hem normal hem de saldırı akışı özelliklerinden oluşan etiketlenmemiş veri seti kullanılır. Bu çalışmada değerlendirme metriklerinin hesaplanmasında test veri setindeki etiketlerin dikkate alındığını belirtmek önemlidir.

2.3.1. Veri seti seçimi

İhlal tespiti yapmak için kullanılan yöntemlerinin performanslarını değerlendirmek için uygun bir veri setinin seçilmesi gerekir. Literatürde atıfta bulunulan ihlal tespit yöntemleri çeşitli veri setleri aracılığıyla performans değerlendirilmesine tabi tutulur. İlk bölümde de belirtildiği üzere eğitilen modelleri değerlendirmek için birçok araştırmacı KDDCUP99 ve NSLKDD veri setlerini kullanır. Bununla birlikte, KDDCUP99 veri setinin eğitim ve test veri setleri yüksek artıklık (fazlalık, high redundancy) sorunundan muzdariptir. Bu sorunun üstesinden gelmek için KDDCUP99'dan NSL-KDD veri seti oluşturuldu. Ancak NSL-KDD tarafından ele alınmayan diğer bir problem de bu veri setinin ağ trafiğinin gerçekçi bir temsili olmamasıdır [21]. Dahası, bu veri setleri bazı akış tabanlı özellikler yer alsa da çoğunlukla paket tabanlı (içerik) özellikler içerirler. Bu çalışma akış tabanlı özelliklerden gelen saldırıları tespit etmeyi amaçladığından Kyoto 2006+ [65], CTU-

13 [66], UNSW-NB15 [67], CIDDS-001 [68] ve CICIDS2017 [69] birkaç akış tabanlı aday veri setleri arasında yer alır.

Kyoto 2006+; DoS, malware, backscatter, port scans, exploits ve shellcode gibi honeypot'lara karşı gerçekleştirilen çok sayıda saldırı dahil olmak üzere gerçek ağ trafiğini kapsayan halka açık bir veri setidir. Ancak hem istatistiksel bilgiler hem de akış temelli özelliklerle birlikte yalnızca az miktarda veri ve gerçekçi normal kullanıcı davranışı içerir. CTU-13, çeşitli botnet saldırılarını kapsayan 13 senaryoyu karakterize ederek bir kampüs ağından elde edilmiştir ve paket, çift yönlü akış ve tek yönlü akış gibi çeşitli veri formlarında erişilebilir. UNSW-NB15, 31 saat boyunca küçük bir benzetilmiş ağda normal ve kötücül trafik toplanarak paket tabanlı formatta oluşturulmuştur. Ekstra özelliklere sahip akış veri seti şeklinde de sunulan veri seti, eğitim ve test için önceden belirlenmiş ayrımlarla DoS, backdoors, fuzzers, worms veya exploitler gibi dokuz farklı saldırı kategorisini kapsar. CIDDS-001, python komut dosyalarını çalıştırarak normal ve kötücül kullanıcıların davranışlarını sergileyen küçük bir benzetilmiş ağdan elde edilmiştir. Dört haftalık tek yönlü akış tabanlı ağ trafiğinin yanı sıra DoS, SSH brute force ve port taraması dahil birçok ağ saldırılarını kapsar. CICIDS2017, benzetilmiş bir ağ ortamında 5 günlük bir zaman diliminde üretilmiştir. Bunun yanında, paket tabanlı formattaki ağ trafiğinin yanı sıra akış tabanlı formatta da sunulan bu veri seti IP adresleri hakkında ekstra meta verileri ile birlikte 80’den fazla özellik içermekte ve IP adresleri hakkında ekstra meta veri bilgileri ve FTP patator, SSH patator, DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed, Brute force, XSS, SQL Injection, Infiltration, Bot, DDoS and Port Scan dahil geniş bir yelpazede güncel saldırı türlerini de barındırmaktadır.

CICIDS2017, yukarıda belirtilen veri setleri arasından aşağıdaki nedenlerden dolayı önerilen yöntemin performansının değerlendirilmesinde kullanılması amacıyla seçilmiştir: 1) Bir McAfee raporuna göre, son zamanlarda ağlara gerçekleştirilen çeşitli saldırı türlerini içerir [70][30], 2) Güncel bir veri setidir, 3) İstatistiksel olarak bazı parametreler ölçülerek genişletilen akış tabanlı özelliklerden oluşan etiketli veri setidir, 4) Gerçek zamanlı ağ trafiği özelliklerine sahiptir [23], 5) Doğrusal ayrılabilir değildir (non-linearly separable) [23]. Saldırı türü ve güne göre kategorize edilen

CICDS2017'deki akış örneklerinin sayısı Tablo 2.1.'de gösterilmektedir. Tablo 2.1.'den de anlaşıldığı gibi veri setinin tek dezavantajının sınıfların dağılımının oldukça dengesiz olduğu belirtilmelidir.

Tablo 2.1. Saldırı türüne ve gününe göre kategorize edilen CICIDS2017 veri setindeki örnek sayısı

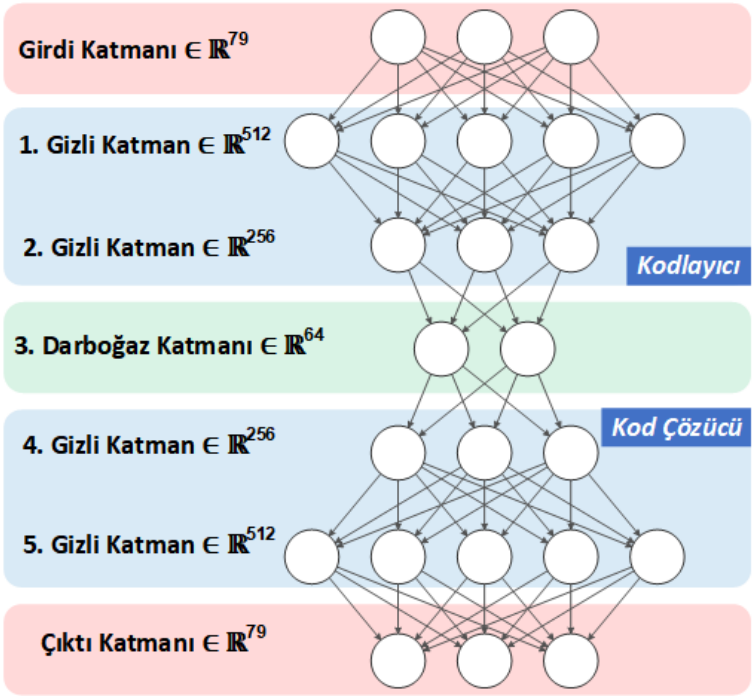
		Akış Sayısı (Günlük)				
	Traffic Türü	**Pazartesi**	**Salı**	**Çarşamba**	**Perşembe**	**Cuma**
NORMAL	*İyicil*	529918	432074	440031	456752	500514
ANOMALİ	*FTP-Patator*		7938			
	SSH-Patator		5897			
	DoS slowloris			5796		
	DoS Slowhttptest			5499		
	DoS Hulk			231073		
	DoS GoldenEye			10293		
	Heartbleed			11		
	Web Attack–Brute force				1507	
	Web Attack–XSS				652	
	Web Attack–SQL Injection				21	
	Infiltration				36	
	Bot					1966
	DDoS					41835
	Port scan					158930
Günlük Toplam		529918	445909	692703	458968	703245
TOPLAM		2830743				

2.3.2. Model konfigürasyonu

AE, VAE ve OCSVM yöntemlerinin model konfigürasyonunun nasıl yapıldığı ve en iyi anormali tabanlı saldırı tespit modelini elde etmek için parametrelerin seçiminin nasıl gerçekleştirildiği bu kısımda tartışılmaktadır.

Bu çalışmada kullanılan yöntemlerin parametreleştirilmiş olması nedeniyle modellerin performansının optimal parametreler ile belirlenmesi gerekmektedir. Eğitim süreci

anormal veriler kullanılarak yürütülmediğinden hiperparametrelerin optimizasyonunda çapraz doğrulama (cross-validation) işlevi gerçekleştirilemez [71]. Böylece, hiperparametrelerin ayarlanması Patterson ve Gibson [72] tarafından bahsedilen öneriler dikkate alınarak yapılır. Bundan sonra, AE ve VAE'nin hiperparametreleri çoğunlukla deneme yanılma yani pratik kural kullanılarak yapılandırılır. OCSVM modelinin konfigürasyonunda kütüphanenin varsayılan değerleri kullanılır.



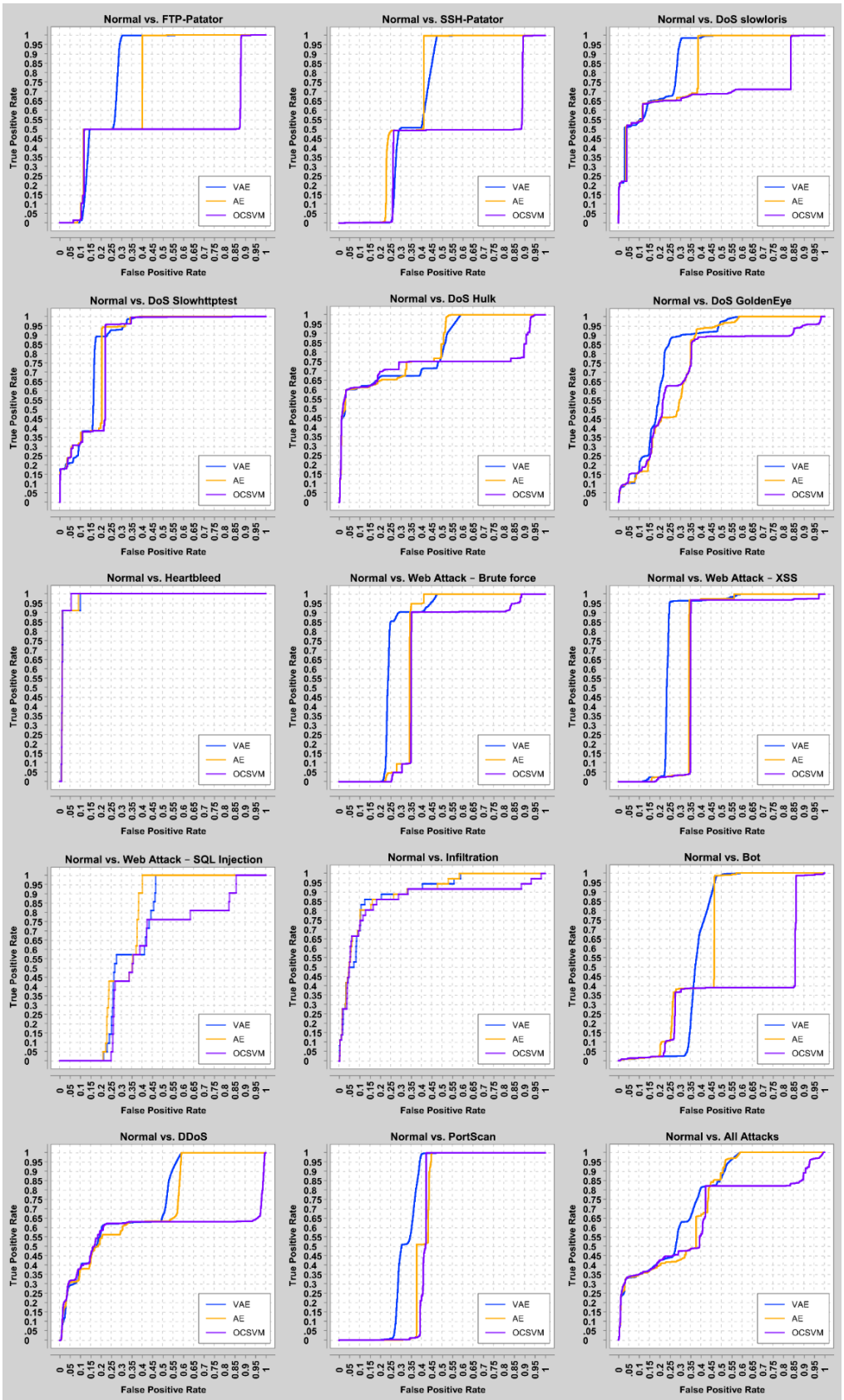
Şekil 2.7. Modellerde kullanılan AE ve VAE'nin mimari diyagramı.

Deneyleri gerçekleştirmek için basit bir derin otokodlayıcı mimarisi biçimi benimsenir. Kullanılan AE ve VAE'nin mimari diyagramı Şekil 2.7.'de gösterilmektedir. Hem AE hem de VAE için kodlayıcı 512 ve 256 boyutlu iki gizli katmana ve kod çözücü ise sırasıyla 256 ve 512 boyutlu olmak üzere iki gizli katmana sahiptir. Hem AE hem de VAE'nin darboğaz katmanının (bottleneck layer – latent dimension - gizli boyut) boyutu 64'tür. Hem AE hem de VAE'nin en iyi performans gösteren modellerinin belirlenmesinde, katman sayıları sabit tutularak katmanlardaki boyutlar deneme yanılma ile belirlendi. Hem AE hem de VAE'nin sinir ağı konfigürasyonunda kullanılan parametre ayarları aşağıdaki gibidir: Eğitim denemelerinde [0.1, 0.01, 0.001] gibi öğrenme hızları (learning rate) ve [0.3, 0.5, 0.9]

gibi momentumlar kullanılır. En iyi performansı veren değerler öğrenme hızı için 0,001 ve momentum için 0,3'tür. L2 regülasyonunun [73] parametresi, yaygın olarak kullanılan 0.001 değerine ayarlanmıştır ve kütüphanede varsayılan olan ağırlık başlatma yöntemi olarak gizli katmanlarda ReLU kullanılıyorsa önerilen [72] Xavier [74] kullanılır. Her iki sinir ağı, büyük veri setleri ve gerçek değerli çıktılar için önerilen bir seçim olan eşlenik gradyan optimizasyon (conjugate gradient optimization) algoritması kullanan geri yayılım algoritması ile eğitilir [64]. Nesterovs güncelleyici [75], yerel minimumumdan kaçmak ve optimizasyon sürecine [64] yönelik daha iyi çözümler keşfetmek için öğrenme prosedürünü destekleyen momentumu kullandığı için seçilmiştir. VAE konfigürasyonunda kullanılan ek parametreler aşağıdaki gibidir. Sızdıran Doğrultulmuş Doğrusal Birim (Leaky Rectified Linear Unit) [76], gizli katmanlarda aktivasyon fonksiyonu olarak kullanılır. Geleneksel ReLU'nun "ölen ReLU" problemini ve sigmoid / tanh'ın kaybolan gradyan problemini çözdüğü için sigmoid veya tanh yerine tercih edilir [64]. Modellenen veri tipi gerçek değerli olduğundan Gauss rekonstrüksiyon dağılımı hiperbolik tanjant (tanh) pzx aktivasyon fonksiyonu ile kullanılır [50]. Yaygın olarak kullanılan ve [49], [77] gibi çalışmaların temelini oluşturan AE konfigürasyonunda kullanılan ek parametrelerde çıktı katmanı dışında sigmoid aktivasyon fonksiyonu ve çıktı katmanında ise ortalama kare hata kaybı (mean square error loss) fonksiyonu ile soft-max aktivasyon fonksiyonu kullanılmaktadır [73]. Hem AE hem de VAE sinir ağları deeplearning4j kütüphanesi [78] ile gerçeklenmiş ve 8192 batch (toplu iş) boyutu ile 1000 epoch süresince eğitildi.

OCSVM'de γ (gamma) 0.5, C maliyet parametresi 1 ve v (nu) parametresi 0.5 olan varsayılan parametrelerle ile birlikte radyal temel fonksiyonu ise çekirdek fonksiyonu olarak kullanıldı. LibSVM kütüphanesi [79] kullanılarak model oluşturulmuştur.

Bir anormali skoru olarak VAE'de RP, AE'de RE ve OCSVM'de ise herhangi bir eşik (threshold) uygulanmadan tahmin (prediction) skoru kullanılır (puan 0'dan küçükse, örnek bir anormali aksi takdirde normaldir).



Şekil 2.8. ROC eğrileri

2.3.3. Performans değerlendirmesi

Başlangıçta, tüm değerleri [0,1] aralığına getirmek için birlik tabanlı normalleştirme (unity-based normalization) olarak da adlandırılan özellik ölçeklendirme yöntemi [80] kullanılarak veri setinin özelliklerinde normalizasyon işlemi gerçekleştirilmiştir. Yöntemlerin performansını değerlendirmek için ayrı eğitim ve test veri setleri kullanılmıştır. Sinir ağını "Pazartesi" günündeki sadece normal akış verileriyle eğitilerek modeller oluşturuldu. Bunun nedeni, modellerin denetimsiz yaklaşımla eğitilmesidir ve veri setindeki son sütun eğitimde kullanılmayan saldırı sınıfına karşılık gelir. Test süreci için hem normal trafik hem de diğer günlerdeki saldırı trafiği kullanıldı. Test sürecinde "normal" veya "iyicil" dışındaki tüm sınıflar "anomali" olarak kabul edilir. Bir saldırı sınıfının performans değerlendirmesinde diğer tüm saldırı sınıfları hariç olmak üzere yalnızca normal akış kayıtları ve o saldırı sınıfının akış kayıtları kullanılarak metrikler hesaplanır. Örneğin “FTP-Patator” saldırısının değerlendirilmesinde “1829371” normal akış kayıtları ve “7938” saldırı akış kayıtları kullanılmaktadır. Yöntemlerin bilinmeyen saldırıları normal trafikten ayırt edip edemeyeceğini anlamak için tüm saldırıları "normal" dışında tek bir "anormali" sınıfında birleştirerek başka bir değerlendirme stratejisi de gerçekleştirilir.

Yöntemlerin performans değerlendirmesi hem ROC hem de AUC ölçütleri kullanılarak gerçekleştirilir. Denetimsiz anomali tespitinde fiili standart olduğu ve yorumlanabilirliği kolaylaştırdığı için AUC ölçütüne dayalı bir değerlendirme gerçekleştirilir [81]. Daha spesifik olarak belirtmek gerekirse performans tüm eşik değerleri aralığında toplanırak tüm ROC eğrisi özetlenir [81]. Sınıf dağılımına duyarlı olmadığından ve aynı zamanda bir yöntemin farklı eşiklerde diğerinden daha iyi performans gösterebileceğini (ROC grafiğindeki farklı FP değerlerine karşılık gelir) göstermek için ROC tabanlı değerlendirme yapılmıştır.

Yöntemlerin AUC sonuçları Tablo 2.2.'de verilmiştir. Ayrıca farklı türde bilinmeyen (yani eğitilmemiş) saldırılara uygulanan algoritmanın ROC eğrileri Şekil 2.8.'de gösterilmektedir. AUC sonuçları ve ROC eğrilerinin yorumlanması aşağıda

açıklanmaktadır. AUC sonuçlarının yorumlanmasında rehber olarak bir akademik puan sistemi de kullanılmıştır.

Tablo 2.2. Belirtilen saldırılar için yöntemlerin AUC sonuçları [(1)]

	AUC (Normal vs. *Anomali*)		
Anomaly Type	**VAE**	**AE**	**OCSVM**
FTP Patator	**0.7955**	0.7429	0.5028
SSH Patator	0.6472	**0.6787**	0.4221
DoS Slowloris	**0.8752**	0.8428	0.7059
Dos Slowhttptest	**0.8664**	0.8529	0.8421
Dos Hulk	0.8155	**0.8275**	0.7361
Dos Goldeneye	**0.8088**	0.7536	0.7228
Heartbleed	0.9805	0.9816	**0.9849**
Web Attack–Brute force	**0.7436**	0.6622	0.6090
Web Attack–XSS	**0.7567**	0.6571	0.6372
Web Attack–SQL Injection	0.6664	**0.6878**	0.5695
Infiltration	**0.8965**	0.8964	0.8588
Bot	0.6197	**0.621**	0.3749
DDoS	**0.7452**	0.7221	0.5840
Portscan	**0.6753**	0.5953	0.5888
Tüm Saldırılar	**0.7596**	0.7338	0.6636

[(1)] *Kalın sayısal değerler en iyi uygulanan yöntemi gösterir.*

Tablo 2.2.'deki AUC değerleri genel olarak incelendiğinde, VAE'nin diğer iki yöntemden daha iyi performans gösterdiği ve OCSVM'nin bazı saldırılar dışında kötü performans gösterdiği anlaşılmaktadır. Dahası, bazı saldırılar her üç yöntem için de mükemmel veya iyi AUC değerlerine sahipken diğerleri, saldırıları veya anormal vakaları normalden ayırt edemediklerini gösteren zayıf veya gerçekçi olmayan performans gösterir. Çeşitli DoS saldırıları ve DDoS gibi yüksek oranlı saldırıların ortaya çıkmasında VAE'nin anormal durumları AE'den daha iyi tespit etmesinin yanısıra her ikisinin de iyi performans sergilediği görülmüştür. "SSH Patator", "Web Attack - SQL Injection", "Bot" ve "Portscan" saldırılarında her üç yöntemin de kötü performans gösterdiği fark edilir. Her biri farklı sayıda saldırı örneği içermesine ve sırasıyla kaba kuvvet, web uygulama saldırısı, bot ve bağlantı noktası taraması gibi farklı saldırı kategorilerinde yer almalarına rağmen üç yöntemin eğitim modellerininin

hiçbirinin bu saldırıları yeterince ayırt edemediği görülebilir. Sonuç olarak, modellerin performansını artırmak ve bu saldırılar arasında farklı akış tabanlı özellikler veya veriler ekleyerek veya daha gelişmiş makine öğrenimi metodolojisi kullanarak bu saldırılar arasında ayrım yapmak için ek araştırma yapılması gerekmektedir. Her üç yöntemin de “Infiltration” ve “Heartbleed” saldırılarında sırasıyla iyi ve mükemmel performanslar sergilediğini görmek ilginçtir. Bunun nedeni, belirli saldırı türlerinin veri setinde az sayıda olması olabilir. Bu nedenle bu saldırıların örnek sayısını artırarak bu varsayımı doğrulamak için daha fazla araştırma yapılması gerekir.

Tek bir AUC sonucunu ve ROC eğrisini hesaplamak için tüm saldırı türlerini bir anomali olarak birleştirerek ağda "Tüm saldırılar" gerçekleştiğinde yöntemlerin nasıl davrandığı değerlendirilebilir. AUC sonuçlarına göre, VAE orta bir performansla diğerlerine göre daha başarılıdır. AE aynı zamanda orta bir performans gösterirken OCSVM'nin performansı zayıftır. Bu durum, VAE ve AE'nin normal ağ trafiğini modelleme yeteneğinin çok çeşitli saldırı kategorilerinden oluşan on dört farklı türde bilinmeyen ihlali tespit etmede oldukça başarılı olduğunu gösterir. Ayrıca, eğitim sürecinde kullanılan veri örneklerinin sayısının (toplam veri setinin yaklaşık %19'u) test sürecinde kullanılan veri örneklerinin sayısından (toplam veri setinin yaklaşık %81'i) çok daha az olduğuna dikkat edilirse hem VAE hem de AE'nin iyi performansları, bu yöntemlerin akış tabanlı özelliklerden ihlalleri veya anomalileri tespit etme konusunda oldukça yetenekli olduğunu açıkça göstermektedir. Akış özelliklerinden normal ağ trafiği modelini oluştururken eğitimde daha çok normal veri örneklerinin kullanılması (yaygın olarak kullanılan bölme oranları eğitim için %70 ve denetimli öğrenme yaklaşımında test için %30'dur) performans oranını daha da artırabilir.

Şekil 2.8.'deki tüm ROC eğrileri incelendiğinde, aynı davranış niteliğine sahip saldırıların VAE'de tüm saldırıların için geçerli olmasa bile benzer ROC eğrileri sergilediği görülmektedir. Örneğin, Web Attack-Brute force”, “Web Attack-XSS” ve “Web Attack-SQL Injection”nun ROC eğrileri bu varsayımı haklı çıkarmaktadır. Bir başka benzer gözlem kaba kuvvet (brute force) kategorisinde “FTP Patator” ve “SSH Patator” için yapılabilir. VAE ve AE yöntemlerinin AUC değerleri iyi performans

verse de bu yöntemlerin herhangi bir ek yöntem olmaksızın gerçek yaşam senaryosunda kullanılmasını zorlaştıran işaretler ROC eğrilerinden açıkça anlaşılmaktadır. Ayrıca, tüm ROC eğrileri neredeyse tüm saldırı türlerinde (Heartbleed hariç) düşük FP değerlerinde yüksek TP değerlerinin elde edilemediğini göstermektedir. Bu gözlem, pratik bir saldırı tespit sisteminde yüksek tespit doğruluğu veya düşük yanlış alarm oranı sağlayan uygun bir eşik değerini belirlemenin kolay olmadığı şeklinde yorumlanabilir.

2.4. Sonuç

Bu çalışmada, OCSVM ile birlikte AE ve VAE derin öğrenme yöntemlerinin saldırı tespit yetenekleri yarı denetimli bir öğrenme stratejisi uygulanarak analiz edilmiştir. Modellerin oluşturulması sadece normal akış tabanlı veriler kullanılarak gerçekleştirildi. Ayrıca modellerin testleri hem normal hem de anormali içeren veri seti kullanılarak gerçekleştirilmiştir. Deneysel sonuçlar, ROC eğrileri ve AUC ölçütleri bakımından hesaplanmıştır. Sonuçlara göre, VAE'nin tespit oranı çoğunlukla AE ve OCSVM'den daha iyidir. Bununla birlikte, yöntemlerin yüksek yanlış alarmları nedeniyle denetimli öğrenme yöntemleriyle desteklenmesi gerektiğinin de belirtilmesi önemlidir. Ayrıca, bazı saldırıların özelliklerinin daha iyi modellenebilmesi için belirli zaman aralıklarında toplanan akış tabanlı özellikler yöntemlerin tespit oranını artırmak için dikkate alınabilir.

BÖLÜM 3. YAZILIM-TANIMLI AĞLAR İÇİN UYARLANABİLİR BİR SALDIRI TESPİT VE ÖNLEME SİSTEMİ

SDN, şu anda aktif olarak kullanılan geleneksel ağ altyapısının sınırlandırmalarını değiştirmeyi vaat eden yeni bir ağ paradigmasıdır [3]. Bu paradigma, yönlendirici (router) ve anahtarlar (switch) gibi trafik iletimi (forwarding) yapan cihazlardan ağın kontrol mantığını (control logic) alarak dikey bütünleşmeyi ayırarak kontrol mantığını mantıksal olarak merkezileştirilmiş denetleyiciye (veya ağ işletim sistemine) yerleştirir [4]. Ağ güvenliği açısından bakıldığında SDN, tüm ağ altyapısındaki veri akışlarının yönlendirilmesi için tek bir kontrol noktası tanımlar. Üst seviye reaktif (duyarlı) güvenlik gözetimi, analizi ve tepki sisteminin hazırlanmasıyla ağ güvenliğini artırmak için SDN mimarisinden istifade edilebilir [5], [6]. SDN paradigmasının değiştirilmiş bir versiyonu olan OpenFlow protokolü kullanılarak uygulanan ağ güvenliği uygulamaları ile basitçe izin vermek veya engellemekten ziyade karmaşık mantığı işleyebilen akışları uygulamak da mümkündür. [7]. Ağ trafiği analizi veya anormallik algılama yöntemleri, merkezi denetleyiciye iletilen güvenlikle ilgili verileri düzenli olarak üretir. Uygulamalar, tüm ağdan toplanan bu geri bildirimi analiz etmek ve ilişkilendirmek için denetleyici üzerinde çalıştırılabilir. Elde edilen analizin sonuçlarına bağlı olarak, yeni veya güncellenmiş güvenlik politikaları, akış kuralları şeklinde ağ bileşenlerine dağıtılabilir. Bu birleşik yaklaşım, ağa yönelik güvenlik tehditlerinin kontrolünü etkili bir şekilde hızlandırabilir ve bunlardan korunabilir [5]. Saldırı tespiti ve saldırı önleme veya azaltma algoritmaları OpenFlow güvenlik uygulamaları olarak uygulanabilir.

Bölümün organizasyonu şu şekilde yapılmıştır. Sonraki bölümde literatürde yer alan SDN ortamlarda akış tabanlı saldırı tespiti üzerine yapılan çalışmalar özetlenmektedir. Üçüncü bölümde, çalışmada kullanılan yöntemlere ilişkin teorik bilgiler verilmektedir.

Deneysel metodoloji ve sonuçlar dördüncü bölümde sunulmaktadır. Sonuç tespitleri bölümün sonunda açıklanmaktadır.

3.1. Kaynak Araştırması

Anomali algılama yönteminin amacı, normal trafiği tanımlamak için istatistiksel bir model üretmektir [82]. Böyle bir durumda, bu modelden herhangi bir sapma bir anomali olarak kabul edilir ve bir saldırı olarak tespit edilir. SDN ortamlarda anomali tabanlı saldırı tespiti yapabilen birçok literatürde önerilmiştir. Bu bölümde, literatürde yer alan çalışmaların kısa bir özeti Jafarian ve ark.'nın [82] yaptığı taksonomi dikkate alınarak akış sayma tabanlı şemalar, bilgi kuramı tabanlı şemalar, entropi tabanlı şemalar ve derin öğrenme tabanlı şemalar ve hibrit şemalar olmak üzere beş kategoriye ayrılarak verilmektedir. SDN ağlarda anomali tabanlı saldırı tespitiyle ilgili çalışmaların daha kapsamlı ve ayrıntılı bir derlemesi [83] ve [82] çalışmalarında bulunabilir.

Akış sayma tabanlı anomali algılama algoritmalarında ağdaki trafik akışları başlangıçta bir alt ağ önekine göre yakalanır ve buna göre toplanır. Bu işlem, sisteme ekstra yük getirmesine rağmen ağdaki anormal durumları doğru bir şekilde algılar [82]. Zhang ve ark. [84], anormallikleri tespit edebilmek için etkili teknikler sağlayan uyarlanabilir bir akış sayma şeması önermektedir. Gözetleme ek yükünü ve anomali algılama doğruluğunu daha iyi dengelemek için hem uzaysal hem de zamansal boyutlar arasında ölçümün ayrıntı düzeyini dinamik olarak değiştiren tahmin temelli bir şema önerdiler. Garg ve ark. [85], akış girdilerinin uyarlanabilir bir biçimde toplamasını ve anomali algılama şemasını önerir. Yazarlar, bu şemayı kullanarak, gözetleme (monitoring) ek yükünün azaldığını ve anomali tespit doğruluğunun arttığını gösterirler. İki adımlı algoritmayı, her bir toplamada bayrak ayarlama adımını ortadan kaldırarak ve toplama kurallarını güncelleyerek tek bir basitleştirilmiş prosedürde birleştirirler. Ha ve ark. [86] örneklenen trafiğin toplam biriktirme hacmini denetleme işlem kapasitesinin altında tutarken, kötü amaçlı trafiğin denetim kapasitesinin kullanımını tamamen iyileştiren geniş ölçekli SDN ağlarda çalışabilen bir saldırı tespit sistemi için bir trafik örnekleme stratejisi sunar. Granby ve ark. [87]

yazılım tanımlı veri merkezinde anomali tespiti için takılabilir bir yazılım platformu olan SDNPANDA'yı (Anomali Algılama Uygulamaları için Yazılım Tanımlı Ağ Platformu) tasarlar. SDNPANDA, ağ gözetleme modülü, ağ saldırı tespit modülü ve ağ saldırı yanıt modülü olan üç denetleyici uygulama modülünden oluşur. Hommes ve ark. [88] DoS saldırılarında OpenFlow anahtarlarının akış tablosu satürasyon sorununu araştırır. Çalışmada, ağın mantıksal topolojisindeki varyasyon farklı saldırı türleri için analiz edilerek ve bir tablo üzerinden uzaklık kriteri ölçülerek saldırı vakaları tespit edilmiştir. Carvalho ve ark. [89], paket akışlarının trafik analizini kullanarak SDN'de gerçek zamanlı bir anomali algılama ve önleme sistemi önerdiler. Önerilen bu şemanın dört aşaması vardır: İlk aşama, OpenFlow protokolü aracılığıyla akışları toplar. İkinci aşama, analiz edilen bölümlerin dijital imzasını oluşturmak için normal trafik davranış profilini çıkarır. Beklenen davranıştan farklı olan şüpheli trafik olaylarını belirlemek için mevcut trafik önceden oluşturulmuş trafik profiliyle karşılaştırılır. Üçüncü aşama, ağ üzerinde anomali önleme işlemi yapılır. Son aşamada ise, sistem sonuçlarını doğrulamak için belirlenen saldırıların raporları oluşturulur. Lee ve ark. [90] iyi yapılandırılmış bir geliştirme ara yüzünü dışa aktaran bir SDN anomali algılama çerçevesi olarak Athena'yı önerdiler. Athena, çok çeşitli anomali algılama hizmetlerini minimum programlama çabasıyla hızlı bir şekilde sentezlemek için genel amaçlı fonksiyonlar sağlar. Ağ veri düzleminin tamamında sofistike anomali algılama hizmetlerini desteklemek için bir anomali algılama geliştirme çerçevesini SDN altyapılarına entegre ettiler. Athena ölçeklenebilir, dağıtılmış SDN dağıtımlarıdır ve makine öğrenimi tabanlı güvenlik uygulamalarının geliştirilmesini destekler. He ve ark. [91], iki aşamalı kümeleme yöntemini kullanarak SDN'de bir anomali algılama şemasını önerdiler. İlk aşama, veri setinin gerekli özelliklerini seçmek için kullanılan bir özellik seçim şemasıdır. Uygun ve ilgili özellikleri seçtikten sonra, yoğunluk zirvesine dayalı bir kümeleme algoritması, indirgenmiş veri setini normal ve yanlış davranış kalıpları olarak sınıflandırır. Carvalho ve ark. [92] anomalileri tespit etmek ve önlemek için SDN'ye dayalı bir ekosistem sundu. Bu şema ağ trafiğini tarar ve proaktif bir biçimde önemli anomalileri tespit eder. Bu anomali algılama şeması, trafik toplama modülü, anormallik algılama modülü, önleme modülü ve raporlama modülü olmak üzere dört modülden oluşur. Peng ve ark. [93], merkezi denetime dayalı olarak SDN için bir DDoS anomali algılama şeması sundular. Bu şema, akış özelliği

vektörlerini normalleştiren bir ön işleme modülü ve bu vektörleri işleyen ve KNN yöntemini kullanan anomali algılama modülü olmak üzere iki modülden oluşur.

Bilgi kuramı yöntemleri, ağ trafiğindeki mevcut bir anomalinin trafik veri setlerinin bilgilerinde değişikliğe yol açtığı varsayımına dayanmaktadır. Mehdi ve ark. [2] yaptığı çalışmada, SDN ortamlarındaki trafik anomali tespiti üzerinde odaklanılmaktadır. SOHO (Small Office Home Office) ortamlarında birden fazla anomali tespit algoritması, uygulanabilirliklerini doğrulamak için deneysel olarak test edilmiştir. Yazarlar, OpenFlow kullanan dağıtık ağ cihazlarının merkezi olmayan kontrolünün anomalinin kaynağına yakın ağ güvenlik problemlerini tespit edebileceği ve sınırlandırabileceğinden bahsetmektedir. Çalışma, sadece düşük ağ trafik hızı kullanan TRW-CB [94], Rate-Limiting [95], Maximum Entropy Detector [96] ve NETAD [97] saldırı tespit algoritmalarının verimliliğiyle ilgili deneysel sonuçları vermektedir ve tespit edilen ağ anomalilerinin önlenmesi (mitigation) ileride yapacakları çalışmalar arasına alınmıştır. Dotcenko ve ark. [7], ağ trafiğini saldırı trafiği ve normal trafik şeklinde sınıflandırmak için ağ trafiğini analiz etme mekanizmasının yanı sıra bulanık mantığın kullanıldığı bir yöntem önerdiler. Bu çalışmada Mehdi ve ark. [2] tarafından yapılan çalışmada olduğu gibi, saldırıların özelliklerinin araştırılmasında TRW-CB ve Rate-Limiting algoritmalarının kullanılmasının yanı sıra trafiğin sınıflandırılmasında bulanık mantık kullanılmıştır. Kokila ve ark. [98], SDN ortamlarında DDoS saldırılarını tespit etme üzerine birbirinden farklı makine öğrenme algoritmalarını ve saldırı tespitindeki performanslarını inceleyen bir araştırma yaptılar. Bu çalışmada, 2000 DARPA [99] veri setini kullanan çeşitli öğrenme yöntemler araştırılmış ve SVM yönteminin önerilen tüm diğer yöntemler arasında %95 tespit oranı ve %8 yanlış alarm oranıyla en iyi performansı gösterdiğini keşfetmişlerdir. Sathya ve Thangarajan [100] veri toplamaktan ve ağ trafiğinin amaçlanan özelliklerinden ziyade NSL-KDD veri setinin anomali tespit sistemine girdi olarak verildiği SDN ağlarda anomali tespiti üzerine bir araştırma yürütmüştür. Yazarlar, normal trafik verilerini ve anomalilerini sınıflandırmak için karar ağacı (Decision Tree) yöntemini kullanmıştır.

Entropi tabanlı anomali algılama şemalarının özellikle bir veri setinin rastgeleliğini belirlemede ağ anormalliklerini tespit etmede etkili olduğu kanıtlanmıştır [82], [101]. Wang ve ark. [102], entropi düzeyine dayalı olarak düşük işlem hacmine sahip DDoS taşırma (flooding) saldırılarını tespit etmek için OpenvSwitch gibi SDN'lerin düşük katmanlı anahtarlarında uygulanabilen bir yöntem önerdiler. Bu durumda, ağ seviyesinde dağıtılmış anomali tespiti gerçekleştirilebilir ve istatistik akışının iletilmesinden denetleyiciye oluşturulan ek yük azaltılır. Giotis ve ark. [103] yaptığı çalışma ise, SDN mimarilerinde anomali tespiti ve azaltımı yapan verimli ve ölçeklenebilir bir mekanizma sunmaktadır. Bu çalışma, [2]'teki çalışmadan farklı olarak OpenFlow istatistikleri yerine sFlow [104] gözetleme (monitoring) verisini kullanarak denetleyici işlem yükünü azaltan ve dolaysıyla daha yüksek hat hızlarında çalışan bir mekanizma sunmakta ve bunların deneysel sonuçlarını vermektedir. Ayrıca anomali tespit edildiğinde OpenFlow protokolü kullanılarak akış tablolarında modifikasyonlar yaparak başarılı bir şekilde ağ anomalisinin azaltıldığı (mitigation) gösterilmektedir. Çalışma, sFlow örnekleme oranına bağlı olarak 500 Mbps'a (1/256 örnekleme) kadar iletmeyi yönetebilen bu mekanizma, günümüz gigabit ağları için yine yetersiz kalmaktadır. Ayrıca örneklenen istatistiksel veriler anomali tespit algoritmasına girdi olarak verildiğinde anomali tespit oranının [2]'tekine göre daha düşük olduğu görülmektedir. Francois ve Festor [105], SDN'de anomali tespiti için anahtar gibi her bir ayrı cihazın akış girdileri SDN denetleyicisi tarafından yakalanır. Denetleyicini sağladığı avantajlara bağlı olarak, saldırı algılandıktan ve saldırı paketlerinin özellikleri belirlendikten sonra saldırının tespit edildiği anahtarda gözetleme gerçekleştirilecektir. Sonraki atlama (next-hop) anahtarının akış girdileri ve saldırı paketlerinin özellikleri saldırı akışlarının geldiği olası yolu tespit etmek için kullanılır.

Derin öğrenme yöntemleri, bilgiye tabanlı öğrenmeye dayalı bir geliştirme algoritmasını içerir ve derin öğrenmeyle ilgili anahtar kelimeler şunlardır: yapay zeka, denetimsiz makine öğrenimi ve çoklu katman öğrenme [106][82]. Öte yandan, derin öğrenme mekanizması, özellikleri ve işlem verilerini kendi başına öğrenme kabiliyeti açısından SDN'lerde uyarlanabilirlik için güçlü bir potansiyele sahiptir [107]. Dey ve Rahman [107], SDN ortamla için derin öğrenmeye dayalı bir ağ ihlal tespit sistemi

önerir. Geçitli Tekrarlayan Ünite Uzun Kısa Süreli Belleğin (Gated Recurrent Unit Long Short Term Memory), ANOVA F-Testi ve REF özellik seçim şeması ile farklı performans değerlendirme parametrelerine dayanan en iyi sınıflandırıcı olduğunu gösterdiler. Niyaz ve ark. [108], SDN ortamlar için ve üç modülden oluşan bir DDoS algılama sistemi önerir. Bu şemada, özellik seçimi ve trafik düzenlemesi için derin öğrenme mekanizması kullanılmıştır. Tang ve ark. [109], SDN'de akış tabanlı anomali tespiti için derin öğrenmeye dayalı bir yöntem önerir. İhlal tespit sistemi için NSL-KDD veri setinin 41 özelliğinden 6 temel özelliğini seçerek bir derin sinir ağı modeli oluşturdular. Garg ve ark. [110], SDN'de şüpheli akışları tespit etmek için hibrit ve gerçek zamanlı çalışan derin öğrenmeye dayalı bir anomali tespit yöntemi önerdiler. Bu yöntem, anormal etkinlikleri tespit etmek için geliştirilmiş kısıtlı Boltzmann makinesi ve gradyan azalma tabanlı destek vektör makinesinden yararlanan bir anomali algılama modülü olan iki modül ve SDN'nin sıkı hizmet kalitesi gereksinimlerini karşılamak için bir uçtan uca veri dağıtım modülünden oluşur. Li ve ark. [111], SDN tabanlı 5G ağlar için akıllı bir hibrit IDS önerdiler. Önerilen sistemde, özellik seçimi için rastgele orman (random forest) yaklaşımının yanında akış sınıflandırması için k-ortalama (k-means) yöntemini kullanıldı. Yazarlar, KDD99 veri setinden seçilen 23 özelliği kullandıklarını belirtmişlerdir.

Bahsi geçen tüm yöntemlere ek olarak SDN ağlarda anomali tabanlı ihlal tespiti için literatürde hibrit şemalarda önerilmiştir. Santos Da Silvo ve ark. [112], SDN için bir anomali algılama ve makine öğrenimi trafik sınıflandırması önerdiler. Anomali tespiti ve sınıflandırması, potansiyel olarak kötü niyetli akışları hızlı bir şekilde tespit etmek için düşük hesaplama maliyeti yöntemlerinin daha sık uygulandığı hafif (lightweight) bir aşama ve bu tür akışların anormal davranışlarına göre sınıflandırılıp analiz edildiği ağır (heavyweight) bir aşama olan iki tamamlayıcı aşamada gerçekleştirilir. Hafif aşamada entropi analizine dayanan bir bilgi kuramı yaklaşımını kullandılar. Öte yandan, ağır aşamada, geçmiş anormalliklerle ilgili tarihsel bilgilerden yararlanarak düzensiz trafiği kategorize etmek için SVM'yi (Destek Vektör Makinesi) kullandılar. Pang ve ark. [113], SDN'de yeni, yüksek verimli ve yüksek doğrulukta çalışan anomali tespit şeması olan FADE'yi önerdiler. Önerilen FADE şeması, minimum akış setinin akış istatistiklerini doğru bir şekilde analiz ederek iletme anomalilerini tanımlar.

FADE, istatistiklerini doğru bir şekilde ölçmek için bu akışlarla ilgili az sayıda özel akış kuralı üretir. FADE, bu ayrılmış akış kurallarının yüklenmesini ve zaman aşımını kontrol eder. Dolayısıyla, aynı akış için üretilen tüm özel akış kuralları aynı paket seti üzerinde çalışır. Seungwon ve Guofei'nin [114] yaptığı çalışmada, geniş ve dinamik bulut ağlarına özel gözetleme (monitoring) hizmetleri sağlayan Cloudwatcher isimli yeni bir çatı (framework) önerilmekte ve basit bir poliçe scripti yazılarak gözetlenecek paketlerin önceden kurulu ağ güvenlik cihazları üzerinden geçişi sağlanacak şekilde saptırılması gibi tüm işlemler gerçeklenebilmektedir. Poliçeler bulut operatörü tarafından yazıldığından saldırılara anında önlem alınması yapılmamaktadır. Cui ve ark. [115], SDN'de DDoS'nin üstesinden gelmek için dört modül içeren bir şema önermişlerdir. Bu modül, saldırı algılama tetikleyicisi, saldırı algılama, saldırıyı geri izleme (traceback) ve saldırı önlemeden oluşur. Saldırı tespit şemasının tetikleyicisi, DDoS saldırısına daha hızlı yanıt vermekten sorumludur. Ek olarak, bu modül denetleyicilerin ve anahtarların iş yükünü azaltır. DDoS saldırısını tespit etmek için sinir ağı tabanlı şema kullanılır. Ayrıca, yazarlar SDN'ler için bir saldırı geriye dönük izleme için yöntem sundu. Braga ve ark. [116] tarafından yapılan çalışmada OpenFlow kullanılarak ağ cihazının iletme düzlemi yönetilmekte ve akış istatistikleri toplanarak sadece veri düzlemine yapılan Distributed Denial of Service (DDoS) saldırıları üzerinde yoğunlaşılmaktadır ve sunulan performans analizi sadece önerilen saldırı tespit algoritmasıyla sınırlı olup sistemin tüm performansıyla bilgi yer almamaktadır. Saldırı tespitinde kullanılan yöntem, bir denetimsiz (unsupervised) yapay sinir ağı çeşidi olan Self Organizing Maps (SOM)'dur. Bu yönteme girdi olarak verilen parametreler trafik akışları istatistikleridir. Ayrıca veri düzlemi trafik anomalilerinin OpenFlow kontrol düzlemine olan etkisi hakkında herhangi bir şeye değinilmemiştir.

3.2. Temel Bilgiler

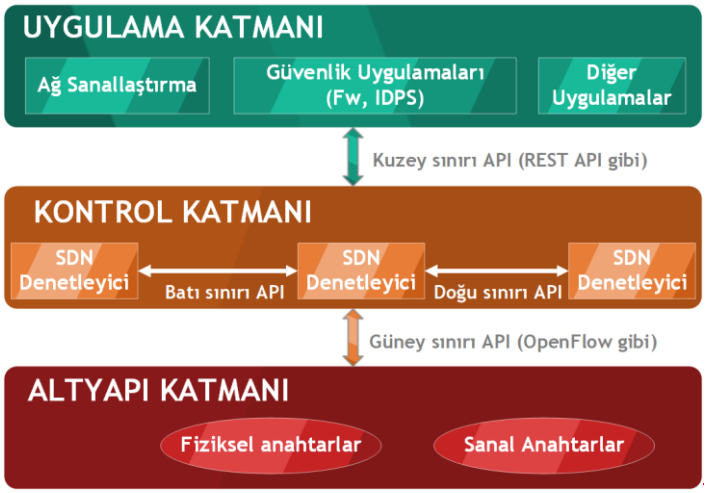
Bu bölümde, Yazılım-Tanımlı Ağ (SDN) mimarisinin temel bileşenlerinden bahsedilerek geleneksel ağlardaki problemleri nasıl çözdüğü üzerinde durulmaktadır. Ayrıca geleneksel ağlara yapılan ağ saldırı çeşitleri ve bu saldırıları tespit etme ve önlemek için geliştirilen Saldırı Tespit ve Önleme Sistemleri (Intrusion Detection and Prevention Systems - IDPS) anlatılarak şu anki ağlardaki dezavantajları sunulmaktadır.

3.2.1. Yazılım tanımlı ağ

Open Networking Foundation'a (ONF) [117] göre, SDN ağ kontrol ve iletme fonksiyonlarını birbirinden ayıran yeni bir mimaridir. Bu, ağ kontrolünün doğrudan programlanabilir olmasına ve altyapının uygulamalar ve ağ hizmetleri için soyutlaştırılmasına olanak sağlar. Böyle bir mimaride, altyapı cihazları, kontrol katmanında bulunan bir denetleyici tarafından önceden tanımlı program mantığına göre anlık olarak üretilen birtakım kurallara göre gelen paketleri işleyen basit iletme motorları olarak görev yaparlar. Denetleyici genellikle uzak bir sunucuda çalışır ve bir takım standartlaşmış komutlar kullanarak iletme elemanlarıyla güvenli bir bağlantı üzerinden haberleşir. ONF, SDN için dikey olarak üç ana fonksiyonel katmana ayıran üst-seviye bir mimari [118] sunar:

- Altyapı Katmanı (Infrastructure Layer): Veri düzlemi (Data plane) olarak da bilinen bu katman, bir açık arayüzle erişilebilen fiziksel ve sanal anahtarları kapsayan İletme Elemanlarından (Forwarding Elements) oluşur.
- Kontrol Katmanı (Control Layer): Kontrol düzlemi (Control plane) [118] olarak da bilinen bu katman, bir açık arayüz üzerinden ağ iletme davranışını yönetmek için açık API'ler üzerinden birleştirilmiş kontrol fonksiyonelliği sağlayan bir takım yazılım-tabanlı SDN denetleyicilerden oluşur. Üç haberleşme arayüzü, denetleyicilerin birbirleriyle etkileşim içine girmesine izin verir: Güneysınırı (Southbound), Kuzeysınırı (Northbound) ve Doğu/Batısınırı (East/Westbound)
- Uygulama Katmanı (Application Layer): Bu katman, ana olarak SDN haberleşmesi ve ağ servislerini kullanan son-kullanıcı uygulamalarından oluşur.

Şekil 3.1.'de kontrol katmanı, uygulama katmanı, altyapı katmanı ve bu üç katman arasındaki haberleşme arayüzleri gibi bu mimarideki anahtar kısımlar detaylı bir şekilde gösterilmektedir.



Şekil 3.1. SDN mimarisi [119]

SDN denetleyici, bu üç katmanla üç açık arayüz (open inerface) ile etkileşimde bulunur:

- Güneysınırı (Southbound): Bu haberleşme arayüzü, denetleyicinin altyapı katmanında bulunan iletme elemanlarıyla etkileşimde bulunmasına imkân sağlar. ONF'nin idame ettirdiği OpenFlow protokolü, ONF'ye göre SDN çözümleri üretmek için temel bir elemandır ve böyle bir etkileşimin umut verici bir gerçeklenmesi olarak görülebilir.
- Kuzeysınırı (Northbound): Bu haberleşme arayüzü, uygulama katmanındaki uygulamalar tarafından kullanılması için denetleyicilerde yer alan evrensel ağ soyutlama ve diğer fonksiyonellikleri açığa vurarak denetleyicilerin programlanabilirliğine imkân verir. Bu, protokolden daha çok ağı programlamaya ve yönetmeye izin veren yazılım API'si olarak görülmektedir. Bunun için bir standartlaşma çabası olmamakla birlikte, birçok marka, uygulamaların kullanabilmeleri için REST-tabanlı API'leri denetleyicilerine bir programlama arayüzü sağlamak için sunmaktadır.
- Doğu/Batısınırı (East/Westbound): Haberleşme arayüzü olarak düşünülen bu arayüz, henüz kabul görmüş bir standart tarafından desteklenmemektedir. Bu, esasen yüksek elverişlilik için durumu senkronize etmede denetleyici grupları arası haberleşmeye olanak tanımak için düşünülmüştür.

SDN mimarisinde kullanışlı olabilmesi için, iletme elemanlarının (genellikle anahtarlar) bir güneysınırı API'sini (özellikle OpenFlow) desteklemesi gerekir. OpenFlow anahtarlar iki çeşitte karşımıza çıkmaktadır: Yazılım-tabanlı (örneğin, Open vSwitch) ve donanım-tabanlı gerçeklemeler (örneğin. NetFPGA). Yazılım anahtarları genellikle çok iyi tasarlanmış ve tüm özellikleri içerirler. Fakat, en son gerçeklenenleri bile yavaş olmaktan muzdariptir. Donanım-tabanlı OpenFlow anahtarlar genellikle ASIC'ler olarak gerçeklenmektedir. Çok sayıda port için hat hızında iletme sağlamalarına rağmen yazılım gerçeklemelerinden farklı olarak esneklik ve özellik tamamlığından yoksundurlar.

OpenFlow-etkin anahtar üç ana elemana [120] bölünebilir. Bu elemanlar; donanım katmanı (veya veri yolu), yazılım katmanı (veya kontrol yolu) ve OpenFlow protokolüdür:

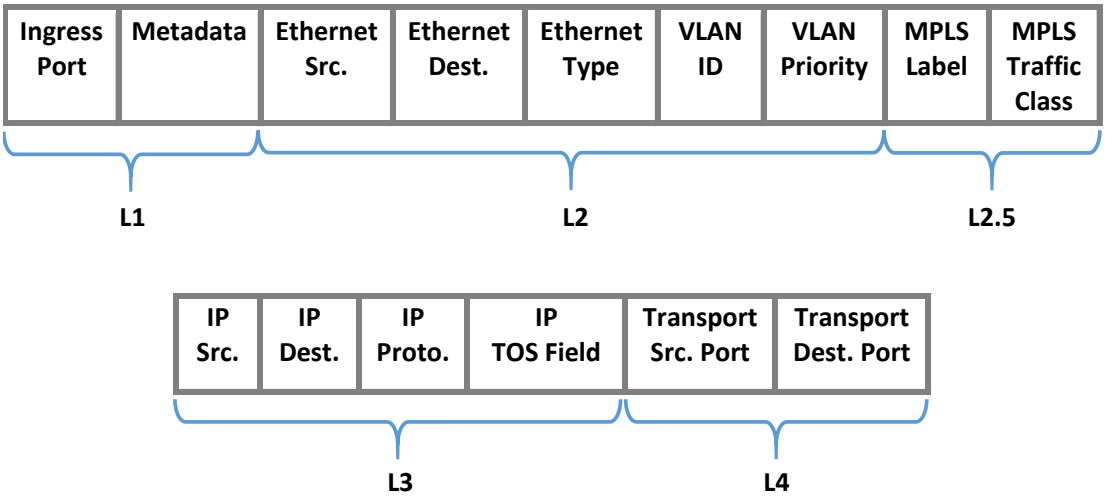
a. Veri yolu, paket araması ve iletmesi yapan bir veya daha fazla akış tablosu ve grup tablosundan oluşur. Bir akış tablosu, akışı nasıl işleyeceğini anahtara söyleyen eylemlerle ilişkili akış girdilerinden oluşur. Akış tabloları genellikle denetleyici tarafından doldurulur ve akış iletmenin ek yöntemlerini açıklamaya izin verir.
b. Kontrol yolu, işaret ve programlama amaçları için anahtarı denetleyiciye bağlayan bir kanaldır. Komutlar ve paketler, OpenFlow protokolü kullanılarak bu kanal aracılığıyla değiştirilir.
c. OpenFlow protokolü, anahtarlar ve denetleyiciler arasında haberleşmeyi sağlar. Değiştirilen mesajlar alınan paketler, gönderilen paketler, istatistik toplama, belirli akışlarda gerçekleştirilecek eylemler… v.b. gibi bilgileri içerebilir.

OpenFlow-etkin bir anahtardaki birkaç alandan oluşan bir akış tablosu girdisi aşağıdaki gibi sınıflandırılabilir:

a. Eşleşme alanları (Matching Fileds), bir 15-tuple paket başlığı, ingress portu ve opsiyonel olarak paket meta-datasına dayalı paketleri eşleştirmek için

kullanılır. Şekil 3.2.'de OSI L1-4 katmanlarına göre gruplandırılmış paket başlık alanları gösterilmektedir.

b. Akış girdisi önceliği (Priority of flow entry), akış girdisinin eşleşme sıralamasını önceliklendirir.
c. Eylem seti (Action set), başlık eşleştiğinde paketlerde gerçekleştirilecek spesifik eylemleri gösterir.
d. Sayıcılar (Counters), akış istatistikleri kaydı tutmak için kullanılır. (Her bir akıştaki paket ve bayt sayısı ve son paketin akışla eşleştiği zaman).
e. Süre aşımları (Timeouts), maksimum süreyi veya akışın anahtar tarafından geçersiz kılınmadan önceki boş süreyi tanımlar.



Şekil 3.2. OpenFlow'da akış [119]

OpenFlow mesajları üç ana tipte kategorize edilebilir. Bunlar; denetleyici-anahtar arası, asenkron ve simetriktir. Denetleyici tarafından başlatılan ve anahtarların durumunu gözetlemek veya yönetmek için kullanılan mesajlar denetleyici-anahtar arası mesajlardır. Bir anahtar ağ olaylarında denetleyiciyi güncellemek için asenkron mesajlar başlatabilir ve anahtarın durumunu değiştirir. Son olarak simetrik mesajlar, herhangi bir istek olmadan anahtar ya da denetleyici tarafından başlatılır. Örneğin, denetleyici-anahtar bağlantısı canlılığını kontrol etmek için bu tip mesaj kullanılır. OpenFlow anahtara bir ingress paketi varır varmaz boruhattı işlemeye göre akış tablolarında bir arama gerçekleştirilir. Akış tablosu girdisi, eşleşen alanlar ve önceliğine göre tespit edilir. Eğer paketteki değerler, girdideki alanların değeriyle

eşleşiyorsa paket gelen akış tablosu girdisiyle eşleşmiş olur. ANY (alan yoksa veya wildcard alanı) değerli bir akış tablosu girdi alanı, başlıktaki tüm mümkün değerlerle eşleşir. Sadece en yüksek önceliğe sahip akış girdisi seçilmelidir. Aynı önceliğe sahip birden fazla akış girdisi eşleşmesi durumunda seçilen akış girdisi açıkca tanımsızdır. Böyle bir senaryoya çözüm için, OpenFlow spesifikasyonu, yeni akış girdisinin var olan bir girdiyle eşleşip eşleşmediğini doğrulamaya imkân veren opsiyonel bir mekanizma sunar. Bu şekilde, bir paket, bir akışla (mikroakış) tam olarak eşleştirilebilir, wildcard alanlarıyla bir akışa (makroakış) eşleştirilebilir veya herhangi bir akışla eşleşmez. Eşleşmenin bulunması durumunda, eşleşme akış tablosu girdisinde tanımlanan birtakım eylemler gerçekleştirilir. Eşleşme bulunmaması durumunda ise, anahtar paketi (veya sadece başlığı) karar vermesi için denetleyiciye iletir. Yönetim düzleminde yer alan ilişkili ilkeye (poliçe) danışıldıktan sonra, anahtarın akış tablosuna yeni girdi eklenmesi için denetleyici anahtara cevap verir. Sonuncu girdi, aynı akıştaki sonraki paketlerin yanı sıra kuyruğa alınmış paketi de yönetmek için anahtar tarafından kullanılır.

Denetleyici, SDN ağların çekirdeğini oluşturarak bir uçtaki ağ cihazları ile diğer uçtaki uygulamalar arasında oturur. SDN denetleyici, anahtar cihazlarına akış girdilerini yükleyerek ağdaki her akışı yönetme sorumluluğuna sahiptir. Proaktif ve reaktif olmak üzere iki farklı akış kurulum (setup) modu vardır. Proaktif ayarda, akış kuralları, akış tablolarına önceden yüklenir. Bu yüzden akış kurulum (setup) işlemi, bir akışın ilk paketi OpenFlow anahtara varmadan önce yapılır. Proaktif kurulumun ana avantajları, denetleyiciyle irtibat kurma sıklığının azalması ve dolayısıyla kurulum gecikmesinin göz ardı edilebilir derecede düşük olmasıdır. Bununla birlikte, anahtarların akış tablolarını taşırabilir (overflow). Reaktif kurulum moduna gelince, bir akış kuralı, denetleyici tarafından akış tablosuna ancak ve ancak akış tablosunda bir girdi bulunmadığında yerleştirilir ve bu da bir akışın ilk paketi OpenFlow anahtara varır varmaz gerçekleştirilir. Bu yüzden sadece bir paket anahtar ile denetleyici arasındaki haberleşmeyi tetikler. Bu akış girdileri, önceden tanımlı bir durgunluk süreaşımından sonra geçersiz kılınır ve tablodan silinir. Akış kurulum isteğine cevap vermek için, denetleyici ilk önce uygulama katmanındaki ilkelerle bu akışı kıyaslar ve yapılması

gereken eylemlere karar verir. Ondan sonra bu akış için bir yol hesaplar ve bu yola ait olan her bir anahtara isteği başlatan da dahil olmak üzere yeni akış girdilerini yükler. Anahtarlardaki trafiğe genel bir bakış atmak istenirse, denetleyici ile anahtarlar arasında istatistikler iletilir. Anahtardan denetleyiciye istatistik göndermenin iki yolu vardır. Bunlar, itme-tabanlı (push-based) ve çekme-tabanlı (pull-based) akış gözetlemedir. İtme-tabanlı yaklaşımda, yeni bir akış oluşturma, süreaşımı veya oluşan boşluklardan dolayı tablo girdisini kaldırma gibi belirli olaylar hakkında denetleyiciyi her bir anahtarın bilgilendirmesi için istatistikler gönderilir. Bu mekanizma, girdi süreaşımından önce akış davranışı hakkında (öyle ki bu akış tarifeleme için uygun değildir) denetleyiciye bilgi vermez. Çekme-tabanlı yaklaşımda, denetleyici verilen bir akış spesifikasyonuyla eşleşen birtakım akışlar için sayıcıları biriktirir. Bu yaklaşım aynı zamanda isteğe bağlı olarak bir özel sembol (wildcard) spesifikasyonuyla örtüşen tüm akışların toplandığı bir rapor istek edebilir. Bu anahtar-denetleyici arası bant genişliğinden tasarruf ederken, diğer taraftan denetleyicinin belirli akışların davranışları hakkında çok fazla şey öğrenmesini etkisiz kılar. Çekme-tabanlı yaklaşım, istatistik toplamaya dayalı ölçeklenebilirlik ve güvenilirlik işlevlerini etkileyebileceği için, denetleyicinin yaptığı istekler arasındaki gecikmede iyileştirme gerektirir.

Literatürde en çok öne çıkan OpenFlow denetleyicileri örnek olarak NOX [121], POX, Beacon [122], Floodlight [123] ve OpenDaylight [124] örnek olarak verilebilir. Bu denetleyicilerin birçoğu OpenFlow versiyon 1.0'ı şu anda desteklemektedir.

OpenFlow protokolünü ve SDN protokollerini destekleyen anahtarların kullanılmasına olanak sağlayan ns-3 platformunda entegre edilen OpenFlow-etkin anahtarlar [125], fs-sdn [126] ve EstiNet [127] gibi simülasyon araçları ve Mininet [128], Mininet-HiFi [129], Mininet-WiFi [129] ve MaxiNet [130] gibi emülasyon araçları geliştirilmiştir. SDN tabanlı ağların modellenmesine imkân sağlayan ilk emülasyon ortamı olan Mininet'in amacı, SDN protokollerini ve diğer denetleyici uygulamalarını prototipleme ve değerlendirme için hızlı ve basitlik sağlamaktır. Mininet, sanallaştırılmış kapsayıcılarda yazılım tabanlı OpenFlow anahtarları kullanır ve bu şekilde, donanım tabanlı OpenFlow anahtarlarının kesin eşdeğer anlamlarını sağlar.

Başka bir deyişle, Mininet'te uygulanan ve test edilen denetleyici veya uygulamaları, herhangi bir değişikliğe gerek kalmadan gerçek bir OpenFlow özellikli ağda konuşlandırılabilir. Bu araçların özellik tabanlı karşılaştırılması [131]'de bulunabilir.

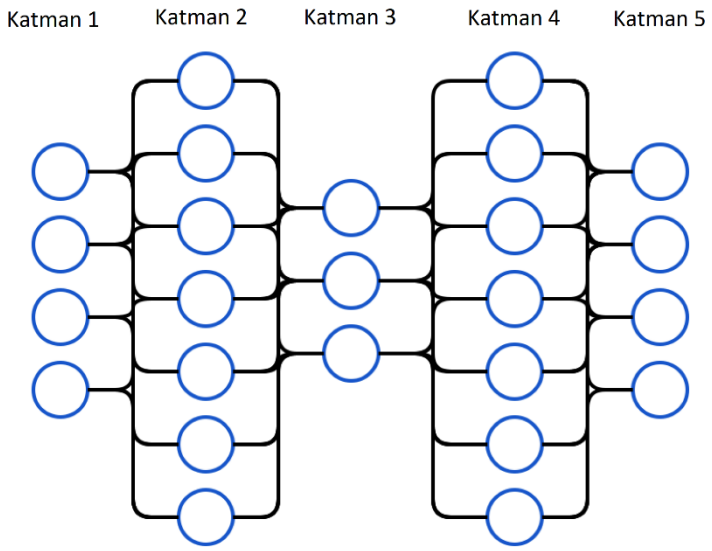
3.2.2. Anomali tespiti

3.2.2.1. Tekrarcı sinir ağı tabanlı anomali tespiti

Görüntü ve konuşma işlemedeki çeşitli uygulamalar veri sıkıştırma yetenekleri için Tekrarcı Sinir Ağı'nı (Replicator Neural Network - RNN) kullanmış olsa da [132][133], anomali tespit aracı olarak kullanımını öneren ilk çalışma Hawkins et al. [134] tarafından önerilir.

RNN, olağan regresyon modelinin bir varyasyonudur. RNN'ler sinir ağlarıdır ve başlangıçta bir sıkıştırma tekniği [133] olarak önerilen otomatik kodlayıcıların [135] belirli örnekleridir. Normalde, girdi vektörleri, çok katmanlı algılayıcı sinir ağlarında istenen çıktı vektörleriyle eşlenir. Bununla birlikte, RNN için, girdi vektörleri aynı zamanda çıktı vektörleri olarak da kullanılır. Diğer bir deyişle RNN, çıktıdaki girdi modellerini yeniden üretmeye çalışır. Eğitim sırasında, RNN'nin ağırlıkları, tüm eğitim modelleri için ortalama kare hatasını (veya ortalama rekonstrüksiyon hatasını) en aza indirecek şekilde ayarlanır. Sonuç olarak, ortak modellerin eğitimli RNN tarafından iyi bir şekilde yeniden üretilmesi daha olasıdır, böylece aykırı değerleri temsil eden modeller eğitimli bir RNN tarafından daha az iyi yeniden üretilecek ve daha yüksek bir rekonstrüksiyon hatasına sahip olacaktır. Rekonstrüksiyon hatası, bir verinin dışta kalmasının ölçüsü olarak kullanılır.

Standart RNN, giriş ve çıkış katmanı dahil olmak üzere 5 katmana sahiptir. Bir RNN'nin nasıl göründüğüne dair bir örnek Şekil 3.3.'te gösterilmektedir [136]. Katmanlar 2 ve 4 tamamen bağlıdır ve bir sigmoid veya tanh aktivasyon fonksiyonu kullanır. Bu iki katmanın aynı sayıda birime sahip olması gerekmez. Katman 3, tipik olarak giriş katmanından daha az birime sahiptir ve N adımlı [134] basamak aktivasyon fonksiyonunu kullanır. Giriş katmanındaki (Katman 1) ve çıktı katmanındaki (Katman 5) birim sayısı aynı olmalıdır.



Şekil 3.3. Tekrarcı sinir ağının örnek bir diyagramı [136]

Cordero et al., ağ akışlarındaki anomalileri tespit etmek için RNN kullanan bir yöntem önerir [136]. Bu yöntemde öncelikle normal ağ akışını temsil eden modelini oluşturmak için bir RNN [134] kullanılır. RNN'ler, girdi ve çıktı katmanları dahil olmak üzere toplam beş katmana sahip, tamamen bağlı çok katmanlı algılayıcılardır. Bu çok katmanlı algılayıcının amacı, gözlemlenen girdilerin nasıl kopyalanacağını öğrenmektir. Bu nedenle, çıkışların sayısı girişlerin sayısı ile aynıdır. Orijinal RNN'nin yalnızca üç katmana indirilerek [137] basitleştirilebileceği gösterilmiş olsa da, aşırı uydurmayı önlemek için kullandığımız dropout düzenleme (dropout regularization) tekniği [138], orijinal beş katman mevcut olduğunda daha iyi sonuçlara yol açar. Dropout, eğitim sırasında olasılıkla birimleri ve bağlantılarını sinir ağından kaldırır. Bu, ağı, ağ birimlerini birlikte uyarlamaya daha dayanıklı hale getirir [138]. Başka bir deyişle, ağ birimleri başka bir özelliğin varlığına bağlı öğrenme özelliklerinden kaçınır ve bunun yerine genel olarak çalışan yararlı özellikleri öğrenir. Dropout, 2 ve 4. katmanlara uygulanır.

Bir RNN'deki tüm katmanlar tamamen bağlıdır. Katman 2 ve 4, tanh doğrusal olmayan bir aktivasyon fonksiyonuna sahiptir. Çıktı katmanının doğrusal veya özdeş bir aktivasyon fonksiyonu vardır. Orijinal RNN ile burada uygulanan arasındaki tek fark, orta katmanın (Katman 3) aktivasyon fonksiyonunda yatmaktadır. Orijinal RNN,

teorik olarak veri örneklerini küme bileşenleriyle temsil ederek girdi verilerinin boyutluluğunu azaltmaya çalışan adım adım bir aktivasyon fonksiyonu kullanır. Daha spesifik olarak, her küme, verilerin üzerinde bulunduğu bir eksen haline gelir. Adımsal (stepwise) aktivasyon fonksiyonu ilginç teorik yeteneklerine rağmen, gradyan azalmasına dayalı geri yayılma (backpropagation) yöntemleri adım adım (stepwise) fonksiyonuyla iyi çalışmaz [136]. Gradyan azalma (gradient descent) yöntemleri, arama sürecini yönlendirmek için gradyanı kullanır. Aşamalı fonksiyonlardaki gradyanın bileşenleri neredeyse her zaman sıfır olduğundan, öğrenme süreci durur. Bu aktivasyon fonksiyonu yerine, RNN'lerin orta aktivasyon fonksiyonu olarak iyi çalıştığı kanıtlanmış standart bir sigmoid aktivasyon fonksiyonu kullanır [139].

Diğer çok katmanlı algılayıcılar gibi, RNN'ler örnek veri yığınlarıyla (batch) eğitilebilir. RNN modellerini eğitmek için kullanılan yığınlar (batch), farklı ağ akışlarından çıkarılan özelliklerden oluşur. Kullanılan araçlara bağlı olarak eğitim için ağdan birçok farklı özellik çıkartılabilir. Seçilen özelliklerin sayısı, girdi sinirlerinin (ve çıktı sinirlerinin) sayısını belirler. Çıkartılan akış özellikleri her eğitim adımında veya çağında (epoch) RNN'ye girdi olarak verilir. Öğrenme sürecinin genelleme yeteneklerini değerlendirmek için bir doğrulama seti kullanılır. RNN eğitildikten sonra, RNN, anomali skorları (AS) hesaplandığı bir normallik modeli olarak kullanılabilir. Belirli bir eşiğin üzerindeki AS'ler anormal olarak kabul edilir. AS, öğrenme probleminin daha resmi bir modelini tanıttıktan sonra resmi olarak tanımlanır.

Bir RNN'nin ağırlıkları, normal modelinin parametrelerini tanımlar. Tüm N tane örnekten oluşan eğitim setini $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ olsun. D özellikten oluşan bir akış $\mathbf{x} = \{x^1, x^2, \dots, x^D\}$ şeklinde ve RNN'nin çıktısı ise $f(\mathbf{x}) = \hat{\mathbf{x}} + \vec{\epsilon}$ şeklinde temsil edilsin. $\hat{\mathbf{x}}$ vektörü, $\mathbf{x}$'in rekonstrüksiyonu ve $\vec{\epsilon} = \{\epsilon^1, \epsilon^2, \dots, \epsilon^D\}$ vektörü ise rekonstrüksiyonun kalıntıları (residual) veya hata bileşenleridir. Sinir ağının ağırlık parametreleri, bir gradyan azalma (gradient descent) tekniği ile geri yayılma kullanılarak güncellenir. Bu tür teknikler, örneğin, standart Stokastik Gradyan Azalması (SGD) veya Nesterov momentumlu SGD gibi uyarlanabilir varyasyonları

içerir [140]. Öğrenme sürecinde en aza indirilen kayıp fonksiyonu Denklem 3.1'deki gibi tanımlanır.

$$\mathbf{L} = \frac{1}{N}\sum_{i=1}^{N} \left(\mathbf{x}_i - (\hat{\mathbf{x}}_i + \vec{\epsilon}_i)\right)^2 \quad (3.1)$$

Gradyan azalma ile geri yayılımın amacı $\boldsymbol{L}$ 'i en aza indirmek olduğundan, ağ $\hat{\mathbf{x}} \approx \mathbf{x}$ ve $\epsilon \approx 0$ gibi bir ağırlık kombinasyonu bulmaya çalışır. Önemsiz kimlik çözümünü (trivial identity solution) öğrenmekten kaçınmak için $f(\mathbf{x}) = \mathbf{x}$, bırakma (dropout) tekniğiyle [138] her öğrenme yinelemesinde birimleri rastgele kaldırarak ağ boyunca gürültü eklenir.

$\vec{\epsilon} = \mathbf{x} - \hat{\mathbf{x}}$ kalıntı değeri (residual value), bir özellik setinin ne kadar anormal olduğunu belirleyen AS'yi hesaplamak için kullanılır. Akış özellikleri olan $\mathbf{x}$ setinin AS'si Denklem 3.2'deki gibi tanımlanır.

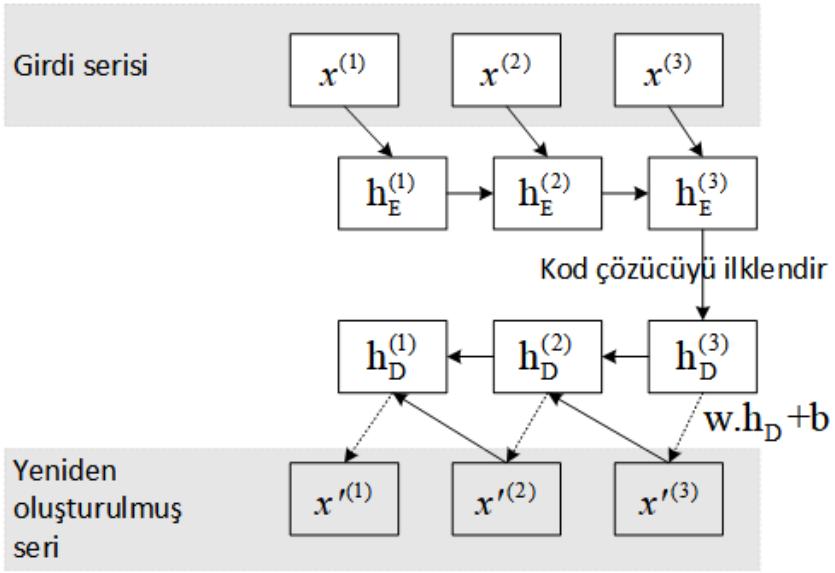
$$AS(\mathbf{x}) = \frac{1}{D}\sum_{i=1}^{D} \left(x^i - \hat{x}^i\right)^2 = \frac{1}{D}\sum_{i=1}^{D} \left(\epsilon^i\right)^2 \quad (3.2)$$

Bir $\mathbf{x}$ ağ akışının anormal olup olmadığına karar vermek için, AS'nin bir akışın normal kabul edilemeyeceği kadar yüksek olup olmadığını belirlemek için bir eşik seçilir. Bu eşik değerini, aykırı değerler kaldırıldıktan sonra eğitim sırasında gözlemlenen en büyük E rekonstrüksiyon hatası olarak belirlenmiştir.

3.2.2.2. Anomali algılama için LSTM tabanlı kodlayıcı kod çözücü

LSTM ağları [141], el yazısı tanıma, konuşma tanıma ve duygu analizi gibi birçok dizi öğrenme görevi için kullanılan tekrarlayan modellerdir. LSTM kodlayıcı-kod çözücü modelleri, makine çevirisi gibi sıralı öğrenme görevleri için yakın zamanda önerilmiştir [142][143]. LSTM tabanlı bir kodlayıcı, bir giriş dizisini sabit boyutluluğun vektör temsiline eşlemek için kullanılır. Kod çözücü, hedef diziyi üretmek için bu vektör gösterimini kullanan başka bir LSTM ağıdır. Doğal dil üretimi ve yeniden inşası [144], ayrıştırma [145], resim yazısı [146] için başka varyantlar önerilmiştir.

Malhotra ve ark. [147], zaman serilerinde anomali algılama için EncDecAD adında LSTM tabanlı bir Kodlayıcı Kod Çözücü şeması önerir. Bu sinirsel ağ mimarisinde kodlayıcı, giriş zaman serisinin bir vektör gösterimini öğrenir ve kod çözücü, zaman serisini yeniden oluşturmak için bu gösterimi kullanır. LSTM tabanlı kodlayıcı kod çözücü, hedef zaman serisinin giriş zaman serisinin kendisi olduğu "normal" zaman serilerinin örneklerini yeniden oluşturmak için eğitilir. Daha sonra, herhangi bir gelecek zamandaki rekonstrüksiyon hatası o noktada anomali olasılığını hesaplamak için kullanılır. Yalnızca normal diziler kullanılarak eğitilen bu tür bir kodlayıcı-kod çözücü modelinin çoklu sensör zaman serilerindeki anomalileri tespit etmek için kullanılabileceği gösterilir. Buradaki önsezi, kodlayıcı-kod çözücü çiftinin yalnızca eğitim sırasında normal örnekleri görmüş ve öğrenmiş olmasıdır. Anormal bir dizi verildiğinde onu iyi bir şekilde yeniden oluşturmada başarısız olur. Bu nedenle, normal diziler için oluşan rekonstrüksiyon hatalarına kıyasla anormal diziler daha yüksek rekonstrüksiyon hatalarına yol açabilir. EncDecAD, eğitim için yalnızca normal dizileri kullanır. Bu, özellikle anormal verilerin mevcut olmadığı veya seyrek olduğu senaryolarda kullanışlıdır.



Şekil 3.4. LSTM kodlayıcı kod çözücü çıkarım adımları

EncDecAD yönteminin çalışma prensibi matematiksel olarak şu şekilde ifade edilir. L uzunluğunda bir $X = \{x^{(1)}, x^{(2)}, \ldots, x^{(L)}\}$ zaman serisi verilsin. Her bir $\mathbf{x}^{(i)} \in R^m$ noktası, t_i zaman örneğindeki m değişkenlerinin okumaların m boyutlu bir

vektörüdür. Bu tür birden fazla zaman serisinin mevcut olduğu veya daha geniş bir zaman serisinden L uzunluğunda bir pencere alınarak elde edilebileceği senaryo burada ele alınmaktadır. Normal zaman serisinin yeniden oluşturulması için öncelikle LSTM Kodlayıcı-Kod Çözücü modelini eğitilir. Verilen test zaman serisindeki bir noktanın anormal olma olasılığını elde etmek için rekonstrüksiyon hataları kullanılır. Öyle ki her bir $\mathbf{x}^{(i)}$ noktası için anormal olan noktanın bir $a^{(i)}$ anomali skoru hesaplanır. Daha yüksek bir anomali skoru veya yüksek rekonstrüksiyon hatası, noktanın anormal olma olasılığının daha yüksek olduğunu gösterir.

Normal zaman serilerinin örneklerini yeniden oluşturmak için bir LSTM kodlayıcı-kod çözücü eğitilir. LSTM kodlayıcı, giriş zaman serisinin sabit uzunlukta vektör temsilini öğrenir ve LSTM kod çözücüsü, mevcut gizli durumu ve önceki zaman adımında tahmin edilen değeri kullanarak zaman serisini yeniden oluşturmak için bu gösterimi kullanır. Verilen bir X için, $\mathbf{h}_E^{(i)}$, her bir $i \in \{1,2,\dots,L\}$ ve $\mathbf{h}_E^{(i)} \in R^c$ için t_i zamanındaki gizli durumudur. Kodlayıcının gizli katmanındaki LSTM birimlerinin sayısı c ile temsil edilir. Kodlayıcı ve kod çözücü, zaman serilerini ters sırada yeniden yapılandırmak için birlikte eğitilir ([143]'dekine benzer). Örneğin, $\{\mathbf{x}^{(L)}, \mathbf{x}^{(L-1)}, \dots, \mathbf{x}^{(1)}\}$ hedef zaman serisidir. Kodlayıcının son durumu $\mathbf{h}_E^{(L)}$, kod çözücünün başlangıç durumu olarak kullanılır. Hedefi tahmin etmek için LSTM kod çözücü katmanının üstündeki doğrusal bir katman kullanılır. Eğitim sırasında, kod çözücü $\mathbf{h}_D^{(i-1)}$ durumunu elde etmek için girdi olarak $\mathbf{x}^{(i)}$'yi kullanır ve ardından hedef $\mathbf{x}^{(i-1)}$'e karşılık gelen $\mathbf{x}'^{(i-1)}$'yi tahmin eder. Çıkarım sırasında, $\mathbf{x}'^{(i-1)}$ tahmini değeri, $\mathbf{h}_D^{(i-1)}$ elde etmek ve $\mathbf{x}'^{(i-1)}$'yi tahmin etmek için kod çözücüye girdi olarak verilir. Model, $\sum_{X \in s_N} \sum_{i=1}^{L} \|\mathrm{x}^{(i)} - \mathrm{x}'^{(i)}\|^2$ hedefini en aza indirecek şekilde eğitilir. Burada, normal eğitim dizileri seti s_N ile temsil edilir. Şekil 3.4., $L = 3$ uzunluğundaki bir dizi için LSTM Kodlayıcı-Kod Çözücü rekonstrüksiyon modelindeki çıkarım adımlarını göstermektedir. Kodlayıcının t_i zamanındaki $\mathbf{h}_E^{(i)}$ gizli durumunu elde etmek için t_i zaman örneğindeki $\mathbf{x}^{(i)}$ değeri ve kodlayıcının $t_i - 1$ zamanındaki $\mathbf{h}_E^{(i-1)}$ gizli durumu kullanılır. Girdi dizisinin sonundaki kodlayıcının $\mathbf{h}_E^{(3)}$ gizli durumu, kod çözücünün $\mathbf{h}_D^{(i-1)}$ başlangıç durumu olarak kullanılır, $\mathbf{h}_D^{(3)} = \mathbf{h}_E^{(3)}$.

$\mathbf{x}'^{(3)} = \mathbf{w}^{\mathrm{T}}\mathbf{h}_D^{(3)} + \mathbf{b}$'ü hesaplamak için kodlayıcının tepesindeki $b \in R^m$ bias vektörü ve $c \times m$ boyutundaki w ağırlık matrisli doğrusal bir katman kullanılır. Kod çözücü, bir sonraki gizli durum $\mathbf{h}_D^{(i-1)}$ 'yi elde etmek için $\mathbf{h}_D^{(i)}$ ve $\mathbf{x}'^{(i)}$ tahminini kullanır.

Anomali olasılığının hesaplanması şu şekilde yapılır. Öncelikle, normal zaman serisini dört set zaman serisine bölünür: s_N, v_{N1}, v_{N2}, ve t_N. Anormal zaman serileri ise v_A and t_A şeklinde iki sete ayrılır. LSTM kodlayıcı-kod çözücü rekonstrüksiyon modelini öğrenmek için s_N seri seti kullanılır. v_{N1} seti, kodlayıcı-kod çözücü modelini eğitirken erken durdurma işlemi için kullanılır. t_i için rekonstrüksiyon hata vektörü $\mathbf{e}^{(i)} =| \mathbf{x}^{(i)} - \mathbf{x}'^{(i)}$ şeklinde formülize edilir. v_{N1} setindeki dizilerdeki noktaların hata vektörleri maksimum olabilirlik kestirimi kullanarak $\mathcal{N}(\mu, \Sigma)$ normal dağılımının $\boldsymbol{\mu}$ ve Σ parametrelerini tahmin etmek için kullanılır. Ardından herhangi bir $\mathbf{x}^{(i)}$ noktası için anomali skoru $a^{(i)} = \left(\mathbf{e}^{(i)} - \boldsymbol{\mu}\right)^T \boldsymbol{\Sigma}^{-1} \left(\mathbf{e}^{(i)} - \boldsymbol{\mu}\right)$ kullanılarak hesaplanır. Denetimli bir ayarda, eğer bir $a^{(i)} > \tau$ ise, serideki bir noktanın "anormal", aksi takdirde "normal" olduğu varsayılır. Yeterli miktarda anormal seri mevcut olduğunda, $F_\beta = (1 + \beta^2) \times P \times R/(\beta^2 P + R)$'yi maksimize etmek için olasılık değerlerinin üzerinden τ bir eşik değeri öğrenilir. Burada P kesinlik, R duyarlılık, "anomali" pozitif sınıf ve "normal" negatif sınıftır. Bir pencere anormal bir model içeriyorsa, pencerenin tamamı "anormal" olarak etiketlenir. Bu, anormalliğin kesin konumunun bilinmediği birçok gerçek dünya uygulamasında yararlıdır. Anormal olarak etiketlenen bir dizideki gerçek anormal noktaların oranı yüksek olmadığından ve dolayısıyla daha düşük duyarlılık (recall) beklendiğinden $\beta < 1$ olduğu varsayılır. τ ve c parametreleri, v_{N2} ve v_A'daki validasyon serileri kullanılarak maksimum F_β ile belirlenir.

3.3. SDN Ağlar İçin Anomali Tabanlı Saldırı Tespit Sistemi Mimarisi

SDN, son birkaç yılda geleneksel ağ yapısını değiştiren bir paradigma olarak karşımıza çıkmaktadır. Bu paradigma geleneksel ağ iletme cihazlarından farklı olarak kontrol düzlemini veri düzleminden ayırır ve kontrol mantığını merkezi bir denetleyicide toplayarak iletme cihazları üzerinde merkezi bir kontrol noktası tanımlar.

Ağ güvenliği açısından bakıldığında SDN, ağ altyapısı üzerinde eşi benzeri görülmemiş bir kontrol sunar ve tüm ağ altyapısındaki veri akışlarının yönlendirilmesi için tek bir kontrol noktası tanımlar. Üst seviye reaktif (duyarlı) güvenlik gözetimi, analizi ve tepki sisteminin hazırlanmasıyla ağ güvenliğini artırmak için SDN mimarisinden istifade edilebilir [5]. Merkezi denetleyici bu sistemin anahtar bileşenidir. SDN paradigmasının şekillenmiş hali olan OpenFlow protokolü kullanılarak gerçeklenmiş ağ güvenliği uygulamalarıyla, izin verme veya yasaklamadan daha ziyade karmaşık mantığı işleyebilen akışlar da gerçeklenebilir. Bu tip uygulamalarla, karmaşık karantina prosedürleri veya özel muamele için kötücül ağ akışlarını yeniden yönlendirmeyi sağlayan mantık gerçeklenebilir [5].

Saldırı tespiti ve saldırı önleme algoritmaları OpenFlow güvenlik uygulaması olarak gerçeklenebilir. Ağa yerleştirilen trafik analizi veya anomali-tespit yöntemleri, düzenli olarak merkezi denetleyiciye aktarılan güvenlikle ilgili veriler üretir. Ağın tamamından alınan bu geri beslemeyi analiz etmek ve ilişkilendirmek için denetleyicide uygulamalar çalıştırılabilir. Elde edilen analiz sonuçlarına bağlı olarak yeni veya güncellenmiş güvenlik ilkeleri, akış kuralları (flow rules) şeklinde ağ bileşenlerine (örneğin; yönlendirici, anahtar) dağıtılabilir. Bu birleştirilmiş yaklaşım, ağa karşı yapılan güvenlik tehditlerinin kontrolünü ve bunlardan korunmayı etkili ve verimli bir şekilde hızlandırabilir [5]. Burada önemli olan nokta bu uygulamaların gerçeklenmesi genellikle geleneksel ağlardakilere göre daha kısa ve etkili olmasıdır.

Güvenlikle ilgili bazı bilgiler iletme cihazları sorgulanarak elde edilebilir. OpenFlow-etkin bir anahtardaki birkaç alandan oluşan bir akış tablosu girdisi, aşağıdaki gibi sınıflandırılabilir [119]:

a. Eşleşme alanları (Matching Fileds), bir 15-tuple paket başlığı, ingress (giriş) portu ve opsiyonel olarak paket meta-datasına dayalı paketleri eşleştirmek için kullanılır.
b. Akış girdisi önceliği (Priority of flow entry), akış girdisinin eşleşme sıralamasını önceliklendirir.

c. Eylem seti (Action set), başlık eşleştiğinde paketlerde gerçekleştirilecek spesifik eylemleri gösterir.
d. Sayıcılar (Counters), akış istatistikleri kaydı tutmak için kullanılır. (Her bir akıştaki paket ve bayt sayısı ve son paketin akışla eşleştiği zaman).
e. Süre aşımları (Timeouts), maksimum süreyi veya akışın anahtar tarafından geçersiz kılınmadan önceki boş süreyi tanımlar.

Burada her bir akışa karşılık gelen paket sayısı ve bayt miktarı gibi akış istatistikleri kullanılarak ağdaki anomalilerin tespiti gerçekleştirilebilir. Literatürde yer alan anomali tespit yöntemleri incelendiğinde bu istatistiklerin kullanılmasıyla başarılı bir şekilde anomalilerin tespit edildiği görülmektedir.

SDN mimarisi kullanılarak gerçeklenecek saldırı tespit sisteminde, literatürdeki yöntemler incelenirken aşağıdaki parametreler göz önünde bulundurulur:

a. Gerçek zamanlı: Denetleyici, akış kurallarını iletme cihazlarına yüklenerek n programlanabilmesi saldırı anında aktif müdahale yapmaya olanak tanımaktadır. Belirli zaman aralıklarıyla toplanan akış istatistikleri analiz edilerek gelen yeni bir akışın anormal olup olmadığına karar verilmesi ve sonuca göre anahtardaki akış girdisinin güncellenmesi gerçek zamana yakın bir şekilde yapılması gerekir.
b. Doğruluk oranı: Bu saldırı tespit sisteminde aktif bir müdahale hâsıl olacağından anormal veya saldırı ihtimali bulunmayan normal akışların da kesilmesi tehlikesi mevcuttur. Bu durum ancak yüksek doğrulukta çalışan bir yöntemle engellenebilir.
c. Akış tabanlı: OpenFlow-etkin bir anahtarda bir iletme cihazlarında akışlar girdi olarak tutulduğundan tespit yönteminin de akış tabanlı olması gerekir.
d. Saldırı çeşitleri: Hizmet engelleme (Denial of Service), DDoS ve Tarama (Probe) gibi birçok saldırı türünlerini tanıyan bir yöntem olmasıdır.

3.3.1. SAnDet sistem mimarisi

SDN ortamlarda çalışabilmesi için tasarlanmış ve bu mimarinin sunduğu imkanları aktif olarak kullanabilen SAnDet (SDN Anomali Detector)'te bazı prensipler göz önünde bulundurulmalıdır. SAnDet'in tasarımında Giotis ve ark.'nın [103] belirlediği birkaç özelliğin yanı sıra OpenFlow'un sunduğu imkanlar göz önünde bulundurularak aşağıdaki maddeler halinde verilen anahtar prensipler dikkate alınmıştır:

a. Veri toplama, anomali tespit ve önlemeyi (mitigation-hafifletme) modüler tasarımla birbirinden ayırma.
b. OpenFlow-etkin Katman 2 ve Katman 3 cihazlar ile uyumluluk.
c. Gerçek-zamanlı ortamlarda hızlı anomali tespit ve önleme (mitigation) birbirinden ayrı veri ve kontrol düzleminin kullanılması.
d. İstatistik toplamaya dair OpenFlow'un yeteneklerinden faydalanılması.

Çalışma, genellikle OpenFlow anahtarda bir akış girdisi olarak yer alan ve dört spesifik değişken ile ilişkili 12-tuple akış tanımı setine dayanmaktadır. Bu değişkenler; i) herhangi bir paket ilişkili akış girdisiyle eşleştiğinde nasıl iletileceğini belirten eylem kuralı, ii) son paket eşleşmesinden sonra bir akışın geçersiz kılınmasını temsil eden soft timeout değişkeni, iii) akış girdisi işlenmesinden bu yana bu akışla eşleşen paket sayısı, iv) paket eşleşme işleminde çakışma olma durumunda hangi akış kuralının saptanacağının belirten ve her bir akış girdisine atanan bir spesifik öncelik.

SAnDet'in mimarisi üç ana modülden oluşmaktadır [103]:

- İstatistik Toplayıcı (Collector) Modülü: Toplayıcı modülü, akış-tabanlı anomali tespiti için gerekli olan veri toplamadan sorumludur. Bu modül, periyodik olarak akış bilgisini toplar ve bu verileri Anomali Algılama modülüne aktarır. Veri toplamayla ilişkili iki farklı yöntem literatürde kullanılmıştır. Bunlardan birincisi, periyodik olarak switch sorgulanarak gelen cevaplar biriktirme şeklinde çalışan OpenFlow yaklaşımıdır. İkincisi ise, paket örneklemeden yararlanan bir akış gözetleme (monitoring) mekanizmasıdır. Giotis ve ark. marka-bağımsız olan sFlow kullanılır [103].

- Anomali Algılama Modülü: Toplayıcı modülünden sağlanan veri belirli periyodik zaman aralıklarıyla Anomali Tespit modülüne gönderilir. 10 sn zaman penceresi olarak seçilmiştir. Giotis ve arkadaşları [103] entropi tabanlı algoritma kullanmıştır. Fakat bu modülde, istatiksel anomali tespiti, makine öğrenmesi tabanlı anomali tespiti ve veri madenciliği tabanlı anomali tespiti algoritmalarından herhangi biri de kullanılabileceğini de belirtmişlerdir. DDoS, Worm yayılımı ve Portscan saldırılarını başarıyla tespit ettiklerini de belirtmişlerdir.

- Anomali Önleme (mitigation) Modülü: Anormali önleme modülü, istenen kötü amaçlı trafiği engellemek için OpenFlow anahtarının akış tablosuna akış girdileri ekleyerek (veya mevcut olanları değiştirerek) tespit edilen saldırıların veya ihlallerin etkisiz hale getirilmesi amaçlanır [103]. Bu akış girdileri, akış tablosundaki diğer akışlardan daha yüksek önceliğe sahiptir. Kötü niyetli anormal davranışa benzeyen zararsız ağ akışlarını engellemekten kaçınmak için, bir beyaz liste tablosuna ana bilgisayarla ilgili belirli kurallar ekleme işlemi yapılır. Bunun yanında bir kara liste (tehlikeli adres) tablosuna kötücül olduğu kesin olan kurallar eklenir.

Şekil 3.5. SAnDet'in mimarisel gösterimi

3.3.2. Veri toplama ve özellik çıkartma

OpenFlow yaklaşımı, akış istatistiklerinin anahtarlardan toplanması için OpenFlow protokolünü kullanır. OpenFlow protokolünün zorunlu kıldığı gibi, periyodik akış istatistiği istekleri, OpenFlow anahtarında bulunan tüm akış girdilerinin karşılık gelen sayaçlarıyla birlikte olağan toplanması için denetleyici tarafından gerçekleştirilir

[103]. Anahtarlarda yer alan akışların sayaçları, yalnızca bir yönlendirme arama süreci akış tablosunun belirli bir akış girdisiyle eşleştiğinde güncellenir. Bu nedenle, doğal bir OpenFlow ortamında akış istatistiklerinin toplanması, OpenFow denetleyicisinin içinde bulunan paket yönlendirme mantığıyla sıkı bir şekilde bağlantılıdır. İletme (forwarding) mantığının anomali algılama tekniği tarafından dikte edildiği durumumuzda, Katman 3 ve 4 protokol alanları kullanılır. Bu nedenle, akış tablosunun gerekli akış girdileri, tek bir akışa indirgenecek şekilde ileri ve geri yönde akışlar kaynak IP, hedef IP, kaynak port, hedef port, protokol alanları dikkate alınarak tek bir akış girdisi oluşturulur. Akışların kalan alanları joker karakterlerle doldurulur, bu nedenle ilgili alanların herhangi bir değeri ile eşleşir.

```
active_flows // Belirli zaman aralığındaki aktif akışlar
t // akış istatistiklerinin sorgulanacağı zaman aralığı
F // akış istatistik tablosu

if e is initialization event then
  active_flows <- 0, U <- 0
end if

if e is PacketIn event then
   f.stats_values <- 0
   f <- new_flow (src_ip,dst_ip,src_port,dst_port,ip_proto,stats_values)
   add f to F
else if e is FlowRemoved event then
  f.stats_values <- e.stats_values
  f.isFlowRemoved <- true
else if e is timeout t event then
   for all flows f1 in F
       // eğer istatistik tablosundaki
       // herhangi bir akış girdisi switchte aktifse
       if f1.isFlowRemoved is false then
            send a FlowStatsRequest to edge_switch
       end if
   end for
else if e is a FlowStatsReply event for flow f then
   F.f.stats_values = e.stats_values
end if
```

Şekil 3.6. Akış istatistik toplama algoritması

OpenFlow yaklaşımı, bir akış girdisi anahtardan silindiğinde bu bilgiyle birlikte akış girdisine ait sayaç değerleri gibi diğer verileri denetleyiciye ileten bir mesaj (FlowRemoved) gönderir. OpenFlow yaklaşımı ile istatistik toplama, bir anahtar ilişkili OpenFlow denetleyicisinden gelen bir akış istatistikleri isteğine (akış

istatistikleri için anahtarı sorgulayan OFFlowStatsRequest mesajıdır) cevap verdiğinde (OFFlowStatsReply) gerçekleştirilebilir. Bu iki mesajın birlikte kullanılmasıyla akış girdilerinin verilerin toplanması gerçekleştirilebilir [148]. Başlangıç olarak gelen her yeni paket denetleyicide bir tabloya eklenir. Daha sonra belirli bir zaman dilimde akış girdisi silindiğinde mesaj göndermek üzere programlanan bu girdilerden mesaj beklenir. Eğer bu zaman diliminde tablodaki tüm girdilerden bu mesajlar geldiyse tüm akış istatistiklerinin elde edildiği varsayılır ve bu veriler algılama kodülüne aktarılır. Eğer en az bir akış girdisinden silindi mesajı gelmediyse istatistik toplama mesajı OpenFlow anahtara gönderilerek cevap mesajı beklenir. Bu yaklaşımın algoritması Şekil 3.6.'daki gibi verilebilir. Sonuç olarak, anahtar, akış tablosu içeriğinin büyük parçalarıyla yanıt verir. Her yığın, her akışa karşılık gelen paket sayaçlarıyla birlikte akış girdilerinin bir bölümünü içerir. Bununla birlikte, akış tablosu kayıtlarının paket sayaçları, her bir kuralın somutlaştırılmasından bu yana toplam paket sayısını depolarken, anomali algılama algoritması yalnızca belirli bir zaman penceresi sırasında her akış girdisiyle eşleşen paket sayısı gibi sorgulama sonucu sayaçlardan gelen sayaç içeriklerini gerektirir. Bu nedenle, belirlenen zaman penceresine karşılık gelen paket sayısını bulmak için, daha önceki

Tablo 3.1. Akış özellikleri

	Özellik Adı	**Tanımlama**	**Akış Yönü**
Temel	Kaynak IP		
	Hedef IP		
	Protokol tipi	TCP, UDP ve ICMP	
	Servis Tipi (ToS)		-
	Kaynak Port		
	Hedef port		
	Bayt Sayısı		İleri (Forward) Geri (Backward)
	Paket sayısı		İleri yönlü ve geri yönlü
	Saniye olarak aktif süre		İleri yönlü ve geri yönlü
	Saniyedeki paket oranı		İleri yönlü ve geri yönlü
	Saniyedeki bayt oranı		İleri yönlü ve geri yönlü

Tablo 3.1. (Devamı)

	Özellik Adı	**Tanımlama**	**Akış Yönü**
Türetilmiş özellikler 1 [8], [116]	Akış başına ortalama paket sayısı (APf)	Aşağıdaki denklem ortalama değerini hesaplamak için kullanılır. Burada P, akış başına paket sayısı ve n akış sayısıdır. $$APf = \frac{1}{n}\sum_{i=1}^{n} P_i$$ DoS saldırılarının temel özelliklerinden biri, saldırganın gerçek kaynağını izleme görevini çok zorlaştıran kaynak IP sahtekarlığıdır. Bir yan etki, az sayıda paket içeren, yani akış başına yaklaşık 3 paket içeren akışların üretilmesidir. Bu, genellikle akış başına daha yüksek sayıda paket içeren normal trafikten farklıdır	İleri yönlü Geri yönlü
	Akış başına ortalama bayt (ABf)	Bu özellik hesaplanırken aşağıdaki denklem kullanılır, B burada bayt miktarını temsil eder. $$ABf = \frac{1}{n}\sum_{i=1}^{n} B_i$$	İleri yönlü Geri yönlü
	Akış başına ortalama süre (ADf)	Burada D anahtardaki bir akışın süresini ve n akış sayısını temsil eder. $$ADf = \frac{1}{n}\sum_{i=1}^{N} D_i$$	İleri yönlü Geri yönlü
	Tekil akışların büyümesi (growth) (GSf)	$$GSf = \frac{Num\_Flows - (2 * Num\_Pair\_Flows)}{interval}$$	İleri yönlü
	Farklı portların büyümesi (GDP)	$$GDP = \frac{Num\_Ports}{interval}$$	İleri yönlü
	Aynı kaynak IP'den gelen akış sayısı	Saldırgan, bir alt küme içindeki olası her adrese ICMP ping paketleri gönderebilir ve hangi makinenin yanıt verdiğini görmek için bekleyebilir.	İleri yönlü

önceki zaman pencerelerindeki için akış tablosunun durumunun bir kaydı tutulmalı ve her akış girdisi için geçerli verilerle karşılaştırılmalıdır. OpenFlow istatistik toplama yöntemini kullanarak, toplama sürecinde örnekleme olmadığından, anahtardan geçen genel ağ trafiğini tüm ayrıntılarıyla toplamak ve analiz etmek mümkündür [103]. Bu nedenle, bu yöntem düşük ila orta trafik hacimli ağları izlemek için başarıyla

uygulanmıştır [116][2]. Tablo 3.1.'deki türetilmiş özellikler, bir akışın karşılık gelen çifti olan akış için de hesaplanır ve ayrı ayrı veri setine eklenir.

3.3.3. Saldırı algılama ve önleme

Önceki bölümde bahsedilen akışlardan çıkartılan özellikleri girdi olarak alan ve ağda saldırı olup olmadığını tespit eden bir yöntemin geliştirilmesi gerekir. Son zamanlarda popüler olan ve birçok problemin çözülmesine önemli ölçüde katkı sağlayan derin öğrenme tabanlı yöntemler tespit yöntemi olarak tercih edilerek performansları incelenmiştir.

```
white_list // whitelist flows
black_list // blacklist flows
detected_flows // detected flows

for all flows f in detected_flows  do
   if white_list.contains(f) then
      do nothing
   else if black_list.contains(f) then
      do nothing // already in blacklist
   else
      // write drop rule to the switch for f
      create match m object
      m.srcIp = f.srcIp
      m.destIp = f.destIp
      m.srcPort = f.srcPort
      m.destPort = f.destPort
      m.proto = f.procto
      m.action = drop
      m.idleTimeout = default_timeout * delay_ratio
      m.hard_timeout = default_hard_timeout

      write m to the switch
   end if
end for
```

Şekil 3.7. Anomali önleme algoritması

SAnDet mimarisinde hem bilinen hem de bilinmeyen (zero-day) saldırıların algılaması için anomali tabanlı teknikler üzerinde yoğunlaşılmıştır. Derin öğrenme yöntemlerinde RNN ve EncDecAD literatürde yapılan çalışmalara göre veri setlerindeki anomalileri oldukça başarılı şekilde tespit edebilmişlerdir. Ağdan toplanan akış istatistiklerinin yanı sıra bunlardan türetilen özellikler bu iki yönteme girdi olarak verilmekte ve tespit performansları belirli değerlendirme ölçütlerine göre yapılmaktadır. Anomali tabanlı

methodların değerlendirilmesinde değerlendirme ölçütü olarak AUC'un kullanılması önerilmesine rağmen sınıf dağılımı dengesiz (imbalanced) olan veri setlerinde doğruluk (accuracy) ölçütünü de dikkate almak gerektiği belirtilmiştir [61].

3.4. Performans Değerlendirmesi

Bu bölümde, OpenFlow protokolünün saldırı algılama ve önleme mekanizması olarak performans değerlendirmesi yapılmaktadır. Bunun yanında, SDN denetleyici (Floodlight) yeteneklerini kullanarak, tespit edilen kötü amaçlı trafiği önlemek için OpenFlow'un sağladığı faydalar araştırılmaktadır.

Floodlight [123], modüler mimariye sahip açık kaynak kodlu bir OpenFlow denetleyicisidir. Floodlight denetleyicisinin sağladığı API vasıtasıyla, periyodik ve aperiyodik veri toplama, toplananan akış girdilerinin temel özelliklerinden yeni özelliklerin türetilmesi ve akış tablosu modifikasyon görevlerinden sorumlu farklı üç bileşenin gerçeklenmesi yapıldı. İstatistiklerin toplanması ve yeni özelliklerin türetilmesi işlemini yapan İstatistik Toplayıcı modülü gerçeklenmiştir. RNN ve EncDecAD yöntemlerinin her ikisinden de geliştirilen modellerinin uygulanabileceği ve bununla birlikte farklı tespit algoritmasının kolayca entegre edilebileceği anomali algılayıcı modülü gerçeklenmiştir. Bu işlemin sonuçları daha sonra saldırı karşı önlemleri sağlamak için Anomali Önleme modülüne sevk edilir. Bu durum, gerekli bilgileri Anomali Önleme modülüne iletebildiği sürece, kullanıcıya tercih ettiği anomali algılama yöntemini geliştirme veya kullanma olanağı sağlar. Performans değerlendirme denemeleri, ISCX2012 veri setinde yer alan ağ trafiğinin bir kısmıyla gerçekleştirilmiştir.

3.4.1. Deneysel ortamın hazırlanması ve trafik üretimi

Anomali tespit ve önleme fonksiyonları için Java dilinde yazılmış Floodlight modülleri olarak gerekli tüm algoritmalar gerçeklendi. Deneylerin yapılabilmesi için OpenFlow-etkin anahtarlara ihtiyaç vardır. Bu amaçla, gerekli trafik yüklerini işleyebilen bir yazılım anahtarı olan Open vSwitch'i [149] kullanılır. Çalışmanın deneysel ortamı, açık kaynak kodlu ve literatürde sıkça kullanılan bir araç olan Mininet

[128] emülasyon yazılımı ile sağlanmıştır. Bu emülasyon ortamı, sunduğu özellikler açısından donanım anahtarlarından üstün olan OpenFlow vSwitch [149] destekler ve SDN destekli ağ prototiplerini geliştirmek için tasarlanmıştır. Emülasyon ortamı, 16 GB RAM'i, dört çekirdekli 3 GHz işlemcisi olan bir sunucuda barındırıldı. Ayrıca sağladığı işlevsellik ve açık kaynak olması nedeniyle Floodlight [123] denetleyici yazılımı kullanılmaktadır. Şekil 3.8., yukarıda belirtilen iki yaklaşımı değerlendirmek için kullanılan deneysel kurulumu ve gerçeklenen Floodlight uygulamalarını gösterir. Denetleyici yazılımı, sunucu üzerindeki sanal bir makinede çalıştırılır. Performans değerlendirmesi için yakalanan ağ trafiği tek bir anahtara enjekte edilir (bu durumda yazılım tabanlı Open vSwitch). Önerilen mekanizmanın birden çok OpenFlow etkin anahtarlara ve buna karşılık gelen önleme kurallarıyla daha genel ağ topolojilerine uygulanabileceğinin açıkça belirtilmesi gerekir.

OpenFlow tabanlı istatistik toplamada bazı parametrelerin ayarlanması gerekir. Özellikle, her akış girdisinin esnek süre aşımı (idle timeout) değişkeni için belirli bir değer ayarlanması gerekir. Bunun nedeni, toplayıcı modülün ilgili akış istatistiklerini toplaması için herhangi bir akış girdisinin süre aşımına uğramasının istenmesidir. Bu nedenle, çalışmada benimsenen zaman penceresi yani istatistik toplama peryodu 10 saniye, esnek süre aşımı 3 saniye olarak ayarlanır.

Tablo 3.2. Üretilen trafikten toplanan tüm akışların sayısı

Gün	**Normal**	**Saldırı**	
		DoS	***Portscan***
11 Haziran Cuma	3452552	0	0
12 Haziran Cumartesi	557885	210569	252874
16 Haziran Çarşamba	2213469	0	0
Toplam	*5820763*	*463443*	

Saldırı algılama ve önleme mekanizmasının performansını değerlendirmek için ISCX2012 veri setinin normal veya iyicil (benign) trafiğini kullanılır. Daha açık bir biçimde belirtmek gerekirse "11 Haziran Cuma" ve "16 Haziran Çarşamba" günlerinde yakalanan trafik normal ağ trafiği olarak kullanılır. Bunun yanında

“Cumartesi (Saturday)” trafik izini içeren iyicil trafik ise normal olarak oynatılarak belirtilen DoS ve port tarama saldırıları belirli zaman aralıklarında gerçekleştirilir.

ISCX2012 veri seti [150], ağ emülasyon ortamındaki trafiği bir haftadan uzun bir süredir yakalayarak 2012 yılında oluşturulmuştur. Yazarlar, normal ve kötü amaçlı ağ davranışıyla bir saldırı algılama veri seti oluşturmak için dinamik bir yaklaşım olan α profilleri, β profilleri e-posta yazma veya internette gezinme gibi normal kullanıcı davranışını karakterize ederken saldırı senaryolarını tanımlar. Bu profiller, paket tabanlı ve çift yönlü akış tabanlı biçimde yeni bir veri seti oluşturmak için kullanılır. Dinamik yaklaşım, sürekli yeni bir veri seti oluşturma olanağı sağlar. Bu veri seti SSH kaba kuvvet, DoS veya DDoS gibi çok çeşitli saldırıları içerse de bu saldırıları içeren trafik izleri bu çalışmada tcpreplay aracının saldırı karakteristiğini bozması nedeniyle kullanılmamıştır.

Tablo 3.3. Zaman aralığında toplanan akışlardan üretilen örnek sayısı

Gün	**Normal**	**Saldırı**	
		DoS	***Portscan***
11 Haziran Cuma	8640	0	0
12 Haziran Cumartesi	8275	55	35
16 Haziran Çarşamba	8644	0	0
Toplam	*25559*	*90*	

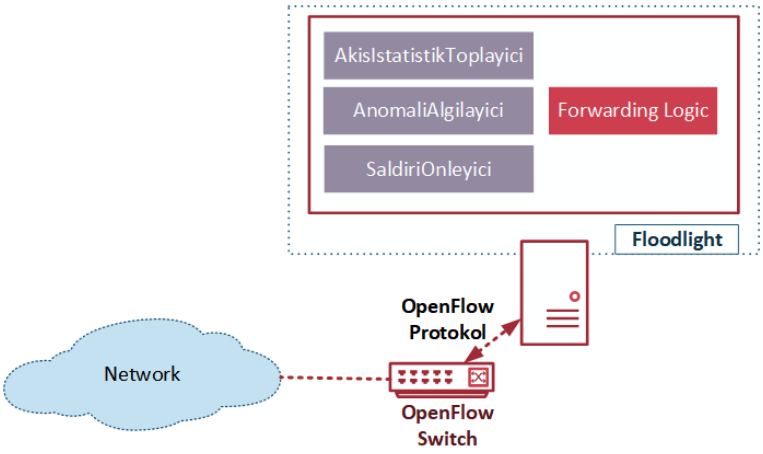
Bu trafik iz dosyaları, OpenFlow yaklaşımının doğruluğunu ve algılama yeteneklerini değerlendirmek amacıyla deneylerde kullanılmıştır. Tcpreplay [151] aracı, yakalanan paket iz dosyalarını yeniden oynatmak için kullanılır ve oluşturulan trafiği belirli bir Ethernet bağlantı noktasına enjekte eder. Tcpreplay, yakalanan trafiği yakalandığı hızda yeniden oynatabilme yeteneğine sahiptir. Saldırı gerçekleştirmek için, rasgele protokol alanı değerlerine sahip paket dizilerinin gönderilmesine izin veren ve programlanabilir bir yazılımsal araç olan hping3 [152] kullanıldı. Böylece, bir DoS saldırısını gerçekleştirmek için önceden tanımlanmış bir hedef IP adresi ve porta paketlerin gönderimi sağlanır. Son olarak, hping3 aracı vasıtasıyla port tarama (portscan) senaryosu için belirli bir kaynak ve hedef IP adresine sahip kaynak ve hedef portlar rasgele seçilerek saldırı gerçekleştirilir. Üretilen trafikten toplanan tüm

akışların sayısı Tablo 3.2.'te ve belirli zaman aralığında (10 saniye) toplanan akışlardan türetilen özellikleri içeren örnek sayısı ise Tablo 3.3.'te gösterilmiştir.

3.4.2. Anomali algılama ve önleme

Daha az bilgi, zaman ve çaba gerektirmesi nedeniyle etiketlenmemiş verilerin ihlal (izinsiz giriş) tespit sistemi için elde edilmesi etiketli verilere göre daha kolay olduğu için kısıtlı kümeleme SSL stratejisi seçilmiştir. Ayrıca, bu strateji kullanılan tespit yöntemlerinin denetimsiz eğitim doğasına daha uygundur.

Şekil 2.6.'da görüldüğü gibi, eğitim ve test veri seti olmak üzere veri seti iki kısma ayrılır. SSL stratejisi uygulanarak ağ trafiğinin normal profilini oluşturmak için sadece normal akış özelliklerini içeren etiketli veri seti eğitim aşamasında kullanılır. Test aşamasında ise hem normal hem de saldırı akışı özelliklerinden oluşan etiketlenmemiş veri seti kullanılır. Bu çalışmada değerlendirme metriklerinin hesaplanmasında test veri setindeki etiketlerin dikkate alındığını belirtmek önemlidir.



Şekil 3.8. Deney ortamı

Bu çalışmada kullanılan yöntemlerin parametreleştirilmiş olması nedeniyle modellerin performansının optimal parametreler ile belirlenmesi gerekmektedir. Eğitim sürecine anormal veriler kullanılarak dahil edilmediğinden hiperparametrelerin optimizasyonunda çapraz doğrulama (cross-validation) işlevi gerçekleştirilemez [71]. Böylece, hiperparametrelerin ayarlanması Patterson ve Gibson [72] tarafından

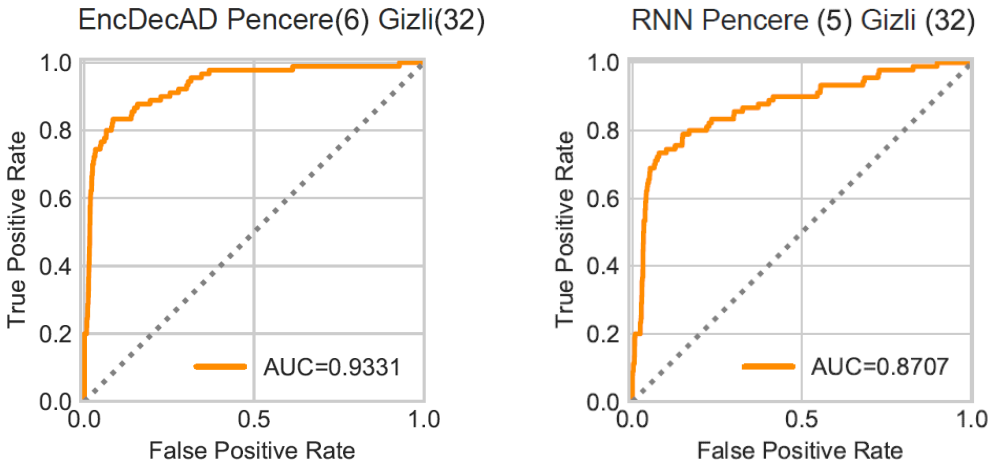
bahsedilen öneriler dikkate alınarak yapılır. Bundan sonra, RNN ve EncDecAD'nin hiperparametreleri çoğunlukla deneme yanılma yani pratik kural kullanılarak yapılandırılır.

Tablo 3.4. RNN ve EncDecAD performans değerlendirme sonuçları

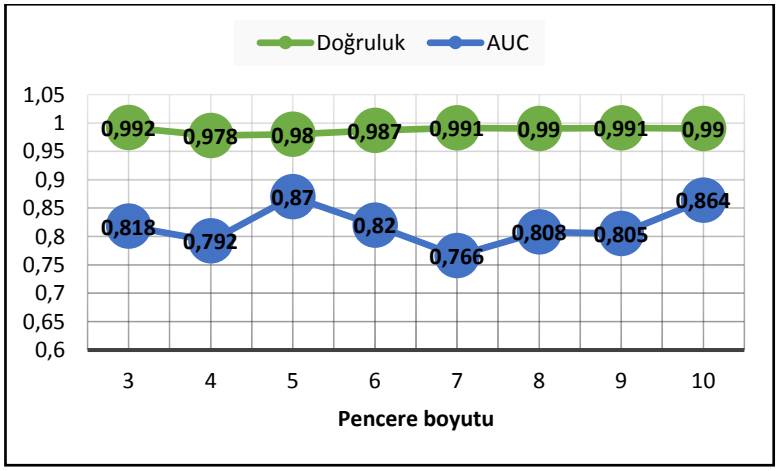
Yöntem	Pencere Boyutu	Darboğaz katman boyutu	Doğruluk	AUC
EncDecAD	6	32	**0.993**	**0.933**
EncDecAD	4	48	**0.993**	0.932
EncDecAD	10	8	0.992	0.931
EncDecAD	7	32	0.992	0.924
EncDecAD	8	32	**0.993**	0.920
EncDecAD	5	96	0.99	0.911
EncDecAD	9	8	0.992	0.911
EncDecAD	3	48	0.992	0.886
RNN	5	32	0.98	**0.87**
RNN	10	64	0.99	0.864
RNN	6	32	0.987	0.820
RNN	3	8	**0.992**	0.818
RNN	8	64	0.99	0.808
RNN	9	48	0.991	0.805
RNN	4	32	0.978	0.792
RNN	7	64	0.991	0.766

Deneyleri gerçekleştirmek için basit bir derin otokodlayıcı mimarisi biçimi benimsendi. Kullanılan RNN'nin mimarisel diyaragramı Şekil 3.3. ve EncDecAD'nin ise mimarisel diyagramı Şekil 3.7.'de (pencere boyutu 3 olduğunda) gösterilmektedir. RNN'nin darboğaz katmanında (bottleneck layer) ve EncDecAD'nin gizli katmanlarında 8,16, 32, 48, 64, 80, 96 gibi farklı boyut değerleri kullanılarak modeller eğitilip geliştirildi. RNN ve EncDecAD'nin her ikisinde en iyi performans gösteren modellerinin belirlenmesinde katman sayıları sabit tutularak yani belirtilen mimarilere sadık kalınarak katmanlardaki boyutlar deneme yanılma ile belirlendi. Hem RNN hem de EncDecAD'nin sinir ağı konfigürasyonunda kullanılan parametre ayarları aşağıdaki gibidir: Eğitim denemelerinde [0.1, 0.01, 0.001] gibi öğrenme hızları (learning rate) kullanılır. En iyi performansı veren değerler öğrenme hızı için 0,001'dir. Kütüphanede varsayılan olan ağırlık başlatma yöntemi olarak gizli katmanlarda ReLU kullanılıyorsa önerilen [72] Xavier [74] kullanılır. Her iki sinir ağı, büyük veri setleri ve gerçek değerli çıktılar için önerilen bir seçim olan eşlenik gradyan optimizasyon (conjugate gradient optimization) yöntemini kullanan geri yayılım (back propagation) algoritması ile eğitildi [64]. Adam güncelleyici [75], yerel minimumumdan kaçmak ve

optimizasyon sürecine yönelik daha iyi çözümler keşfetmek için seçilmiştir. EncDecAD konfigürasyonunda kullanılan ek parametreler aşağıdaki gibidir. Gizli katmanlarda aktivasyon fonksiyonu olarak tanh fonksiyonu kullanılır. Yaygın olarak kullanılan ve [136] çalışmasında da tavsiye edilen RNN konfigürasyonunda kullanılan ek parametrelerde çıktı katmanı dışında sigmoid aktivasyon fonksiyonu ve çıktı katmanında ise ortalama kare hata kaybı (mean square error loss) fonksiyonu ile soft-max aktivasyon fonksiyonu kullanıldı. RNN ve EncDecAD modellerinin her ikisi de Pytorch kütüphanesi [153] ile gerçeklenmiş ve 16 batch (toplu iş) boyutu ile 50 epoch süresince eğitildi. Anomali skoru olarak RNN ve EncDecAD yöntemlerinin her ikisinde de rekonstrüksiyon hatası kullanıldı.

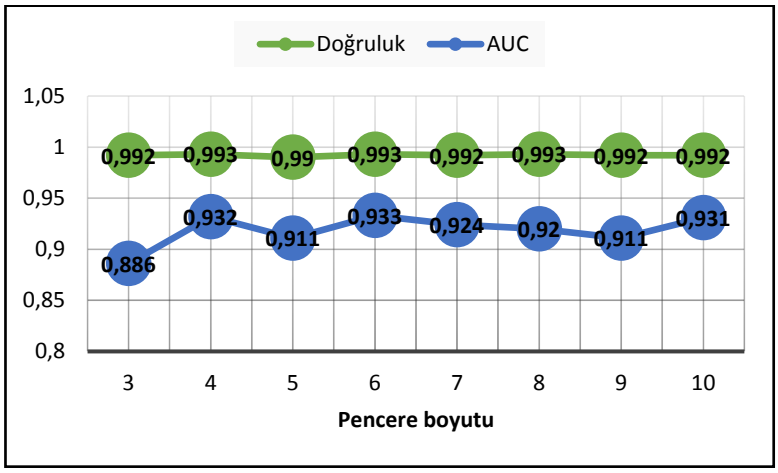


Şekil 3.9. EncDecAD ve RNN'nin en yüksek AUROC sonucu için ROC eğrileri



Şekil 3.10. RNN'de zaman penceresine göre doğruluk ve AUC'un değişim grafiği

Başlangıçta, tüm değerleri [0,1] aralığına getirmek için birlik tabanlı normalleştirme (unity-based normalization) olarak da adlandırılan özellik ölçeklendirme yöntemi [80] kullanılarak trafik izleri oynatılan ve çıkartılan özelliklerinde normalizasyon işlemi gerçekleştirilmiştir. Yöntemlerin performansını değerlendirmek için ayrı eğitim ve test veri setleri kullanılmıştır. Sinir ağını "Cuma" ve "Çarşamba" günlerindeki sadece normal akış verileriyle eğitilerek modeller oluşturuldu. Bunun nedeni, modellerin denetimsiz yaklaşımla eğitilmesidir ve veri setindeki son sütuna karşılık gelen etiket eğitimde kullanılmayan saldırı sınıfına karşılık gelir. Test süreci için hem normal trafik hem de diğer günlerdeki saldırı trafiği kullanıldı. Test sürecinde "normal" veya "iyicil" dışındaki tüm sınıflar yani saldırılar "anomali" olarak kabul edilir. Bir saldırı sınıfının performans değerlendirmesinde diğer tüm saldırı sınıfları hariç olmak üzere yalnızca normal akış kayıtları ve o saldırı sınıfının akış kayıtları kullanılarak metrikler hesaplandı.



Şekil 3.11. EncDecAD'de zaman penceresine göre doğruluk ve AUC'un değişim grafiği

Yöntemlerin performans değerlendirmesi ROC, AUC ve doğruluk ölçütleri kullanılarak gerçekleştirilir. Denetimsiz anomali tespitinde fiili standart olduğu ve yorumlanabilirliği kolaylaştırdığı için AUC ölçütüne dayalı bir değerlendirme gerçekleştirilir [81]. Sınıf dağılımına duyarlı olmadığından ve aynı zamanda bir yöntemin farklı eşiklerde diğerinden daha iyi performans gösterebileceğini (ROC grafiğindeki farklı FP değerlerine karşılık gelir) göstermek için ROC tabanlı değerlendirme yapılmıştır. Yöntemlerin AUC sonuçları Tablo 3.4.'te verilmiştir.

Sonuçlar incelendiğinde EncDecAD yönteminin farklı zaman penceresi ve darboğaz katmanlarında RNN'den daha iyi sonuç verdiği görülür.

Tablo 3.5. SAnDet'in önceki çalışmalarla karşılaştırması

Referans	**Yöntem**	**Özellik sayısı**	**Saldırı türleri**	**Denetleyici**	**Veri seti**	**Sonuç**
Niyaz ve ark. [108]	Stacked Autoencoder	TCP, UDP ve ICMP akışlarından özellikler	DDoS	POX	Farklı ortamlardan toplanmış trafik veri seti	Doğruluk % 95.65
Tang ve ark. [109]	Derin Sinir Ağı	6 özellik	DoS, U2R, R2L, Probes	OpenFlow Denetleyici	KDD	Doğruluk % 75.75
Tang ve ark. [154]	Tekrarlayıcı Sinir Ağı	6 özellik	DoS	POX	NSL-KDD	Doğruluk %89
Braga ve ark. [116]	SOM	6 özellik	DDoS	NOX	KDD99	Doğruluk % 98.57
Kokila ve ark. [98]	SVM	Akış istatistikleri	DDoS	SDN denetleyici	DARPA	% 95.11
Abubakar ve ark. [155]	Sinir Ağı	7 özellik	DoS, U2R, R2L ve Tarama	OpenDaylight	NSLKDD	Doğruluk %97
Wang ve ark. [156]	SVM	Özellik indirgeme tekniği kullanır	DDoS	Ryu	KDD99	Doğruluk % 99
Önerilen yaklaşım (SAnDet)	EncDecAD (LSTM)	21 özellik	DoS ve Port tarama	Floodlight	ISCX2012 normal trafik ve hping3 ile saldırı gerçekleştirme	Accuracy % 99.3 ve AUC % 93.3

3.5. Sonuç

Bu çalışmada, SAnDet diye adlandırılan saldırı tespit ve önleme mimarisi önerilmiş ve bir denetleyici uygulaması olarak gerçeklenmiştir. Daha açık bir ifadeyle, OpenFlow protokolünden istifade eden bir istatistik toplama modülü, EncDecAD yöntemini saldırıları algılamak için kullanan anomali tabanlı bir saldırı algılama modülü ve bir sadırı tespit edildiğinde OpenFlow protokolü vasıtasıyla saldırı önlemesi yapabilen saldırı önleme modülü bir SDN denetleyici uygulaması olarak gerçeklenmiştir. Anomali tespiti için kullanılan RNN ve EncDecAD yöntemlerinin saldırı tespit yetenekleri yarı denetimli bir öğrenme stratejisi uygulanarak analiz edilmiştir. Modellerin oluşturulması sadece normal akış tabanlı veriler kullanılarak gerçekleştirildi. Ayrıca modellerin testleri hem normal hem de anormali içeren zaman serisi veri seti kullanılarak gerçekleştirilmiştir. Deneysel sonuçlar, ROC eğrileri, AUC ve doğruluk ölçütleri kullanılarak elde edilmiştir. Sonuçlara göre, EncDecAD'nin saldırıları tespit oranının literatürde önerilen diğer yöntemlere kıyasla daha yüksek olduğu görülmüştür.

İleriki çalışma olarak, veri setinde yer alan saldırı çeşitlerinin artırılması sonucu yöntemin nasıl bir sonuç vereceğinin araştırılması önemlidir. Bunun yanında, istatistik toplama süre aralığının değişiminin sistem performansını hem denetleyici yükü açısından hem de saldırı tespit oranı açısından nasıl etkileyeceği de başka bir araştırma konusudur.

BÖLÜM 4. SONUÇ

Bu tez çalışmasında, OCSVM ile birlikte AE ve VAE derin öğrenme yöntemlerinin saldırı tespit yetenekleri yarı denetimli bir öğrenme stratejisi uygulanarak analiz edilmiştir. Modellerin oluşturulması sadece normal akış tabanlı veriler kullanılarak gerçekleştirildi. Ayrıca modellerin testleri hem normal hem de anormali içeren veri seti kullanılarak gerçekleştirilmiştir. Deneysel sonuçlar, ROC eğrileri ve AUC ölçütleri bakımından hesaplanmıştır. Sonuçlara göre, VAE'nin tespit oranı çoğunlukla AE ve OCSVM'den daha iyi olduğu gözlemlenmiştir.

Bunun yanında, SDN mimarisinin sunduğu yeteneklerden istfade ederek çalışabilen ve akış özelliklerini girdi olarak alan bir anomali tespit ve önleme sistemi olan SAnDet mimarisi önerilmiş ve denetleyici uygulaması olarak gerçeklenmiştir. İstatistik toplayıcı, anomali algılayıcı ve anomali önleme şeklide üç temel modülden oluşan bu sistem, OpenFlow anahtarlardan toplanan akış özellikleri kullanılarak bilinmeyen saldırıları algılamak için Otokodlayıcının özel bir çeşidi olan RNN ve özellikle zamana dayalı bir veri serisi girdi olarak verildiğinde başarılı sonuçlar üretebilen LSTM ağların özel bir çeşidi olan EncDecAD yöntemleri kullanılmıştır ve bu yöntemler literatürleki diğer önerilenlerle karşılaştırılmıştır. Performans değerlendirmesi yapıldığında farklı saldırı türleri de dahil olmak üzere ağ trafiği verilerinden çıkarılan akış tabanlı özellikler bir zaman serisi olarak modellere girdi olarak verilmiştir ve ROC eğrileri ve AUC sonuçları vasıtasıyla performans analiz etmek için ayrıntılı olarak incelenmiştir. Deneysel sonuçlar, EncDecAD'nin RNN'den doğruluk ölçütüne göre literatürde önerilen yöntemlerden daha iyi performans sergilediğini göstermektedir. Bunun yanında, SAnDet uygulaması aktif bir saldırı tespit sistemi olarak çalışarak anomalilerin yani tespit edilen saldırıların engellenmesini başarılı bir şekilde sağlamıştır.

Gelecekte yapılabilecek araştırma konuları aşağıdaki maddeler halinde sıralanabilir:

- Denetimsiz olarak eğitilmiş anomali tabanlı yöntemlerin önemli bir dezavantajı yüksek yanlış alarmlar üretmesidir. Bu durum bir saldırı önleme sisteminde geçerli (normal, legitimate) akışların engellenerek normal kullanıcılara sunulan hizmetlerin aksamasına neden olur. Literatürde önerilmiş ve yüksek tespit oranına sahip denetimli öğrenme yöntemleriyle desteklenmesi sonucu ilk aşamada anomali tespit yapılıp ikinci aşamada ise denetimli bir yöntem veya yöntemler yardımıyla saldırı türü belirlenerek yanlış alarmların düşürülmesi sağlanabilir.
- Ağdaki trafikten akış tabanlı özelliklerin otomatik çıkartımı işleminin gerçekleştirilmesi diğer bir ileriki çalışma olarak değerlendirilebilir. Bu durumda, belirtilen bazı saldırı karakteristiğine göre belirlenmiş özelliklerden ziyade literatürde yaygın olarak rastlanan derin öğrenme teknikleri kullanılarak otomatik özellik çıkartma işleminin yapılması bilinmeyen veya sıfırıncı gün saldırılarına karşı önemli ölçüde performans artışı sağlayabilir.
- OpenFlow tabanlı çalışan istatistik toplama yaklaşımı denetleyici üzerinde ve aynı zamanda denetleyici-anahtar arası haberleşmede önemli bir iş yükü oluşturmaktadır. Bu iş yükünü azaltacak ve yüksek doğrulukta ağla ilgili istatistik verilerini toplayabilecek ve OpenFlow'un sunduğu imkanlardan istifade eden yeni yöntemlerin geliştirilmesi de önem arz etmektedir.

KAYNAKLAR

[1] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, Network Anomaly Detection: Methods, Systems and Tools, Commun. Surv. Tutorials, IEEE, vol. 16, no. 1, pp. 303–336, 2014.

[2] S. A. Mehdi, J. Khalid, and S. A. Khayam, Revisiting traffic anomaly detection using software defined networking, in Lecture Notes in Computer Science, vol. 6961 LNCS, pp. 161–180, 2011.

[3] How SDN will shape networking, https://www.youtube.com/watch?v=c9-K5O_qYgA., Erişim Tarihi: 20.01.2018.

[4] H. Kim and N. Feamster, Improving network management with software defined networking, IEEE Commun. Mag., vol. 51, no. 2, pp. 114–119, 2013.

[5] S. Scott-Hayward, G. O'Callaghan, S. Sezer, SDN Security: A Survey, in Future Networks and Services (SDN4FNS), 2013 IEEE SDN for, pp. 1–7, 2013.

[6] S. Scott-Hayward, S. Natarajan, and S. Sezer, A Survey of Security in Software Defined Networks, Commun. Surv. Tutorials, IEEE, vol. PP, no. 99, p. 1, 2015.

[7] S. Dotcenko, A. Vladyko, and I. Letenko, A fuzzy logic-based information security management for software-defined networks, in Advanced Communication Technology (ICACT), 16th International Conference on, pp. 167–171, 2014.

[8] Y. Abuadlla, G. Kvascev, S. Gajin, and Z. Jovanović, Flow-based anomaly intrusion detection system using two neural network stages, Comput. Sci. Inf. Syst., vol. 11, no. 2, pp. 601–622, 2014.

[9] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, Autoencoder-based feature learning for cyber security applications, in Proceedings of the International Joint Conference on Neural Networks, vol. 2017-May, pp. 3854–3861, 2017.

[10] B. Zhang, Y. Yu, and J. Li, Network intrusion detection based on stacked sparse autoencoder and binary tree ensemble method, 2018 IEEE International Conference on Communications Workshops, ICC Workshops 2018 - Proceedings, pp. 1–6, 2018.

[11] B. Abolhasanzadeh, Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features, 7th Conference on Information and Knowledge Technology, IKT 2015, pp. 1–5, 2015.

[12] S. N. Mighan and M. Kahani, Deep Learning Based Latent Feature Extraction for Intrusion Detection, 26th Iranian Conference on Electrical Engineering, ICEE 2018, pp. 1511–1516, 2018.

[13] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, A Deep Learning Approach to Network Intrusion Detection, IEEE Trans. Emerg. Top. Comput. Intell., vol. 2, no. 1, pp. 41–50, 2018.

[14] U. Cekmez, Z. Erdem, A. G. Yavuz, O. K. Sahingoz, and A. Buldu, Network anomaly detection with deep learning, 26th Signal Processing and Communications Applications Conference (SIU), pp. 1–4, 2018.

[15] R. C. Aygun and A. G. Yavuz, A stochastic data discrimination based autoencoder approach for network anomaly detection, 25th Signal Processing and Communications Applications Conference (SIU), pp. 1–4, 2017.

[16] S. Naseer et al., Enhanced Network Anomaly Detection Based on Deep Neural Networks, IEEE Access, vol. 6, pp. 48231–48246, 2018.

[17] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, Autoencoder-based network anomaly detection, Wireless Telecommunications Symposium (WTS), pp. 1–5, 2018.

[18] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, Erişim Tarihi: 30.01.2019.

[19] F. Farahnakian and J. Heikkonen, A deep auto-encoder based approach for intrusion detection system, International Conference on Advanced Communication Technology, ICACT, vol. 2018-Febru, pp. 178–183, 2018.

[20] https://www.unb.ca/cic/datasets/nsl.html., Erişim Tarihi: 30.01.2019.

[21] J. McHugh, Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory, ACM Trans. Inf. Syst. Secur., vol. 3, no. 4, pp. 262–294, 2000.

[22] M. F. Umer, M. Sher, and Y. Bi, Flow-based intrusion detection: Techniques and challenges, Comput. Secur., vol. 70, no. Supplement C, pp. 238–254, 2017.

[23] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, An overview of IP flow-based intrusion detection, IEEE Commun. Surv. Tutorials, vol. 12, no. 3, pp. 343–356, Sep. 2010.

[24] R. Beghdad, Critical study of neural networks in detecting intrusions, Comput. Secur., vol. 27, no. 5–6, pp. 168–175, Oct. 2008.

[25] S. Sui, L. Li, and C. N. Manikopoulo, Flow-based Statistical Aggregation Schemes for Network Anomaly Detection, Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control, ICNSC'06, pp. 786–791, 2006.

[26] Q. A. Tran, F. Jiang, and Ji. Hu, A Real-Time NetFlow-based Intrusion Detection System with Improved BBNN and High-Frequency Field Programmable Gate Arrays, IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, pp. 201–208, 2012.

[27] Z. Jadidi, V. Muthukkumarasamy, and E. Sithirasenan, Metaheuristic algorithms based Flow Anomaly Detector, 19th Asia-Pacific Conference on Communications, APCC 2013, pp. 717–722, 2013.

[28] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection, 2018.

[29] N. Marir, H. Wang, G. Feng, B. Li, and M. Jia, Distributed abnormal behavior detection approach based on deep belief network and ensemble SVM using spark, IEEE Access, vol. 6, pp. 59657–59671, 2018.

[30] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, Deep Learning Approach for Intelligent Intrusion Detection System, IEEE Access, vol. 7, pp. 41525–41550, 2019.

[31] H. J. Liao, C. H. Richard Lin, Y. C. Lin, and K. Y. Tung, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications, vol. 36, no. 1. pp. 16–24, 2013.

[32] P. Winter, E. Hermann, and M. Zeilinger, Inductive Intrusion Detection in Flow-Based Network Data Using One-Class Support Vector Machines, 4th IFIP International Conference on New Technologies, Mobility and Security, pp. 1–5, 2011.

[33] C. Wagner, J. François, R. State, and T. Engel, Machine learning approach for IP-flow record anomaly detection, Lecture Notes in Computer Science vol. 6640 LNCS, no. PART 1, pp. 28–39, 2011.

[34] M. F. Umer, M. Sher, and Y. Bi, A two-stage flow-based intrusion detection model for next-generation networks, PLoS One, vol. 13, no. 1, p. e0180945, 2018.

[35] A. Shubair, S. Ramadass, and A. A. Altyeb, KENFIS: KNN-based evolving neuro-fuzzy inference system for computer worms detection, J. Intell. Fuzzy Syst., vol. 26, no. 4, pp. 1893–1908, 2014.

[36] K. A. P. Costa, L. A. M. Pereira, R. Y. M. Nakamura, C. R. Pereira, J. P. Papa, and A. Xavier Falcão, A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks, Inf. Sci. (Ny)., vol. 294, pp. 95–108, 2015.

[37] A. Lakhina, M. Crovella, and C. Diot, Mining anomalies using traffic feature distributions, Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM ’05, p. 217, 2005.

[38] P. Casas, J. Mazel, and P. Owezarski, UNADA: Unsupervised network anomaly detection using sub-space outliers ranking, in Lecture Notes in Computer Science, vol. 6640 LNCS, no. PART 1, pp. 40–51, 2011.

[39] F. Hosseinpour, P. V. Amoli, F. Farahnakian, J. Plosila, T. Hämäläinen, and C. Author, Artificial Immune System Based Intrusion Detection: Innate Immunity using an Unsupervised Learning Approach, Int. J. Digit. Content Technol. its Appl., vol. 8, no. 5, 2014.

[40] A. Satoh, Y. Nakamura, and T. Ikenaga, A flow-based detection method for stealthy dictionary attacks against Secure Shell, J. Inf. Secur. Appl., vol. 21, pp. 31–41, Apr. 2015.

[41] S. Thaseen and C. A. Kumar, An analysis of supervised tree based classifiers for intrusion detection system, Proceedings of the 2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering, PRIME 2013, pp. 294–299, 2013.

[42] D. Zhao et al., Botnet detection based on traffic behavior analysis and flow intervals, Comput. Secur., vol. 39, no. PARTA, pp. 2–16, 2013.

[43] F. Haddadi, D. Runkel, A. Nur Zincir-Heywood, and M. I. Heywood, On botnet behaviour analysis using GP and C4.5, GECCO 2014 - Companion Publication of the 2014 Genetic and Evolutionary Computation Conference, pp. 1253–1260, 2014.

[44] M. Stevanovic and J. M. Pedersen, An efficient flow-based botnet detection using supervised machine learning, International Conference on Computing, Networking and Communications, ICNC 2014, pp. 797–801, 2014.

[45] Y. Kanda, R. Fontugne, K. Fukuda, and T. Sugawara, ADMIRE: Anomaly detection method using entropy-based PCA with three-step sketches, Comput. Commun., vol. 36, no. 5, pp. 575–588, 2013.

[46] M. H. Haghighat and J. Li, Edmund: Entropy Based Attack Detection and Mitigation Engine Using Netflow Data, Proceedings of the 8th International Conference on Communication and Network Security, pp. 1–6, 2018.

[47] G. E. Hinton and R. S. Zemel, Autoencoders, minimum description length and Helmholtz free energy, in Advances in neural information processing systems, vol. 6, pp. 3–10, 1994.

[48] H. Bourlard and Y. Kamp, Auto-association by multilayer perceptrons and singular value decomposition, Biol. Cybern., vol. 59, no. 4–5, pp. 291–294, 1988.

[49] M. Sakurada and T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, p. 4, 2014.

[50] J. An and S. S. Cho, Variational autoencoder based anomaly detection using reconstruction probability, Seoul, Korea, 2015.

[51] D. P. Kingma and M. Welling, Auto-encoding variational bayes, arXiv Prepr. arXiv1312.6114, 2013.

[52] B. E. Boser, I. M. Guyon, and V. N. Vapnik, A Training Algorithm for Optimal Margin Classifiers, Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152, 1992.

[53] J. C. Platt, Fast training of support vector machines using sequential minimal optimization, Adv. kernel methods, pp. 185–208, 1999.

[54] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput., vol. 13, no. 7, pp. 1443–1471, 2001.

[55] Y. Wang, J. Wong, and A. Miner, Anomaly intrusion detection using one class SVM, Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, pp. 358–364, 2004.

[56] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol., vol. 2, no. 3, p. 27, 2011.

[57] O. Gu, P. Fogla, D. Dagon, W. Lee, and B. Škorić, Measuring intrusion detection capability: An information-theoretic approach, Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ASIACCS '06, vol. 2006, pp. 90–101, 2006.

[58] Y. Xin et al., Machine Learning and Deep Learning Methods for Cybersecurity, IEEE Access, p. 1, 2018.

[59] K. A. Spackman, Signal Detection Theory: Valuable Tools For Evaluating Inductive Learning, Proceedings of the Sixth International Workshop on Machine Learning, Elsevier, pp. 160–163, 1989.

[60] T. Fawcett, ROC Graphs: Notes and Practical Considerations for Researchers, HP Labs Tech Rep. HPL-2003-4, pp. 1–38, 2004.

[61] H. He and Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications. Wiley, 2013.

[62] A. Tharwat, Classification assessment methods, Appl. Comput. Informatics, pp. 1–13, 2018.

[63] Thomas G. Tape, The Area Under an ROC Curve, http://gim.unmc.edu/dxtests/roc3.htm, Erişim Tarihi: 18.02.2019.

[64] X. Zhu and A. B. Goldberg, Introduction to Semi-Supervised Learning, Synth. Lect. Artif. Intell. Mach. Learn., vol. 3, no. 1, pp. 1–130, 2009.

[65] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation, Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, pp. 29–36, 2011.

[66] S. García, M. Grill, J. Stiborek, and A. Zunino, An empirical comparison of botnet detection methods, Comput. Secur., vol. 45, pp. 100–123, 2014.

[67] N. Moustafa and J. Slay, UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), Military Communications and Information Systems Conference, 2015.

[68] Ring, Markus, Sarah Wunderlich, Dominik Grüdl, Dieter Landes, Andreas Hotho. Flow-based benchmark data sets for intrusion detection, In Proceedings of the 16th European conference on cyber warfare and security, pp. 361-369. 2017.

[69] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization, Proceedings of the 4th International Conference on Information Systems Security and Privacy, pp. 108–116, 2018.

[70] C. Beek et al., McAfee Labs Threats Report: September 2017, 2017.

[71] V. L. Cao, M. Nicolau, and J. McDermott, Learning Neural Representations for Network Anomaly Detection, IEEE Trans. Cybern., vol. 49, no. 8, pp. 3074–3087, 2019.

[72] J. Patterson and A. Gibson, Deep Learning: A Practitioner's Approach. O'Reilly Media, 2017.

[73] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[74] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249–256, 2010.

[75] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, On the importance of initialization and momentum in deep learning, 2013.

[76] A. L. Maas, A. Y. Hannun, and A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in in ICML Workshop on Deep Learning for Audio, Speech and Language Processing, pp. 1–8, 2013.

[77] C. Zhou, R. C. Paaenroth, and R. C. Paffenroth, Anomaly detection with robust deep autoencoders, Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, vol. Part F1296, pp. 665–674, 2017.

[78] Deeplearning4j, https://deeplearning4j.org/, Erişim Tarihi: 23.02.2019.

[79] LIBSVM -- A Library for Support Vector Machines, https://www.csie.ntu.edu.tw/~cjlin/libsvm/, Erişim Tarihi: 23.02.2019.

[80] S. Aksoy and R. M. Haralick, Feature normalization and likelihood-based similarity measures for image retrieval, Pattern Recognit. Lett., vol. 22, no. 5, pp. 563–582, 2001.

[81] W. J. Krzanowski and D. J. Hand, ROC Curves for Continuous Data. CRC Press, 2009.

[82] T. Jafarian, M. Masdari, A. Ghaffari, and K. Majidzadeh, A survey and classification of the security anomaly detection mechanisms in software defined networks, Cluster Comput., pp. 1–19, 2020.

[83] Y. Hande and A. Muddana, A survey on intrusion detection system for software defined networks (SDN), Int. J. Bus. Data Commun. Netw., vol. 16, no. 1, pp. 28–47, 2020.

[84] Y. Zhang, An adaptive flow counting method for anomaly detection in SDN, Proceedings of the ninth ACM conference on Emerging networking experiments and technologies. ACM, Santa Barbara, California, USA, pp. 25–30, 2013.

[85] G. Garg and R. Garg, Detecting anomalies efficiently in SDN using adaptive mechanism, International Conference on Advanced Computing and Communication Technologies, ACCT, vol. 2015-April, pp. 367–370, 2015.

[86] T. Ha et al., Suspicious traffic sampling for intrusion detection in software-defined networks, Comput. Networks, vol. 109, Part, pp. 172–182, 2016.

[87] B. R. Granby, B. Askwith, and A. K. Marnerides, SDN-PANDA: Software-Defined Network Platform for ANomaly Detection Applications, IEEE 23rd International Conference on Network Protocols, pp. 463–466, 2015.

[88] S. Hommes, R. State, and T. Engel, Implications and detection of DoS attacks in OpenFlow-based networks, IEEE Glob. Commun. Conf. GLOBECOM, pp. 537–543, 2014.

[89] L. F. Carvalho, G. Fernandes, J. J. P. C. Rodrigues, L. S. Mendes, and M. L. Proenca, A novel anomaly detection system to assist network management in SDN environment, IEEE Int. Conf. Commun., 2017.

[90] S. Lee, J. Kim, S. Shin, P. Porras, and V. Yegneswaran, Athena: A Framework for Scalable Anomaly Detection in Software-Defined Networks, 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 249–260, 2017.

[91] D. He, S. Chan, X. Ni, and M. Guizani, Software-Defined-Networking-Enabled Traffic Anomaly Detection and Mitigation, IEEE Internet Things J., vol. 4, no. 6, pp. 1890–1898, Dec. 2017.

[92] L. F. Carvalho, T. Abrão, L. de S. Mendes, and M. L. Proença, An ecosystem for anomaly detection and mitigation in software-defined networking, Expert Syst. Appl., vol. 104, pp. 121–133, 2018.

[93] H. Peng, Z. Sun, X. Zhao, S. Tan, and Z. Sun, A Detection Method for Anomaly Flow in Software Defined Network, IEEE Access, 2018.

[94] S. E. Schechter, J. Jung, and A. W. Berger, Fast detection of scanning worm infections, Lect. Notes Comput. Sci., vol. 3224, pp. 59–81, 2004.

[95] J. Twycross and M. M. Williamson, Implementing and Testing a Virus Throttle, in Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12, p. 20, 2003.

[96] M. M. Williamson, Throttling viruses: Restricting propagation to defeat malicious mobile code, Proceedings - Annual Computer Security Applications Conference, ACSAC, vol. 2002-January, pp. 61–68, 2002.

[97] M. V. Mahoney, Network traffic anomaly detection based on packet bytes, Proceedings of the 2003 ACM symposium on Applied computing - SAC '03, p. 346, 2003.

[98] R. T. Kokila, S. Thamarai Selvi, and K. Govindarajan, DDoS detection and analysis in SDN-based environment using support vector machine classifier, in 6th International Conference on Advanced Computing, ICoAC 2014, pp. 205–210, 2015.

[99] 2000 DARPA Intrusion Detection Scenario Specific Datasets | MIT Lincoln Laboratory, https://www.ll.mit.edu/r-d/datasets/2000-darpa-intrusion-detection-scenario-specific-datasets, Erişim Tarihi: 13.12.2020.

[100] R. Sathya and R. Thangarajan, Efficient anomaly detection and mitigation in software defined networking environment, Electronics and Communication Systems (ICECS), 2015 2nd International Conference on, pp. 479–484, 2015.

[101] C. Yang, Anomaly network traffic detection algorithm based on information entropy measurement under the cloud computing environment, Cluster Comput., vol. 22, no. 4, pp. 8309–8317, 2019.

[102] R. Wang, Z. Jia, and L. Ju, An entropy-based distributed DDoS detection mechanism in software-defined networking, Proceedings - 14th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom 2015, vol. 1, pp. 310–317, 2015.

[103] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, Combining Open Flow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments, Comput. Networks, vol. 62, pp. 122–136, 2014.

[104] N. Visible and H.- Packard, Traffic Monitoring using sFlow, Analysis, http://www.sflow.org/sFlowOverview.pdf, Erişim Tarihi: 13.12.2019.

[105] J. Francois and O. Festor, Anomaly traceback using software defined networking, IEEE International Workshop on Information Forensics and Security (WIFS), pp. 203–208, 2014.

[106] L. Deng and D. Yu, Deep learning: Methods and applications, Foundations and Trends in Signal Processing, vol. 7, no. 3–4. Now Publishers Inc, pp. 197–387, 2013.

[107] S. K. Dey and M. M. Rahman, Flow based anomaly detection in software defined networking: A deep learning approach with feature selection method, in 4th International Conference on Electrical Engineering and Information and Communication Technology, pp. 630–635, 2019.

[108] Q. Niyaz, W. Sun, and A. Y. Javaid, A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN), arXiv Prepr. arXiv1611.07400, 2016.

[109] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, Deep learning approach for Network Intrusion Detection in Software Defined Networking, Proceedings - 2016 International Conference on Wireless Networks and Mobile Communications, WINCOM 2016: Green Communications and Networking, pp. 258–263, 2016.

[110] S. Garg, N. Kumar, J. J. P. C. Rodrigues, and J. J. P. C. Rodrigues, Hybrid deep-learning-based anomaly detection scheme for suspicious flow detection in SDN: A social multimedia perspective, IEEE Trans. Multimed., vol. 21, no. 3, pp. 566–578, 2019.

[111] J. Li, Z. Zhao, and R. Li, Machine learning-based IDS for softwaredefined 5G network, IET Networks, vol. 7, no. 2, pp. 53–60, 2018.

[112] A. Santos Da Silva, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN, Proc. NOMS 2016 - 2016 IEEE/IFIP Netw. Oper. Manag. Symp., no. Noms, pp. 27–35, 2016.

[113] C. Pang, Y. Jiang, and Q. Li, FADE: Detecting forwarding anomaly in software-defined networks, IEEE International Conference on Communications, 2016.

[114] S. Seungwon and G. Guofei, CloudWatcher: Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?), IEEE International Conference onNetwork Protocols (ICNP), pp. 1–6, 2012.

[115] Y. Cui et al., SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks, J. Netw. Comput. Appl., vol. 68, pp. 65–79, 2016.

[116] R. Braga, E. Mota, and A. Passito, Lightweight DDoS flooding attack detection using NOX/OpenFlow, Proceedings - Conference on Local Computer Networks, LCN, pp. 408–415, 2010.

[117] Open Networking Foundation, https://www.opennetworking.org/, Erişim Tarihi: 13.06.2018.

[118] ONF and O. N. Foundation, Software-Defined Networking: The New Norm for Networks, 2012.

[119] Y. Jarraya, T. Madi, and M. Debbabi, A Survey and a Layered Taxonomy of Software-Defined Networking, Commun. Surv. Tutorials, IEEE, vol. PP, no. 99, p. 1, 2014.

[120] N. McKeown et al., OpenFlow: Enabling Innovation in Campus Networks, SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69–74, 2008.

[121] NOX SDN Controller, http://www.noxrepo.org/, Erişim Tarihi: 05.12.2017.

[122] D. Erickson, The Beacon OpenFlow controller, HotSDN 2013 - Proceedings of the 2013 ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, pp. 13–18, 2013.

[123] Floodlight OpenFlow Controller, http://www.projectfloodlight.org/floodlight/, Erişim Tarihi: 13.01.2017.

[124] OpenDaylight | A Linux Foundation Collaborative Project, http://www.opendaylight.org/, Erişim Tarihi: 13.01.2017.

[125] OpenFlow switch support — ns-3 vns-3-dev documentation, https://www.nsnam.org/docs/release/3.13/models/html/openflow-switch.html, Erişim Tarihi: 13.01.2017.

[126] M. Gupta, J. Sommers, and P. Barford, Fast, accurate simulation for SDN prototyping, Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, Hong Kong, China, pp. 31–36, 2013.

[127] W. Shie-Yuan, C. Chih-Liang, and Y. Chun-Ming, EstiNet openflow network simulator and emulator, Commun. Mag. IEEE, vol. 51, no. 9, pp. 110–117, 2013.

[128] Mininet Team, Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet, http://mininet.org/, Erişim Tarihi: 13.01.2017.

[129] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, Reproducible network experiments using container-based emulation, Proceedings of the 8th international conference on Emerging networking experiments and technologies, pp. 253–264, 2012.

[130] P. Wette, M. Draxler, A. Schwabe, F. Wallaschek, M. H. Zahraee, and H. Karl, Maxinet: Distributed emulation of software-defined networks, Networking Conference, 2014 IFIP, pp. 1–9, 2014.

[131] S. Zavrak and M. Iskefiyeli, A feature-based comparison of SDN emulation and simulation tools, Proc. Int. Conf. Eng. Technol.(ICENTE), pp. 214–217, 2017.

[132] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, A learning algorithm for Boltzmann machines, Cogn. Sci., vol. 9, no. 1, pp. 147–169, 1985.

[133] [133] R. Hecht-Nielsen, Replicator neural networks for universal optimal source coding, Science (80-.)., vol. 269, no. 5232, pp. 1860–1863, 1995.

[134] S. Hawkins, H. He, G. Williams, and R. Baxter, Outlier detection using replicator neural networks, Lecture Notes in Computer Science, vol. 2454 LNCS, pp. 170–180, 2002.

[135] L. Deng and D. Yu, Deep learning: methods and applications, Found. trends signal Process., vol. 7, no. 3–4, pp. 197–387, 2014.

[136] C. G. Cordero, S. Hauke, M. Muhlhauser, and M. Fischer, Analyzing flow-based anomaly intrusion detection using Replicator Neural Networks, 14th Annual Conference on Privacy, Security and Trust, pp. 317–324, 2016.

[137] H. A. Dau, V. Ciesielski, and A. Song, Anomaly detection using replicator neural networks trained on examples of one class, in Asia-Pacific Conference on Simulated Evolution and Learning, pp. 311–322, 2014.

[138] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res., vol. 15, no. 1, pp. 1929–1958, 2014.

[139] L. Tóth and G. Gosztolya, Replicator neural networks for outlier modeling in segmental speech recognition, International Symposium on Neural Networks, pp. 996–1001, 2004.

[140] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, On the importance of initialization and momentum in deep learning, International conference on machine learning, pp. 1139–1147, 2013.

[141] S. Hochreiter and J. Schmidhuber, Long Short-Term Memory, Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[142] K. Cho et al., Learning phrase representations using RNN encoder-decoder for statistical machine translation, arXiv Prepr. arXiv1406.1078, 2014.

[143] I. Sutskever, O. Vinyals, and Q. V Le, Sequence to sequence learning with neural networks, Advances in neural information processing systems, pp. 3104–3112, 2014.

[144] J. Li, M.-T. Luong, and D. Jurafsky, A hierarchical neural autoencoder for paragraphs and documents, arXiv Prepr. arXiv1506.01057, 2015.

[145] O. Vinyals, Ł. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton, Grammar as a foreign language, Advances in neural information processing systems, pp. 2773–2781, 2015.

[146] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, Scheduled sampling for sequence prediction with recurrent neural networks, in Advances in Neural Information Processing Systems, pp. 1171–1179, 2015.

[147] Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G., LSTM-based encoder-decoder for multi-sensor anomaly detection, arXiv preprint arXiv:1607.00148, 2016.

[148] S. R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, PayLess: A low cost network monitoring framework for Software Defined Networks, 2014 IEEE Network Operations and Management Symposium (NOMS), pp. 1–9, 2014.

[149] Open vSwitch, http://openvswitch.org/, Erişim Tarihi: 23.02.2019.

[150] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Comput. Secur., vol. 31, no. 3, pp. 357–374, 2012.

[151] Tcpreplay, http://tcpreplay.synfin.net/, Erişim Tarihi: 23.02.2019.

[152] Hping - Active Network Security Tool, http://www.hping.org/, Erişim Tarihi: 16.11.2019.

[153] PyTorch, https://pytorch.org/, Erişim Tarihi: 16.11.2020.

[154] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, Deep Recurrent Neural Network for Intrusion Detection in SDN-based Networks, in 2018 4th IEEE Conference on Network Softwarization and Workshops, NetSoft 2018, pp. 462–469, 2018.

[155] A. Abubakar and B. Pranggono, Machine learning based intrusion detection system for software defined networks, Proceedings - 2017 7th International Conference on Emerging Security Technologies, EST 2017, pp. 138–143, 2017.

[156] P. Wang, K. M. Chao, H. C. Lin, W. H. Lin, and C. C. Lo, An Efficient Flow Control Approach for SDN-Based Network Threat Detection and Migration Using Support Vector Machine, IEEE 13th International Conference on e-Business Engineering (ICEBE), pp. 56–63, 2016.

ÖZGEÇMİŞ

Sultan Zavrak, 1987 yılında Trabzon'da doğdu. İlk, orta ve lise eğitimini Trabzon'da tamamladı. 2004 yılında Fatih Lisesi'nden mezun oldu. Karadeniz Teknik Üniversitesi Bilgisayar Mühendisliği lisans programını 2010 yılında, yüksek lisans programını ise 2013'te bitirdi. Sakarya Üniversitesi Bilgisayar ve Bilişim Mühendisliği Anabilim Dalı'nda doktora programına devam etmektedir. Düzce Üniversitesi Bilgisayar Mühendisliği Bölümü'nde 2011 yılında araştırma görevlisi olarak başladığı görevine halen devam etmektedir. Bilgisayar ağları, ağ güvenliği ve makine öğrenmesi alanlarında araştırmalarını sürdürmektedir.