
GAZİ ÜNİVERSİTESİ
Gazi
1926

DERİN ÖĞRENME YÖNTEMLERİ İLE KABLOSUZ AĞLARA SIZMA TESPİTİ

Emre HALİSDEMİR

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

EYLÜL 2023

ETİK BEYAN

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

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

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

Emre HALİSDEMİR

21/09/2023

DERİN ÖĞRENME YÖNTEMLERİ İLE KABLOSUZ AĞLARA SIZMA TESPİTİ

(Doktora Tezi)

Emre HALİSDEMİR

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Eylül 2023

ÖZET

Bu tez çalışmasında, siber güvenlik mimarisinin en önemli bileşenlerinden olan ağ sızma tespit sistemlerinin derin öğrenme algoritmaları kullanılarak geliştirilmesi hedeflenmiştir. Sağladığı uzun ömürlü kısa dönem bellek ile gradyanın yok olması problemine çözüm olarak sunulan Long Short Term Memory (LSTM) tabanlı bir model geliştirilmiş, model performansının iyileştirilmesi amacıyla yeni düzenlileştirme teknikleri uygulanmıştır. LSTM modelinin NSL-KDD verisetine uygulanması ile %83,18 kesinlik değeri elde edilmiştir. Uzun süreli kısa dönem belleğin model performansına etkisinin gözlemlenmesi amacıyla, Gated Recurrent Unit (GRU) ve basit Recurrent Neural Network (RNN) modelleri geliştirilerek elde edilen sonuçlar tartışılmıştır. Bununla birlikte, NSL-KDD verisetinde yer alan sayısal değerlerden yararlanılarak literatürde ilk defa kullanılan bir yöntem ile imaj veriseti elde edilmiş ve görüntü sınıflandırmada etkin olarak kullanılan Convolutional Neural Network (CNN) tabanlı model geliştirilerek %90,99 oranında bir kesinlik seviyesine ulaşılmıştır. Bunun literatürde bu alanda yapılan çalışmalardan daha iyi bir sonuç olduğu gösterilmiştir. Özellik seçiminin işlem maliyetine etkisinin incelenmesi amacıyla, Chi-square, Principal Component Analysis (PCA) ve lokal algoritma özellik seçim metotları ile elde edilen veriseti özellik kümeleri ile LSTM modeli test edilerek değerlendirmede bulunulmuştur. Ardından, bir derin öğrenme uygulamasının başarısındaki en önemli bileşenlerden biri olan verisetine yoğunlaşılmış, CICIDS 2017, CICIDS-001, UGR'16 ve UNSW-NB15 gibi güncel verisetlerinin özellikleri incelenmiştir. Bunlardan UNSW-NB15 verisetinin LSTM modeline uygulanması neticesinde elde edilen sonuçlar paylaşılmıştır. Literatüre katkı sağlayacak yeni ve erişilebilir bir veriseti üretimi amacıyla, NATO CCDCOE (Cooperative Cyber Defence Centre of Excellence) tarafından her yıl düzenlenen ve siber savunma alanında önemli bir yeri olan Kilitli Kalkanlar (Locked Shields) tatbikatı ağ trafiğinden yararlanılarak veriseti üretimi gerçekleştirilmiştir. Bu amaçla tatbikata ilk defa sanal mavi takım eklenmesi ve verisetinin bu takımın ağ trafiği kullanılarak üretilmesi sağlanmıştır. Kayıt etiketlemede özgün bir yöntem kullanılarak daha güvenilir bir veriseti elde edilmiştir.

Bilim Kodu	:	92432
Anahtar Kelimeler	:	Yapay zekâ, derin öğrenme, bilgi güvenliği, ağ sızma tespiti
Sayfa Adedi	:	159
Danışman	:	Prof. Dr. Hacer KARACAN

WIRELESS NETWORK INTRUSION DETECTION USING DEEP LEARNING METHODS

(Ph. D. Thesis)

Emre HALİSDEMİR

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

September 2023

ABSTRACT

In this thesis, we aimed to develop intrusion detection systems, which are crucial components of cybersecurity architecture, utilizing deep learning algorithms. A Long Short Term Memory (LSTM) model, which addresses the problem of vanishing gradient through its long-term short-term memory, was developed, and novel regularization techniques were implemented to enhance the model performance. When the LSTM model was applied to the NSL-KDD dataset, an accuracy value of 83.18% was achieved. To observe the impact of long-term short-term memory on model performance, Gated Recurrent Unit (GRU) and simple Recurrent Neural Network (RNN) models were developed, and the obtained results were discussed. Additionally, an image dataset was obtained using a method utilizing numerical values from the NSL-KDD dataset, which is used for the first time in the literature, and a Convolutional Neural Network (CNN)-based model, commonly used for image classification, was developed, achieving an accuracy level of 90.99%. To examine the impact of feature selection on computational cost, the dataset obtained using Chi-square, Principal Component Analysis (PCA), and local algorithm feature selection methods were tested with the LSTM model, and evaluations were done. Next, we focused on the dataset, which is one of the most important components for the success of a deep learning model, so the features of recent datasets such as CICIDS 2017, CICIDS-001, UGR'16, and UNSW-NB15 were examined. The results obtained from applying the UNSW-NB15 dataset to the LSTM model were shared. To contribute to the literature by generating a new publicly available dataset, we produced a dataset using the network traffic of the Locked Shields exercise, which is organized annually by NATO CCDCOE (Cooperative Cyber Defence Centre of Excellence) and holds an important place in the field of cyber security. For this purpose, a virtual blue team was implemented into the exercise for the first time, and the dataset was generated using the network traffic of this team. A novel method was used for labeling to obtain a more reliable dataset.

Science Code	:	92432
Key Words	:	Artificial Intelligence, deep learning, information security, network intrusion detection
Page Number	:	159
Supervisor	:	Prof. Dr. Hacer KARACAN

TEŞEKKÜR

Tez çalışmasının her aşamasında beni sıkılmadan dinleyen ve değerli tavsiyeleri ile tezin hazırlanmasına büyük katkı veren değerli danışman hocam Sayın Prof. Dr. Hacer KARACAN'a, tez izleme komite toplantılarımızda yaptıkları çok kıymetli girdiler ile tezin olgunlaşmasını sağlayan Sayın Prof. Dr. M. Ali AKCAYOL'a, her zaman yanımda olan ve beni destekleyen sevgili eşim Derya HALİSDEMİR ve değerli aileme teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

KISALTMALAR

Bu çalışmada kullanılmış kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
ADAM	Automated Digital Analysis and Measurement
AE	Autoencoders
AICA	Autonomous Intelligent Cybersecurity Agents
ARPANET	The Advanced Research Projects Agency Network
CCDCOE	Cooperative Cyber Defence Centre of Excellence
CNN	Convolutional Neural Network
CyCon	International Conference on Cyber Conflict
DARPA	Defense Advanced Research Projects Agency
DBN	Deep Belief Networks
DDoS	Distributed Denial of Service
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DoS	Denial of Service
DR	Detection Rate
ERSPAN	Encapsulataed Remote Switched Port Analyzer
FAR	False Alarm Rate
fMRG	fonksiyonel Manyetik Rezonans Görüntüleme
FN	False Negatif
GRU	Gated Recurrent Unit
IDS	Intrusion Detection Systems
KLD	Kullback-Leibler Divergence
K-NN	K-Nearest Neighbour
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MIT	Massachusetts Institute of Technologies

Kısaltmalar	Açıklamalar
MSE	Mean Square Error
MSLE	Mean Squared Logarithmic Error
NDAE	Nonsymmetric Deep Autoencoder
NCP	Network Control Protocol
NSL-KDD	Network Security Laboratory – Knowledge Discovery and Data Mining
PCA	Principal Componen Analysis
PCAP	Packet Capture
PNN	Probabilistic Neural Network
R2L	Remote to Local
RBM	Restricted Boltzman Machine
RELU	Rectified Linear Unit
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
TCP/IP	Transmission Control Protocol/Internet Protocol
TP	True Pozitif
U2R	User to Root
UDP	User Datagram Protocol
VBT	Virtual Blue Team
YSA	Yapay Sinir Ağı

1. GİRİŞ

İnternetin temeli, 1960'lı yılların başında MIT (Massachusetts Institute of Technologies) üniversitesinden Leonard Kleinrock'un paket anahtarlama üzerine yazdığı makale ile atılmıştır. Thomas Merrill tarafından Massachusetts'te bulunan TX-2 bilgisayarı ve Kaliforniya'daki Q-32 arasında düşük hızlı dial-up telefon hattı üzerinden yapılan bağlantı ile önemli bir ilerleme kaydedilmiştir. 1960'lı yılların sonundan 1970'li yılların başına kadar DARPA'da (Defense Advanced Research Projects Agency) yapılan çalışmalar neticesinde ARPANET (The Advanced Research Projects Agency Network) adı verilen ilk internet ağı kurulmuş, bilgisayar bağlantısı gerçekleştirilmiş ve ilk host-to-host protokolü olan NCP (Network Control Protocol) kullanıma verilmiştir [1].

İnternet ilk dönemde sadece üniversiteler arası iletişim ve askeri amaçlarla kullanılmıştır. Zamanla TCP/IP (Transmission Control Protocol/Internet Protocol), DHCP (Dynamic Host Configuration Protocol), UDP (User Datagram Protocol) gibi protokollerin geliştirilmesi, kablolama ve anahtar altyapısının hızla yaygınlaşması, servis sağlayıcılar tarafından sunulan hizmet maliyetinin düşmesi gibi gelişmeler neticesinde herkes tarafından erişilebilir hale gelmiştir. Bankacılık siteleri, öğrenci bilgi sistemleri, çevrimiçi alışveriş portalları, e-posta servisleri, sosyal medya ağları veçevrimiçi oyunlar gibi çok kullanıcılı birçok uygulama internet üzerinden servis sunmaya başlamıştır. Ayrıca çoğu kurum, kullanıcılarına verilen çevrimiçi servislerin yanısıra, kurum içi işlemlerin büyük bölümünü internet ortamında yürütmeye başlamıştır.

İnternetin yaygınlaşması ve üzerinde işlenen verinin değerli hale gelmesi, zararlı yazılım ve tehditlerin artmasına neden olmuştur. Online Trust Alliance tarafından Temmuz 2019'da yayınlanan 2018 Siber Olay ve İhlal Trendleri Raporu'nda, 2018 yılı içerisinde 2 milyondan fazla siber olayın gerçekleştiği, bu olayların mali etkisinin 45 milyar dolardan fazla olduğu belirtilmektedir [2]. Siber olay ve ihlaller nedeniyle oluşan mali zararın yanısıra, maddi karşılığı olmayan kötü sonuçlar da meydana gelmektedir. Kurum/kuruluş, hatta bazı durumlarda devlet itibarının zedelenmesi, internet ortamında işlenen veya depolanan verinin korunması ihtiyacını ortaya çıkarmıştır.

Temel hedefi değerli verinin korunması olan bilgi güvenliği şemsiyesi altında, personel güvenliği, doküman güvenliği, fiziki güvenlik ve bilgi sistemleri güvenliği faaliyetleri gerçekleştirilmektedir. Bilgi sistemleri güvenliği kapsamında; şahsi bilgisayarları etkileyen virüs, trojan, arka kapı tehditleri, gateway'ler üzerine yapılan DoS (Denial of Service) atakları, dosya sunucularına yönelik veri bütünlüğü tahrip saldırıları, kimliklendirme sunucuları üzerinden yapılan kimlik hırsızlıkları, isimlendirme sunucularına yönelik DNS (Domain Name Server) atakları gibi çok çeşitli güvenlik tehtidi ile başa çıkmak gerekmektedir. Süreç içerisinde bu tehditleri önlemek için çeşitli güvenlik çözümleri geliştirilmiştir. Bununla birlikte, etkin ve tam bir bilgi sistemi güvenliği, farklı güvenlik bileşenlerinin birlikte kullanımı ile sağlanmaktadır. Bu nedenle güvenlik duvarları (firewall), hacklemeye karşı geliştirilen uygulamalar ve antivirüs yazılımlarının yanısıra, bu önlemleri aşarak sisteme yapılabilecek sızma girişimlerinin önceden tespiti için ağ sızma tespit sistemleri (Intrusion Detection Systems -IDS) kullanımına ihtiyaç duyulmaktadır [3]. Bilgisayar ağlarına yapılan sızma girişimleri (network intrusion), bilgi sistemleri güvenliğinin kötücül yöntemler kullanılarak zafiyete uğratılması şeklinde gerçekleştirilmekte olup, ağ sızma tespiti (network intrusion detection) bu zararlı aktivitelerin belirlenerek bilgi sistemlerinin gizlilik, bütünlük ve erişilebilirliğinin sağlanması ve sürdürülmesini hedeflemektedir [4].

Ağ sızma tespit metodolojilerine bakıldığında imza-tabanlı tespit (signature-based detection -SD), anomaly-tabanlı tespit (anomaly-based detection -AD) ve durum protokol analizi (stateful protocol analysis -SPA) olmak üzere üç ana kategoriye ayrıldığı görülmektedir. Liao ve diğerleri (2013) çalışmalarında, ağ güvenliğinin sağlanmasında daha etkin ve kesin sonuçların elde edilmesi için metodolojilerin hibrit şekilde kullanımının uygun olacağını ifade etmektedir [5]. Aynı çalışmada sızma tespit yaklaşımlarının iki kategoride ele alındığı, bunların anomali tespiti ve kötü niyetli kullanım tespiti olduğu ifade edilmekte, bu iki metodoloji de üç alt gruba ayrılarak hesaplamaya dayalı yaklaşım, yapay zekâ ve biyolojik yaklaşımlar olarak belirlenmektedir. Ayrıca, tespit yaklaşımları; istatistiğe dayalı (istatistik, uzaklık tabanlı, Bayesian tabanlı, oyun teorisi), pattern tabanlı (pattern eşleştirme, klavye izleme, dosya sistemi kontrolü), kural tabanlı (kural tabanlı, veri madenciliği, model/profil tabanlı, Support Vector Machine (SVM)), durum tabanlı (durum geçiş analizi, Markov süreç modeli, protokol analizi) ve heuristik tabanlı (sinir ağları, bulanık mantık, genetik algoritma, bağışıklık sistemi, swarm zekası) olmak üzere alt başlıklara ayrılmaktadır. Gupta ve diğerleri (2013), kablosuz ağlara yapılabilecek saldırıları, erişim kontrolü saldırıları, gizlilik

saldırıları, bütünlük saldırıları, yetkilendirme saldırıları ve erişilebilirlik saldırıları olarak sınıflandırmakta ve her bir saldırı türüne yönelik örnekleri paylaşmaktadır [6].

Bilgisayar ağlarına sızmada kullanılan tekniklerin sürekli gelişim göstermesi ve karmaşık hale gelmesi, geçmişe oranla daha kabiliyetli ve etkin yöntemlere olan ihtiyacı arttırmaktadır [7]. Geçmişte SVM [6, 8, 9], K-Nearest Neighbour (K-NN) [10,11], Decision Tree [12-14], Naive Bayes, [15-17] sınıflandırması gibi birçok makine öğrenme yöntemi kullanılmış, bazı çalışmalarda bu yöntemlerin hibrit kullanımı [18-20] gerçekleştirilmiştir. Son dönemde yapılan çalışmalarda ise Yapay Sinir Ağı (YSA) tabanlı Derin Öğrenme (Deep Learning) tekniklerinin ağ sızma tespitinde kullanılmaya başladığı görülmektedir.

Tez çalışmasında, derin öğrenme yöntemleri ile kablosuz ağlara sızma tespiti üç problem alanında ele alınmaktadır. Öncelikle ağ sızma tespitinde kullanılabilecek optimum derin öğrenme modelinin belirlenmesine odaklanılmış, bu amaçla derin öğrenmenin matematiksel temelleri ve yöntemlerin özelliklerine yoğunlaşılarak detaylı bir inceleme yapılmıştır. Çeşitli modeller üzerinde testler gerçekleştirilerek elde edilen sonuçlar paylaşılmıştır. Hiperparametrelerin doğru belirlenmesi ve kullanımı bir derin öğrenme modelinin başarısına doğrudan etki etmektedir. Bu nedenle, hiperparametre özelliklerine yönelik detaylı inceleme yapılmış, optimum hiperparametre değerleri uygulamaya yansıtılarak yapılan testler neticesinde elde edilen sonuçlar sunulmuştur. Bununla birlikte, model başarısının arttırılması için düzenlileştirme işlemlerinin yapılması gerektiği belirlenmiş, çeşitli teknikler incelenerek uygun düzenlileştirme teknikleri modellere uyarlanmıştır. İkinci problem alanı olarak verisetinde özellik seçiminin model başarısına etkisi incelenmiştir. Bu kapsamda üç farklı özellik seçim algoritmasının NSL-KDD (Network Security Laboratory – Knowledge Discovery and Data Mining) verisetine uygulanması neticesinde elde edilen özellik setleri kullanılarak test işlemleri gerçekleştirilmiş, elde edilen sonuçlar paylaşılarak değerlendirmede bulunulmuştur. Modelin, eğitim, doğrulama ve testinde kullanılan verisetleri son problem alanı olarak belirlenmiş, ağ sızma tespit sistemlerinde yaygın olarak kullanılan verisetleri ile ilgili inceleme sonuçları paylaşılmıştır. Bir verisetinin nicelik ve nitelik olarak iyi olması, model başarısı açısından oldukça önemlidir. Model ne kadar güçlü olursa olsun, derin öğrenme uygulamalarında başarının temelinde kullanılan verisetinin kalitesi yatmaktadır. Bu, sadece ağ sızma tespiti için değil, tüm yapay zekâ problemleri için geçerlidir. Bazı problemler için genel veriseti bulmak oldukça zordur ve verisetinin araştırmacı tarafından üretilmesi gerekebilir. Tez çalışmasında, bilinen verisetlerinden farklı

olarak özgün bir verisetinin kullanım durumunun incelenmesi amacıyla, siber savunma alanında önemli bir yeri olan ve her yıl NATO CCDCOE (Cooperative Cyber Defence Centre of Excellence) tarafından düzenlenen Locked Shileds tatbikatında yakalanan ağ trafiğinin işlenmesi ile elde edilen örnek verisetinin modele uygulanması neticesinde alınan sonuçlar paylaşılmıştır.

Hazırlanan tez çalışmasının katkıları aşağıda sunulmuştur:

Derin öğrenme yöntemlerinin ağ sızma tespitinde kullanımı göreceli olarak yeni bir çalışma alanıdır. Bu kabulle tez çalışmasında derin öğrenme yöntemlerinin ağ sızma tespitinde kullanımına yönelik detaylı bir literatür araştırması sunulmaktadır. Bununla birlikte, incelenen çalışmalarda kullanılan yöntemler arasından uygun olanları belirlenmiş, daha iyi sınıflandırma oranı elde etmek için model üzerinde çeşitli iyileştirme çalışmaları yapılmıştır. Derin öğrenme uygulamalarında veriseti boyutu, katmanlardaki hücre sayısı, dropout oranı, mini-batch-size, epoch (tekrar sayısı), optimizasyon algoritması, öğrenme oranı, momentum katsayısı, aktivasyon fonksiyonları gibi çeşitli hiperparametreler kullanılmaktadır. Bu parametre değerlerinin uygun şekilde belirlenmesi model performansını önemli ölçüde etkilemektedir. Yapılan incelemede, öncelikle doğru hiper-parametrelerin belirlenmesine odaklanılması gerektiği, bu nedenle başlangıç aşamasında kurulan modelde gizli katman sayısı veya tekrar sayısının düşük tutulmasının uygun olduğu belirlenmiştir. Tezde ayrıca hiperparametreler ve bunların model başarısına etkileri yapılan testlerle ortaya koyulmuş, her hiperparametre için değerlendirmede bulunulmuştur. Ayrıca düzenlileştirme işlemlerinin, modelin aşırı öğrenme problemine çözüm olarak sunulduğu tespiti ile çeşitli düzenlileştirme işlemleri uygulanarak model performansının arttırılması sağlanmıştır. Bununla birlikte, RNN (Recurrent Neural Network) tabanlı modellerde, uzun süreli bellek probleminin çözümünde kullanılan hücresel kapıların, IDS model performasına etkisinin incelenmesi amacıyla, LSTM (Long Short-Term Memory) yönteminin yanı sıra, simple RNN ve GRU (Gated Recurrent Unit) yöntemlerini kullanan modeller geliştirilmiş, bu modellerin performans kıyaslaması sunulmuştur.

Literatürdeki bazı çalışmalarda, geliştirilen modelin hız ve başarı seviyelerinin arttırılması için özellik seçim algoritmaları ile veriseti boyutunda azaltmaya gidildiği görülmüş, model başarısına etkisinin belirlenmesi amacıyla Chi-square, PCA (Principal Componen Analysis)

ve lokal algoritma tekniklerinin NSL-KDD verisetine uygulanması ile elde edilen özellik setleri kullanılarak çeşitli testler gerçekleştirilmiş ve elde edilen sonuçlar paylaşılmıştır.

Derin öğrenme modellerinin görüntü sınıflandırma kabiliyetlerinden etkin şekilde yararlanmak amacıyla, görüntü sınıflandırma kabiliyeti oldukça yüksek olan Convolutional Neural Network (CNN) tabanlı bir model geliştirilmiş, modelin eğitim ve testinde kullanılmak üzere nümerik tabanlı NSL-KDD veriseti imaj verisetine dönüştürülmüştür. Dönüştürme işleminde literatürde karşılaşılan çalışmalardan farklı bir yöntem kullanılmıştır. Yapılan testler neticesinde elde edilen sonuçların önceki sonuçlara oranla oldukça iyi bir seviyeye ulaştığı gözlemlenmiştir. Elde edilen bulgular teze yansıtılmıştır.

Tezde, ağ sızma tespitinde kullanılan verisetleri ile ilgili araştırma sonuçları paylaşılmış, verisetlerinin genel karakteristikleri hakkında değerlendirmede bulunulmuştur. Tezin ayırt edici hedeflerinden biri, özgün bir veriseti üretimi olarak belirlenmiştir. Bu kapsamda, tez çalışmasında veriseti üretim metodolojisi paylaşılmış, bu metodolojinin diğer promlemlere ait veriseti üretiminde de kullanılabileceği değerlendirilmiştir. Literatürde yer alan verisetleri incelenmiş, bunların artı ve eksi yönleri ortaya koyulmuştur. Bu bilgiler doğrultusunda, özgün, yeni ve birçok artı özellik barındıran bir veriseti üretimi için araştırma gerçekleştirilmiş, siber güvenlik tatbikatlarının kaliteli veriseti üretimi için oldukça verimli bir altyapı sunduğu tespit edilmiştir. Ancak üretilen verisetinin gizlilik gerekçeleri ile yayınlanamadığı, bu problemin aşılması gerektiği görülmüştür. Bu sorunun aşılması amacıyla tez çalışması kapsamında hazırlanan ve International Conference on Cyber Conflict (CyCon) konferansında yayınlanan bir makale ile, Locked Shields tatbikat altyapısından istifade edilerek özgün ve erişime açık bir veriseti üretebilmek için bir öneride bulunulmuştur. Bu öneride tatbikata, diğer takımlardan bağımsız sanal bir mavi takım eklenmesi, bu takımın ağ trafiğinin elde edilmesi, elde edilen bu trafik kullanılarak güvenlik endişeleri barındırmayan yeni ve erişilebilir bir veriseti üretilmesi hedeflenmiştir. Anılan öneri doğrultusunda kurulan çalışma grubu tarafından yapılan çalışmalar neticesinde, tatbikata sanal mavi takım eklenmiş, bu sanal mavi takımın ağ trafiği yakalanmış, veri dönüştürme ve etiketleme işlemleri gerçekleştirilmiş, neticede yaklaşık iki milyonu zararlı olmak üzere toplamda yaklaşık 16 milyon kayıt içeren özgün ve kaliteli bir ağ sızma tespit veriseti üretilmiştir. Üretilen veriseti, Random Forest makine öğrenme yöntemi tabanlı bir modelin eğitimi ve testinde kullanılmış, %98,02 seviyesinde F1 skoru elde edilerek verisetinin IDS geliştirme çalışmalarında kullanılabileceği gösterilmiştir.

Bu bilgiler doğrultusunda tezin ikinci bölümünde derin öğrenmenin matematiksel temelleri ile ilgili bilgi sunulmaktadır.

Üçüncü bölümde derin öğrenme yöntemlerinin ağ sızma tespit sistemlerinde kullanımına yönelik literatür inceleme sonuçları paylaşılmakta, tezde kullanılan modelin geliştirme süreci ve kullanılan platformlar hakkında bilgi verilmekte, model başarısına etki eden hiperparametreler tanıtılmakta, bunlarla ilgili detaylı analizler paylaşılmakta ve hiperparametrelerin model başarısına etkisi değerlendirilmektedir. Bununla birlikte, tez kapsamında geliştirilen modelin geliştirme süreci hakkında bilgi verilmekte, modelin NSL-KDD verisetine uygulanması neticesinde elde edilen sonuçlar paylaşılmakta, ayrıca veriseti üzerinde özellik azaltımı yapılmasının model başarısına etkisi, elde edilen sonuçlar göz önünde bulundurularak değerlendirilmektedir. Bununla birlikte, RNN ve GRU yöntemlerinin çalışma prensiplerine yönelik bilgi sunulmakta, bu yöntemler kullanılarak geliştirilen modeller ile LSTM modelinin kıyaslaması yapılarak elde edilen sonuçlar paylaşılmaktadır.

Dördüncü bölümde, özgün veri görselleştirme işlemleri ve evrişimli sinir ağlarının ağ sızma tespitinde kullanımına ilişkin detaylı bilgi sunulmaktadır. Bu kapsamnda, nümerik verinin imaj verisine dönüştürülmesi, imaj verisinin evrişimli sinir ağı tabanlı modelin eğitimi ve testinde kullanılması, evrişimli sinir ağı modeline düzenlileştirme tekniklerinin entegre edilmesi ve test işlemleri neticesinde elde edilen sonuçların literatürdeki diğer sonuçlarla kıyaslanmasına ilişkin bilgiler paylaşılmaktadır.

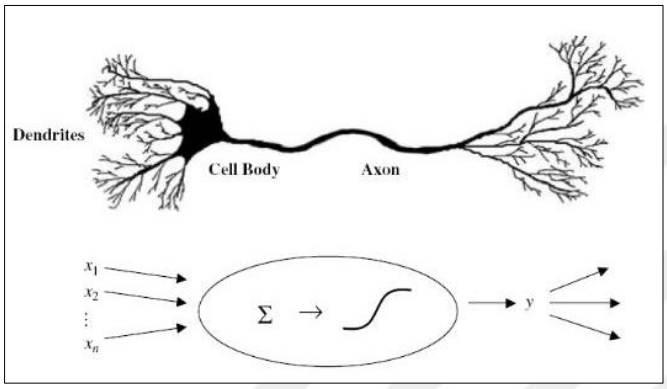
Beşinci bölümde literatürde sıklıkla kullanılan ağ sızma tespit verisetlerine yönelik inceleme sonuçları paylaşılmakta, bu verisetlerinin üretim yöntemleri, özellikleri, artı ve eksi yönleri detaylı bir şekilde anlatılmaktadır. Kaliteli bir ağ sızma tespit verisetinin sahip olması gereken özelliklere ilişkin bilgi sunulmakta, belirli atak kategorilerine karşı model geliştirme çalışmalarında hangi verisetlerinin kullanılabileceğine ilişkin değerlendirme yapılmaktadır. Yapılan bu incelemenin ardından, siber savunma tatbikatları hakkında bilgi paylaşımında bulunulmakta, bu tatbikatların özgün veriseti üretimi için neden oldukça elverişli olduklarına ilişkin değerlendirme sunulmaktadır. Bu değerlendirmenin ardından, özgün veriseti üretim adımları hakkında bilgi verilmekte, kullanılan teknikler detaylı bir şekilde paylaşılmaktadır. Bu bölümün sonunda Locked Shields tatbikat trafiğinden istifade edilerek tasnif dışı ağ sızma tespit veriseti üretimine yönelik yapılan çalışmalar sunulmaktadır.

Altıncı bölümde, tez kapsamında geliştirilen modelin güncel veriseti UNSW-NB15 ile eğitimi ve testi neticesinde elde edilen sonuçlar paylaşılmakta, bu sonuçlar diğer çalışma sonuçları ile kıyaslanarak bir değerlendirme sunulmaktadır. Ayrıca, Random Forest algoritması temelli bir modelin Locked Shields tatbikatından elde edilen özgün veriseti ile eğitilmesi ve testi neticesinde elde edilen sonuçlar paylaşılarak bir değerlendirmede bulunulmaktadır.

Yedinci ve son bölümde tez çalışmasında elde edilen bulgular değerlendirilmekte, bundan sonra yapılabilecek çalışmalar hakkında araştırmacılara önerilerde bulunulmaktadır.

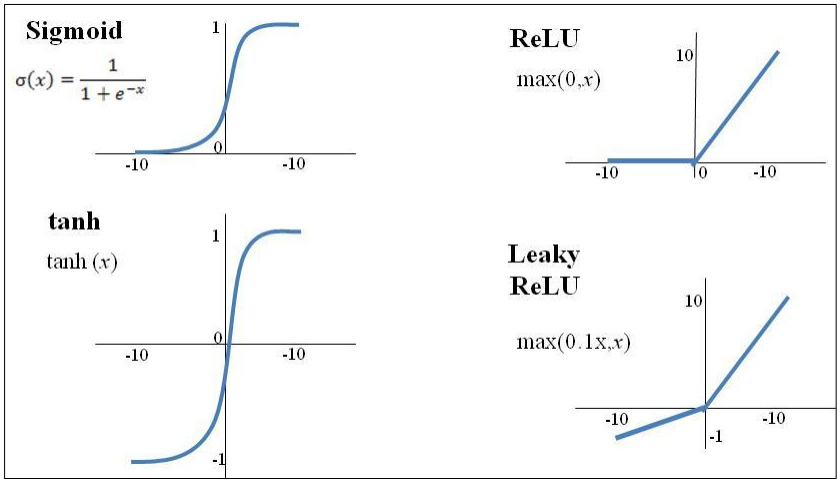
2. TANIMLAR

Birçok araştırmacı ve uzman tarafından yapay zekânın gelişimine önemli katkısı olacağı değerlendirilen derin öğrenme yöntemleri, Yapay Sinir Ağı (YSA) temellidir. YSA'nın geçmişi yıllar öncesine dayanır. McCulloch ve Pitts tarafından 1943 yılında önerilen [21] YSA ile Şekil 2.1'de görüldüğü üzere insan vücudunun sinir sistemi modellenmeye çalışılır. YSA birçok makine öğrenme probleminin çözümünde başarı ile uygulanmıştır [22].



Şekil 2.1. İnsan sinir sisteminde bulunan sinir hücresinin YSA modeli [23].

Sinir hücreleri yani nöronlar işlem birimini oluşturur ve kendi üzerine iletilen girdileri alıp işledikten sonra bir karar vermeye yarar. Nöron üzerine gelen birçok bilgi toplanarak bir değer oluşturulur, bu değer belirlenen eşiğin üzerinde ise nöron aktif hale geçer. Nöron aktivasyonu için çeşitli fonksiyonlar kullanılabilir. Şekil 2.2'de bazı aktivasyon fonksiyonlarının matematiksel grafik gösterimleri sunulmaktadır.

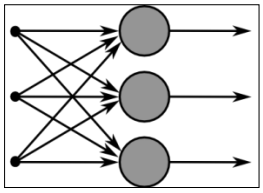


Şekil 2.2. Bazı YSA aktivasyon fonksiyonları.

YSA'lar temel olarak;

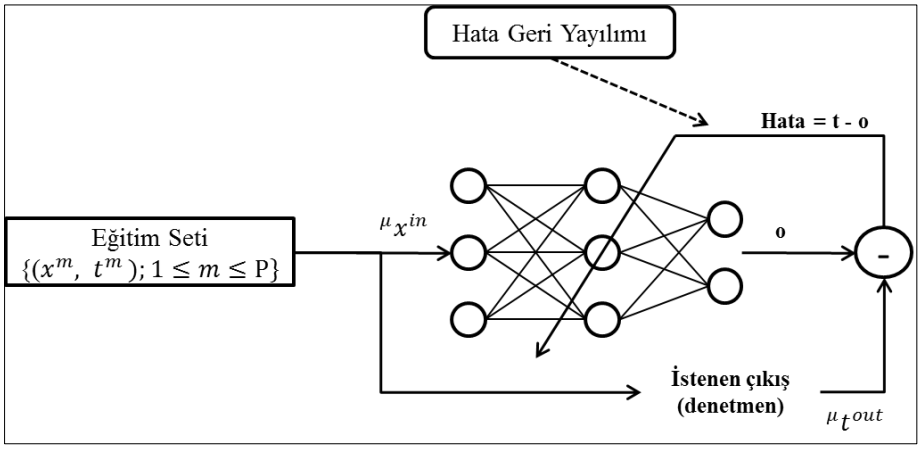
1. Tek katmanlı ileri beslemeli (Single layered feed forward)
2. Çok katmanlı ileri beslemeli (Multi layered feed forward)
3. Geri beslemeli (Recurrent)

şeklinde sınıflandırılır.



Şekil 2.3. Tek katmanlı ileri beslemeli YSA [24].

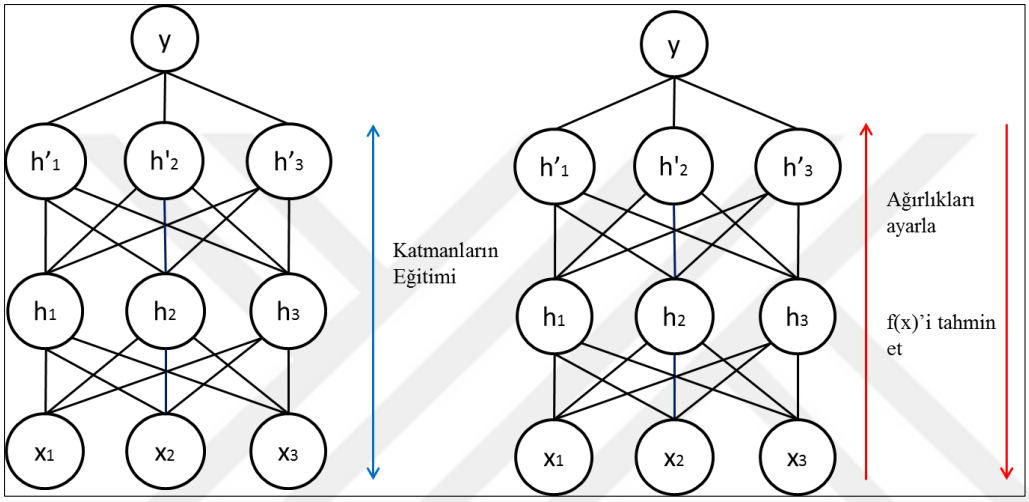
Tek katmanlı YSA perceptron olarak isimlendirilir. Böyle bir YSA'nın aktivasyon fonksiyonu örneğin sigmoid ise YSA sigmoid perceptron olarak adlandırılır. Bir perceptronda çıktı sayısı kadar nöron bulunur. Perceptronun AND, OR ve NOT kapıları için çalıştığı ancak XOR kapısında çalışmadığı, bir başka deyimle lineer sınıflandırma problemlerinin çözümüne imkân sunduğu ancak nonlineer sınıflandırma problemlerinin çözümünde kullanılamadığı görülmektedir. Bu problemin giderilmesi amacıyla YSA'lara gizli katman eklenmesi ve hedeflenen sonucun elde edilebilmesi için ağırlık değerlerinin ayarlanmasına bağlı eğitim çözümleri (backpropagation, forward propagation) kullanılarak sistemin eğitilmesi yoluna gidilir. Şekil 2.4'te çok katmanlı bir yapay sinir ağının denetimli öğrenme (supervised learning) yapısı görülmektedir.



Şekil 2.4. Çok katmanlı YSA'nın denetimli öğrenme yapısı [22].

2.1. Derin Öğrenme

Derin öğrenme algoritmalarının temel özelliği, birden çok gizli katman içeren bir YSA'nın, her katmanının istenilen tahmin değerinin elde edilebilmesi için ayrı ayrı eğitilmesi, daha sonra da geriye yayılma (backpropagation) yöntemi uygulanarak ağırlık değerleri üzerinde hassas ayarlamaların yapılmasıdır [25]. Şekil 2.5'te derin öğrenmenin temel çalışma yapısı görülmektedir.



Şekil 2.5. Derin öğrenme tabanlı sistemlerin temel çalışma yapısı [25].

Derin öğrenme, birçok soyut seviyeden oluşan verinin gösteriminin öğrenilmesini sağlayan birçok işlem seviyesi içeren hesaplama modellerinin çalışmasına imkân sunar ve geniş veri setleri içerisindeki karmaşık yapıyı geriye yayılma yöntemi ile çözümlemeyi sağlar. Geriye yayılma ile makinenin, önceki katmandaki gösterimden yararlanarak her katmandaki iç parametreleri nasıl değiştirmesi gerektiği belirlenir [26]. Derin öğrenme alanındaki önemli isimlerden Yann LeCun derin öğrenmeyi, "parametirize edilmiş fonksiyonel modüllerden oluşan ağlar inşa etme ve bu ağları gradyan tabanlı optimizasyon ile örnekler üzerinden eğitmek" olarak tanımlamaktadır.

Tez hazırlık sürecinde yapılan incelemede, derin öğrenmenin bilişsel nitelikli insan davranışlarını taklit edecek seviyede başarılı yöntemler sunduğu, büyük bir hesaplama gücü ile karmaşık problemlerin çözümüne önemli katkı sağladığı gürülmüştür. Çeşitli derin öğrenme yöntemleri incelenmiş, her yöntemin belirli bir alanda daha etkin olduğu tespit edilmiştir. Bu kapsamda, etiketsiz büyük boyutlu verinin eğiticisiz öğrenme ile eğitilerek

daha düşük boyutlu veri gösterimi sağlayan, bir başka ifadeyle özellik çıkarmaya imkân sunan Restricted Boltzman Machine (RBM) ve Autoencoders (AE); etiketli veri ile eğitilen ve yaygın olarak metin işleme, zaman serisi analizi ve ses tanımada kullanılan Recurrent Neural Networks (RNN), görüntü ve nesne tanımada kullanılan Deep Belief Networks (DBN) ve Convolutional Neural Networks (CNN), sağladığı programlanabilir kısa süreli hafıza özelliği ile uzun metinlerin çevirisinde ve diğer biçok uygulamada etkinlikle kullanılan RNN tabanlı Long Short Term Memory (LSTM) en sık kullanılan derin öğrenme yöntemleri olarak öne çıkmaktadır [27].

Daha önce vurgulandığı gibi derin öğrenme yöntemleri YSA temellidir ve diğer makine öğrenme yöntemleri gibi lineer cebir bilgisi gerektirir. Derin öğrenmede kullanılan verisetleri, üç veya daha büyük boyutlu diziler olan tensörler ile ifade edilir. Yukarıda belirtilen derin öğrenme yöntemlerine hâkim olunması için skaler, vektör, matris ve tensör gibi terimler ile bunlar üzerinde yapılan bazı matematiksel hesaplamalar hakkında bilgi sahibi olmak gerekir.

Skaler: Tek bir sayıdır. Derin öğrenme açısından sadece bir eleman taşıyan sıfır boyutlu tensörlere skaler denir. Integer, float32 ve float64 gibi sayı formatları ile ifade edilir.

$x_1 = 5$

Vektör: Bir sayı dizisidir. Bu sayı dizisi içindeki her bir sayının indisi vardır ve sayılara bu indisler vasıtasıyla erişilebilir. Derin öğrenme açısından tek eksene sahip tek boyutlu tensörlere vektör denir. Vektörün içerisinde çeşitli tiplerde sayılar saklanabilir. Örneğin;

A = ([5, 6, 2, 4, 8, 3])

A beş boyutlu bir vektördür. Bir vektör için boyut, aynı eksen üzerindeki eleman sayısını göstermektedir.

Matris: İki boyutlu sayı dizileridir ve her bir elemanı, bir yerine iki indis ile ifade edilir. Örneğin $\boldsymbol{A}_{1,3}$ ifadesi, $\boldsymbol{A}$ matrisinin birinci satır üçüncü sütununda yer alan elemanı göstermek için kullanılır. Derin öğrenme açısından iki eksene sahip iki boyutlu tensörlere matris denir. Matris içerisinde de istenilen tipte sayılar saklanabilir.

A = ([[2, 3, 8, 7, 4],
[4, 2, 5, 3, 0],
[8, 1, 6, 9, 2]])

A, 3x5 boyutlu bir matristir. Bazı durumlarda birden fazla harften oluşan matrisli ifadelerin indekslenmesi gerekmektedir. Bu durumlarda $A_{i,j}$ şeklinde indisleme yapılır. Bu ifade $\boldsymbol{A}$ matrisinin i. satır ve j. sütununda yer alan elemanı ifade etmektedir.

Tensör: Matrisler iki eksene sahip dizilerdir ve bazı durumlarda ikiden fazla ekseni olan dizilere ihtiyaç duyulur. Bu durumda tensör, "düzenli bir ızgara üzerine yerleştirilmiş ve değişken sayıda ekseni bulunan dizi" olarak tanımlanabilir [28].

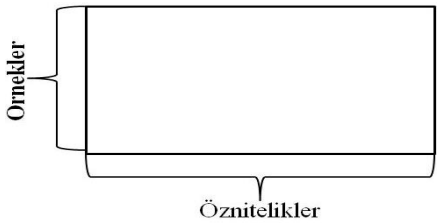
A = ([[[2, 4, 3, 5],
[3, 9, 1, 0],
[1, 3, 5, 4]],

[[4, 7, 3, 9],
[8, 5, 2, 1],
[6, 1, 4, 2]],

[[2, 5, 6, 3],
[1, 5, 6, 2],
[4, 9, 0, 3]]])

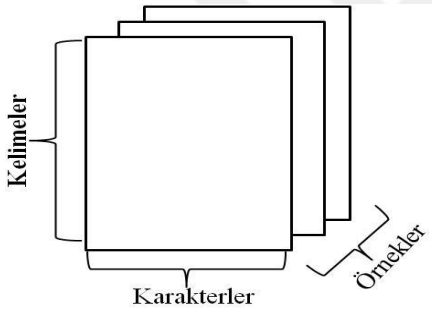
A, 3 boyutlu bir tensördür ve A tensörünün (i, j, k) koordinatlarındaki bir elemanı, $A_{i,j,k}$ şeklinde tanımlanır. Yukarıdaki tensörün eksen sayısı 3'tür ve şekli (3, 3, 4) olarak ifade edilir. Derin öğrenme uygulamaları genel olarak çok hesaplamalı tensör işlemlerinin yapıldığı matematiksel işlemlerdir. Tensör işlemleri çeşitli verisetleri üzerinde gerçekleştirilir. Vektör serisi, zaman serisi ya da ardışık veri, görüntüler, video başlıca veri tensörü kategorileridir [29].

Vektör serisi bir matris, bir başka ifadeyle 2 boyutlu bir tensördür ve ilk eksen örnek, ikinci eksen ise öznitelik eksenidir. Örneğin IDS verisetinde her satırda çeşitli öznitelik değerlerine sahip birçok kayıt vardır. Öznitelik sayısı 41, kayıt sayısı 125000 olması durumunda tüm veriseti (125000, 41) şeklinde olacaktır. Özniteliklere bağlı sınıflandırma değerlerinin pozitif (1) veya negatif (0) olması durumunda 125000 kayıt için (125000, 1) şeklinde bir vektörün sınıflandırma değerlerini ifade ettiği söylenebilir.



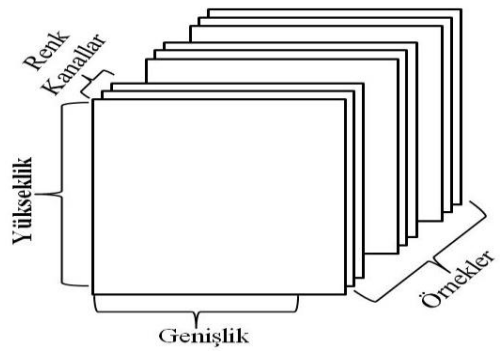
Şekil 2.6. Vektör serisi veriseti tensör yapısı [28].

Zaman serisi ve ardışık verinin gösterimi için 3 boyutlu bir tensör kullanılır. İlk eksen öznitelik bilgilerini, ikinci eksen zaman veya ardışık veriyi, üçüncü eksen ise örnekleri göstermektedir. Örneğin her biri 128 farklı karakterden biri olan 280 karakterlik bir tweet dizisinde her tweet (280, 128) şeklinde iki boyutlu bir matris olacağından 100000 tweet bulunan veriseti (100000, 280, 128) şeklinde bir tensör olacaktır.



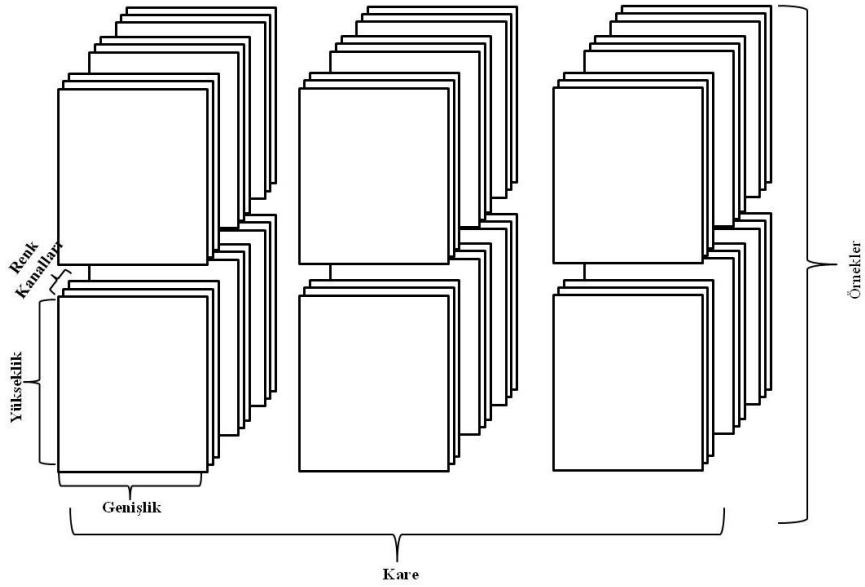
Şekil 2.7. Ardışık veri içeren verisetinin tensör yapısı [28].

Görüntü verileri genelde genişlik, yükseklik ve renk kanallarından oluşan üç boyutlu tensörler ile ifade edilir. Örnekler ile görüntü verisi gösterimi dört boyutlu bir tensör haline gelir. Örneğin 3 renk kanalı olan 256 x 256 boyutunda 128 örnek içeren bir verisetinin gösterimi (128, 256, 256, 3) şeklinde dört boyutlu bir tensör ile gösterilir.



Şekil 2.8. Görüntü verisetine ait dört boyutlu tensör yapısı [28].

Her görüntü yükseklik, genişlik ve kanallardan oluşan üç boyutlu bir tensör ile ifade edilmektedir ve bir video karesinin gösterimi için bu tensöre kare boyutu eklenerek dört boyutlu bir tensör elde edilir. Bir video yığınının gösterimi için ise bu dört boyutlu tensöre örnek boyutu da eklenerek videonun beş boyutlu bir tensör ile gösterimi sağlanır. Örneğin 60 saniyelik, 144 X 256 boyutunda ve 3 renk kanalından oluşan görüntüleri içeren 4 kare ile örneklenmiş (toplam 240 kare) bir videonun gösterimi için (4, 240, 144, 256, 3) şeklinde bir tensöre ihtiyaç duyulmaktadır.



Şekil 2.9. Video verisetine ait beş boyutlu tensör yapısı.

Derin öğrenme uygulamalarında eğitim, doğrulama ve test için kullanılan her veriseti bir matris veya tensör ile ifade edilir. Bu tensörler üzerinde lineer matematik, sayısal hesaplama ile olasılık ve bilgi kuramına dayalı işlemler gerçekleştirilerek öğrenme ve tahmin işlemleri yapılır. Sinir ağlarının çeşitli şekillerde kullanılması ile elde edilen derin öğrenme yöntemleri, bir verisetine yönelik birbirini takip eden basit adımlar ile çok boyutlu uzayda karmaşık dönüşümlerin gerçekleştirilmesini sağlar. “Derin öğrenme, her katmanda verinin sahip olduğu karmaşıklığı biraz daha azaltarak, birbirini takip eden katmanlarda karmaşık verinin kolay takip edilebilir görünümlerini elde edebilmektedir.” [29] Tensör elemanlarına yönelik işlemler, iki farklı şekle sahip tensör olması durumunda gerçekleştirilen yayma işlemleri, tensörler arası iç çarpım işlemi, tensör satır ve sütun ayarlaması yaparak istenen şekle gelecek biçimde tensör şekil değiştirme gibi tensörlerin tüm elemanlarına uygulanan işlemler ile gerçekleştirilebilmektedir.

2.2. Gradyan Temelli Eniyileme

Derin öğrenme uygulamaları, bir en iyileme probleminin çözümünü içerir ve amaç fonksiyonuna ait değişkenin küçük adımlar ile değiştirilmesi neticesinde fonksiyonun en iyi değerinin elde edilmesini hedefler. En iyileme problemlerinin genelde en küçültme olarak karşımıza çıktığı görülmektedir.

$y = f(x)$ şeklinde bir fonksiyon olması durumunda, bu fonksiyonun türevi olan $f'(x)$, fonksiyonun x noktasındaki eğimini ifade eder ve çıkış değerinde yapmak istediğimiz değişiklik için girdide yapılacak değişikliği $f(x+\epsilon) \approx f(x) + \epsilon f'(x)$ ifade eder. $f(x)$ fonksiyonunun değeri, x'in türevin tersi yönde küçük adımlarla ilerletilmesi ile küçültülebilir. Bu işleme gradyan inişi denir. $f'(x) = 0$ olması durumunda eğimin sıfır olduğu anlaşılır ve hangi yönde ilerlenileceği ile ilgili bir bilgi bulunmaz. Bu noktalar durağan ve kritik noktalardır. Yerel minimum, yerel maksimum, genel minimum, genel maksimum ve eyer noktaları kritik noktalardır. "Derin öğrenme kapsamında, genellikle birçok yerel minimum içerebilecek düz bölgelerle çevrili birçok eyer noktasına sahip, eniyilemeye uygun olmayan fonksiyonlar eniyilenir." Bu nendele çoğu durumda en küçük olmasa da küçük bir f değeri yeterli olmaktadır. Şekil 2.10'da bir fonksiyona ait kritik noktalar görülmektedir [28, 29].

Şekil 2.10. Fonksiyon türev değerinin sıfır olduğu kritik noktalar.

Amaç bir f fonksiyonunu en hızlı şekilde en küçültmek olduğundan, fonksiyonun en hızlı azaldığı yönün bulunması gerekir. Bu işlem yönlü türev kullanılarak gerçekleştirilir. Buna en dik iniş yöntemi ya da gradyan inişi (gradient descent) adı verilir ve her adımda

$x' = x - \epsilon \nabla_x f(x)$ noktasını önermektedir. Burada ϵ öğrenme oranıdır ve pozitif skaler bir değerdir [28, 29].

2.3. Geriye Yayılım Algoritması

Girdi vektörü ağ üzerinde yer alan düğümler üzerinde işlenerek bir çıktı üretecek şekilde ileri doğru bir bilgi akışı gerçekleşir. Buna ileri yayılım denir ve eğitim esnasında bir maliyet değeri üretilene kadar ileri yayılım işlemi devam eder. Eğitim işleminin amacı üretilen maliyet değeri ile istenen maliyet değeri arasında yakınsama sağlamak olduğundan, ağ içerisindeki ağırlık değerlerinin güncellenmesi gerekir ve elde edilen maliyetten geriye doğru bir bilgi akışı sağlanarak gradyan hesaplama işlemi gerçekleştirilir. Bu nedenle geriye yayılım gradyan hesaplama metoduna verilen addır. Bu gradyan değeri bir başka algoritma tarafından kullanılarak öğrenme işlemi gerçekleştirilir.

Geri yayılım, bazı işlemlerin özel bir sıra ile kullanımı neticesinde zincir türev hesaplayan bir algoritmadır. "Uygulamada sinir ağları birbirini takip eden zincirleme tensör işlemleri bulundurur ve her birinin bilinen basit bir türevi vardır." [28]

x bir gerçek sayı, f ve g ise iki fonksiyon olsun. $y = g(x)$ ve $z = f(g(x)) = f(y)$ olması durumunda zincir kuralına göre;

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx} \tag{2.1}$$

olacaktır. $x \in R^m$, $y \in R^n$ olsun ve g R^m'den R^n'e, f ise R^n'den R'ye eşleştirme yapsın. Eğer $y = g(x)$ ve $z = f(y)$ olursa bu durumda;

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_i}\frac{\partial y_i}{\partial x_i} \tag{2.2}$$

olur. Bu ifade vektör olarak aşağıdaki şekilde yazılır.

$$\nabla_x z = \left(\frac{\partial y}{\partial x}\right)^T \nabla_y z \tag{2.3}$$

Burada $\frac{\partial y}{\partial x}$, g'nin n x m boyutlu matristir. Bu ifade, x değişkeninin gradyanının, $\frac{\partial y}{\partial x}$ matrisi ile $\nabla_y z$ gradyan değerinin çarpımı neticesinde bulunabileceğini göstermektedir. Geri yayılım algoritması da graf üzerindeki her operasyon için buna benzeyen matris-gradyan çarpımından oluşmaktadır [29].

3. UYGULAMA MODELİ GELİŞTİRİLMESİ

Derin öğrenmenin matematiksel temellerine ilişkin incelemenin ardından, tezin bu bölümünde derin öğrenme yöntemlerinin ağ sızma tespit sistemlerinde kullanımına ilişkin bilgi sunulmaktadır. Derin öğrenme tabanlı ağ sızma tespit sistemi geliştirme çalışmaları nispeten yeni bir konudur. Yapılan literatür incelemesinde, Türkiye'de derin öğrenme uygulamalarının daha çok görüntü tanıma üzerine yoğunlaştığı; uydu görüntülerinin tanınması [30], sürücüsüz araçlarda kullanılmak üzere Convolutional Neural Network (CNN) ile yol çizgilerinin tespiti [31], CNN kullanılarak uydu görüntülerinden bina tespiti [32], CNN kullanılarak video kavram sınıflandırması [33], biyomedikal veri sınıflandırmasında derin öğrenme yöntemlerinin kullanımı [34], histopatoloji imgelerinden göğüs kanseri teşhisi [35], ultrason görüntülerinden yararlanılarak metastasis teşhisi [36], fonksiyonel Manyetik Rezonans Görüntüleme (fMRG) ile beyin okuma [37], videolarda derin öğrenme tabanlı derinlik kestirimi [38], derin öğrenmeye dayalı yüz ve vücut biyometrilerinin tümleştirilmesi [39], ses tanıma [40] gibi çalışmaların gerçekleştirildiği görülmüştür. Yapılan incelemede bilgisayar ağlarında anormallik tespitine yönelik öne çıkan dört çalışma tespit edilmiştir. Bu çalışmaların ilkinde [41], otomatik kodlayıcı tabanlı deterministik otomatik kodlayıcı ve gürültü giderici otomatik kodlayıcı modelleri önerilmekte, otomatik kodlayıcıların anomali tespit performanslarının arttırılması için stokastik bir anomali eşik değeri tespit yöntemi önerilmektedir. Çalışma kapsamında geliştirilen modellerin NSL-KDD veriseti üzerinde testine yönelik sonuçlar paylaşılmaktadır. Diğer çalışmada ise [42], bulut ağ üzerinde yapılan ağ sızma girişimlerinin DNN, RNN-LSTM ve CNN derin öğrenme metodları ile tespitine yönelik kapsamlı bir çalışma sunularak, yöntemlerin farklı verisetleri üzerinde uygulanması neticesinde elde edilen sonuçlar paylaşılmaktadır.Bir başka çalışmada [43], UNSW-NB15 ve CICIDS 2017 gibi güncel verisetleri kullanılarak Deep Feed-ForwardNeural Network, Random Forest ve Gradient Boosting Tree derin öğrenme yöntemleri ile ağ sızma tespit sistemi geliştirildiği görülmektedir. Son çalışmada [44], gözetimli ve gözetimsiz derin öğrenme algoritmalarının sıfırıncı gün saldırılarını tespit etmedeki etkinliğinin analiz edildiği belirtilmekte, analizlerin farklı derin öğrenme modelleri üzerinde yapıldığı ve algoritma performanslarının birbirleri ile kıyaslandığı ifade edilmektedir.Bununla birlikte uluslararası alanda yapılan çalışmalar incelenmiş ve önemli olarak değerlendirilenlere yönelik sonuçlar aşağıda paylaşılmıştır.

3.1. Literatür İncelemesi

İncelenen ilk çalışmada [45], sızma tespitinde yaşanan gereksiz bilgi, büyük boyutlu veri, uzun zaman alan eğitim süreci, lokal optimuma düşme gibi problemlere odaklanıldığı ifade edilerek Deep Believe Network (DBN) ve Probabilistic Neural Network (PNN) kullanan bir metot önerilmektedir. Çalışmada öncelikle DBN kullanılarak veri boyutu azaltılmakta, daha sonra en iyi öğrenme performansının elde edilmesi için her seviyedeki gizli katmanda bulunan düğüm sayısı optimizasyonu için swarm optimizasyon algoritması kullanılmaktadır. Daha sonra PNN ile düşük boyutlu veri sınıflandırması yapılmaktadır. Son aşamada önerilen yöntem KDD CUP 1999 veri seti üzerinde test edilmektedir. Sonuçta önerilen yöntemin geleneksel PNN, PCA-PNN ve optimize edilmemiş DBN-PNN yöntemlerine göre daha iyi sonuç verdiği ifade edilmektedir. Çalışmaya yönelik yapılan değerlendirmede; saldırı tespit sistemlerinde kullanılan veri seti boyutunun büyük olması ve yöntemin canlı ortamda kullanılacağı değerlendirildiğinde veri boyutunun daha da artacağı, veri etiketlenme imkânı olmamasına bağlı olarak sistemin hızlı bir şekilde eğitilebilmesi için eğiticisiz öğrenme ve özellik azaltımına ihtiyaç duyulduğu, bununla birlikte en iyi sonucun elde edilmesi için gizli katmanlarda ihtiyaç duyulan düğüm sayısının optimize edilmesi gerektiği, tüm işlemler neticesinde düzgün bir sınıflandırma yapılması için uygun bir sınıflandırma yöntemi kullanılması gerektiği görülmektedir. Çalışma sonuçları Çizelge 3.1'de yer almaktadır.

Çizelge 3.1. Optimize Edilmiş DBN-PNN sonuçlarının diğer yöntemlerle kıyaslanması [45].

Metotlar	Deneysel Sonuçlar			
	Çalışma Süresi (sn)	Tespit Kesinliği (%)	Tespit Oranı (%)	Yanlış Alarm Oranı (%)
Optimize edilmemiş DBN-PNN	5,71	99,31	91,75	0,375
Optimize edilmiş DBN-PNN	5,48	99,14	93,25	0,615
PCA-PNN	6,16	98,28	89	1,33
PNN	35,38	99,04	89,25	0,55

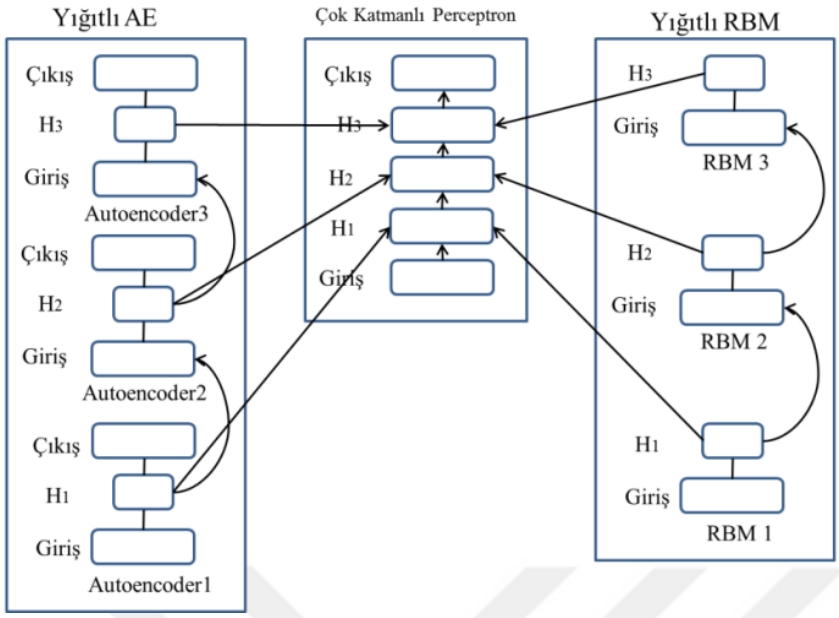
Bir diğer çalışmada [46], bilgisayar ağlarındaki anomaliler, nokta anomalisi, kapsam anomalisi ve müşterek anomali olmak üzere üçe ayrılmaktadır. Bu anomali çeşitlerinin ağ güvenliği kapsamında ele alınan DoS, Probe, U2R (User to Root), R2L (Remote to Local)

atakları ile ilişkili olduğunu ifade edilmektedir. DoS ataklarının müşterek anomali ile, Probe ataklarının kapsam anomalisi ile, U2R ve R2L ataklarının ise nokta anomalisi ile ilişkilendirilebileceği, bunların dışında ağda, daha önceden tanımlanmamış yeni tip atakların da olabileceği, bunların ancak ağ trafiğindeki anormalliklerin anlaşılması ile tespit edilebileceği belirtilmektedir.

Çalışmada, anomali tabanlı saldırı tespit sistemlerinin bir model oluşturmak üzere normal ağ trafiğinde eğitilmesi gerektiği, oluşturulan modelin yeni olayların sınıflandırılması veya anormal ağ trafiği tespitinde kullanılabileceği ifade edilmektedir. Bahse konu modelin en önemli özelliğinin dinamik ağ ortamına adapte olması ve genel bir çözüm sunması olduğu ifade edilmektedir. Diğer bir önemli hususun, eğitim faaliyetlerinde kullanılan etiketli verinin sisteme sağlanması olduğu vurgulanmakta, ağ üzerindeki normal davranışların etiketlenebileceği, ancak saldırıların etiketlenmesinin mümkün olmadığı, bu nedenle yarı eğiticili veya eğiticisiz öğrenme tekniklerinin kullanılması gerektiği vurgulanmaktadır.

Çalışmada anomali tabanlı ağ saldırı tespit sistemi için bir DBN yapısı oluşturulduğu, bunun ön eğitimi için Stacked RBM ve Stacked Autoencoder yöntemleri kullanıldığı ifade edilmektedir. Stacked AE yönteminin ön eğitimde, Stacked RBM kullanımına oranla daha iyi sonuç verdiği, aynı zamanda AE kullanımının daha uygulanabilir olduğu, buna bağlı olarak kullanılacak parametrelerin daha kolay tespit edildiği belirtilmektedir. RBM'in ise daha karmaşık bir hesaplama işlemi gerektirdiği vurgulanmaktadır. Ön eğitimin tamamlanmasını takiben özellik azaltımı gerçekleştirildiği, gereksiz bilginin filtrelendiği, çok katmanlı yapay sinir ağı yapısı kullanılarak eğiticili hassas ayarlama işleminin geri yayılım ile sağlandığı ve böylece en iyi ağırlık değerlerinin elde edildiği ifade edilmektedir.

Çalışmada KDDCup99 veri setinin kullanıldığı, bu veri seti içerisinde 41 özelliğe sahip yaklaşık 4.900.000 kayıt olduğu, simüle edilen saldırıların DoS, Probe, U2R ve R2L ataklarından biri olduğu ifade edilmektedir. Veri setinin eğitim, doğrulama ve test olacak şekilde üç bölüme ayrıldığı vurgulanmıştır.

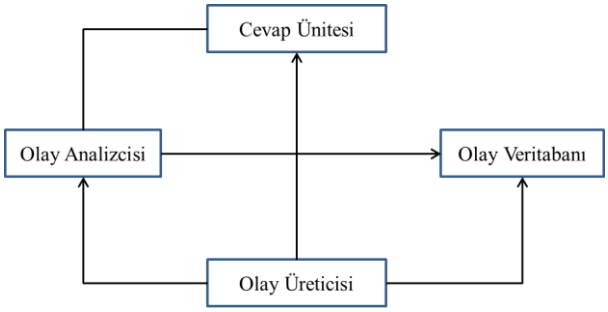


Şekil 3.1. Van ve arkadaşları tarafından önerilen STS'nin yapısı [46].

Sonuç olarak çalışmada, Şekil 11'de verilen mimari kullanılarak sadece atakların tespit edilmediği, aynı zamanda Normal, DoS, Probe, U2R ve R2L olacak şekilde beş sınıfa ayrıldığı, böylece atak tipi bazında sınıflandırma ve tespit yapıldığı ifade edilmektedir.

Bir başka çalışmada [47], relevance derin öğrenme tekniği kullanılarak kesinlik değerinin arttırılmasını sağlayan bir yöntem önerilmektedir. Çalışmada öğrenme algoritması olarak RBM kullanıldığı ifade edilmekte, önerilen metodun bilinmeyen sızma ve ataklara karşı yüksek seviyede ortalama tespit oranı ve yine yüksek seviyede ortalama hata tespit oranı sağladığı ifade edilmektedir.

Çalışmada saldırı tespit sistemleri, yazılım ve donanımın birlikte çalıştığı yapılar olarak tanımlanmaktadır. Bununla birlikte saldırı tespit sistemlerinin çalışma prensibi; bilginin birçok yoldan toplanması ve analizi, saldırı aktivitelerinin üretimi için temel karakteristiklerin çıkarımı, tespit edilen anormal davranışa karşı ağ bağlantısının kesilmesi, olayların kaydedilmesi ve uyarı mekanizmalarının devreye girmesini içeren bir dizi otomatik yanıtın üretilmesi olarak ifade edilmektedir.



Şekil 3.2. Li Jing ve Wang Bin'in sunduğu saldırı tespit modeli [47].

Önerilen relevance derin öğrenme temelli modelin iki aşamadan oluştuğu, ilk aşamada saldırı verisinin ilişkilendirildiği, ikinci aşamada ise saldırı verisi öğreniminin gerçekleştiği vurgulanmaktadır. İlk aşamada saldırı verisi sınıflandırması ile işlem karmaşıklığının azaltıldığı, tespit veri popülasyonunun crossover mutasyon ve selection işlemleri ile optmize edildiği vurgulanmaktadır. Yazarlar uygulamalarının saldırı tespitinin normal trafik tespitine oranının 10:1 olduğunu, çok büyük bant genişliğine ve hıza sahip bir ağda ortalama tespit oranının %50'den fazla, hata oranının ise yaklaşık %1,5 olduğunu ifade etmektedirler.

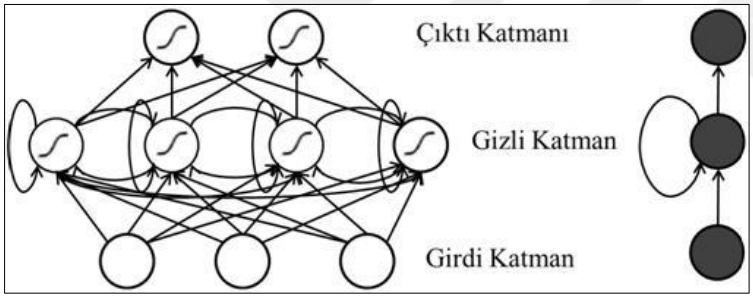
Bir diğer çalışmada [48], RNN tabanlı bir saldırı tespit sistemi (RNN-IDS) önerilmektedir. Önerilen modelin performans gösterimi için ikili ve çoklu sınıflandırma yapılmakta, bu sınıflandırma sonuçları YSA, random forest, SVM ve daha önce aynı veri seti üzerinde diğer araştırmacılar tarafından uygulanan makine öğrenme yöntem sonuçları ile kıyaslanmaktadır. Önerilen modelin yüksek kesinlik değerine sahip olduğu, diğer geleneksel makine öğrenme yöntemlerine oranla hem ikili hem de çoklu sınıflandırma için yüksek performans sağladığı, bu sayede saldırı tespiti için yeni bir araştırma metodu sunduğu belirtilmektedir.

Saldırı tespit sistemlerinin aslında ikili ve çoklu sınıflandırma problemine odaklandığı, daha önceki çalışmalarda olduğu gibi çoklu sınıflandırma işleminin dört çeşit saldırının (DoS, Probe, U2R ve R2L) belirlenmesi için gerçekleştirildiği ifade edilmektedir. Saldırı tespitinde ana motivasyonun sınıflandırıcının kesinlik değerinin arttırılması olduğu vurgulanmaktadır.

Derin öğrenme yöntemlerinin Profesör Hinton tarafından 2006 yılında önerilmesini takiben özellikle görüntü tanıma, ses tanıma, hareket tespiti gibi alanlarda sıklıkla ve başarı ile kullanıldığı, RNN'in ise son yıllarda bilgisayarla görüntüleme, doğal dil işleme (NLP), semantik anlamlandırma, ses tanıma, dil modelleme, çeviri, resim tanılama ve insan aktivitelerinin tanımlanması alanlarında başarı ile kullanıldığı belirtilmektedir.

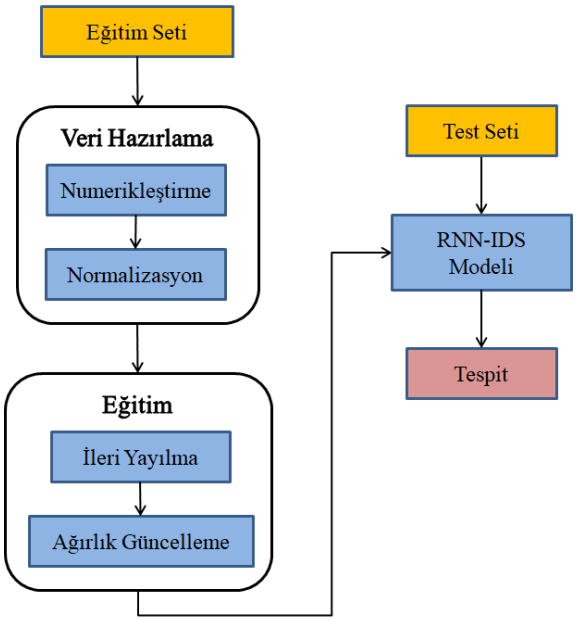
Çalışmada, veriden daha iyi temsili veri elde edilerek daha iyi modellerin oluşturulması konusunda derin öğrenmenin sağladığı avantajlardan yararlanarak, ağlarda saldırı tespitine yönelik RNN-IDS yöntemi önerilmektedir. Modelin hem ikili hem de çoklu sınıflandırma için performansının değerlendirildiği, kesinlik değerine etki eden nöron sayısı ve öğrenme oranlarının iyileştirilmesine yönelik çalışma yapıldığı ifade edilmektedir. Ayrıca geleneksel makine öğrenme yöntemlerinin (naive bayes, random forest, multi-layer perceptron (MLP), SVM vb.) NSL-KDD veri setine uygulanması ile elde edilen sonuçlar çalışma sonuçları ile kıyaslanmaktadır. Önerilen RNN-IDS çözümünün diğer geleneksel makine öğrenme yöntemlerine oranla daha başarılı olduğu belirtilmektedir.

Kullanılan derin öğrenme algoritması RNN hakkında bilgi verilmekte, bu yöntemde girdi, gizli ve çıktı katmanlarının bulunduğu, en önemli işlemlerin gizli katmanda yapıldığı vurgulanmaktadır. Şekil 3.3'te kullanılan RNN mimarisi görülmektedir.



Şekil 3.3. RNN mimarisi [48].

RNN mimarisi incelendiğinde, girdi katmanından gizli katmana tek yönlü bir besleme olduğu, gizli katmanda bulunan nöronların birbirleri çift yönlü, çıktı katmanındaki nöronlarla tek yönlü bağlantıları olduğu görülmektedir. Önerilen modelin çalışma yapısı Şekil 3.4'te sunulmuştur.



Şekil 3.4. RNN-IDS modelinin çalışma yapısı [48].

RNN-IDS modelinin çalışma yapısı incelendiğinde; veri setinin eğitim ve test olmak üzere iki bölüme ayrıldığı, eğitim işleminden önce nümerikleştirme ve normalizasyon işlemlerinin yapıldığı, eğitim aşamasında öncelikle ileri yayılım (forward propagation), daha sonra ağırlık güncelleme işleminin yapıldığı, eğitimi tamamlanan RNN-IDS modelinin test verisi ile test edildiği görülmektedir.

Çalışmada, 2009 yılında üretilen NSL-KDD veri setinin birçok saldırı tespit sistemi geliştirme çalışmasında kullanıldığı, bu veri setinin KDD Cup 99 veri setindeki gereksiz veri problemine çözüm getirdiği, böylece eğitim ve test verisetlerindeki veri miktarlarının daha uygulanabilir hale getirildiği ifade edilmektedir.

Çalışmada kullanılan nümerikleştirme ve normalizasyon metotları açıklanmakta, kullanılan RNN yöntemine yönelik metodolojiden bahsedilmektedir. Aktivasyon fonksiyonu olarak sigmoid ve SoftMax fonksiyonlarının kullanıldığı ifade edilmekte, kullanılan ileri yayılım ve ağırlık güncelleme algoritmaları sunulmaktadır. Modelin testi neticesinde elde edilen ikili ve çoklu sınıflandırma sonuçları Çizelge 3.2'de paylaşılmaktadır. Sunulan RNN-IDS yönteminin özellikle NSL-KDD veri seti kullanılarak yapılan çoklu sınıflandırmada geleneksel makine öğrenme yöntemlerine kıyasla düşük false pozitif içerecek şekilde daha yüksek kesinlik ve tespit oranı sağladığı ifade edilmektedir.

Çizelge 3.2. Çoklu ve ikili sınıflandırma sonuçları[48].

(a) Çoklu sınıflandırma sonuçları.

Gerçek Sınıf \ Tahmin Edilen Sınıf	Normal	DoS	R2L	U2R	Probe
Normal	9377	88	2	6	238
DoS	1011	6227	125	0	95
R2L	2058	0	680	6	10
U2R	149	0	11	23	17
Probe	231	166	5	0	2019

(b) İkili sınıflandırma sonuçları.

Gerçek Sınıf \ Tahmin Edilen Sınıf	Anomali	Normal
Anomali	9362	3471
Normal	298	9413

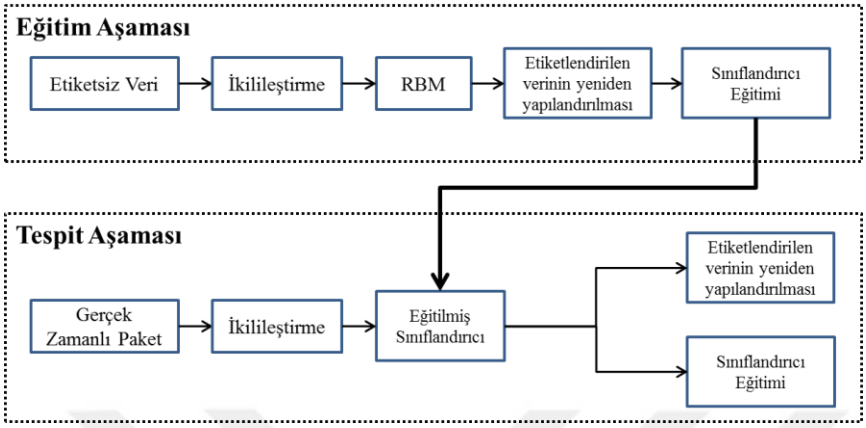
Çalışma incelendiğinde, KDD Cup 99 veri seti yerine NSL-KDD veri setinin kullanıldığı, veri seti yapısının net bir şekilde gösterildiği, veri seti üzerinde yapılan ön işlemlerden daha açık şekilde bahsedildiği, bir derin öğrenme yönteminin tam anlamı ile sınıflandırma işleminde kullanıldığı görülmüştür.

Bir başka çalışmada [49], saldırı tespit sistemlerindeki kesinlik değerinin arttırılması için sadece sınıflandırıcı performansının değil, aynı zamanda eğitim için kullanılan veri içerisindeki gürültü ve aykırılıkların yönetiminin de önemli olduğu vurgulanmakta, RBM'in sınıf etiketleri olmayan verinin eğiticisiz eğitiminde kullanılan bir yöntem olduğu ifade edilmektedir. RBM'in olasılıksal bir üretici model olduğu, eğitilen olasılığına bağlı olarak üretilen yeni veriyi içerdiği vurgulanmaktadır. RBM vasıtasıyla üretilen yeni verinin gürültü ve aykırılıklardan arındığı, temizlenmiş verinin ağ saldırı tespit sistemine girdi olarak verilmesi durumunda gürültü ve aykırılıkları içeren verinin eğitime olan negatif etkisinin elimine edildiği ifade edilmektedir.

Çalışmada RBM metodu kullanılarak KDD Cup 99 veri setindeki gürültü ve aykırılıkların temizlendiği ve yeni veri seti üretildiği, daha sonra gürültü ve aykırılıkları içeren eski hali ile yeni halinin kıyaslandığı, neticede yeni verinin uygulanması ile performans artışının sağlandığı belirtilmektedir.

Önerilen saldırı tespit sisteminin çalışma yapısı Şekil 3.5'te sunulmuştur. Yapı incelendiğinde, etiketsiz verinin öncelikle ikilileştirmeye (binarization) tabi tutulduğu, RBM vasıtasıyla gürültü ve aykırılıklardan arındırıldığı, yeniden yapılandırılan verinin etiketlendirildiği ve eğitim sınıflandırıcısına girdi olarak verildiği, eğitim aşamasının

tamamlanmasını takiben üretilen modelin gerçek zamanlı ağ paketlerindeki saldırıların tespitinde kullanıldığı görülmektedir.



Şekil 3.5. Seo ve arkadaşları tarafından önerilen saldırı tespit sistemi [49].

Çalışma incelendiğinde, saldırı tespit sistemlerindeki sınıflandırma probleminden ziyade verinin sınıflandırıcı tarafından daha işlenebilir hale dönüştürülmesi, bu sayede tespit oranının arttırılmasına odaklanıldığı görülmektedir. Ayrıca etiketsiz verinin eğiticisiz eğitimine imkân sunan RBM yöntemi ile verideki gürültü ve aykırılıklar giderilerek canlı sistemlerdeki büyük boyutlu veri daha işlenebilir hale getirilip günümüz saldırılarının çeşitlenmesine bağlı yanlış pozitif oranının azaltılabileceği hususu çalışmanın önemli özelliği olarak görülmektedir.

Başka bir çalışmada [50], LSTM-RNN tabanlı çok kanallı saldırı tespit sisteminden bahsedilmektedir. İncelenen bu çalışmada konu ile ilgili bir literatür çalışması da yer almaktadır. Bu literatür çalışmasında, bilinen makine öğrenme yöntemlerinin, yapay sinir ağlarının ve derin öğrenme yöntemlerinin kullanıldığı birçok çalışmadan, gerçek zamanlı IDS çözümlerinden, endüstriyel kontrol sistemleri ve bulut bilişim alanlarında yaşanan saldırılarda derin öğrenme yöntemlerinin kullanıldığı uygulamalardan bahsedilmektedir.

Çalışmada sunulan çözümün en ilginç yanının çok kanallı eğitim bölümü olduğu düşünülmektedir. Eğitim aşamasında öncelikle veri hazırlama, daha sonra çoklu özellik azaltımı (feature extraction), son olarak ise çok kanallı eğitim işleminin yapıldığı; test aşamasında ise veri hazırlama, çoklu özellik azaltımı ve saldırı tespit adımlarının gerçekleştirildiği görülmektedir. Yapılan incelemede oluşturulan kanalların, tüm tipler (All

Types), temel ve içerik-tabanlı tip (Basic and Content-based Type) ile temel ve trafik-tabanlı tip (Basic and Traffic-based Type) olduğu görülmüştür. Çalışmada NSL-KDD veritabanı kullanılmış, bu kapsamda kullanılan eğitim ve test kayıt miktarları atak tipi bazında tabloya yansıtılmıştır. Yapılan tüm çalışmaların TensorFlow ortamında, Intel(R) Core(TM) i5-7200U CPU 2.50GHz ve NVIDIA GeForce 920MX GPU donanım özelliklerine sahip platform üzerinde gerçekleştirildiği vurgulanmaktadır.

Çizelge 3.3. Sonuçlar tablosu [50].

Metot	Tespit Oranı (%)	Yanlış Alarm Oranı (%)	Kesinlik (%)
GRNN	59,12	12,46	87,54
PNN	96,33	3,34	96,66
RBNN	69,83	6,95	93,05
KNN	45,74	46,49	90,74
SVM	87,65	6,12	90,4
Bayesian	77,6	17,57	88,46
Çalışma Sonucu	99,23	9,86	98,94

Öğrenme oranı, kayıp fonksiyonu gibi değerlerin denemeler neticesinde belirlendiği ifade edilmekte, elde edilen sonuçların kıyaslanması amacıyla diğer makine öğrenme yöntemleri ile yapılan çalışmaların sonuçları Çizelge 3.3'te sunulmaktadır. Çizelge incelendiğinde, önerilen çözümün oldukça başarılı olduğu görülmektedir.

Bir başka çalışmada [51] TensorFlow ortamında GPU kullanılarak yapılan bir uygulama görülmektedir. Burada KDDCUP 99 ve NSL-KDD veri setlerinin kullanıldığı, bunun nedeninin geliştirilen uygulamanın literatürdeki diğer uygulamalarla kıyaslanması olduğu ifade edilmektedir. Nonsymmetric Deep Autoencoder (NDAE) ile eğiticisiz özellik öğrenme işlemi gerçekleştirildiği, sınıflandırma işleminde ise Random Forest (RF) algoritması kullanıldığı vurgulanmaktadır. Yığıtlı NDAE'lerin gücünden ve RF algoritmasının kesinlik ve hız özelliklerinden yararlanıldığı, böylece derin öğrenme ve basit makine öğrenmesi bileşimi hibrit bir yapı kurulduğu ifade edilmektedir.

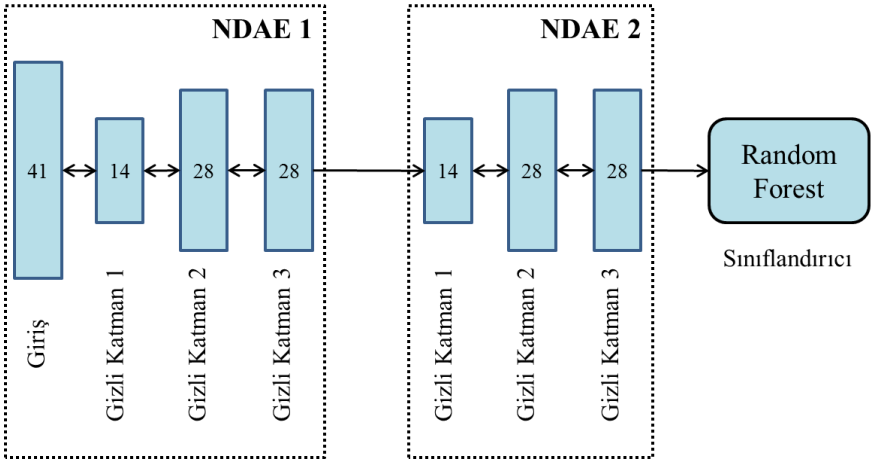
Çalışmada sunulan çözümü diğer çözümlerden ayıran özellikler iki başlıkta toplanmıştır:

- Eğiticisiz özellik öğrenme yöntemi NDAE'nin diğer auto encoder'lardan farklı olarak simetrik olmayan boyut azaltma yeteneğine sahip olduğu, bu sayede Deep Belief Network (DBN) gibi bilinen yöntemlere oranla daha iyi sınıflandırma sonuçları sağladığı,

- Sunulan sınıflandırıcının yığıtlı NDAE ve RF'nin birleşimi hibrit bir modele sahip olduğu, bu iki yöntemin birleştirilmesi ile özellikle eğitim süresinin azaltıldığı ifade edilmektedir.

Ayrıca, bilgi sistemlerinde son dönemde yaşanan gelişmelere bağlı olarak ağ saldırı tespit sistemleri ile ilgili bazı sorunlarla karşılaşıldığı, bunların başında internet üzerinde işlenen veri boyutunun çok fazla artmasının geldiği, bir ağ paketinin 6.72 ns gibi çok kısa bir sürede saldırı tespit sistemi tarafından incelenmesi gerektiği, bu durumda kesinlik, verimlilik ve etkinliğin istenen seviyede sağlanmasının mümkün olmadığı ifade edilmektedir. Artan veri boyutunun yanı sıra, saldırı tespitinin doğru şekilde yapılması için kesinlik değeri yüksek karmaşık uygulamaların geliştirilmesi gerektiği, ancak bu ihtiyacın beraberinde mali, işlemsel ve zaman bedellerini getirdiği vurgulanmaktadır. Bununla birlikte artan ağ çeşitliliğine bağlı olarak cihaz sayısı ve trafik normalliğinin belirlenememesi; davranışsal karakteristiklerin belirlenmesinin ağların dinamikliğine bağlı olarak zorlaşması, mevcut genel veri setlerinde düşük frekanslı atak kayıtlarının az olması nedeniyle istenen seviyede eğitim sağlanamaması neticede düşük frekanslı atakların belirlenmesinde güçlük yaşanması sorunları ile karşılaşıldığı ifade edilmiştir.

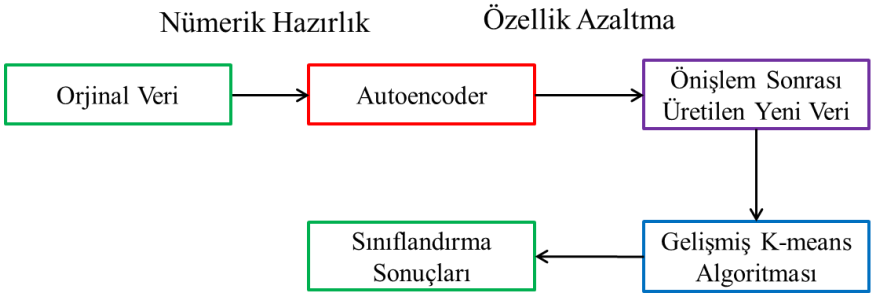
Auto encoder'ların PCA (Principle Component Analysis) yöntemine oranla özellik seçim işleminde daha başarılı olduğu, yığıtlı AE'lerin derin öğrenme tabanlı bir sistem sağladığı, bunun sıradan AE'ye göre iki adet simetrik DBN içerdiği, bunların içerisinde tipik olarak encoding için dört veya beş katmanın yine decoding için dört veya beş katmanın olduğu vurgulanmıştır.Yazarların sunduğu model Şekil 3.6'da sunulmuştur.



Şekil 3.6. Yığıtlı NDAE modeli [51].

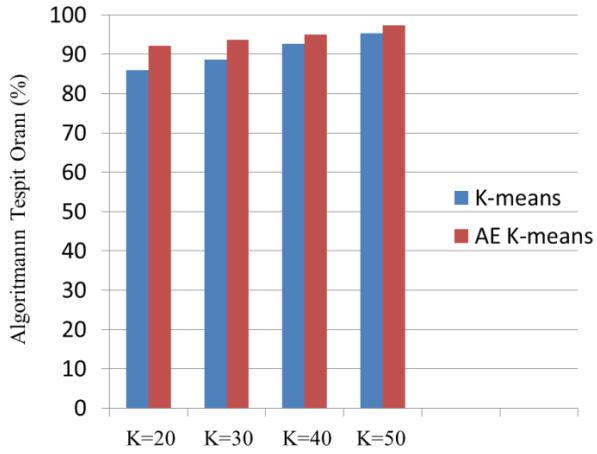
Model incelendiğinde, yukarıda da bahsedildiği üzere simetrik olmayan AE'ler ve RF algoritmasından yararlanılarak hibrit bir model sunulduğu görülmektedir. Her NDAE içerisinde üç adet gizli katman bulunmaktadır. Her gizli katman içerisinde karşılıklı olarak aynı sayıda düğüm vardır. Düğümler veriseti içerisindeki özellik sayısına işaret etmektedir. Bilindiği üzere KDDCup99 ve NSL-KDD veri setlerindeki özellik sayıları başlangıç için 41'dir. Yazarlar, sundukları çözümde, AE özelliklerinden yararlanarak özellik azaltımı gerçekleştirildiğini, gerçekleştirilen birçok hesaplama neticesinde Şekil 16'da görülen düğüm sayılarının belirlendiğini ifade etmektedirler. Sayıların belirlenmesinde karşılıklı doğrulama ile birçok kombinasyonun değerlendirildiği, neticesinde en etkin değerlerin belirlendiği ifade edilmektedir. Bu sayede performans değerlendirmesinin overfitting olmadan gerçekleştirilebildiği vurgulanmaktadır. Modellerinin çalışması neticesinde elde edilen sınıflandırma değerinin 0.995999 +/- 0.000556 olduğu ve bunun da oldukça iyi bir sonuç olduğu belirtilmektedir.

Bir diğer çalışmada [52], yukarıdakine benzer şekilde özellik azaltımı için AE, sınıflandırma için ise K-means algoritması kullanılmaktadır. Sunulan çalışmanın, büyük boyutlu veri işlemede ve sadece K-means algoritması kullanımına kıyasla doğru tespit oranının artmasına fayda sağladığı belirtilmektedir. Çalışmada KDDCup'99 veri seti kullanıldığı belirtilmekte olup sunulan çözümün çalışma mimarisi Şekil 3.7'de sunulmuştur.



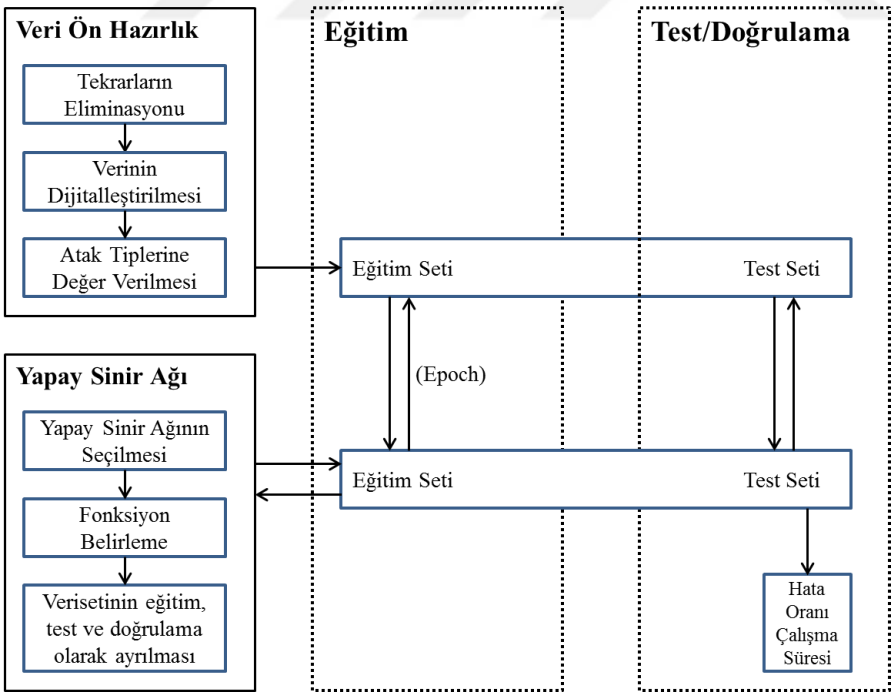
Şekil 3.7. AE-Kmeans Çözümü [52].

Veri seti üzerinde ön işlem yapıldığı, AE kullanarak 41 olan özellik sayısının 18'e düşürüldüğü, verinin standardizasyon ve normalizasyona tabi tutulduğu ifade edilmektedir. Geliştirilen çözümün, küme merkez değerine (k) bağlı olarak elde edilen sonuçları Şekil 3.8'de paylaşılmaktadır. Grafik incelendiğinde, AE-Kmeans algoritmasının, klasik K-means algoritmasına göre daha iyi sonuç verdiği görülmektedir.



Şekil 3.8. K-means ve AE-Kmeans sonuçlarının kıyaslaması [52].

Türkiye'de yapılan bir çalışmada [53] YSA tabanlı bir IDS sisteminin eğitiminde kullanılan bazı fonksiyonların verimlilik durumlarının değerlendirildiği ve yapılan testler neticesinde bunlardan birinin ön plana çıktığı vurgulanmaktadır. Çalışmada bahse konu eğitim algoritmaları hakkında bilgi verilmekte ve geliştirilen çözümün şeması sunulmaktadır. Sunulan şema Şekil 3.9'da görülmektedir.



Şekil 3.9. Sunulan Sistemin Blok Diyagramı [53].

Diyagramda görüldüğü üzere öncelikle veri ön işleme gerçekleştirilmektedir. Çalışmada KDDCup'99 veri seti kullanıldığı, veri setinin büyük boyutlu olduğu, bu nedenle çalışmada

veri setinin bir bölümünün kullanıldığı ifade edilmektedir. Uygulama geliştirme ortamı olarak MATLAB kullanıldığı, beşer düğümden oluşan iki adet gizli katman içeren bir YSA modeli oluşturulduğu belirtilmektedir. Toplam 8 adet eğitim fonksiyonunun (trainc, trainlm, trainbfg, trainscg, traincgp, trainoss, trainbr, trainr) aynı boyuttaki veri seti üzerinde 10'ar defa test edildiği; neticesinde maksimum, minimum, ortalama ve standart sapma değerlerinin belirlendiği, trainscg fonksiyonunun çalışma süresinin diğerlerine oranla çok daha iyi olduğu ancak doğruluk oranının kabul edilebilir olmadığı ifade edilmektedir. trainlm fonksiyonunun en iyi doğruluk oranını verdiği, bu fonksiyonun çalışma süresinin de kabul edilebilir seviyede olduğu vurgulanmaktadır.

Diğer bir çalışmada [54], Recurrent Neural Network (RNN) üzerine Long Short Term Memory (LSTM) mimarisinin uygulanması ve elde edilen IDS modelinin KDDCup'99 veri seti kullanılarak eğitimine dayalı bir çalışma yapıldığı ve neticesinde derin öğrenme tekniklerinin ağ sızma tespiti için etkin olduğunun teyit edildiği ifade edilmektedir. Çalışmada, otomatik ağ sızma tespit sistemlerinin inşasında çeşitli makine öğrenme yöntemlerinin kullanıldığı, son dönemde YSA tabanlı sistem kullanımının yaygınlaştığı belirtilmektedir. İleri beslemeli YSA'nın geriye yayma algoritması kullanılarak eğitimi ile elde edilen IDS çözümünün, YSA ve SVM algoritmalarının hibrit şekilde kullanımının, RNN tabanlı çözümlerin ve bunların aynı zamanda SQL tabanlı atakları da tespit ettiğinin belirlendiği ifade edilmekte, YSA'nın bu alanda kullanılabilecek iyi bir yaklaşım olmasına karşı derin öğrenme algoritmalarının daha iyi sonuçlar verdiğinin 2015 yılında yazarlar tarafından geliştirilen çözüm ile gösterildiği vurgulanmaktadır.

Sunulan LSTM-RNN çözümünde recurrent gizli katmandaki hücrelerin LSTM hücreleri ile değiştirildiği, bu sayede LSTM hücrelerinde yer alan üç kapı sayesinde gradient'in kaybolması probleminin çözüldüğü vurgulanmaktadır. KDDCup'99 veri setindeki kayıt sayısının fazla olması (4.898.431 adet) nedeniyle bu veri setinin %10'luk bölümünün kullanıldığı ifade edilmektedir.

RNN'in sıralı veri eğitiminde en bilinen yöntemlerden biri olduğu, ancak RNN'in uzun adım boyutu nedeniyle eğitim için kullanımda sorun yaşandığı ifade edilmektedir. Makalede RNN yönteminden ve gradient'in kaybolması probleminden bahsedilmekte, LSTM'in bu problemin giderilmesini nasıl sağladığı açıklanmaktadır. RNN'in bilinen ileri beslemeli yapay sinir ağlarının gelişmiş hali olduğu, bilinen ileri beslemeli yapay sinir ağlarından

farklı olarak çevrimsel bağlantılarının bulunduğu ve bu özelliklerin RNN'i sıralamaların modellenmesi için daha güçlü bir araç haline getirdiği belirtilmektedir. Giriş dizisinin $X = (x_1,x_2,...,x_T)$ olarak verildiği bir durumda, geleneksel bir RNN'in t=1'den T'ye kadar olacak şekilde gizli vektör dizisini $H = (h_1,h_2,...,h_T)$ ve çıkış vektör dizisini $Y = (y_1, y_2, ..., y_T)$ aşağıdaki şekilde hesapladığı belirtilmektedir:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \quad (3.1)$$

$$y_t = W_{hy}h_t + b_y \quad (3.2)$$

Burada $\boldsymbol{\sigma}$ nonlineerlik fonksiyonu, W ağırlık matrisi ve b ise sabit terimdir.

Yaygın RNN değişken uzunluklu dizi girdilerini tutmak için Geri Beslemeli Eğitim Zamanını (Back Propagation Training Time -BPTT) kullanmaktadır. BPTT'de model öncelikle eğitim verisi ile eğitilir. Daha sonra çıktı hata gradient değeri her zaman adımı için saklanır. Bu gradient değerinin BPTT algoritması ile eğitim esnasında ortadan kaybolması (vanishing) nedeniyle RNN'in eğitimi zordur. RNN algoritmasında gradient değerinin kaybolması sorununun giderilmesi için LSTM'in kullanıldığı belirtilmektedir. Çalışmada, bir LSTM hücresinde yer alan üç kapı ve hücre durumunun hesaplanmasında aşağıdaki eşitlikler kullanılmaktadır:

$$i_t = \sigma(W_{xi}\,x_t + W_{hi}\,h_{t-1} + W_{ci}\,c_{t-1} + b_i) \quad (3.3)$$

$$f_t = \sigma\,(W_{xf}\,x_t + W_{hf}\,h_{t-1} + W_{cf}\,c_{t-1} + b_f) \quad (3.4)$$

$$c_t = f_t\,c_{t-1} + i_c\,tanh\,(W_{xc}\,x_t + W_{hc}\,h_{t-1} + b_c) \quad (3.5)$$

$$o_t = \sigma(W_{xo}\,x_t + W_{ho}\,h_{t-1} + W_{co}\,c_t + b_o) \quad (3.6)$$

$$h_t = o_t\,tanh(c_t) \quad (3.7)$$

Bu eşitliklerde $\boldsymbol{\sigma}$ lojistik sigmoid fonksiyonu, i, f, o ve c sırasıyla girdi kapısı, unutma kapısı, çıktı kapısı ve hücre durumudur. W değerleri ilgili bağlantılar için kullanılan ağırlık matris değerleridir.

LSTM'de yukarıda bahsedilen üç kapı bilgi akış kontrolünü sağlamaktadır. Girdi kapısı girdi oranının belirlenmesini sağlar. Hücre durumu hesaplanırken bu oran Eş. 3.3 üzerinde etkilidir. Unutma kapısı, önceki hafızanın (h_{t-1}) değerini geçirir veya geçirmez. Önceki

hafıza oranı Eş. 3.4 ile hesaplanır ve Eş. 3.5'te kullanılır. Çıktı kapısı, hafıza hücresi çıktısının geçirilip geçirilmeyeceğine karar verir. Eş.3.6 bu durumu göstermektedir. LSTM yöntemiyle, bahse konu 3 kapı kullanılarak gradient'in kaybolması probleminin çözülebileceği ifade edilmektedir. LSTM-RNN mimarisinde tekrarlı gizli katmanların LSTM hücreleri ile değiştirildiği vurgulanmaktadır. Bununla birlikte geliştirilen uygulamanın Intel(R) Core(TM) i7-4790 3.60GHz özellikli işlemci, GTX Titan X GPU, 8 GB RAM ve Ubuntu 14.04 işletim sistemine sahip bir platformda çalıştığı belirtilmektedir. IDS başarı değerlendirmesinde, tespit oranı (Detection Rate -DR) ve yanlış alarm oranı (False Alarm Rate -FAR) değerlerinin kullanıldığı, DR'nin doğru tespit edilen sızma örnekleri olduğu, FAR'ın ise yanlış sınıflandırılmış normal örnekler olduğu belirtilmektedir. Buna göre;

- DR = TP/(TP + FN) (3.8)
- FAR = FP/(TN + FP) (3.9)

olarak ifade edilmektedir. Bu durumda DR değerinin artması ve FAR değerinin düşmesi performans artışı anlamına gelmektedir. Bu nedenle modelin verimliliği;

- VERİMLİLİK = DR * FAR (3.10)

olarak ifade edilmektedir.

Çalışmada, öğrenme oranı ve gizli katman boyutu ile ilgili çeşitli denemeler yapıldığı, neticesinde 0.01 öğrenme oranı ve 80 gizli katman boyutunun en iyi sonucu verdiği vurgulanmaktadır. Test için, kdddup.data.txt. dosyası içinden 10 adet rastgele seçilmiş 5000 kayıt içeren veri seti kullanıldığı, ortalama DR değerinin 0.9879003 olduğu, bir başka deyişle yapılan atakların yaklaşık %98,8'inin başarı ile tespit edildiği savunulmaktadır. Bununla birlikte, test veri setlerinde atak çeşitlerinden biri olan U2R tipinde toplam 30 adet kayıt bulunması nedeniyle, bu atak çeşidinin tespit edilemediği, DoS atak çeşidi ve normal kayıtların ise başarı ile tespit edildiği vurgulanmaktadır.

Bir diğer çalışmada [55], daha önce LSTM-RNN tabanlı bir yapıdan yararlanıldığı, elde edilen tecrübeler ışığında Nadam optimizer ile kullanılan LSTM-RNN modelinin çok daha iyi performans sergilediği belirtilmektedir. Geliştirilen modelin sızma tespitinde kullanıldığı

ve neticede %97,54 oranında kesinlik, %98,95 oranında tespit ve %9,98 gibi kabul edilebilir bir yanlış alarm oranı elde edildiği ifade edilmektedir.

Geçmişteki hibrit makine öğrenme tekniklerinin kullanıldığı sızma tespit sistemlerinde, yüksek yanlış alarm oranı ve/veya yavaş tespite bağlı performans sorunlarının görüldüğü ifade edilmektedir. Bu çalışmada, sadece sınıflandırma performansında değil, aynı zamanda her atak için sınıflandırma sonuçlarının iyileştirilmesinin de hedeflendiği belirtilmektedir.

Çalışmada kullanılan modelin LSTM-RNN temeline dayandığı, LSTM-RNN modeli eğitiminde genelde Stochastic Gradient Descent Optimization ile BPTT algoritması kullanıldığı, çalışma kapsamında bunlardan farklı bazı optimizer çeşitlerinin belirlendiği ve uygulandığı ifade edilmektedir. Gradient descent optimizasyonunda kullanılan Adagrad, Adadelta, RMSprop, Adam, Adamax ve Nadam algoritmaları hakkında bilgi verilmekte, Nadam'ın çalışmada kullanıldığı ifade edilmektedir.

Çalışmada KDD Cup99 veri seti kullanıldığı, eğitim işlem maliyetinin Mean Square Error (MSE) ile hesaplandığı, modelin sınıflandırma performansının hesaplanmasında ise confusion matrix kullanıldığı vurgulanmaktadır. MSE'nin tahmin edilenin hedeflenen değerden ne kadar farklı olduğunun hesaplanmasında kullanıldığı, bu değerin her pattern'deki hatanın karelerinin toplamının ortalaması olduğu belirtilmektedir. Örneğin i'inci örnek için p tahmin edilen, a ise gerçek değer ise MSE aşağıdaki gibi hesaplanmaktadır.

$$MSE = \frac{\sum_{i=0}^{n}(p_i + a_i)^2}{m} \tag{3.11}$$

Hata matrisinde (Confusion matrix) dört farklı durumun hesaplandığı belirtilmektedir.

Doğru Pozitif (True Positive -TP): x ile gösterilmekte olup, modelin pozitif değerleri (saldırı var) doğru tespit ettiğini ifade etmektedir. Bir başka deyişle saldırı doğru sınıflandırılmıştır.

Yanlış Negatif (False Negative -FN): y ile gösterilmekte olup, modelin pozitif değerleri (saldırı var) yanlışlıkla negatif (saldırı yok) olarak sınıflandırdığını belirtmektedir. Bir başka deyişle saldırının olduğu durumlar yanlışlıkla saldırı yokmuş gibi algılanmıştır.

Yanlış Pozitif (False Positive -FP): z ile gösterilmekte olup, modelin negatif değerleri (saldırı yok) yanlışlıkla pozitif (saldırı var) olarak sınıflandırdığını belirtmektedir. Bir başka deyişle saldırı olmayan durumlar yanlışlıkla saldırı varmış gibi algılanmıştır.

Doğru Negatif (True Negative -TN): t ile gösterilmekte olup, modelin negatif değerleri (saldırı yok) doğru bir şekilde sınıflandırdığını göstermektedir. Bir başka deyişle saldırı olmadığı doğru bir şekilde sınıflandırmıştır.

Yukarıda bahsedilen değerler confusion matrix'e aktarılarak modelin performans değerlendirmesinde kullanılan aşağıdaki ifadeler elde edilmiştir.

Doğru Pozitif Oranı = Tespit Oranı (Detection Rate) = x/(x+y) : Bu değer pozitif örneklerin doğru sınıflandırılması oranını ifade etmektedir. Burada FN (y) değerlerinin çok olması istenilen bir şey değildir. Çünkü bir saldırının yanlışlıkla saldırı yokmuş gibi sınıflandırılması sistemi büyük riske sokacaktır.

Hassasiyet (Precision) = x/(x+z) : Bu değer bir örneğin doğru sınıflandırılma oranını göstermektedir. Bir başka deyişle doğru sınıflandırılmış saldırı örnekleri sayısının, kendi değeri ile yanlışlıkla saldırı var diye sınıflandırılan temiz örneklerin sayısının toplamına bölümü ile elde edilen orandır. Precision bir örneğin doğru sınıflandırma olasılığını verir.

Duyarlılık (Recall) = (x+t) / (x+y+z+t) : Bu değer doğru sınıflandırılmış saldırı ve temiz örnek sayılarının toplamının tüm sınıflandırma değerleri toplamına bölünmesi neticesinde elde edilen orandır. Recall, modelin doğru tahmin ettiği test sonuçlarının oranını vermektedir.

Yanlış alarm oranı (False Alarm Rate -FAR) = z / (z+t) : Yanlış alarm oranı, yanlışlıkla saldırı olarak tespit edilen temiz örnek sayısının, kendi değeri ve doğru tespit edilen temiz örnek sayısının toplamına bölümü ile elde edilmektedir.

Tespit oranı değerinin artması buna karşı FAR değerinin azalması performans artışı anlamına gelmektedir. Bu nedenle çalışmada verimlilik (Efficiency) metriği de kullanılmaktadır. Bu metriğin kullanımı ile IDS modelinin rahatlıkla değerlendirildiği ifade edilmektedir.

Çalışmada kullanılan hiper-parametre değerleri Çizelge 3.4'te sunulmaktadır ve bu değerlerin model eğitim sonuçlarının geliştirilmesinde büyük öneme sahip olduğu vurgulanmaktadır.

Çizelge 3.4. Modelin hiper-parametre değerleri [55].

Parametre İsimleri	Değer
Öğrenme Oranı	0,01
Gizli Katman Sayısı	80
Tekrar (Epoch) Sayısı	500
Batch Boyutu	100

Çalışmanın ilk aşamasında hiper-parametre değerlerinin belirlendiği, ikinci aşamada ise bu değerler kullanılarak LSTM modelinin uygulandığı ifade edilmektedir. İkinci aşamada, LSTM-RNN modelinin yukarıda bahsedilen altı farklı optimizer ile uygulandığı bu optimizer'lar içinde kullanılan parametre değerlerinin paylaşıldığı vurgulanmaktadır.

Çalışmada en uygun optimizer'ı bulabilmek için KDDCUP'99 verisetine altı optimizer'ın da uygulandığı ölçümlerin iki durum için gerçekleştirildiği vurgulanmaktadır. İlk durumda beş atak tipinin ortalama sınıflandırma yüzdesinin hesaplandığı, ikinci durumda ise sınıflandırma performansının bazı metriklerin işlenmesi ile değerlendirildiği ifade edilmektedir.

İlk durum için yapılan değerlendirme sonucunda, Nadam optimizer'ın 0.002 öğrenme oranı ile uygulanması durumunda diğerlerine göre daha iyi sınıflandırma oranı sağladığı belirtilmektedir. İlk durum için elde edilen sonuçlar Çizelge 3.5'te görülmektedir.

Çizelge 3.5. Kullanılan optimizer tipine bağlı olarak elde edilen sınıflandırma sonuçları [55].

Optimizer	DoS	Probe	R2L	U2R	Normal
RMSprop	0,9672	0,59131	0,51041	0,5	0,89945
Adagrad	0,98272	0,63132	0,56719	0	0,91213
Adadelta	0,98131	0,5819	0,52	0	0,81508
Adam	0,98288	0,60313	0,51401	0	0,90597
Adamax	0,98362	0,68257	0,66236	0,5	0,94464
Nadam	0,98435	0,77034	0,66702	0,5	0,95716

İkinci durum için de aynı sonucun geçerli olduğu belirtilmekte ve sonuçlar Çizelge 3.6'da paylaşılmaktadır.

Çizelge 3.6. Kullanılan optimizer tipine bağlı olarak elde edilen metrik sonuçları [55].

Optimizer	Kesinlik	Duyarlılık	FAR	Hassasiyet	Efektiflik
RMSprop	0,9496	0,97709	0,15361	0,95987	7,0753
Adagrad	0,96786	0,99142	0,12277	0,96887	8,5766
Adadelta	0,94401	0,99074	0,23128	0,9442	5,40848
Adam	0,95915	0,98723	0,1287	0,9675	8,20554
Adamax	0,97376	0,98621	0,10642	0,97595	9,30781
Nadam	0,9754	0,9895	0,0998	0,9769	9,9808

Çalışma neticesinde elde edilen sonuçlar ile diğer IDS model sonuçlarının kıyaslanması Çizelge 3.7'de sunulmaktadır.

Çizelge 3.7. Geliştirilen modelin diğer yöntemlerle kıyaslanması [55].

Sınıflandırıcı	Precision	Tespit Oranı	Kesinlik	FAR
FNN	92,47	86,89	97,35	2,65
GNNN	87,08	59,12	93,05	12,46
RBNN	69,56	59,12	93,05	12,46
Jordan ANN	-	62,9	-	37,09
Hessian-free RNN	-	95,37	-	2,1
SGD'li LSTM RNN	-	98,88	93,93	10,04
Sunulan Sınıflandırıcı	97,69	98,95	97,54	9,98

Yukarıda bahsedilen çalışmaların yanı sıra son dönemde gerçekleştirilen çalışmalar [56-59] incelenmiştir. Yapılan inceleme neticesinde, ağ saldırı tespit sistemlerinin yüksek kesinlik değerinde sınıflandırma yapması gerektiği görülmüştür. İncelenen çalışmalarda, yüksek kesinlik değeri elde edilmesi amacıyla, bazen sadece derin öğrenme yöntemlerinin, bazen de derin öğrenme yöntemleri ile klasik makine öğrenme yöntemlerinin birlikte kullanıldığı hibrit yapıların sunulduğu belirlenmiştir. İncelemede ayrıca, AE ve RBM gibi derin öğrenme yöntemlerinin veri seti boyutu azaltılması ve verinin temizlenmesi işlemlerinde kullanıldığı, RNN ve DBN gibi derin öğrenme tekniklerinin ise eğitim ve sınıflandırma işlemlerinde kullanıldığı görülmüştür. Derin öğrenme yöntemlerinin büyük boyutlu veriyi işleyerek yüksek öğrenme oranları göz önünde bulundurulduğunda, özellik azaltma işlemi yapmanın tekniğin doğasına aykırı olduğu, ancak ön işlem esnasında uygulanacak doğru özellik azaltım işlemleri ile çalışma süresinin azaltılabileceği belirlenmiştir. Gradient'in kaybolması problemine etkili bir çözüm sunan RNN tabanlı

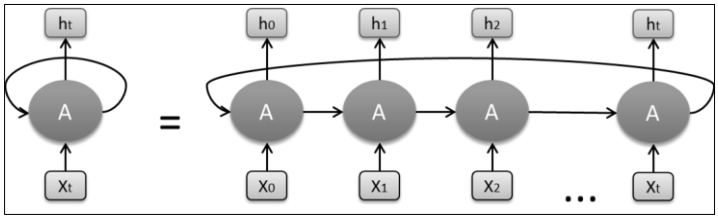
LSTM yönteminin son dönemde yapılan çalışmalarda sıklıkla kullanılmaya başladığı ve elde edilen sınıflandırma sonuçlarının oldukça başarılı olduğu gözlemlenmiştir.

Bununla birlikte, KDDCUP'99 ve NSL-KDD genel veri setlerinin modelin diğer modellerle etkin şekilde kıyaslanabilmesi amacıyla yaygın olarak kullanıldığı, veri setlerinin ön işleme tabi tutulduğu, veri seti içerisinde bulunan sembolik değerlerin sayısal değerlere dönüştürüldüğü, verinin normalizasyona tabi tutulduğu, eğitim ve test veri seti boyutları ile ilgili standart bir uygulama olmadığı belirlenmiştir.

Ayrıca, uygulama geliştirme ortamı olarak genelde TensorFlow'un kullanıldığı, derin öğrenme uygulamaları için yüksek işlemci gücü gerektiğinden GPU kullanımının yaygın olduğu görülmüştür.

3.2. Model Geliştirme Çalışması

Ağ sızma tespit sistemleri, belirli ağ trafik parametrelerine bağlı bir dizi içeriğin normal veya saldırı kaydı şeklinde sınıflandırılmasını hedefler. Bir başka ifadeyle bir sınıflandırma probleminin çözümüne odaklanır. Sınıflandırma işlemlerinde bir girdinin belirlenen k adet sınıftan hangisine ait olduğunun tespitine çalışılır. Bunun için öğrenme algoritmasının $f : R^n \rightarrow \{1,\ldots,k\}$ fonksiyonu belirlemesi gerekmektedir. Böylece algortima *x* girdisinin, $y = f(x)$ olacak şekilde *y* olarak belirlenen kategoriye ayrılmasını sağlar. Sınıflandırma problemlerinde sıralı bilgi işlenen ve bir bilginin önceki işlem sonucuna bağlı olduğu durumlar, çözülmesi gereken bazı problemleri doğurmaktadır. Örneğin insanlar bir yazı okurken kelimeleri önceki kelimelere bağlı olarak anlamlandırmakta, bir kelime okunduktan sonra metni anlamak için bir süre unutulmamakta, böylece kalıcı düşünceler üretilmektedir. Yapay zekâ açısından bakıldığında YSA'nın hücre çıkış verisini tutmaması nedeniyle bu problemin çözümünde kullanımı mümkün görülmemektedir. Bir derin öğrenme yöntemi olan Recurrent Neural Network (RNN) bu problemin çözümüne önemli bir katkı sağlamaktadır. Çünkü RNN, bir sonrakine mesaj geçiren aynı yapay sinir ağının birden çok kopyasıdır ve sonraki ağa bilgi aktarımı gerçekleştiğinden unutma probleminin çözümüne katkı sağlamaktadır. Basit bir RNN yapısı Şekil 3.10'da görülmektedir.

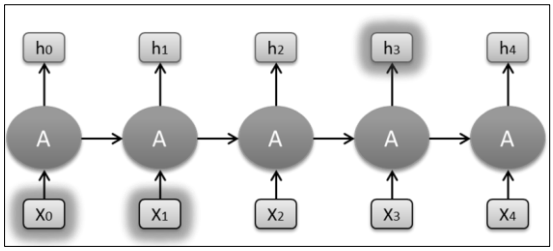


Şekil 3.10. RNN mimarisi.

Yukarıdaki şekilde görüleceği üzere RNN'in doğal olarak oluşturduğu zincir yapısı dizi veya listeler ile ilişkili problemlerin çözümünde başarılı bir şekilde kullanılmaktadır [60, 61].

RNN'in ortaya çıkmasında, önceki düğümde yapılan işlem sonucu üretilen bilginin mevcut işleme girdi olması fikri yatmaktadır [62]. Ancak tahmin edilecek değer ile bu değeri tahmin etmeyi sağlayan bilgi arasındaki mesafenin fazla olması durumunda RNN etkin şekilde çalışmamaktadır. Örneğin 8'inci düğümdeki işlem, 2'nci düğümdeki işlem neticesinde elde edilen bilgiye dayanıyorsa, kurulan RNN mimarisinde unutma durumu oluşacağından istenen sonuç elde edilememektedir.

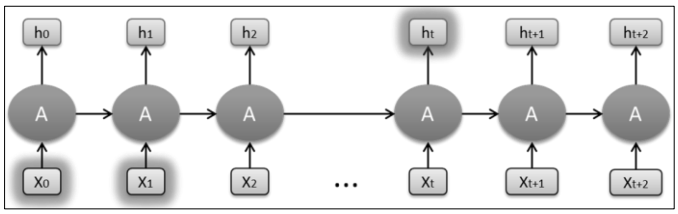
Örneğin "Mavi bir denize benzer gökyüzü" cümlesinde, "gökyüzü" kelimesi önceki kelimelere bağlı olarak tahmin edilmeye çalışıldığında RNN'in başarı ile çalıştığı görülmektedir [63].



Şekil 3.11. RNN'in başarılı bir şekilde çalıştığı durum [63].

Ancak bazı durumlarda tahmin edilmeye çalışılan kelime ile bu tahmini gerçekleştirmede kullanılan anahtar kelime arasında birçok kelime hatta cümle bulunabilir. Tahmin edilecek içerik ile tahmin etmeyi sağlayan içerik arasında mesafe fazla olması durumunda RNN'in bilginin bağdaştırılması ile ilgili sorun yaşadığı görülmektedir. Örneğin "Ben Türkiye'de doğdum. … Akıcı bir şekilde Türkçe konuşurum." içeriğinde tahmin edilecek kelime Türkçe, bu tahminin gerçekleştirilmesine ilişkin kelime ise Türkiye'dir. Türkçe ile Türkiye

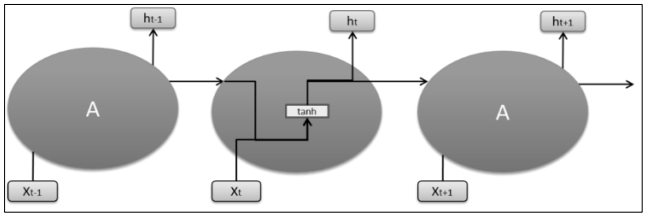
arasında bir ilişki kurulması gerekir ancak iki kelime arasında yer alan içeriğin oluşturduğu mesafe bu ilişkinin RNN ile kurulmasını engeller. Bu durum Şekil 3.12'de görülmektedir.



Şekil 3.12. RNN'in başarısız olduğu durum [63].

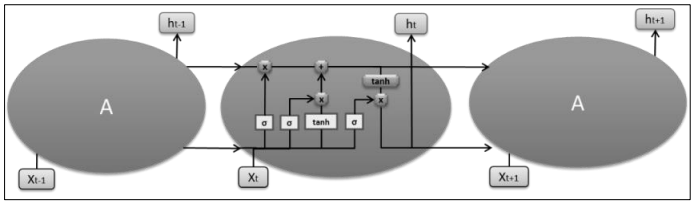
Hochreiter ve Schmidhuber 1997 yılında bu problemi, RNN'in hatırlama süresini uzatarak çözmüştür [64]. Mantıksal ve lineer bileşenlerden oluşan bir hafıza hücre birimi tanımlamışlar ve arzu edilen verinin istenilen zamanda bu hücreye yazılmasını, istenilen süre boyunca hücrede tutulmasını ve istenildiğinde hücreden okunmasını sağlamışlardır [65]. Bu işlemler aşağıda detaylı bahsedilecek kapılar ile gerçekleştirilmektedir.

Standart bir RNN'de tekrar eden modül bir katman içermektedir. Şekil 3.13'te RNN'in tekrar eden modülünün içeriği görülmektedir.



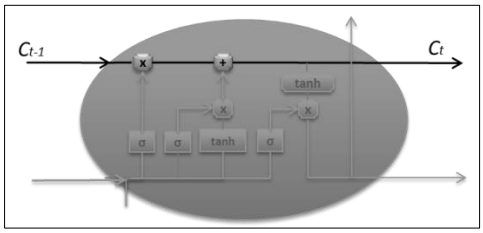
Şekil 3.13. Standart RNN yapısı [63].

LSTM de bu zincir yapısına sahiptir ancak tekrar eden modül RNN'e göre daha komplekstir ve bir sinir ağı katmanı yerine dört katmandan oluşur. Bu katmanlar birbirleri ile özel bir şekilde iletişim halindedir. LSTM'in tekrar eden modülü Şekil 3.14'te sunulmuştur.



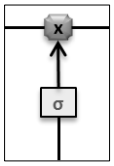
Şekil 3.14. LSTM hücresi [63].

LSTM için en önemli bileşen eğrisel ve kendi kendine döngüye sahip hücre durumudur (cell state). Hücre durumu verinin iletilmesinde kullanılan bir değerdir. Veri hiç değiştirilmeden de aktarılabilir.



Şekil 3.15. Hücre durumu (cell state) [63].

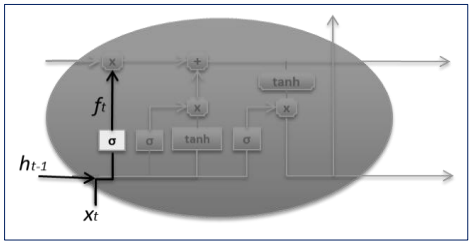
LSTM hücre durumuna dikkatli bir şekilde düzenlenmiş kapılar vasıtasıyla bilgi ekleme veya çıkarma kabiliyetine sahiptir. Kapılar, bilginin opsiyonel olarak iletimine izin verilmesini sağlar. Kapılar, bir sigmoid yapay ağ katmanından ve noktasal çarpım işleminden oluşmaktadır.



Şekil 3.16. LSTM Kapısı [63].

Sigmoid katmanı sıfır ile bir arasında değerler üretir ve bu değerler bilginin ne kadarının iletileceğini belirler. Sıfır olması durumunda hiç bilgi iletimi olmaz, bir olması durumunda ise tüm bilgi iletilir. LSTM'de bu kapılardan üç adet vardır ve hücre durumunun korunması ve kontrolünü sağlar. LSTM'in çalışma mantığı aşağıda adım adım sunulmaktadır.

İlk adımda hücre durumundan hangi bilginin atılacağı belirlenir. Bu işlem, unutma kapısı (forget gate) vasıtasıyla yapılır. Unutma kapısı Şekil 3.17'de görüleceği üzere h_{t-1} ve x_t değerlerine bakar ve hücre durumu içerisindeki her C_{t-1} sayısı için 0 ile 1 arasında bir değer üretir. Bunlardan 1 "bu değeri tamamen tut" ve 0 ise "bu değeri tamamen sil" anlamına gelmektedir. x_t güncel girdi vektörünü, h_t tüm LSTM hücrelerinin çıktılarını içeren güncel gizli katman vektörünü temsil eder.

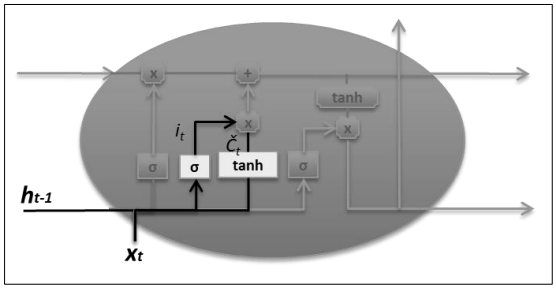


Şekil 3.17. Unutma (forget) kapısı [63].

Unutma kapısına girdiler neticesinde üretilen f fonksiyonu aşağıda yer almaktadır. Bu eşitlikte b_f unutma geçitleri için önyargı değerlerini, W_f ise yineleme ağırlıklarını gösterir.

$$f_t = \sigma (W_f [h_{t-1}, x_t] + b_f) \tag{3.12}$$

Bir sonraki adım ise hücre durumuna hangi yeni bilginin yazılacağının belirlenmesidir. Bu işlem iki adımdan oluşmaktadır. İlk olarak girdi kapısı (input gate) olarak adlandırılan bir *sigmoid* katmanı tarafından hangi değerlerin güncelleneceği belirlenir. Daha sonra bir *tanh* katmanı, yeni aday değerler için $\check{C}_t$ olarak isimlendirilen ve hücre durumuna eklenebilen taşıyıcı bir vektör oluşturur. Daha sonra girdi kapısından elde edilen değer ile bu vektör birleştirilerek durum güncellemesi üretilir.



Şekil 3.18. Girdi kapısı ve durum güncellemesi [63].

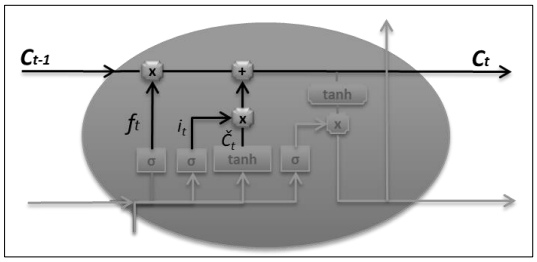
Girdi kapısı sonucu ve durum güncellemesi için kullanılan eşitlikler aşağıda yer almaktadır.

$$i_t = \sigma (W_i [h_{t-1}, x_t] + b_i) \tag{3.13}$$

$$\check{C}_t = tanh (W_c [h_{t-1}, x_t] + b_c) \tag{3.14}$$

Bir sonraki adımda eski hücre durumu C_{t-1} değeri yeni C_t değerine güncellenir. Bu aşamada eski durum C_{t-1} unutma (forget) kapısından elde edilen değer ile çarpılır. Bunun amacı daha

önceden unutulmasına karar verilen değerlerin unutulmasının sağlanmasıdır. Daha sonra bu değere i_t ile $\check{C}_t$'nin çarpımından elde edilen değer eklenir. Bu işlem, yeni aday değerlerin elde edilmesini sağlar.

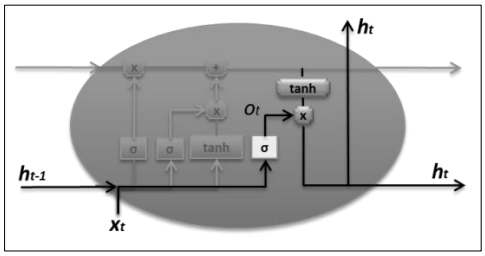


Şekil 3.19. Hücre durumunun güncellenmesi [63].

Hücre durumunun güncellenmesini sağlayan eşitlik aşağıda yer almaktadır.

$$C_t = f_t C_{t-1} + i_t \check{C}_t \tag{3.15}$$

Son olarak çıktı değerinin belirlenmesi gerekmektedir. Bu çıktı değeri hücre durumunun filtrelenmiş haline bağlı olarak belirlenir. Bu aşamada öncelikle hücre durumunun hangi bölümlerinin çıktı olacağını belirlemek amacıyla bir *sigmoid* fonksiyonu çalıştırılır. Daha sonra hücre durumu bir *tanh* işlemine tabi tutularak değerin -1 ile +1 aralığına alınması sağlanır. Son olarak *sigmoid* fonksiyonundan elde edilen değer ile *tanh* işlemine tabi tutulan hücre durumu çarpılarak çıktı olarak sadece istenilen bölümlerin elde edilmesi sağlanır.



Şekil 3.20. Çıktı değerinin elde edilmesi [63].

Çıktı değerinin elde edilmesinde kullanılan eşitlikler aşağıda yer almaktadır.

$$o_t = \sigma\,(W_o\,[h_{t-1}, x_t] + b_o) \tag{3.16}$$

$$h_t = o_t * \tanh(C_t) \tag{3.17}$$

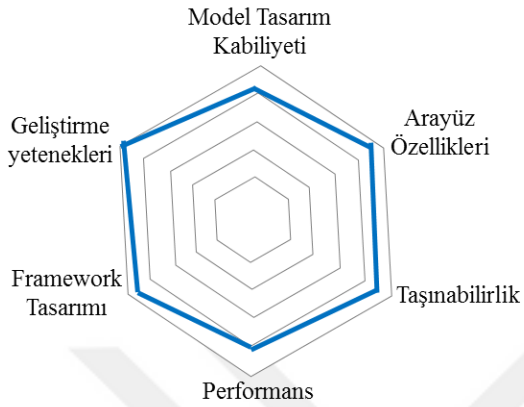
LSTM basit RNN katmanının farklı bir versiyonudur ve birçok zaman adımının taşınmasını sağlayan bir yol eklemeyi sağlar. Bir başka ifadeyle, işlenmeye çalışılan diziye paralel çalışan taşıyıcı bir yol sağlar ve dizideki bir bilgi herhangi bir zamanda taşıyıcı yol üzerine aktarılabilir. Aktarılan bilgi sonraki adıma ulaştırılır ve ihtiyaç duyulduğunda tekrar yerine taşınabilir. Böylece eski bilginin gerektiğinde kullanılması sağlanırken, aynı zamanda ana dizi ile ilgili işlemlerde gradyanın yok olması probleminin oluşmasını engeller [28]. LSTM ağlarının, mesafe içeren bağımlılıkların öğrenilmesinde diğer basit yinelemeli ağlara göre daha başarılı olduğu görülmektedir. LSTM'in diziler üzerinde gerçekleştirilen işlemlere yönelik sağladığı işlem gücünün, diğer birçok problemde olduğu gibi, IDS geliştirmede de kullanılabileceği görülmektedir.

3.3. Uygulama Geliştirme Platformunun Belirlenmesi

Derin öğrenmenin birçok alanda hızlı şekilde yaygınlaşması neticesinde, teknoloji firmaları ve araştırma ekipleri tarafından, bağımsız ve özgünderin öğrenme modeli geliştirme platformları üretilmeye başlanmıştır. Caffe, Caffe2, Tensorflow, Theano, MXNet, CNTK, Torch, PyTorch, Pylearn2, Scikit-learn, Matlab, Chainer ve Deeplearning4j platformları bunlardan bazılarıdır. Bu alanda yapılan bir çalışmada [66], yaygın olarak kullanılan sekiz platformun altı farklı açıdan (model tasarım kabiliyeti, arayüz özellikleri, taşınabilirlik, performans, framework tasarımı ve geliştirme için sunulan özellikler) değerlendirildiği görülmektedir.

Derin öğrenme platformlarının kullanım amaçlarına göre yapılan değerlendirmede; C++ ile kod geliştirme işlemlerine yatkın olunması durumunda Caffe ve MXNet'in kullanılabileceği, bunlardan MXNet'in bellek alanı açısından daha maliyet etkin olduğu belirtilmektedir. Python programlama diline hâkim olunması durumunda Theano'nun önerildiği, Theano'nun bilimsel araştırma çalışmalarının yürütülmesine ve karmaşık sinir ağlarının geliştirilmesine uygun olduğu ifade edilmektedir. Torch'un iyi eğitilmiş modellerde daha karmaşık uygulamalar geliştirmek için, Tesorflow'un ise çok fazla RNN kullanımına ihtiyaç duyulması durumunda etkin olduğu vurgulanmaktadır. Paralel eğitim ihtiyacı varsa CNTK aracının kullanılması gerektiği, Matlab'ın iyi bir araç olmasına karşın arayüzünün program çalıştırma süresini uzattığı ifade edilmektedir. Eğer bir derin öğrenme algortiması mevcut bir projeye entegre edilmek istenirse Deeplearning4j ile başlanmasının faydalı olacağı belirtilmektedir.

Çalışmada sunulan grafikte, Tensorflow platformunun altı farklı açıdan değerlendirildiğinde diğerlerine oranla daha başarılı olduğu görülmektedir. Grafiğin Tensorflow ile ilgili bölümü Şekil 31'de sunulmaktadır.



Şekil 3.21. Tensorflow'un altı açıdan değerlendirmesi [66].

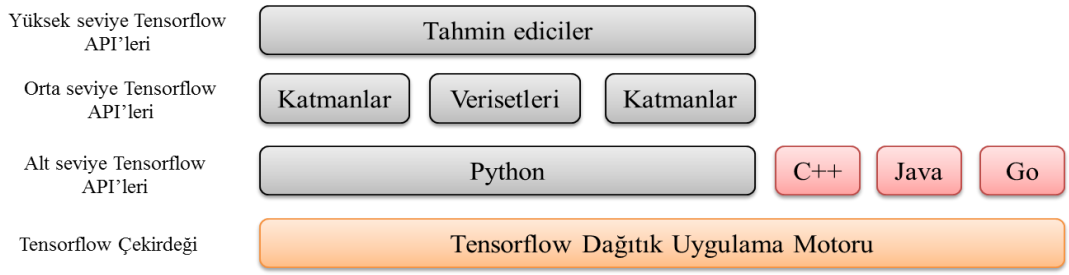
Bir diğer çalışmada [67], derin öğrenme platform ve kütüphanelerinin özelliklerini içeren bir çizelge sunulmaktadır. Bu çizelge incelendiğinde, tüm araçların açık kaynak lisanslı olduğu, bazılarının açık kaynak olmasının yanısıra farklı lisansları da içerdiği (örneğin Tensorflow için Apache 2.0, Keras için MIT, CNTK için Microsoft gibi), araçların genelde C++ ve Python ile geliştirildiği, sadece Torch için Lua, PyTorch için C'nin de kullanıldığı, hesaplama grafının genelde statik olduğu, Tensorflow için küçük bir dinamik graf desteğinin olduğu, Pytorch, MXNet ve Chainer için dinamik hesaplama grafı bulunduğu, arayüzlerin hepsinde Python kullanıldığı, Python'un yanısıra başka dillerde de arayüzlerin olduğu, en çok kullanılan dilin C++ olduğu, en popüler geliştirme aracının Tensorflow olduğu, tüm plaformların akademik ve endüstri amaçlı kullanılabildiği, CNTK'in kısıtlı, Caffe2'nin ise tam mobil çözüm desteğine sahip olduğu ifade edilmektedir.

Bir başka çalışmada [68], derin öğrenme platformlarının, etkin lineer cebir hesaplamaları için çok thread'li paralelliğe imkân sunan çeşitli kütüphaneler kullandığı belirtilmektedir. Çalışma kapsamında değerlendirmeye tabi tutulan MXNet ve Pytorch'un OpenBlas, Tensorflow'un ise Eigen kütüphanelerini kullandıkları ifade edilmektedir. Her üç platformun GPU ile hızlandırılmış hesaplamalar için NVIDIA CUDA aracını kullanıdıkları belirtilmektedir. Çalışmada, Rysia adı verilen bir derin öğrenme platformu kıyaslama ortamı geliştirildiği, her üç platformda, ileri beslemeli ağ ile el yazısı sınıflandırması, evrişimsel ağlar ile görüntü sınıflandırması ve LSTM ağları ile sentiment analizi yapıldığı, her üç

platformda da aynı hiperparametrelerin kullanıldığı belirtilmektedir. Farklı platformlarda bahse konu uygulamaların çalıştırılması neticesinde elde edilen sonuçlar, grafiklerle kıyaslamalı şekilde gösterilmektedir. Pytorch'un çıkarımsal yükler için en iyi performansı sunduğu, buna karşı model yapısının ve yük paylaştırma stratejisinin platform başarısını olumsuz etkilediği vurgulanmaktadır.

Derin öğrenme platformlarının kıyaslanmasına yönelik benzer çalışmalarda [69-71], çeşitli uygulamalar üzerinden, eğitim ve test süreleri, GPU ve CPU performansları, dinamik veya statik uygulama yetenekleri, aynı eğitim neticesinde sınıflandırma sonuçları gibi özelliklerin değerlendirildiği görülmüştür. İnceleme neticesinde TensorFlow platformunun Apache 2.0 açık kaynak kod lisansı olması, Google firmasının makine öğrenmesi ve derin öğrenme araştırmalarını destekleyen makine zekâsı birimi tarafından desteklenmesi, Python gibi güçlü bir programlama dili ile uygulama geliştirmeye imkân sunması, CPU, GPU, mobil cihaz ve yüzlerce düğümden oluşan büyük ölçekli dağıtık sistemler üzerinde çalışma desteğinin olması, veri akış programlama için sunduğu birçok kütüphane ile derin öğrenme araştırma ve geliştirme işlemlerine temel sağlaması, çok boyutlu dizilerde matematiksel tanımlamaları etkin bir şekilde çalıştırması, yatırıma açık en popüler platform olmasının yanısıra, RNN tabanlı uygulama geliştirmeye müsait olması gibi özelliklerinin ön plana çıktığı görülmüştür. Bununla birlikte, uygulama geliştirme çalışmalarında yaşanabilecek aksaklıkların çözümünde yol gösterici olabilecek bir kullanıcı grubunun (https://stackoverflow.com/) ve örnek uygulama temin ortamının (https://github.com/) bulunduğu belirlenmiştir. Ayrıca https://www.TensorFlow.org/ sitesinde kurulumdan uygulama geliştirmeye kadar birçok alanda yeterli seviyede teknik destek sağlandığı görülmüştür. Bu nedenlerle Tensorflow, tez çalışması kapsamında geliştirme platformu olarak belirlenmiştir.

Tensorflow programlama mimarisi Şekil 31'de görülmektedir. Mimaride birçok API bulunmaktadır. TensorFlow platformunda uygulama geliştirilirken tahmin ve veriseti API'lerinin kullanımı işlem kolaylığı sağlamaktadır.



Şekil 3.22. TensorFlow programlama mimarisi.

3.4. NSL-KDD Veriseti Ön İşlemleri

Model geliştirme sürecinin ilk adımı veri setinin hazırlanmasıdır. Veri seti, modelin eğitileceği ve doğru sonuçları öğreneceği verileri içerir. Veri seti, eğitim, doğrulama ve test setleri olarak ayrılır. Eğitim seti, modelin öğrenmesi, doğrulama seti modelin doğruluğunu test etmek vetest seti ise modelin gerçek dünya performansını ölçmek için kullanılır. Bu kapsamda literatürdeki diğer uygulama sonuçları ile tez kapsamında geliştirilen uygulamaların başarı oranlarının doğru şekilde kıyaslanabilmesi amacıyla, NSL-KDD verisetinin tez kapsamında kullanılabileceği belirlenmiştir.

NSL-KDD veri seti, ağ güvenliği alanında makine öğrenmesi algoritmalarının performansını değerlendirmek için sıklıkla kullanılan bir veri setidir. NSL-KDD veriseti, 1999 yılında ABD Hava Kuvvetleri Komutanlığı bilgi sistem ağlarındaki veri kullanlarak üretilen KDDCUP'99 verisetindeki üç önemli sorunu gidermek amacıyla, M.Tavallaee ve diğerleri tarafından 2009 yılında yayınlanmıştır [72]. Bahse konu sorunlardan ilki, veriseti içerisinde yüksek oranda gereksiz kayıt bulunmasıdır. Eğitim verisetinde %78, test verisetinde ise %75 oranında tekrarlı kayıt bulunmaktadır. Eğitim setinde bu kadar gereksiz veri bulunması, verisetinin dengesiz olmasına, bu nedenle öğrenme algoritmalarının sıklık oranı yüksek verileri öğrenmeye meyilli hale gelmesine, böylece ağ için daha tehlikeli olan U2R gibi daha az yoğunluğa sahip atakların öğrenilememesine neden olmaktaydı. İkinci sorun KDDCUP'99 verisetinin sadece hizmet reddi saldırılarını içermesi, bu nedenle, saldırı tespiti algoritmalarının gerçek hayatta kullanılabilirliği sınırlı hale gelmesidir.

M. Tavallaee ve diğerleri (2009) tarafından yapılan çalışmada, toplam 21 farklı eğitilen makine ile eğitim ve test setlerindeki kayıtların sınıflandırıldığı, eğitim setindeki kayıtların %98 seviyesinde, test setindekilerin ise %86 seviyesinde başarı ile etiketlendirildiği

belirtilmektedir. Buradan, veriseti içerisindeki kayıt çeşitliliğinin azlığı nedeniyle KDDCUP'99 veriseti ile eğitilen basit makine öğrenme algoritmalarının bile yüksek oranda sınıflandırma yapabildiği sonucu çıkarılmaktadır. Son sorun ise KDDCup'99 veri setindeki bazı özniteliklerin gereksiz, bazılarının ise yetersiz olmasıdır. Bunun, algoritmaların doğruluğunu ve performansını etkilediği gözlemlenmiştir.

KDDCUP'99 verisetinin yukarıda belirtilen eksikleri,verisetinden seçilen kayıtlar ile üretilen NSL-KDD veriseti ile giderilmiştir. NSL-KDD'nin gerçek ağlardan üretilen verisetlerini temsil eden mükemmel bir veriseti olmadığı, ancak araştırmacılara ağ sızma tepit modellerinin kıyaslanması için oldukça yardımcı olduğu vurgulanmaktadır.

$KDDTrain^{+}$ eğitim seti 125.973, $KDDTest^{+}$ test seti ise 22.544 kayıttan oluşmakta, yukarıda bahsedilen 21 sınıflandırıcı tarafından doğru sınıflandırılan kayıtları içermeyen $KDDTest^{-21}$ ise 11.850 kayıt içermektedir. Ayrıca eğitim setinin ilk %20'lik bölümünden KDDTest verisetinin üretildiği, yedi farklı makine öğrenme yönteminin (J48, Naive Bayes, NB Tree, Random Forest, Random Tree, Multi-layer Percepton ve SVM) KDDTest, $KDDTest^{+}$ ve $KDDTest^{-21}$ versetleri üzerine uygulanması neticesinde en yüksek performansın KDDTest, en düşük performansın ise $KDDTest^{-21}$ veriseti ile elde edildiği belirtilmektedir.

NSL-KDD verisetleri, geçmişte olduğu gibi günümüzde de ağ sızma tespit sistemi çalışmalarında [73-77] sıklıkla kullanılmaktadır. NSL-KDD verisetinde yer alan normal ve saldırı kayıt sayılarına yönelik bilgi, Çizelge 3.8'de sunulmuştur.

Çizelge 3.8. NSL-KDD verisetlerinde bulunan kayıtların dağılımı.

	Toplam	Normal	DoS	Probe	R2L	U2R
$KDDTrain^{+}$	125973	67343	45927	11656	995	52
$KDDTest^{+}$	22544	9711	7458	2421	2754	200
$KDDTest^{-21}$	11850	2152	4342	2402	2754	200

NSL-KDD verisetinde toplam 41 özellik alanı bulunmaktadır. Bu özelliklerden üçü sembolik, diğerleri nümerik değerler içermektedir. NSL-KDD veriseti içerisinde bulunan özelliklere ait bilgi Çizelge 3.9'da sunulmuştur.

Çizelge 3.9. NSL-KDD verisetinde bulunan özelliklere ait bilgi.

No	Özellik	Tip/Etiket	No	Özellik	Tip/Etiket
1	duration	Nümerik / T	22	is_guest_login	Nümerik/ İ
2	protocol_type	Sembolik / T	23	count	Nümerik / AT
3	service	Sembolik / T	24	srv_count	Nümerik / AT
4	flag	Sembolik / T	25	serror_rate	Nümerik / AT
5	src_bytes	Nümerik / T	26	srv_serror_rate	Nümerik / AT
6	dst_bytes	Nümerik / T	27	rerror_rate	Nümerik / AT
7	land	Nümerik / T	28	srv_rerror_rate	Nümerik / AT
8	wrong_fragment	Nümerik / T	29	same_srv_rate	Nümerik / AT
9	urgent	Nümerik / T	30	diff_srv_rate	Nümerik / AT
10	hot	Nümerik / İ	31	srv_diff_host_rate	Nümerik / AT
11	num_failed_logins	Nümerik / İ	32	dst_host_count	Nümerik / H
12	logged_in	Nümerik / İ	33	dst_host_srv_count	Nümerik / H
13	num_compromised	Nümerik / İ	34	dst_host_same_srv_rate	Nümerik / H
14	root_shell	Nümerik / İ	35	dst_host_diff_srv_rate	Nümerik / H
15	su_attempted	Nümerik / İ	36	dst_host_same_src_port_rate	Nümerik / H
16	num_root	Nümerik / İ	37	dst_host_srv_diff_host_rate	Nümerik / H
17	num_file_creations	Nümerik / İ	38	dst_host_serror_rate	Nümerik / H
18	num_shells	Nümerik / İ	39	dst_host_srv_serror_rate	Nümerik / H
19	num_access_files	Nümerik / İ	40	dst_host_rerror_rate	Nümerik / H
20	num_outbound_cmds	Nümerik / İ	41	dst_host_srv_rerror_rate	Nümerik / H
21	is_host_login	Nümerik / İ			

T: Her TCP bağlantısının temel özelliklerine ait alan
İ: Domain bilgisi önerilenbir bağlantıdaki içerik özelliklerine ait alan
AT: İki saniyelik zaman çerçevesi kullanarak çalışan ağ trafik özelliklerine ait alan
H: İki saniyeden fazla süren saldırıları değerlendiren host özelliklerine ait alan

Derin öğrenme uygulamasında kullanılan değerlerin nümerik bir matris olması gerektiğinden, sembolik değerler içeren ‘protocol_type’, ‘service’ ve ‘flag’ özniteliklerinin vektörel formata dönüştürülmesi gerekmektedir. Örneğin ‘protocol_type’ alanında ‘tcp’, ‘udp’ ve ‘icmp’ olmak üzere üç farklı sembolik değer bulunmaktadır ve bu değerlerin (1,0,0), (0,1,0) ve (0,0,1) binary vektör değerlerine dönüştürülmesi gerekmektedir. Benzer şekilde ‘service’ alanında 65, ‘flag’ alanında ise toplam 11 farklı sembolik değer bulunmakta ve bunların da ikili vektörlere dönüştürülmesi gerekmektedir. Dönüştürme işlemlerinin ardından 41 olan öznitelik sayısı 117’ye yükselmektedir.

NSL-KDD verisetindeki sembolik değerlerin nümerik vektörlere dönüştürülmesinin ardından, verisetindeki tüm değerlerin nümerikleştirme ve normalizasyona tabi tutulması gerekmektedir. Bu amaçla, veri setindeki “duration”, “src_bytes” ve “dst_bytes” gibi alanların maksimum ve minimum değerleri arasında büyük fark olması nedeniyle, logaritmik ölçekleme metodu kullanılarak daha dar aralıkta değerler elde edilmiştir. Bu işlem 0 ve 1’den farklı değerlere sahip özelliklere, $x = \log_{10}(x)$ şeklinde yeni değerlerin atanması ile yapılmıştır.

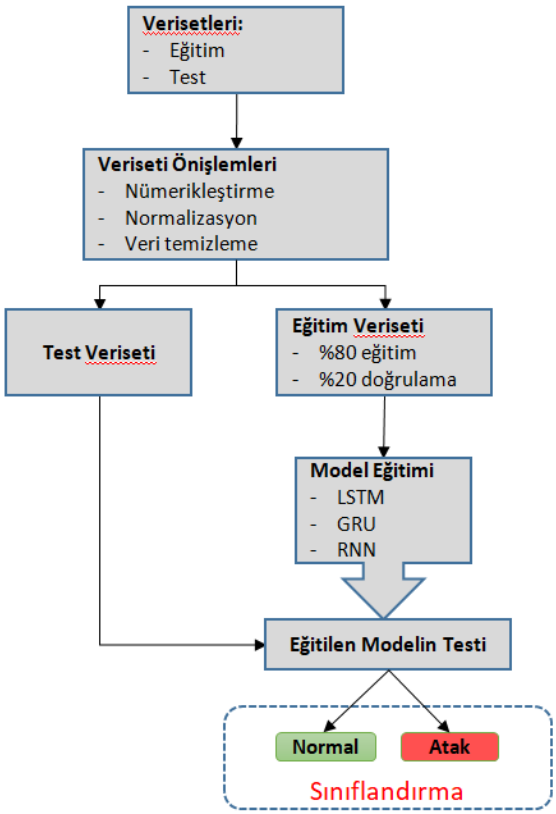
Modele girdi olan değerlerin büyük olması durumunda, katmanlardaki ağırlıkların hesaplanması zorlaşmakta ve zaman almaktadır. Bu nedenle logaritmik ölçekleme sonrası her bir değerlerin [0-1] aralığına haritalanması gerekmektedir. Bu amaçla aşağıdaki eşitlikten yararlanılarak yeni değerler elde edilmiştir. Eşitlikte yer alan Max ifadesi, her özelliğin maksimum değerine, Min ifadesi ise minimum değerine karşılık gelmektedir.

$$Xi = \frac{Xi - Min}{Max - Min} \tag{3.18}$$

İncelenen araştırmaların genelinde olduğu gibi tez çalışmamızda da verisetinin %80'i eğitim, %20'si doğrulama amacıyla kullanılmıştır. Bu yaklaşım derin öğrenme algoritmalarının eğitim ve doğrulama aşamalarında kullanılan standart bir bölütlemedir. Bu bölütleme, eğitim veri setinin algoritmanın öğrenmesi gereken desenleri içerdiği, doğrulama veri setinin ise algoritmanın öğrendiği desenleri değerlendirdiği anlamına gelir. Bu bölütleme yaklaşımı, algoritmanın aşırı uyuma (overfitting) eğilimini azaltmaya yardımcı olur. Eğitim veri setindeki desenleri öğrenen bir algoritma, eğitim veri setindeki örneklerin özelliklerini ezberleyebilir ve bu, başka veri setlerinde performansı düşürebilir. Doğrulama veri seti, algoritmanın eğitim veri setindeki örneklerin özelliklerini ezberlemiş olup olmadığını kontrol etmek için kullanılır. Bu nedenle, veri setinin %80'inin eğitim, %20'sinin doğrulama amacıyla kullanılması, yapay zeka algoritmalarının doğru bir şekilde eğitilmesi ve performanslarının doğru bir şekilde değerlendirilmesi açısından önemlidir.

3.5. Çözüm Mimarisi

Verisetinin belirlenmesini takiben model geliştirme çalışmalarına başlanmıştır. LSTM modeli başlangıçta, bir girdi, bir gizli ve bir de çıktı olmak üzere toplam üç katmandan oluşacak şekilde inşa edilmiştir. Performans testlerine bağlı olarak gizli katman sayısı arttırılıp azaltılabilmektedir. Girdi katmanı ve gizli katmanlarda bulunan ağ hücrelerinin tamamı LSTM hücresidir. Giriş katmanındaki hücre sayısı verisetinde bulunan özellik sayısı ile aynı olacak şekilde ayarlanabilmektedir. Katmanlardaki LSTM hücre sayıları başlangıçta 16 olarak belirlenmiş, model performans değerlendirmesine bağlı olarak değer belirlemesi yapılmıştır. Çıktı katmanı tek hücreden oluşmaktadır ve sınıflandırma işlemi bu katmanda gerçekleşmektedir. Oluşturulan çözüm mimarisinin genel akışı Şekil 3.23'te sunulmuştur.



Şekil 3.23. Geliştirilen çözüm mimarisi.

Model, Tesorflow ortamında Keras bileşenleri kullanılarak geliştirilmiştir. Bu kapsamda kullanılan fonksiyonlar aşağıda sunulmuştur.

- Sequential, modelin sıralı işlem yapacak şekilde tanımlanmasını,
- Add, LSTM ve dense hücrelerinin istenilen aktivasyon fonksiyonları ve hücre sayıları ile modele eklenmesini, ayrıca katmalar arası geçişlerde dropout işlemi yapılmasını,
- Compile, modelin eğitiminde kullanılan optimizer ve kayıp (loss) fonksiyon tiplerinin tanımlanmasını ve modelin derlenmesini,
- Fit, girdi ve çıktı değerlerine bağlı olarak belirlenen tekrar sayısı, doğrulama veri miktarı, batch size gibi hiperparametrelerle model eğitim işleminin yapılmasını,
- Evaluate, modelin test veriseti ile değerlendirilmesini sağlamaktadır.

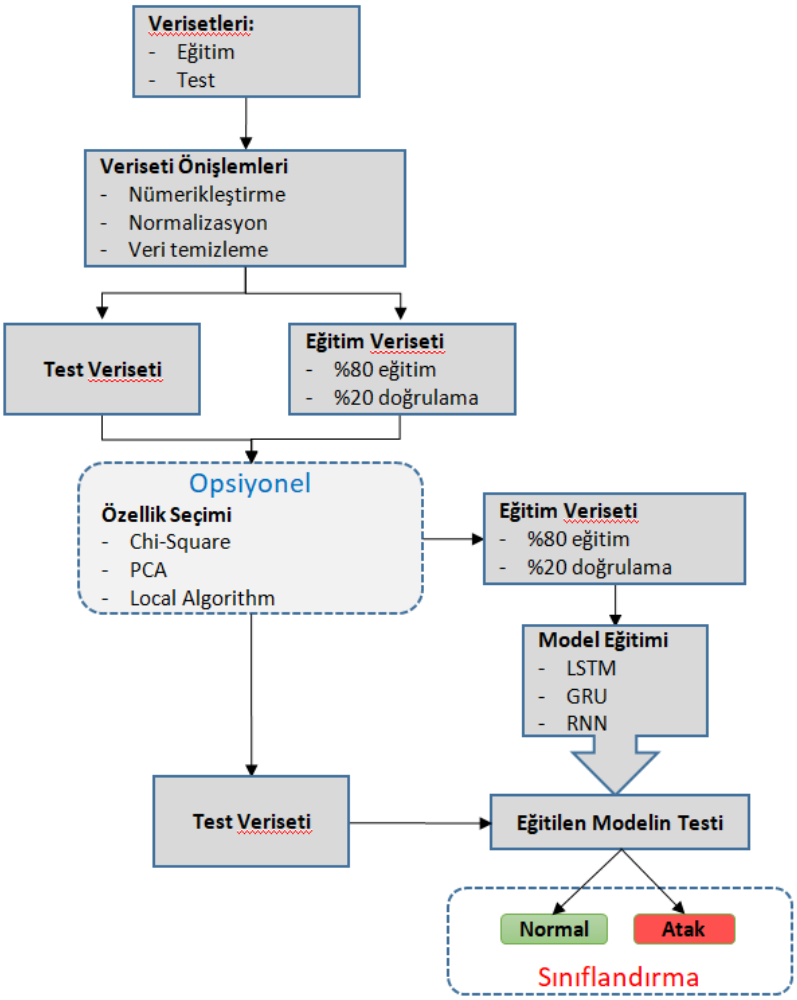
Model başarısının arttırılması amacıyla literatürde görülen çeşitli yöntemler denenmiştir. Bu yöntemlerden ilki K-fold çapraz doğrulama (cross validation) metodudur. Veri setini K parçaya bölmek modelin eğitim süresini K katına çıkarıp sürenin uzamasına neden olabilmektedir. Ancak, K-fold çapraz doğrulama, model performansını daha güvenilir bir

şekilde ölçmek için kullanılan yaygın bir tekniktir. Veri seti küçük olduğunda, modelin eğitimi için yeterli veriye sahip olunamayabilir. K-fold çapraz doğrulama, modelin performansının istatistiksel olarak anlamlı olmasını sağlar ve daha küçük veri setleri ile daha güvenilir bir model oluşturmaya yardımcı olur. Çünkü farklı veri bölümleri kullanılarak modelin performansı tekrar tekrar ölçülür ve sonuçların ortalaması alınır. Böylece overfitting'i önlemeye katkı sağlar. Overfitting, modelin eğitim verilerine çok fazla uyum sağlaması ve gerçek dünya verilerine uygulandığında kötü sonuçlar vermesidir. K-fold çapraz doğrulama, overfitting'i azaltarak modelin genelleştirilebilirliğini artırır.

K-fold çapraz doğrulama işleminde, eğitim veriseti K parçaya ayrılır. K değeri genelde 4 veya 5 olarak belirlenmektedir. Bu durumda model K defa oluşturularak her seferinde veri setinin farklı bölümleri eğitim ve doğrulama amacıyla kullanılır. Böylece modelin sürekli aynı veri ile eğitimi ve doğrulanması yerine, eğitim veriseti içerisindeki farklı bölümlerle eğitilmesi ve doğrulanması sağlanır. Neticede doğrulama skoru, üretilen K adet doğrulama skorunun ortalaması olmaktadır.

Literatürde model başarısına etki eden bir başka yöntemin de özellik seçimi olduğu görülmüştür. Alazab ve diğerleri (2012) tarafından gerçekleştirilen çalışmada [78], NSL-KDD verisetinde özellik azatlımı ile model performansının çalışma süresi ve kesinlik değeri açılarından arttığının gözlemlendiği belirtilmektedir. Veriseti içerisindeki özelliklerin, çok önemli, önemli ve önemsiz olarak sınıflandırıldığı vurgulanmaktadır. PCA (Principal Component Analysis) metodu ile özellik sayısının 12'ye düşürüldüğü, böylece model çalışma süresinin azaldığı, elde edilen kesinlik değerinin bir miktar arttığı ifade edilmektedir. Özellik seçimi ile, veri setindeki gereksiz özellikler çıkarılarak işlemci çalışma süresi ve bellek kullanımı azaltılmaktadır. Gereksiz veya benzer özelliklerin çıkarılması, algoritmanın doğru desenleri öğrenmesine ve overfitting eğiliminin azaltılmasına yardımcı olur. Ayrıca, modelin daha iyi genelleştirme performansı göstermesi ve farklı veri setleri üzerinde daha iyi sonuçlar vermesine yardımcı olmaktadır.

Tez çalışmasında önerilen çözüm mimarisinin veri önişleme bölümüne, opsiyonel özellik seçim aşaması eklenerek Şekil 3.24'te görülen mimari elde edilmiştir.



Şekil 3.24. Özellik seçim aşaması eklenerek elde edilen çözüm mimarisi.

3.6. Hiperparametre Kullanımı ve Model Üzerindeki Etkileri

Derin öğrenme uygulamalarında, veriseti boyutu, model katman sayısı, katmanlardaki hücre sayısı, dropout oranı, mini-batch-size, epoch (tekrar sayısı), optimizasyon algoritması, öğrenme oranı, momentum katsayısı, aktivasyon fonksiyonu gibi çeşitli hiperparametre değerleri kullanılmaktadır. Derin öğrenme uygulamalarında kullanılan hiperparametre değerleri, algoritmanın performansına ve modelin doğruluğuna direk etki etmektedir. Bu değerler, modelin yapılandırılmasına ve eğitilmesine yardımcı olur ve doğru hiperparametre değerleri seçilmezse, modelin performansı düşebilir. Yapılan incelemede, etkin bir derin öğrenme modeli için öncelikle doğru hiperparametre değerlerinin belirlenmesi gerektiği görülmüştür. Bu nedenle model geliştirme işleminin başlangıcında, modelde yer alan gizli katman sayısı ve epoch değeri düşük tutularak model çalışma süresi azaltılmış, defalarca test

işlemi gerçekleştirilerek doğru hiperparametre değerleri araştırılmıştır. Bu testler neticesinde elde edilen bilgiler aşağıda sunulmuştur.

Veriseti Boyutu: Model başarısına etki eden önemli hiperparametrelerden biri, veriseti boyutu ve veriseti içerisindeki veri çeşitliliğidir. Tez kapsamında geliştirilen uygulama performansının görülmesi amacıyla 125973 kayıt içeren $KDDTrain^{+}$ eğitim seti ve 22544 kayıt içeren $KDDTest^{+}$ test veriseti kullanılmıştır. Verisetleri önceki bölümde bahsedilen ön işlemlere tabi tutulmuştur. Model hiperparametrelerinin belirlenmesi aşamasında, eğitim verisetinin %80'i eğitim, %20'si doğrulama amacıyla kullanılmıştır. K-fold yönteminin uygulanması durumunda bu oran fold sayısına bağlı olarak değişmektedir.

Model Katman Sayısı: Katmanlar, diğer hiperparametre değerlerinin tespiti amacıyla başlangıçta bir giriş, bir gizli ve bir çıkış katmanı olarak belirlenmiştir. Yapılan testlerde katman sayıları artırılarak model başarısına etkileri gözlemlenmiştir. Katman sayısının çok arttırılması model karmaşıklık seviyesini arttırmakta ve modeli aşırı uydurmaya meyilli hale getirmektedir. Bu nedenle optimum katman ve hücre sayısının belirlenmesi gerekmektedir. Aşırı uydurmayı önleminin en etkin yollarından biri ağı küçültmek olduğundan, yapılan denemeler ile optimum katman ve hücre sayılarının, bir başka ifadeyle model kapasitesinin belirlenmesi gerekmektedir. Model kapasitesinin yüksek olması, modelin ezberleme gücünü arttıracak, tüm girdileri ve onlara ait hedefler arasındaki eşleştirmeyi öğrenecek, ancak genelleştirme gücü olmayacaktır.

Katmanlardaki Hücre Sayısı: Katmanlardaki LSTM hücre sayılarının sınırlı ölçüde arttırılması ile modelin problem çözme kabiliyetinin bir miktar geliştiği, ancak hücre sayısının çok arttırılması durumunda model kapasitesinin gerekenden fazla arttığı ve modeli ezberlemeye yönlendirdiği görülmüştür. Ayrıca LSTM hücre sayısının fazla olması, model eğitim süresini arttırmaktadır.

Hücre Seyreltme (Dropout) İşlemi: Belirlenmiş eşik değerlerinin altındaki LSTM hücrelerinin seyreltilmesinin (dropout), model eğitim seviyesini arttırdığı gözlemlenmiştir [79]. Bu kapsamda 0.2, 0.3, 0.4 ve 0.5 dropout değerleri ile model eğitilmiştir. Katman düğüm sayılarının artması durumunda dropout değerinin artmasının model eğitim seviyesine katkı sağladığı görülmüştür. Yapılan testlerde en fazla 32 düğümlü LSTM katmanları kullanılmış ve 0.2 dropout değeri ile en yüksek performans elde edilmiştir.

Ağırlık Düzenlileştirme Fonksiyonları: Tezin sonraki bölümlerinde de görüleceği üzere, büyük model ağlarında, eğitim işlemi esnasında model kayıp oranının hızla azalması veya artması, modelin düzensiz ve aşırı uydurmaya meyilli olduğuna işaret etmektedir. Bunu önlemenin bir yolu model kapasitesini azaltmaktır. Aşırı uydurmayı engellemenin bir başka yolu ise, ağa eklenen kısıtlamalar ile ağırlık değerlerinin düşük olmaya zorlanması ve bu sayede ağırlık dağılımının düzenli hale getirilmesidir. Böylece, aşırı öğrenmeye bağlı oluşan ani kayıp oranı düşüşünün önüne geçilir ve modelin daha düzenli öğrenmesi sağlanır. Bu işlem *ağırlık düzenlileştirme* olarak adlandırılır. Bu işlemin amacı kayıp fonksiyonuna bir maliyet ekleyerek büyük değerler almasını engellemektir. Maliyet ekleme işlemi L_1 ve L_2 olmak üzere iki şekilde gerçekleştirilmektedir. L_1 düzenlileştirmede maliyet ağırlıkları, katsayılarının mutlak değerlerine oransal olarak eklenir. L_2 düzenlileştirmede ise maliyet ağırlıkları, katsayılarının karelerine oransal olarak eklenir. Çalışma kapsamında L_2 düzenlileştirme kullanılmıştır.

Mini-batch-size: Model tarafından aynı anda işlenebilen kayıt sayısı mini-batch-size ile ifade edilmektedir. Kayıt, verisetinde bulunan her bir satırı ifade etmektedir. Test çalışmalarına CPU donanıma sahip bir bilgisayar üzerinde başlanmıştır. Bu bilgisayarda yapılan testlerde, mini-batch-size değerinin yüksek tutulması durumunda, veri işleme hızının oldukça düştüğü görülmüştür. Ancak model, GPU donanıma sahip bilgisayarda çalıştırıldığında veri işleme hızının arttığı görülmüştür. Mini-batch-size değerinin düşük olması durumunda modelin eğitimi esnasında üretilen hata oranında dalgalanmalar olduğu gözlemlenmiştir. Bunun, batch-size kadar alınıp işlenen verinin aynı dağılıma sahip olmamasından kaynaklandığı değerlendirilmiştir. Bahse konu dalgalanmanın eğitim işlemlerinin başında daha yüksek olduğu, sonlara doğru azaldığı görülmüştür.

Tekrar Sayısı (Epoch): Bir diğer önemli hiperparametre ise epoch değeridir. Bu değer, eğitim ve doğrulama verisetleri üzerinde eğitim ve doğrulama işlemlerinin kaç defa gerçekleştirileceğini belirlemektedir. Bilindiği üzere her eğitim tekrarından sonra model doğrulanır ve tespit edilen başarı değerine göre geriye yayılım işlemi uygulanarak ağırlık değerleri güncellenir. Her eğitim tekrarında bir önceki adımda elde edilen ağırlık değerlerine bağlı olarak eğitim ve doğrulama yapılarak ağırlık değeri güncelleme işlemi tekrarlanır. Neticede model için en uygun ağırlık değerleri hesaplanır. Model test işlemlerine düşük tekrar sayısı ile başlanmıştır. Tekrar sayısı arttıkça modelin başarı düzeyinin arttığı görülmüştür. Özellikle başlangıçta eğitim ve doğrulama başarı düzeyinin hızla arttığı, belli

bir seviyeye (%90 üzeri kesinlik) gelindikten sonra daha yavaş bir artış olduğu görülmüştür. Bazı durumlarda ise belirli bir tekrar sayısından sonra başarı düzeyinin aşırı uydurmaya bağlı olarak belirgin seviyede azaldığı görülmüştür.

Kayıp (Amaç) Fonksiyonu: Model eğitimi esnasında en küçültülme işleminin uygulandığı fonksiyondur. Belirlenen sınıflandırma işlemi için başarının ölçülmesini sağlar. Amaç fonksiyonu her problem tipi için farklılık gösterir ve doğru amaç fonksiyonunu seçmek model başarısına önemli ölçüde etki eder. Keras'ta aşağıdaki amaç fonksiyonları sıklıkla kullanılmaktadır.

- MSE (Mean Squared Error)
- MSLE (Mean Squared Logarithmic Error)
- MAE (Mean Absolute Error)
- MAPE (Mean Absolute Percentage Error)
- KLD (Kullback-Leibler Divergence)
- Binary Crossentropy
- Categorical Crossentropy
- Categorical Hinge
- Cosine
- Cosine Proximity
- Deserialize
- Logcosh
- Poisson
- Squared Hinge

Yapılan testler neticesinde, model derleme işleminde en iyi sonucu veren kayıp fonksiyonunun *binary_crossentropy*, en kötü sonuç veren kayıp fonksiyonunun ise *MSLE* olduğu gözlemlenmiştir. Geliştirilen uygulamanın amacının bir kaydın özellik değerlerini hesaplayarak bunun saldırı veya normal olduğunu belirlemek olduğu göz önünde bulundurulduğunda, ele alınan problemin bir ikili sınıflandırma problemi olması nedeniyle *binary_crossentropy* kayıp fonksiyonunun en iyi sonucu verdiği değerlendirilmiştir. *Binary_crossentropy* fonksiyonu bilgi kuramından gelmekte ve olasılıklar arası uzaklığı

ölçmektedir. Ağ sızma tespit probleminde bu ölçüm, tahmin edilen ile olması gereken dağılım arasındaki uzaklıktır.

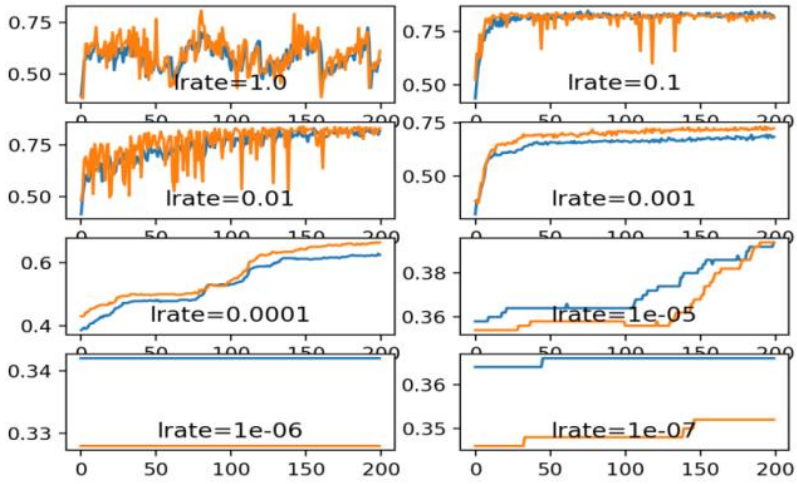
Optimizasyon Algortiması: Önemli hiperparametrelerden biri de optimizasyon algoritmasıdır. Optimizasyon algoritması kayıp fonksiyonuna göre ağın nasıl güncelleneceğini belirler ve modele stokastik gradyan inişi işlemini uygular. Yapılan inceleme neticesinde model derleme işleminde aşağıda belirtilen optimizer tiplerinin sıklıkla kullanıldığı görülmüştür [80].

- ADAM (Automated Digital Analysis and Measurement)
- ADADELTA
- ADAGRAD
- ADAMAX
- NADAM (Nesterov Momentum ADAM)
- RMSPROP (Root Mean Square)
- SGD (Stochastic gradient descent)

Derin öğrenme mimarisinde kullanılan modellerin eğitimi temelde bir optimizasyon problemidir. Bu nedenle kullanılan optimizasyon algoritmasının seçimi önemlidir. Yapılan testlerde yukarıda belirtilen optimizasyon algoritmaları kullanılarak sonuçlar elde edilmiştir. Testler neticesinde, *rmsprop* optimizasyon algoritmasının en iyi sonucu verdiği görülmüştür. Bu nedenle devam eden testlerde *rmsprop* optimizasyon algoritması kullanılmıştır.
Optimizasyon algoritmalarının başarısına; öğrenme oranı, momentum, learning rate decay gibi faktörler etki etmektedir. Örneğin *rmsprop* optimizasyon algoritmasında kullanılan default değerler aşağıda görülmektedir.

opt=Rmsprop(lr=0.002, beta_1=0.9, beta_2=0.99, epsilon=None, schedule_decay=0.004)

Öğrenme Oranı: Optimizasyon algoritması içerisinde tanımlanan öğrenme oranının optimum değerinin belirlenmesi model başarısı için önemlidir. Optimizasyon algoritması parametre değerlerinin çoklu sınıflandırma problemine etkisini belirlemek için yapılan bir çalışmaya ait grafikler Şekil 3.25'te sunulmuştur [81].



Şekil 3.25. Öğrenme oranı değerine bağlı kesinlik değerleri grafik tablosu [81].

Tablonun x ekseni epoch değerlerini, Y ekseni ise modelin sınıflandırma kesinlik oranını göstermektedir. Öğrenme oranı değerinin 0,1, 0,01 ve 0,001 olarak belirlendiği durumlarda, eğitim ve test kesinlik oranlarının arttığı görülmektedir. En iyi sonucun ise 0,001 öğrenme oranı değerinde alındığı görülmektedir.

Gerçekleştirilen testlerde öncelikle rmsprop optimizasyon algoritmasının default değerleri kullanılarak sonuçlar elde edilmiştir. Daha sonra öğrenme oranı 0,01 olarak belirlenerek 50 tekrarda sonuçlar alınmıştır. Neticede, algoritmanın default değerlerinin daha iyi sonuç verdiği görülmüştür.

Momentum: Yapılan incelemede, sınıflandırma problemlerinde momentum değerinin 0.9 veya 0.99 olduğu durumlarda başarılı sonuçlar elde edildiği tespit edilmiştir. Bu nedenle testlerde bu değerler kullanılmıştır.

Aktivasyon Fonksiyonu: Model performansına etki eden bir diğer önemli hiperparametre aktivasyon fonksiyonudur. LSTM tabanlı IDS uygulamasında aktivasyon fonksiyonlarının giriş ve gizli katmalarda bulunan LSTM hücrelerinde, model derleme işleminde ve çıkış katmanı olan dense hücresinde kullanıldığı görülmektedir. Keras'ta kullanılan aktivasyon fonksiyonları aşağıda sunulmuştur [82].

- Softmax *(keras.activations.softmax(x, axis= -1))*
- ELU (Exponential Linear Unit) *(keras.activations.elu(x, alpha=1.0))*

- SELU (Scaled Exponential Linear Unit) *(keras.activations.selu(x))*
- Softplus *(keras.activations.softplus(x))*
- Softsign *(keras.activations.softsign(x))*
- RELU *(Rectified Linear Unit) (keras.activations.relu(x, alpha=0.0, max_value=None, threshold=0.0))*
- Tanh (Hyperbolic tangent) *(keras.activations.tanh(x))*
- Sigmoid *(keras.activations.sigmoid(x))*
- Hard_sigmoid *(keras.activations.hard_sigmoid(x))*
- Exponential *(keras.activations.exponential(x))*
- Linear *(keras.activations.linear(x))*

Yapılan testlerde, LSTM hücrelerinde *relu*, dense hücresinde ise *sigmoid* fonksiyonunun en iyi sonucu verdiği tespit edilmiştir. Buna karşı dense hücresinde *softmax* fonksiyonunun en kötü sonucu verdiği görülmüştür.

3.7. Geliştirilen Modelin Testi ve Elde Edilen Sonuçlar

Test işlemleri Intel Core i7-10510U 2,3 GHz işlemci, 16 GB RAM, 64 bit Windows 10 işletim sistemine sahip bilgisayar üzerinde icra edilmiştir. Uygulama geliştirme işlemleri Anaconda Navigator platformunda Tensorflow altyapısında Spyder Python geliştirme ortamında Python programlama dilinde yapılmıştır. Test sonuçlarını içeren çizelgelerin ilk sütununda her test için kullanılan hiperparametre değerleri; ikinci, üçüncü ve dördüncü sütunlarda sırasıyla, elde edilen kesinlik (accuracy), hassasiyet (precision) ve duyarlılık (recall) değerleri (%), son sütunda ise eğitim ve doğrulama işlemlerine ait "tekrar sayısı – kesinlik" grafiği görülmektedir.

İşlem süresinin kısaltılması amacıyla model test işlemlerine katman sayısı, katmanlardaki hücre sayısı ve eğitim tekrar sayıları düşük tutularak başlanmıştır. Bu şekilde yapılan test işlemleri neticesinde elde edilen sonuçlara bağlı olarak, LSTM aktivasyon fonksiyonu relu, dropout oranı 0.2, dense aktivasyon fonksiyonu sigmoid, optimizasyon algoritması rmsprop, kayıp fonksiyonu binary_crossentropy hiperparametre değerlerinin en başarılı sonuçları verdiği tespit edilmiş ve sonraki testlerin tamamında bu değerler sabit tutulup diğer hiperparametre değerleri değiştirilerek sonuçlar elde edilmiştir. Bununla birlikte giriş

katmanında yer alan hücre sayısı, kullanılan verisetlerinin özellik sayısı olarak belirlenmiştir. Ayrıca giriş katmanı, katmanlar sayılırken hesaba katılmamıştır. Bir başka ifadeyle tabloda 6 katman olması durumu, 1 giriş katmanı ve buna ilave 6 katman anlamına gelmektedir. Eğitim verisetinin %80'i eğitim, %20'si ise doğrulama amacıyla kullanılmıştır. Testlerde kullanılan sabit hiperparametre değerleri Çizelge 3.10'da sunulmuştur.

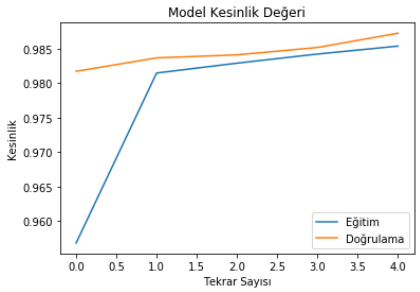
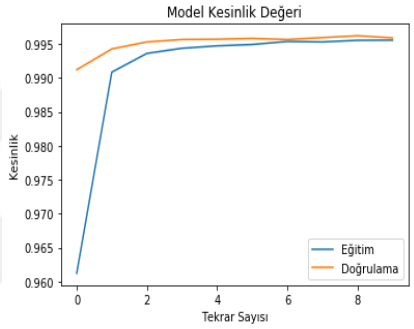
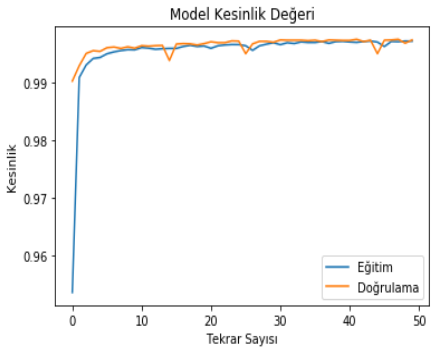
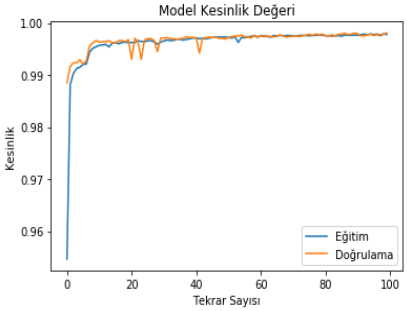
Çizelge 3.10. Testlerde kullanılan sabit hiperparametre değerleri.

Hiperparametre	Değer	Hiperparametre	Değer
Dropout	*0,2*	*Kayıp fonksiyonu*	*Binary_crossentropy*
LSTM aktivasyon fonksiyonu	*Relu*	*Eğitim kayıt oranı (%)*	*80*
Dense aktivasyon fonksiyonu	*Sigmoid*	*Doğrulama kayıt oranı (%)*	*20*
Model optimizer	*Rmsprop*	*Giriş katmanı hücre sayısı*	*Veriseti özellik sayısı*

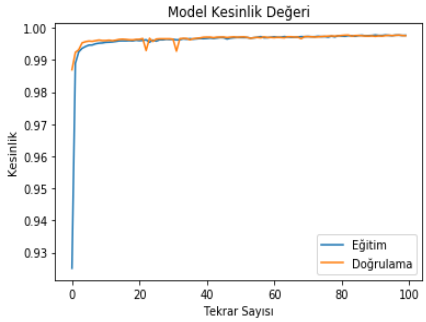
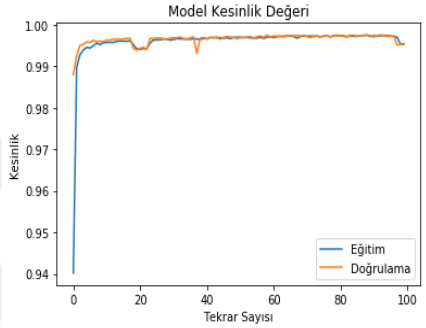
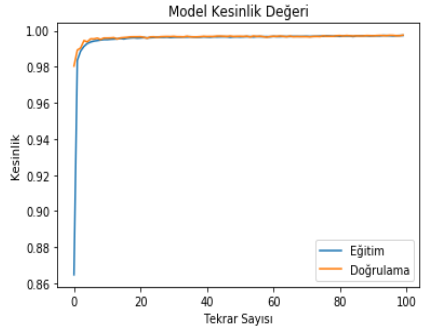
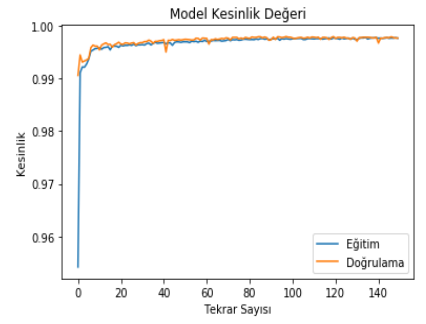
Yapılan ilk testte, katman sayısı 3 ile sınırlandırılmış, gizli katmanlardaki hücre sayısı 8 olarak belirlenmiştir. Toplam 5 eğitim tekrarı ve 128 batch size ile yapılan test neticesinde Çizelge 3.11'de görülen sonuçlar elde edilmiştir. Katman, hücre ve tekrar sayısı değerlerinin bir miktar arttırılması neticesinde model başarısı gözlemlenmiş, eğitim kesinlik değerinin bir miktar, test kesinlik değerinin ise belirgin şekilde arttığı görülmüştür. Diğer değerlerde belirgin bir değişiklik olmamış, doğrulama hassasiyet değerinde ihmal edilebilir bir azalma olmuştur.

Bu noktadan sonra katman, hücre ve tekrar sayıları ile batch size değerinin arttırılmasına bağlı olarak eğitim ve doğrulama kesinlik değerlerinde belirgin bir artış olmuş, test kesinlik değerinde ise beklenmeyen şekilde azalma gerçekleşmiştir. Hassasiyet ve geri çağırma değerleri artmıştır. Sonraki aşamada katman sayısı 6, hücre sayısı 32, tekrar sayısı 100 ve batch size 256 olduğu durumda en yüksek kesinlik değerlerine ulaşılmıştır. Bu testin grafiği incelendiğinde, 20'nci ve 40'ıncı tekrar civarında özellikle doğrulama grafiğinde beklenmeyen sapmalar belirmiş, yaklaşık 45'inci tekrardan sonra grafiğin stabil hale geldiği görülmüştür. Bu aşamada elde edilen en yüksek test kesinlik değeri %80,47, hassasiyet oranı %92,84 ve duyarlılık oranı %71,19 olarak tespit edilmiştir.

Çizelge 3.11. LSTM Modelinin Çeşitli Hiperparametre Değerleri ile Testi.

Hiperparametreler	Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
*Katman Sayısı : **3*** *Katmanlardaki Hücre Sayısı: **8*** *Tekrar Sayısı: **5*** *Batch_size: **128***	Eğitim: 98,54 Doğrulama: 98,72 Test: 77,88	Eğitim: 98.93 Doğrulama: 99,20 Test: 92,36	Eğitim: 97.91 Doğrulama: 98,05 Test: 66,08	
*Katman Sayısı : **4*** *Katmanlardaki Hücre Sayısı: **16*** *Tekrar Sayısı: **10*** *Batch_size: **128***	Eğitim: 99,55 Doğrulama: 99,59 Test: 79,75	Eğitim: 99.66 Doğrulama: 99,59 Test: 92,66	Eğitim: 99.38 Doğrulama: 99,52 Test: 69,96	
*Katman Sayısı : **5*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **50*** *Batch_size:**256***	Eğitim: 99,72 Doğrulama: 99,74 Test: 79,29	Eğitim: 99.82 Doğrulama: 99,85 Test: 92,79	Eğitim: 99.58 Doğrulama: 99,60 Test: 68,99	
*Katman Sayısı : **6*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **100*** *Batch_size: **256***	Eğitim: 99,78 Doğrulama: 99,81 Test: **80,47**	Eğitim: 99.84 Doğrulama: 99,86 Test: 92,84	Eğitim: 99.68 Doğrulama: 99,74 Test: 71,19	

Çizelge 3.11. (Devamı) LSTM Modelinin Çeşitli Hiperparametre Değerleri ile Testi.

Hiperparametreler	Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
*Katman Sayısı : **7*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **100*** *Batch_size:**512***	Eğitim: 99,77 Doğrulama: 99,76 Test: 80,10	Eğitim: 99.85 Doğrulama: 99,80 Test: 92,66	Eğitim: 99.66 Doğrulama: 99,69 Test: 70,63	
*Katman Sayısı : **8*** *Katmanlardaki Hücre Sayısı: **64*** *Tekrar Sayısı: **100*** *Batch_size: **512***	Eğitim: 99,54 Doğrulama: 99,53 Test: 79,10	Eğitim: 99.84 Doğrulama: 99,81 Test: 92,55	Eğitim: 99.18 Doğrulama: 99,17 Test: 68,83	
*Katman Sayısı : **8*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **100*** *Batch_size:**1024***	Eğitim: 99,73 Doğrulama: 99,75 Test: 78,86	Eğitim: 99.81 Doğrulama: 99,81 Test: 92,81	Eğitim: 99.61 Doğrulama: 99,66 Test: 68,15	
*Katman Sayısı : **6*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **150*** *Batch_size:**256***	Eğitim: 99,77 Doğrulama: 99,77 Test: 79,85	Eğitim: 99.87 Doğrulama: 99,85 Test: 92,68	Eğitim: 99.63 Doğrulama: 99,63 Test: 70,14	

Model katman sayısı artışının model karmaşıklığını arttırdığı, eğitim ve test süresini uzattığı görülmüştür. Katmanlardaki hücre sayılarının artışı da benzer etkiyi yaratmaktadır. Model katman sayısının ve hücre sayısının arttırılması model başarı oranını bir miktar arttırmıştır. Ancak katman ve hücre sayılarının çok artması durumunda aşırı uydurma durumlarının

meydana geldiği görülmüştür. Yapılan denemelerde model için katmanlardaki optimum hücre sayısının 32 olduğu tespit edilmiştir.

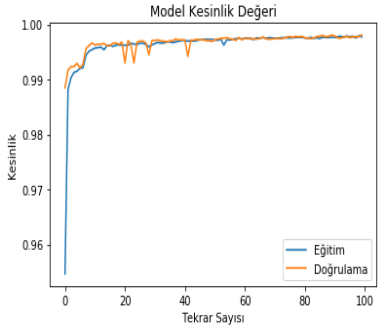
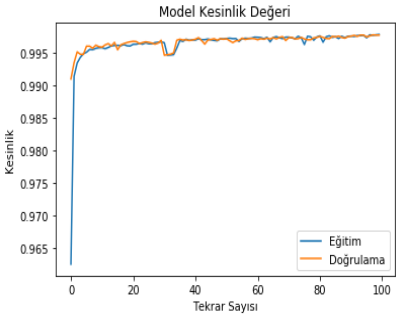
Aktivasyon fonksiyonu, optimizasyon algoritması ve kayıp fonksiyonunun model başarısına direk etki ettiği belirlenmiştir. Örneğin, dense katmanındaki aktivasyon fonksiyonunun *softmax* olması durumunda model başarı kesinlik değerinin yaklaşık %65 seviyesine gerilediği gözlemlenmiştir. Batch size değerinin arttırılmasının model çalışma süresini kısalttığı, model başarı oranına ise olumsuz etkisi olduğu görülmüştür.

Yapılan testler esnasında, eğitim ve doğrulama sonuçlarının parametre değerlerine göre iyileştiği, ancak test sonuçlarının arzu edilen stabiliteye ulaşmadığı gözlemlenmiştir. Bunun nedeninin eğitim ve test verisetlerindeki kayıtlar arasındaki uyumsuzluktan kaynaklanabileceği düşünülmüştür. Bu nedenle veriseti incelenerek gereksiz özelliklerin çıkarılması yoluna gidilmiştir. Bu kapsamda öncelikle eğitim ve test verisetleri içerisindeki aşağıda belirtilen alanlarda yer alan tüm değerlerin "0" olduğu tespit edilmiş ve bu özelliklere ait alanlar verisetlerinden çıkarılmıştır.

- Service 30 – "4_443"
- Service 42 – "host6s"
- Service 50 – "day42"
- Service 65 – "4_8001"
- Num_outbound_cmds

Bununla birlikte test verisetindeki değerler ile ilgili bir literatür incelemesi yapılmıştır. Neticede normalizasyon işleminde kullanılan maksimum ve minimum değerlerin $KDDTrain^{+}$ ve $KDDTest^{+}$ verisetlerinde ayrı ayrı belirlenmesinin model başarı seviyesine etki ettiği görülmüş ve bu değerler daha kapsamlı veriseti olan $KDDTrain^{+}$ verisetine göre belirlenmiştir. Bu işlemden sonra verisetine yönelik yukarıdaki tabloda görülen en başarılı testte kullanılan hiper-parametreler ile test işlemleri tekrarlanmıştır. Veriseti üzerinde değişiklik yapılmadan önceki sonuçlar ile işlemler sonrası elde edilen sonuçlar Çizelge 3.12'de sunulmuştur.

Çizelge 3.12. Verisetlerindeki gereksiz verinin çıkarılması ve ortak normalizasyon sonrası elde edilen sonuçlar.

Veriseti Üzerinde İşlem Yapılmadan Önceki Sonuçlar				
Hiperparametreler	Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
*Katman Sayısı : **6*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **100*** *Batch_size: **256***	Eğitim: 99,78 Doğrulama: 99,81 Test: **80,47**	Eğitim: 99.84 Doğrulama: 99,86 Test: 92,84	Eğitim: 99.68 Doğrulama: 99,74 Test: 71,19	
Veriseti Üzerinde İşlem Yapıldıktan Sonraki Sonuçlar				
*Katman Sayısı : **6*** *Katmanlardaki Hücre Sayısı: **32*** *Tekrar Sayısı: **100*** *Batch_size: **256***	Eğitim: 99,79 Doğrulama: 99,81 Test: **80,55**	Eğitim: 99.87 Doğrulama: 99,86 Test: 92,88	Eğitim: 99.68 Doğrulama: 99,78 Test: 71,56	

Eğitim ve doğrulama kesinlik değeri grafikleri incelendiğinde, veriseti üzerinde işlem yapıldıktan sonra üretilen grafikte belirgin sapmaların olmadığı 35 ile 40'ıncı tekrarlar arasında iki bölümde de düşüş meydana geldiği, eğitim ve doğrulama kesinlik değerlerinde belirgin bir değişiklik olmadığı, test kesinlik değerinde ise bir miktar artış olduğu görülmektedir. Ayrıca test geri çağırma değerinin de bir miktar arttığı gözlemlenmiştir.

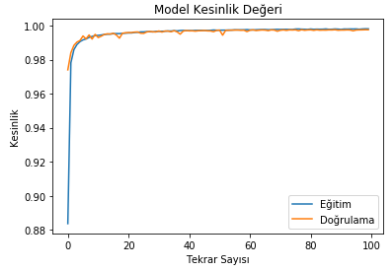
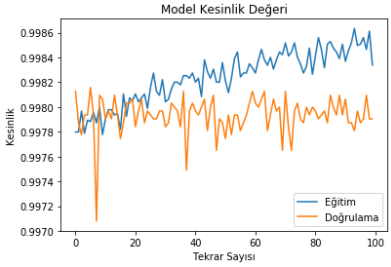
K-fold yönteminin uygulanması ile elde edilen sonuçlar

Yapılan test işlemleri neticesinde, eğitim ve doğrulama kesinlik değerlerinin oldukça yüksek olduğu, ancak test sonuçlarının bu seviyede olmadığı gözlemlenmiştir. Yapılan incelemede, KDDTrain^{+} verisetinin nispeten küçük boyutlu olmasının, modelin arzu edilen eğitim seviyesine ulaşamamasına neden olabileceği sonucu çıkarılmıştır. Sorunun giderilmesi

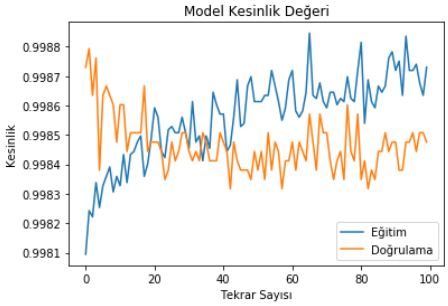
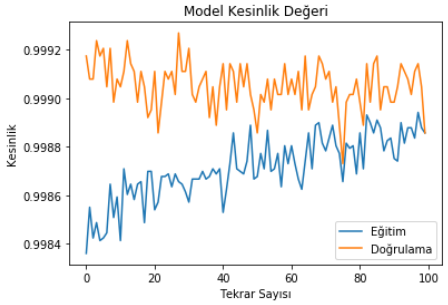
amacıyla, eğitim seti içerisindeki farklı bölümlerde yer alan verilerden yararlanılarak eğitim ve doğrulama yapılmasını, böylece modelin genelleştirilebilirliğini artırarak, modelin gerçek dünya verilerinde daha iyi performans göstermesini sağlayan K-fold yöntemi modele eklenmiştir. Eğitim ve doğrulama işlemleri dört farklı bölümde, en yüksek sınıflandırma sonuçlarının elde edildiği hiperparametre değerleri ile yapılmıştır.

K-fold yönteminin eklenmesi neticesinde yapılan test işlemleri neticesinde, model başarısında belirgin bir artış olduğu gözlemlenmiştir. Çalışmada elde edilen en yüksek sonuçlara göre kıyaslandığında kesinlik değerinin yaklaşık %1,5, hassasiyet değerinin ise yaklaşık %4 oranında artış gösterdiği gözlemlenmiştir. İlk katmanda %80,97 olarak elde edilen kesinlik değerinin son katmana kadar belirli seviyede artış gösterdiği, son katmanda ise bir düşüş yaşandığı görülmektedir. Bunun nedeni, eğitilen ve doğrulama yapılan katmanlarda bulunan verinin, test veriseti ile benzeşmesine bağlı olduğu değerlendirilmiştir. K-fold eğitim işlemleri neticesinde elde edilen sonuçlar Çizelge 3.13'te paylaşılmıştır.

Çizelge 3.13. K-fold eğitim işlemleri neticesinde elde edilen sonuçlar.

Katman	Test Sonuçları	Eğitim/Doğrulama Kesinlik Grafiği
1	Test Kesinlik: 80.97% Test Hassasiyet: 97.08% Test Duyarlılık: 68.63%	
2	Test Kesinlik: 82.82% Test Hassasiyet: 97.13% Test Duyarlılık: 71.94%	

Çizelge 3.13. (Devamı) K-fold eğitim işlemleri neticesinde elde edilen sonuçlar.

Katman	Test Sonuçları	Eğitim/Doğrulama Kesinlik Grafiği
3	Test Kesinlik: 85.22% Test Hassasiyet: 97.33% Test Duyarlılık: 76.12%	
4	Test Kesinlik: 81.71% Test Hassasiyet: 93.06% Test Duyarlılık: 73.34%	
Test Sonuç Ortalamaları Test Kesinlik (Accuracy) Değeri Ortalaması:% **82,68** Test Hassasiyet (Precision) Ortalaması : % 96,15 Test Duyarlılık (Recall) Ortalaması: % 72,51		

Katmanlara ait grafikler göz önünde bulundurularak eğitim ve doğrulama sonuçlarına yönelik yapılan değerlendirme aşağıda sunulmuştur.

- İlk grafikte, eğitim için yaklaşık %88, doğrulama için ise yaklaşık %98'den başlayan kesinlik değerlerinin 1'e doğru yakınsadıkları ve hemen hemen aynı ölçüde artış gösterdikleri görülmektedir. Modelin ilk eğitim ve doğrulama aşamasında olması nedeniyle öğrenilmiş herhangi bir geçmişi olmamasından kaynaklanan bir durum olduğu değerlendirilmektedir.
- İkinci grafikte, model ilk katmandaki eğitim bilgisi ile ikinci katmana geçmekte, eğitim ve doğrulama seviyesi yaklaşık %99,78'den başlamaktadır. Eğitim seviyesinde düzenli bir artış görülmekte iken test seviyesinin dalgalanmalara karşın düz seyrettiği görülmektedir. Bu katmanda sistemin öğrenmeye devam ettiği, doğrulamanın ise ortalama %99,80 seviyelerinde önceki katmana benzer seviyede olduğu anlaşılmaktadır.
- Üçüncü grafikte, model eğitim seviyesinin %99,81'den başlayarak arttığı, doğrulamanın ise başlangıçta %99,88 seviyesinde iken zamanla ortalama %99,85 seviyesine indiği

görülmektedir. Bu katmanda da eğitim seviyesinde düzenli bir artış ile karşılaşılmış ve sistemin öğrenmeye devam ettiği sonucu çıkarılmıştır. Doğrulamanın ise önceki katmana benzer şekilde ancak daha yüksek oranda ortalama %99,85 seviyelerinde devam ettiği görülmüş, doğrulama oranının her katmanda bir miktar artış gösterdiği anlaşılmıştır.

- Dördüncü grafikte, model eğitim seviyesinin bir miktar artışla %99,84 seviyesinden başladığı, düzenli ortalama bir artışla yaklaşık %99,88 seviyesine çıktığı görülmektedir. Doğrulama oranının ise yaklaşık %99,91 seviyelerinde seyrettiği, öğrenmeye bağlı olarak doğrulama oranında da bir artış olduğu görülmektedir.

K-fold yöntemi kullanılırken optimum katman sayısını belirlemek için çeşitli yaklaşımlar mevcuttur. Önceden belirlenmiş standart değerleri kullanmak genelde kabul gören bir yaklaşımdır ancak kullanılan veri setinin boyutu küçükse katman sayısını azaltmak daha uygun olabilir. Bu nedenle farklı katman sayılarını deneyerek her bir katman sayısı için modelin performansını ölçmek zaman alıcı olmasına karşı en doğru sonucun alınmasını sağlamaktadır. Bu doğrultuda katman sayısının model performansına etkisini gözlemlemek için yapılan testlere ait sonuçlar Çizelge 3.14'te sunulmuştur.

Çizelge 3.14. Farklı katman sayıları ile yapılan testlere ait sonuçlar.

<table>
<tr><td colspan="2">5 Katman ile Test</td><td colspan="2">6 Katman ile Test</td></tr>
<tr><td>Katman No.</td><td>Sonuçlar</td><td>Katman No.</td><td>Sonuçlar</td></tr>
<tr><td>1</td><td>Test Kesinlik: 78.99%
Test Hassasiyet: 92.67%
Test Duyarlılık: 68.51%</td><td>1</td><td>Test Kesinlik: 81.65%
Test Hassasiyet: 97.06%
Test Duyarlılık: 69.89%</td></tr>
<tr><td>2</td><td>Test Kesinlik: 81.05%
Test Hassasiyet: 95.05%
Test Duyarlılık: 70.37%</td><td>2</td><td>Test Kesinlik: 83.81%
Test Hassasiyet: 97.23%
Test Duyarlılık: 73.66%</td></tr>
<tr><td>3</td><td>Test Kesinlik: 80.66%
Test Hassasiyet: 92.86%
Test Duyarlılık: 71.53%</td><td>3</td><td>Test Kesinlik: 82.08%
Test Hassasiyet: 93.10%
Test Duyarlılık: 74.00%</td></tr>
<tr><td>4</td><td>Test Kesinlik: 79.58%
Test Hassasiyet: 92.53%
Test Duyarlılık: 69.76%</td><td>4</td><td>Test Kesinlik: 80.94%
Test Hassasiyet: 92.97%
Test Duyarlılık: 71.96%</td></tr>
<tr><td>5</td><td>Test Kesinlik: 78.35%
Test Hassasiyet: 92.35%
Test Duyarlılık: 67.57%</td><td>5</td><td>Test Kesinlik: 81.80%
Test Hassasiyet: 93.00%
Test Duyarlılık: 73.57%</td></tr>
<tr><td colspan="2" rowspan="2"><u>Ortalama Sonuçlar</u>
Test Kesinlik: % 79,26
Test Hassasiyet:% 93,92
Test Duyarlılık: % 69,54</td><td>6</td><td>Test Kesinlik: 81.73%
Test Hassasiyet: 92.93%
Test Duyarlılık: 73.51%</td></tr>
<tr><td colspan="2"><u>Ortalama Sonuçlar</u>
Test Kesinlik: % 82.01
Test Hassasiyet: % 94.38
Test Duyarlılık: % 72.76</td></tr>
</table>

Çizelge 3.14'te görüldüğü gibi beş katman ile eğitim ve test işlemleri neticesinde, dört katman ile gerçekleştirilen test sonuçlarına göre performansta bir düşüş olduğu görülmektedir. Altı katman ile gerçekleştirilen eğitim ve test işlemleri neticesinde ise performansta 5 katman ile yapılan teste oranla bir miktar artış olduğu ancak 4 katman ile yapılan test sonuçları seviyesine çıkılamadığı görülmektedir.

Düzenlileştirme işlemlerinin uygulanması ile elde edilen sonuçlar

Model performansına yönelik yapılan inceleme neticesinde, belirli bir tekrar değerine kadar model eğitim performansında hızlı bir artış olduğu ve bir süre sonra bu artışın oldukça yavaşladığı, hatta azalmanın başladığı görülmüştür. Bu tespitin ardından, bir yapay zekâ problemindeki en önemli konulardan olan en iyileme ve genelleme problemlerine odaklanılmıştır.

En iyileme, bir fonksiyonun en küçük veya en büyük değerini bulmak için yapılan bir optimizasyon işlemidir. Model eğitimi açısından bakıldığında, bir amaç fonksiyonunun en iyi sonucunu veren parametre değerlerini bulmak için kullanılır. Yapay zeka algoritması genellikle çok boyutlu ve karmaşık bir uzayda çalıştığından, doğru optimizasyon algoritması ve parametre seçimi, model başarısı için kritik öneme sahiptir.

Genelleştirme, bir modelin öğrendiklerini, veri seti dışındaki yeni verilerde de doğru bir şekilde uygulayabilme yeteneğidir. Genelleştirme, bir modelin aşırı uyumu (overfitting) engellemesi ve gerçek hayattaki verilerde iyi performans göstermesi açısından önemlidir. Eğitim veri setinde yüksek doğruluk oranları elde eden bir model, test veri setinde de yüksek doğruluk oranları gösterirse, genelleştirme yeteneği yüksektir.

Öğrenme işleminin başında, eniyileme ve genelleştirme arasında bir bağ mevcuttur. Eğitim verisetinden elde edilen düşük kayıp değeri test verisetinde de düşük kayıp değeri getirmektedir. Bu durumda eğitim halen devam ediyor ve modelin öğrenebileceği yeni örüntüler mevcut sonucu çıkarılır. Ancak bir süre sonra, genelleştirme ilerlemesi ve doğrulama metrikleri durur ve sonra düşüşe geçer. Bu durumda model aşırı uydurmaya başlar. Bu noktadan itibaren eğitim verisetine ait örüntüler öğrenilir ancak yeni gelen test verileri yanlış yönlendirilir. Neticede eğitim ve doğrulama sonuçları oldukça yüksek olmasına karşı test verisetinden alınan sonuçlar düşük olur. Bu sorunun giderilmesi için

modelin daha az bilgi ezberlemesine izin verecek hâle getirilmesi gerekmektedir. Bu sayede eniyileme süreci sadece en değerli örüntülere yoğunlaşacak ve böylece genelleştirme oranı artacaktır. Aşırı uydurma probleminin giderilmesine yönelik yapılan bu işlemlere düzenlileştirme denilmektedir [29]. Düzenlileştirme için; ağ kapasitesinin azaltılması, parametre norm cezaları, kısıtlanmış eniyileme olarak norm cezaları, veri kümesi çeşitleme, gürültü giderme, çoklu görev öğrenme, yarı denetimli öğrenme, erken durdurma, parametrelere eşleme ve parametre paylaşımı, seyrek gösterimler, istifleme ve diğer topluluk yöntemleri, iletim sönümü, çekişmeli öğrenme ve tanjant mesafesi, tanjant yayılımı ve manifold tanjant sınıflandırıcısı gibi yöntemler kullanılmaktadır [28].

Bu yöntemlerden en yaygın kullanılanları ağ kapasitesinin azaltılması, parametre norm cezaları ile ağırlıkların düzenlileştirilmesi ve Geoffrey Hinton ve diğerleri tarafından geliştirilen iletim sönümüdür. Tez çalışmasının ayırt edici özelliklerinden biri de model performansına önemli etkisi olan düzenlileştirme yöntemlerinin modele entegrasyonudur. Bu kapsamda yapılan inceleme ve test sonuçları aşağıda sunulmuştur.

Ağ kapasitesinin azaltılması: Aşırı uydurmayı önlemenin en kolay yollarından biri ağ kapasitesinin optimum sonucu verecek şekilde azaltılmasıdır. Ağ kapasitesinin azaltılması, katman sayıları ve katmanlarda bulunan hücre sayılarının azaltılması ile gerçekleştirilir. Böylece model daha az parametreye sahip olur ve aşırı öğrenme riski azalır. Kapasitesi yüksek bir modelin ezberleme oranı daha yüksek olacaktır. Derin öğrenme modelleri eğitim veriseti üzerinde iyi sonuçlar elde etmekte ve bu verisetlerine kolay uyum sağlamaktadır. Ancak asıl yapılması gereken genelleştirme başarısının arttırılmasıdır. Bununla birlikte aşırı uydurmanın önüne geçmek için model kapasitesinin çok düşürülmesi, yetersiz uydurma sorununa neden olacaktır. Bu nedenle yüksek model başarısı için optimum model kapasitesi oluşturmak gerekmektedir.

Ağırlıkların düzenlileştirilmesi: Model kapasitesinin azaltılmasının yanı sıra, kapasitenin sınırlandırılması ile de düzenlileştirme işlemi gerçekleştirilebilir. Bu amaçla, kayıp fonksiyonuna bir norm cezası eklenir.

$$J'(\theta; X, y) = J(\theta; X, y) + \alpha\, \Omega(\theta) \tag{3.19}$$

Burada $\alpha \in [0, \infty)$ norm ceza terimi olan Ω'nın standart kayıp fonksiyonu J'ye göre katkısının ağırlığını belirleyen bir hiperparametredir. α=0 olması durumunda hiç düzenlileştirme eklenmemiş, büyük α değeri seçilmesi durumunda ise daha çok düzenlileştirme gerçekleştirilmiş olur.

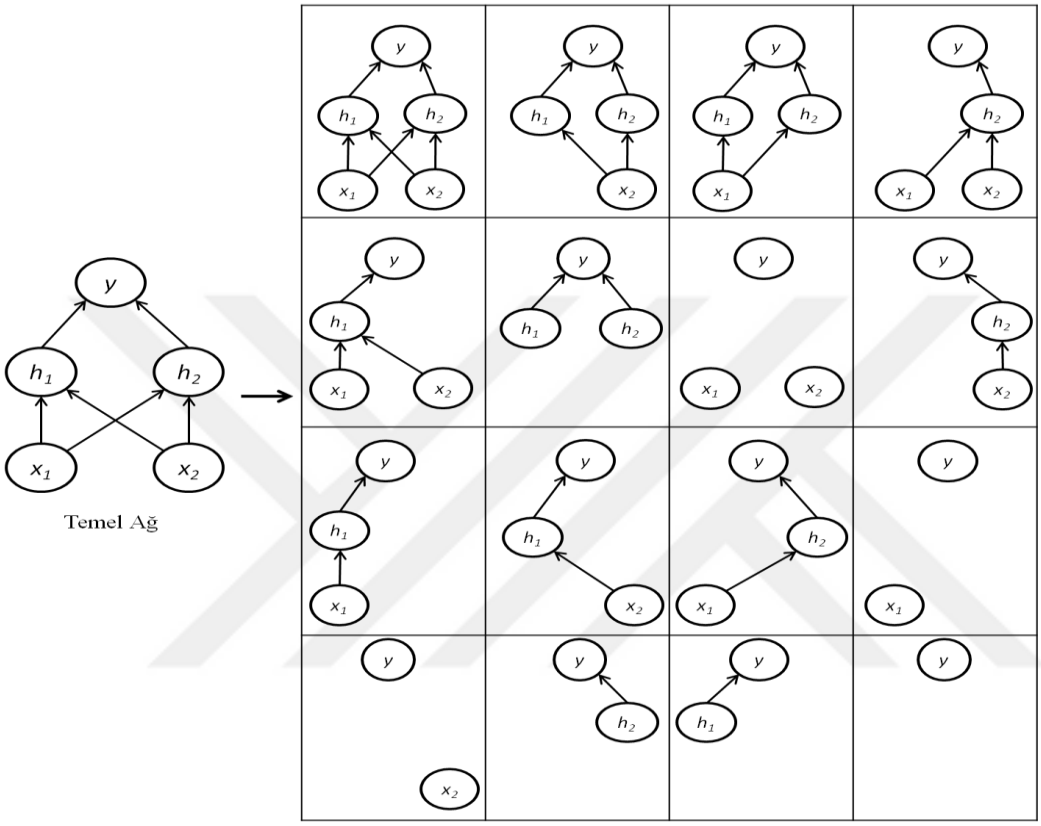
Çalışma kapsamında L_1 ve L_2 parametre normu cezaları incelenmiş, modelde L_2 düzenlileştirme işlemi uygulanmıştır. L2 düzenlileştirme stratejisinde, kayıp fonksiyonuna $\Omega(\theta) = \frac{1}{2}\|w\|_2^2$ düzenlileştirme terimi eklenerek ağırlıklar orjine yaklaştırılır. Bir başka ifadeyle parametreler belirli bir noktaya yaklaşacak şekilde düzenlileştirilebilmektedir. Bu kapsamda gerçek değerlere yakın değerler seçmek sonuçların daha iyi olmasını sağlamaktadır. Sıfır değeri doğru değerin negatif veya pozitif olduğunun belirlenemediği durumlarda varsayılan değer olarak kullanılmakta ve model parametrelerini sıfıra yaklaştırmak daha yaygın bir yöntem olarak karşımıza çıkmaktadır. Uygulamada ağırlık düzenlileştirme katmanlarına, ağırlık düzenlileştirme sınıfının örnekleri parametre olarak gönderilerek kayıp fonksiyonuna bir maliyet eklenerek büyük değerler almasına engel olunmaktadır. Bu işlem aşağıdaki kod satırı ile gerçekleştirilmektedir.

```
kernel_regularizer=regularizers.l2(0.001)
```

Bu ifadenin LSTM dense katmanına eklenmesi ile ağırlık matrisinin her elemanına kendi değeri 0.001 ile çarpılarak ilave edilmekte ve azaltılmış yeni ağırlık değerleri üretilmektedir. Bu norm cezası sadece eğitim zamanında uygulandığından, eğitim aşamasında elde edilen kayıp değeri, test aşamasında elde edilen kayıp değerine oranla daha yüksek olmaktadır.

İletim Sönümü: Bir sinir ağında iletim sönümünün bir katmana uygulanması ile eğitim işlemi esnasında o katmanın öğrendiği bilginin bir bölümü söndürülerek bir sonraki katmana iletilmez. Bu işlemin gerçekleştirilmesi için genelde 0.2 ile 0.5 arasında bir iletim sönüm oranı belirlenir ve bu oran doğrultusunda iletim sönümü gerçekleştirilir. Burada amaç, bir katmanın çıktı değerine rasgele bir gürültü değeri eklenerek, önemsiz örüntülerin ezberlenmesinin önüne geçmektir. İletim sönümü ile temel bir ağda yer alan ve çıktı birimi bulunmayan birimler çıkarılarak üretilen alt ağlar eğitilir. Bahse konu birimlerin çıkarılması çıktı değerleri sıfır ile çarpılarak gerçekleştirilir. I. Goodfellow ve diğerleri tarafından konu aşağıdaki şekilde anlatılmaktadır [28].

Bir sinir ağı birçok alt ağdan oluşmaktadır ve bunların çoğunun çıktı birimi bulunmamaktadır. İletim sönümü ile çıktı birimi olmayan bu alt ağların temel ağdan çıkarılması ile elde edilen alt ağlar topluluğu eğitilir. Şekil 3.26'da bir gizli katmana sahip ağın olası tüm alt ağları görülmektedir.



Şekil 3.26. Bir gizli katmanı olan temel ağ ve olası alt ağları [28].

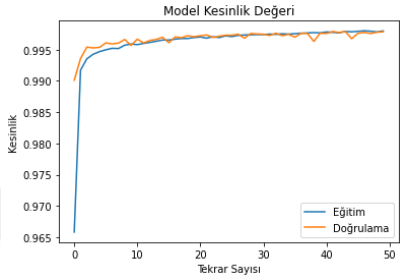
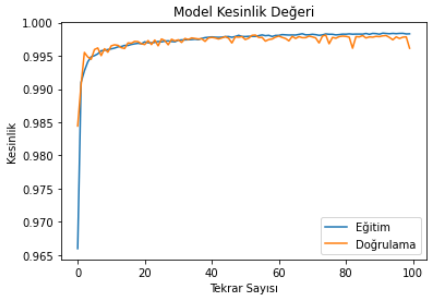
Yukarıdaki bilgiler doğrultusunda düzenlileştirme işlemlerinin modele uygulanması neticesinde elde edilen sonuçlar devam eden bölümde sunulmuştur. Modelde, çalışmanın önceki bölümünde yapılan test işlemleri ile uyum açısından Çizelge 3.15'te görüleceği üzere aynı hiperparametre değerleri kullanılmıştır.

Çizelge 3.15. Kullanılan hiperparametre değerleri.

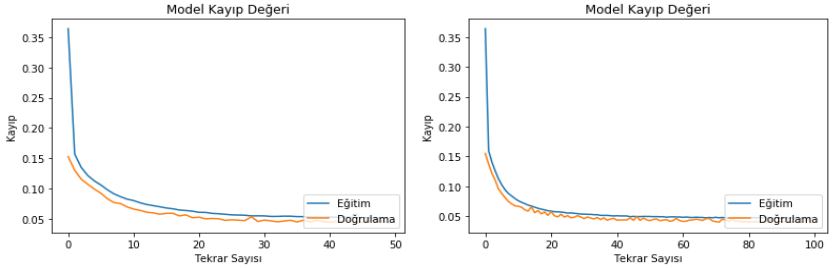
Hiperparametre	Değeri	Hiperparametre	Değeri
Dropout değeri	*0,2*	*Kayıp fonksiyonu*	*Binary_crossentropy*
LSTM aktivasyon fonksiyonu	*Relu*	*Eğitim kayıt oranı (%)*	*80*
Dense aktivasyon fonksiyonu	*Sigmoid*	*Doğrulama kayıt oranı (%)*	*20*
Model optimizer	*Rmsprop*	*Giriş katmanı hücre sayısı*	*#özellik*

Düzenlileştirme işlemleri kapsamında kullanılan parametre değerlerine bağlı olarak elde edilen en iyi sonuçlar Çizelge 3.16'da yer almaktadır. Yapılan test işlemleri neticesinde üç katmanlı, katmanlarda 32 hücre sayısına sahip modelin en iyi neticeyi verdiği görülmüştür.

Çizelge 3.16. Düzenlileştirme işlemleri neticesinde elde edilen sonuçlar.

Düzenlileştirme İşlemleri Neticesinde Elde Edilen Sonuçlar				
Hiperparametreler	Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
Katman Sayısı : 3 *Katmanlardaki Hücre Sayısı: 32* *Tekrar Sayısı: 50* *Batch_size: 256* *Regulizer: L2* *Norm Ceza Değeri: 0.001*	Eğitim: 99,81 Doğrulama: 99,75 Test: **80,68**	Eğitim: 99.82 Doğrulama: 99,71 Test: 92,50	Eğitim: 99.76 Doğrulama: 99,67 Test: 72,49	
Katman Sayısı : 3 *Katmanlardaki Hücre Sayısı: 32* *Tekrar Sayısı: 100* *Batch_size: 256* *Regulizer: L2* *Norm Ceza Değeri: 0.001*	Eğitim: 99,83 Doğrulama: 99,62 Test: **82,30**	Eğitim: 99.86 Doğrulama: 99,32 Test: 92,62	Eğitim: 99,81 Doğrulama: 99,88 Test: 75,17	

Düzenlileştirme işlemleri neticesinde elde edilen test sonuçlarında, kayıp oranının düzenlileştirme işlemi öncesine oranla daha yüksek bir değerden başlayarak daha yavaş azaldığı, önceki testlerde elde edilen çok düşük kayıp oranları yerine bir miktar yüksek kayıp oranı elde edildiği görülmüştür. Elde edilen kayıp oranlarının, önceki testlerde elde edilen kayıp oranlarına göre yüksek olmasına karşı daha iyi sınıflandırma sonuçları elde edilmesi, öğrenmenin daha etkin bir şekilde gerçekleştirildiğine işaret etmektedir. Model kayıp oranlarına yönelik elde edilen eğriler Şekil 3.27'de görülmektedir.



Şekil 3.27. Model kayıp oranı eğrileri.

Katman sayısının azalması ve model kapasitesinin düşmesi, model çalışma hızının belirgin seviyede artmasına neden olmuştur. Böylece daha kısa sürede iyi sonuçların elde edildiği görülmüştür. Çalışmanın devam eden bölümünde, düzenlileştirme işlemleri uygulanan modelin K-fold yöntemi ile testi gerçekleştirilmiştir. Daha önce de vurgulandığı gibi K-fold yöntemi, veriseti içerisindeki farklı bölümlerin doğrulama ve test işlemlerinde kullanılması neticesinde, model eğitim seviyesi ve başarı oranının artmasını sağlamaktadır. Bu kapsamda sonuçların gözlemlenmesi açısından 4, 5 ve 6 fold ile gerçekleştirilen testlere ait sonuçlar Çizelge 3.17'de görülmektedir.

Çizelge 3.17. Düzenlileştirme işlemleri neticesinde K-fold ile elde edilen sonuçlar.

<table>
<tr><th colspan="2">4 Katman ile Test</th><th colspan="2">5 Katman ile Test</th><th colspan="2">6 Katman ile Test</th></tr>
<tr><th>Kt. No</th><th>Sonuçlar</th><th>Kt. No</th><th>Sonuçlar</th><th>Kt. No</th><th>Sonuçlar</th></tr>
<tr><td>1</td><td>Test Kesinlik: % 81,09
Test Hassasiyet: % 92,74
Test Duyarlılık: %72,91</td><td>1</td><td>Test Kesinlik: %79,64
Test Hassasiyet: %92,35
Test Duyarlılık: %70,85</td><td>1</td><td>Test Kesinlik: % 79,37
Test Hassasiyet: % 92,18
Test Duyarlılık: % 70,20</td></tr>
<tr><td>2</td><td>Test Kesinlik: %83,57
Test Hassasiyet: %93,08
Test Duyarlılık: %77,34</td><td>2</td><td>Test Kesinlik: % 79,56
Test Hassasiyet: % 92,58
Test Duyarlılık: % 70,15</td><td>2</td><td>Test Kesinlik: % 81,75
Test Hassasiyet: % 92,89
Test Duyarlılık: % 73,94</td></tr>
<tr><td>3</td><td>Test Kesinlik: %86,04
Test Hassasiyet: %97,99
Test Duyarlılık: %78,63</td><td>3</td><td>Test Kesinlik: % 81,88
Test Hassasiyet: % 92,91
Test Duyarlılık: % 74,47</td><td>3</td><td>Test Kesinlik: % 82,95
Test Hassasiyet: % 97,03
Test Duyarlılık: % 73,03</td></tr>
<tr><td>4</td><td>Test Kesinlik: %82,04
Test Hassasiyet: %94,54
Test Duyarlılık: %74,21</td><td>4</td><td>Test Kesinlik: % 84,15
Test Hassasiyet: % 93,06
Test Duyarlılık: % 78,40</td><td>4</td><td>Test Kesinlik: % 82,02
Test Hassasiyet: % 92,58
Test Duyarlılık: % 75,17</td></tr>
<tr><td colspan="2" rowspan="3"><u>Ortalama Sonuçlar</u>
Test Kesinlik: % 83,18
Test Hassasiyet:% 94,58
Test Duyarlılık: % 75,77</td><td>5</td><td>Test Kesinlik: % 82,75
Test Hassasiyet: % 92,87
Test Duyarlılık: % 76,24</td><td>5</td><td>Test Kesinlik: % 81,92
Test Hassasiyet: % 92,88
Test Duyarlılık: % 74,65</td></tr>
<tr><td colspan="2" rowspan="2"><u>Ortalama Sonuçlar</u>
Test Kesinlik: % 81,59
Test Hassasiyet: % 92,75
Test Duyarlılık: % 74,02</td><td>6</td><td>Test Kesinlik: % 81,90
Test Hassasiyet: % 92,52
Test Duyarlılık: % 74,85</td></tr>
<tr><td colspan="2"><u>Ortalama Sonuçlar</u>
Test Kesinlik: % 81,65
Test Hassasiyet: % 93,35
Test Duyarlılık: % 73,64</td></tr>
</table>

Elde edilen sonuçlar incelendiğinde, en yüksek kesinlik değerinin 4 fold ile gerçekleştirilen test işleminde alındığı görülmektedir. Düzenlileştirme işlemleri öncesi elde edilen sonuçlarla kıyaslandığında 4 ve 5 fold ile gerçekleştirilen test işlemleri neticesinde başarı seviyesinin arttığı, 6 fold uygulanarak alınan test sonuçlarında ise bir miktar azalma olduğu gözlemlenmiştir. Bunun nedeninin veriseti içerisindeki düzensiz veri dağılımından kaynaklandığı değerlendirilmiştir.

3.7.1. Özellik seçim algoritmalarının modele uygulanması

Verisetinde yer alan özelliklerin model başarısına etkisinin gözlemlenmesi için özellik seçimi (feature selection) ile veriseti boyutunun azaltılmasına yönelik literatür incelemesi gerçekleştirilmiştir [83-87]. İncelenen çalışmalarda, özellik seçiminin, model performansına etkisi olmayan özelliklerin ayıklanmasını sağladığı ve böylece model çalışması esnasında bu özellikler için ayrılan kaynak kullanımını önlediği görülmüştür. Bununla birlikte, veriseti içerisindeki gereksiz özelliklerin, ürettikleri gürültü nedeniyle model performansına olumsuz etkilerinin olduğu, büyük boyutlu verisetlerinde bu problemle sıklıkla karşılaşıldığı tespit edilmiştir. Veriseti içerisinde gereksiz özellik bulunmasının, eğitim süresini arttırmanın yanı sıra modeli daha karmaşık hale getirerek aşırı uydurmaya neden olduğu gözlemlenmiştir. Bu nedenle model performansına etkisi olmayan gereksiz özelliklerin ayıklanması ile daha etkin sınıflandırma veya tahmin sonuçları elde etmenin mümkün olduğu değerlendirilmiştir.

Çalışmalar incelendiğinde başta Principal Component Analysis (PCA) olmak üzere çeşitli özellik seçim yöntemlerinin kullanıldığı görülmektedir. Chi-square (chi), Pearson korelasyonu, recursive özellik seçimi (Recursive Feature Selection –RFE), Lasso regression, ağaç-tabanlı özellik seçimi, elastic net, ridge regression, singular value decompoistion, linear discriminant analysis gibi özellik seçim yöntemlerinden yararlanıldığı tespit edilmiştir. Bununla birlikte, lokal arama algoritmalarının problem ve veriseti özelinde uygulanması ile özellik seçimi gerçekleştirilebildiği görülmüştür [87].

Bu konuda yapılan bir çalışmada [83], belirli kritik özelliklerin çıkarılması durumunda sınıflandırma oranlarının düştüğü, chi-square yöntemi kullanılarak yapılan özellik seçim işlemi neticesinde, NSL-KDD veriseti için kritik özellikleri de içeren ve Çizelge 3.18'de görülen 31 özelliğin optimum özellik sınıfı olarak belirlendiği ifade edilmektedir.

Çizelge 3.18'de sunulan 31 özellikten oluşan indirgenmiş verisetinin çalışma kapsamında kullanılan modelin performans oranında önemli seviyede artış sağladığı ifade edilmektedir.

Çizelge 3.18. Chi-square uygulanarak elde edilen NSL-KDD optimum özellik sınıfı [83].

Derece	Özellik	NSL-KDD İçindeki Sırası	Derece	Özellik	NSL-KDD İçindeki Sırası
1	Service	*3*	17	Logged_in	*12*
2	Dst_bytes	*6*	18	Dst_host_Count	*32*
3	Dst_host_diff_srv_rate	*35*	19	Hot	*10*
4	Diff_srv_rate	*30*	20	Dst_host_rerror_rate	*40*
5	Flag	*4*	21	Srv_count	*24*
6	Dst_host_serror_rate	*38*	22	Duration	*1*
7	Dst_host_srv_count	*33*	23	Srv_diff_host_rate	*31*
8	Same_srv_rate	*29*	24	Dst_host_srv_rerror_rate	*41*
9	Count	*23*	25	R error_rate	*27*
10	Dst_host_same_srv_rate	*34*	26	Protocol_type	*2*
11	Dst_host_srv_serror_rate	*39*	27	Srv_r error_rate	*28*
12	Serror_rate	*25*	28	Is_guest_login	*22*
13	Src_bytes	*5*	29	Srv_count	*24*
14	Dst_host_srv_diff_host_rate	*37*	30	Num_compromised	*13*
15	Srv_serror_rate	*26*	31	Num_failed_logins	*11*
16	Dst_host_same_src_port_rate	*36*			

Bir diğer çalışmada [85], PCA özellik seçim yönteminin aşağıda belirtilen beş temel adımın uygulanması ile gerçekleştirildiği belirtilmektedir.

1. Her veri boyutundan anlam çıkarma,
2. Kovaryans matrisinin hesaplanması,
3. Kovaryans matrisinin eugenvektör ve eugenvalue değerlerinin hesaplanması,
4. Bileşenlerin seçimi ve bir özellik vektörünün oluşturulması,
5. Yeni verisetinin çıkarılması.

PCA'in NSL-KDD verisetine uygulanması neticesinde Çizelge 3.19'da sunulan 23 özelliğin belirlendiği vurgulanmaktadır. Chi-square yöntemi kullanılarak elde edilen veriseti ile kıyaslanması amacıyla bu 23 özellik tabloda kırmızı renkte belirginleştirilerek gösterilmiştir.

Çizelge 3.19. PCA uygulanarak elde edilen NSL-KDD optimum özellik sınıfı [85].

Derece	Özellik	NSL-KDD İçindeki Sırası	Derece	Özellik	NSL-KDD İçindeki Sırası
1	**Service**	*3*	17	**Logged_in**	*12*
2	**Dst_bytes**	*6*	18	**Dst_host_Count**	*32*
3	**Dst_host_diff_srv_rate**	*35*	19	Hot	*10*
4	**Diff_srv_rate**	*30*	20	Dst_host_rerror_rate	*40*
5	**Flag**	*4*	21	**Srv_count**	*24*

Çizelge 3.19. (Devam) PCA uygulanarak elde edilen NSL-KDD optimum özellik sınıfı [85].

Derece	Özellik	NSL-KDD İçindeki Sırası	Derece	Özellik	NSL-KDD İçindeki Sırası
6	**Dst_host_serror_rate**	*38*	22	**Duration**	*1*
7	**Dst_host_srv_count**	*33*	23	Srv_diff_host_rate	*31*
8	**Same_srv_rate**	*29*	24	Dst_host_srv_rerror_rate	*41*
9	**Count**	*23*	25	R error_rate	*27*
10	Dst_host_same_srv_rate	*34*	26	**Protocol_type**	*2*
11	**Dst_host_srv_serror_rate**	*39*	27	**Srv_r error_rate**	*28*
12	Serror_rate	*25*	28	Is_guest_login	*22*
13	**Src_bytes**	*5*	29	Srv_count	*24*
14	**Dst_host_srv_diff_host_rate**	*37*	30	**Num_compromised**	*13*
15	**Srv_serror_rate**	*26*	31	**Num_failed_logins**	*11*
16	Dst_host_same_src_port_rate	*36*		**Wrong_fragment**	*8*

Bir diğer çalışmada [87] ise özellik seçim problemi kombinasyonel optimizasyon problemi olarak ele alınmakta ve lokal bir arama algoritması ile özellik seçim işlemi gerçekleştirilmektedir. Lokal özellik şeçim algoritmasına ait pseudocode aşağıda görülmektedir:

1. Bir başlangıç çözümü s_{init} üret,
2. Başlangıç çözümünün maliyetini $C(s_{init})$ hesapla,
3. Mevut en iyiyi başlangıç çözümü yap $s_{best} = s_{init}$,
4. while(1) başla:
   a. Bir biti farklı olan komşu çözümleri bul,
   b. Her komşu çözümün maliyetini hesapla,
   c. Komşu çözümlerden en iyisini belirle s_{bneigh},
   d. if $C(s_{best}) \geq C(s_{bneigh})$
      break
   e. else
      $s_{best} = s_{bneigh}$
5. end

Algoritmanın NSL-KDD eğitim setine uygulanması neticesinde Çizelge 3.20'de mavi ile belirginleştirilen toplam 24 özellik seçilmiştir.

Çizelge 3.20. Lokal algoritma ile elde edilen NSL-KDD optimum özellik sınıfı [87].

Derece	Özellik	NSL-KDD İçindeki Sırası	Derece	Özellik	NSL-KDD İçindeki Sırası
1	**Service**	*3*	17	**Logged_in**	*12*
2	**Dst_bytes**	*6*	18	**Dst_host_Count**	*32*
3	**Dst_host_diff_srv_rate**	*35*	19	**Hot**	*10*
4	**Diff_srv_rate**	*30*	20	**Dst_host_rerror_rate**	*40*
5	**Flag**	*4*	21	**Srv_count**	*24*
6	**Dst_host_serror_rate**	*38*	22	**Duration**	*1*
7	**Dst_host_srv_count**	*33*	23	**Srv_diff_host_rate**	*31*
8	**Same_srv_rate**	*29*	24	Dst_host_srv_rerror_rate	*41*
9	**Count**	*23*	25	**R error_rate**	*27*
10	Dst_host_same_srv_rate	*34*	26	**Protocol_type**	*2*
11	**Dst_host_srv_serror_rate**	*39*	27	**Srv_r error_rate**	*28*
12	**Serror_rate**	*25*	28	Is_guest_login	*22*
13	**Src_bytes**	*5*	29	Srv_count	*24*
14	**Dst_host_srv_diff_host_rate**	*37*	30	**Num_compromised**	*13*
15	**Srv_serror_rate**	*26*	31	**Num_failed_logins**	*11*
16	**Dst_host_same_src_port_rate**	*36*		**Wrong_fragment**	
				Num_file_creations	
				Land	
				Urgent	
				Num_shells	
				Is_host_login	

Tez çalışmasında, üç farklı yöntemin NSL-KDD verisetine uygulanması ile elde edilen azaltılmış özellik sınıfları ile model testleri tekrar edilerek elde edilen sonuçlar diğer model sonuçları ile kıyaslanmış ve model performansı gözlemlenmiştir. Test sonuçları ve bunlara yönelik değerlendirme aşağıda sunulmuştur. Test işlemleri Çizelge 3.21'de yer alan hiperparametre değerleri kullanılarak yapılmıştır.

Çizelge 3.21. Test işlemlerinde kullanılan hiperparametre değerleri.

Hiperparametre	Değeri	Hiperparametre	Değeri
Dropout değeri	*0,2*	*Kayıp fonksiyonu*	*Binary_crossentropy*
LSTM aktivasyon fonksiyonu	*Relu*	*Eğitim kayıt oranı (%)*	*80*
Dense aktivasyon fonksiyonu	*Sigmoid*	*Doğrulama kayıt oranı (%)*	*20*
Model optimizer	*Rmsprop*	*Giriş katmanı hücre sayısı*	*#özellik*

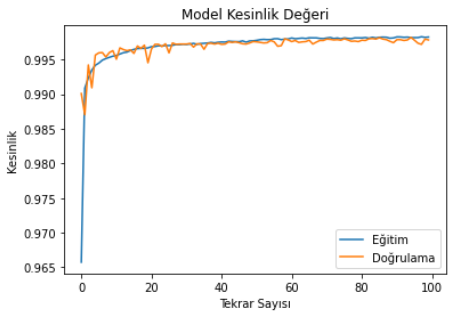
3.7.2. Elde Edilen Sonuçlar

Bahse konu üç algoritma neticesinde elde edilen veri setleri ile gerçekleştirilen test işlemleri neticesinde elde edilen sonuçlar aşağıda sunulumuştur.

Chi-Square özellik çıkarım algoritması ile elde edilen sonuçlar

Chi-square özellik azaltım işlemi neticesinde elde edilen özellikler Çizelge 3.18'de sunulmuştur. Bu kapsamda veriseti ve model üzerinde gerekli düzenlemeler yapılarak testler gerçekleştirilmiş ve Çizelge 3.22'deki sonuçlar elde edilmiştir.

Çizelge 3.22. Chi-square özellik azaltım yöntemi ile elde edilen verisetinin modele uygulanması neticesinde elde edilen sonuçlar.

Veriseti Üzerinde İşlem YapıldıktanSonraAlınan Sonuçlar			
Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
Eğitim: 99,82 Doğrulama: 99,78 Test: 80,95	Eğitim: 99.89 Doğrulama: 99,81 Test: 92,81	Eğitim: 99.79 Doğrulama: 99,66 Test: 72,77	

Çizelge incelendiğinde, özellik azaltma yapılmadan önceki başarı seviyesinden daha düşük bir sonuç elde edildiği görülmüştür. Daha sonra model aynı veri üzerinde dört katman ile tekrar test edilmiş ve Çizelge 3.23'teki sonuçlar elde edilmiştir.

Çizelge 3.23. Chi-square özellik azaltım yöntemi ile elde edilen verisetinin K-fold yöntemi kullanılarak modele uygulanması neticesinde elde edilen sonuçlar.

Değerler	Katman				Ort.
	1	2	3	4	
Kayıp (Loss)	1,81	2,23	3,83	2,76	2,65
Kesinlik (Accuracy)	%80,94	%79,13	%80,51	%81,19	%80,44
Hassasiyet (Precision)	%92,54	%92,38	%92,52	%92,58	%92,50
Duyarlılık (Recall)	%72,86	%69,51	%72,33	%73,15	%71,96

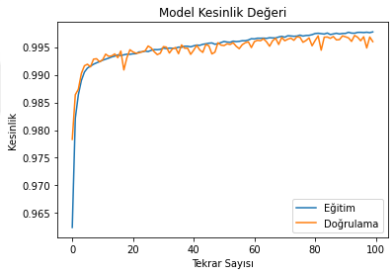
Modelin 4 katman ile çalıştırılması neticesinde kesinlik oranında bir miktar azalma olduğu, ancak belirgin bir etkinin olmadığı görülmüştür. Elde edilen sonuçların, verisetinde özellik azaltma yapılmadan önce elde edilen sonuçlardan daha düşük olduğu görülmüştür. Buna bağlı olarak, Chi-Square özellik azaltım yöntemi ile elde edilen NSL-KDD verisetine

uygulanan LSTM modelinin, özellik azaltma yapılmadan kullanılan NSL-KDD verisetine oranla daha başarısız olduğu sonucu çıkarılmıştır. Bu sonucun, LSTM modelinin başarısının veri miktarı ile doğru orantılı olarak arttığı önermesini desteklediği değerlendirilmiştir.

PCA özellik çıkarım algoritması ile elde edilen sonuçlar

PCA özellik azaltım yöntemi kullanılarak elde edilen NSL-KDD optimum özellik sınıfı, Çizelge 3.19'da sunulmuştur. Modelin bu özellik sınıfı ile elde edilen yeni verisetine uygulanması neticesinde Çizelge 3.24'teki sonuçlara ulaşılmıştır.

Çizelge 3.24. PCA özellik azaltım yöntemi ile elde edilen verisetinin modele uygulanması neticesinde elde edilen sonuçlar.

Veriseti Üzerinde İşlem Yapıldıktan Sonra Alınan Sonuçlar			
Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
Eğitim: 99,77 Doğrulama: 99,60 Test: 78,32	Eğitim: 99.81 Doğrulama: 99,70 Test: 91,17	Eğitim: 99.71 Doğrulama: 99,46 Test: 69,34	

Çizelge incelendiğinde, PCA özellik azaltımı neticesinde elde edilen sonuçların, Chi-square özellik azaltımı ile elde edilen sonuçlara göre daha düşük olduğu görülmektedir. Model başarısının, K-fold yöntemi uygulanması neticesinde görülmesi amacıyla model tekrar test edilmiş ve Çizelge 3.25'teki sonuçlar elde edilmiştir.

Çizelge 3.25. PCA özellik azaltım yöntemi ile elde edilen verisetinin K-fold yöntemi kullanılarak modele uygulanması neticesinde elde edilen sonuçlar.

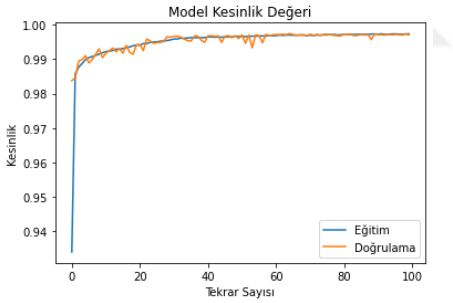
Değerler	Katman				Ort.
	1	2	3	4	
Kayıp (Loss)	2,53	2,13	3,36	4,66	3,17
Kesinlik (Accuracy)	%79,41	%79,14	%79,49	%78,44	%79,12
Hassasiyet (Precision)	%91,52	%91,94	%91,66	%92,34	%91,86
Duyarlılık (Recall)	%70,93	%70,10	%70,67	%68,38	%70,02

Modelin 4 katman ile çalıştırılması neticesinde kesinlik oranında bir miktar artış olduğu, ancak belirgin bir etkinin olmadığı görülmüştür. Elde edilen sonuçların, Chi-square yöntemi ile elde edilen verisetinin ve orijinal verisetinin modele uygulanması neticesinde elde edilen sonuçlardan daha düşük olduğu görülmüştür. Bu sonucun, LSTM model başarısının veri miktarı ile doğru orantılı olarak arttığı önermesini desteklediği değerlendirilmiştir.

Lokal özellik çıkarım algoritması ile elde edilen sonuçlar

Lokal özellik çıkarım algoritması kullanılarak elde edilen NSL-KDD optimum özellik sınıfı Çizelge 3.20'de sunulmuştur. Modelin bu özellik sınıfı ile elde edilen yeni verisetine uygulanması neticesinde Çizelge 3.26'da sunulan sonuçlar elde edilmiştir.

Çizelge 3.26. Lokal algoritma özellik azaltma yöntemi ile elde edilen verisetinin modele uygulanması neticesinde elde edilen sonuçlar.

Veriseti Üzerinde İşlem YapıldıktanSonra Alınan Sonuçlar			
Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Geri Çağırma (Recall) %	Model Kesinlik Grafiği
Eğitim: 99,74 Doğrulama: 99,70 Test: 77,85	Eğitim: 99,70 Doğrulama: 99,74 Test: 91,05	Eğitim: 99,67 Doğrulama: 99,39 Test: 69,12	

Çizelge 3.26 incelendiğinde, model kesinlik başarısında orijinal veri kümesi ve diğer özellik kümelerine oranla azalma olduğu görülmektedir. Model başarısının, K-fold yönteminin verisetine uygulanması neticesinde görülmesi amacıyla model tekrar test edilmiş ve Çizelge 3.27'deki sonuçlar elde edilmiştir.

Çizelge 3.27. Lokal algoritma özellik azaltım yöntemi ile elde edilen verisetinin K-fold yöntemi kullanılarak modele uygulanması neticesinde elde edilen sonuçlar.

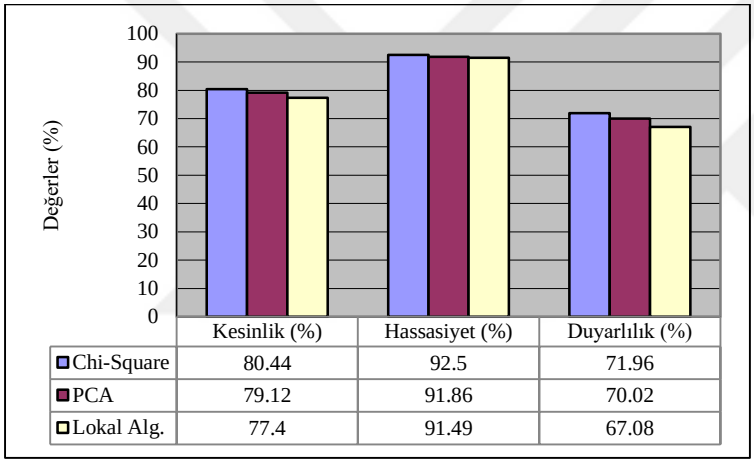
Değerler	Katman				Ort.
	1	2	3	4	
Kayıp (Loss)	2,61	2,68	3,61	3,81	3,17
Kesinlik (Accuracy)	%76,68	%79,40	%77,16	%76,36	%77,4
Hassasiyet (Precision)	%91,75	%92,01	%91,33	%90,88	%91,49
Duyarlılık (Recall)	%65,42	%70,29	%67,01	%65,60	%67,08

Lokal algoritma ile özellik azaltım yöntemi ile elde edilen verisetinin modele uygulanması neticesinde, orijinal verisetinin ve diğer özellik azaltım yöntemleri ile elde edilen verisetlerinin modele uygulanması neticesinde elde edilen sonuçlara oranla daha düşük kesinlik seviyesinde başarı elde edildiği görülmüştür.

Elde sonuçların karşılaştırılması

Üç farklı özellik seçim yöntemi ile üretilmiş özellik kümeleri ile yapılan testler neticesinde elde edilen sonuçların kıyaslamalı olarak gösterildiği grafik Çizelge 3.28'de sunulmuştur.

Çizelge 3.28. Elde edilen sonuçların kıyaslanması.



	Kesinlik (%)	Hassasiyet (%)	Duyarlılık (%)
Chi-Square	80.44	92.5	71.96
PCA	79.12	91.86	70.02
Lokal Alg.	77.4	91.49	67.08

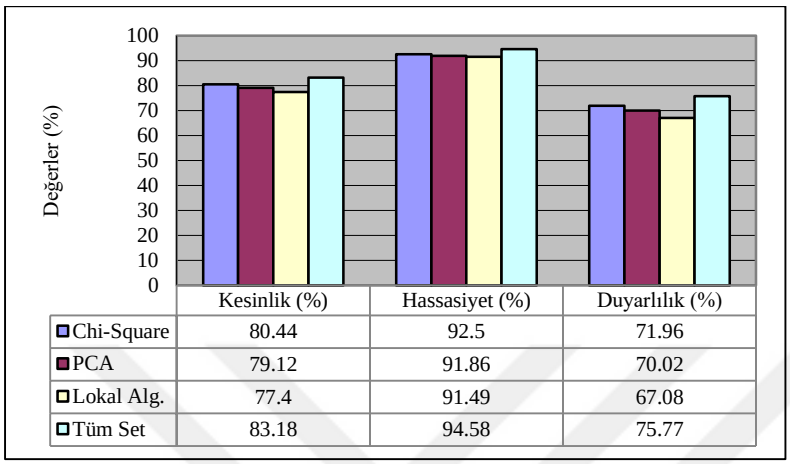
Üç farklı özellik kümesi ile gerçekleştirilen testler neticesinde;

- Chi-square özellik kümesi ile yapılan test neticesinde elde edilen sonuçların, diğer kümelerle yapılan testler neticesinde elde edilen sonuçlara oranla bir miktar artış gösterdiği, ancak değerlerin birbirine oldukça yakın olduğu,
- Verisetlerindeki veri miktarının azalmasının, LSTM model performansını olumsuz etkilediği değerlendirilmiştir.

NSL-KDD veriseti üzerinde özellik seçimi yapılmadan orijinal veriseti ile modelin testi neticesinde alınan sonuçların, özellik seçim algoritmaları ile elde edilen verisetlerinin modele uygulanması neticesinde alınan sonuçlara oranla belirgin derecede başarılı olduğu gözlemlenmiştir. Bu gözlem, LSTM model başarısının, kullanılan veriseti içerisindeki veri miktarı ile doğru orantılı olarak artış gösterdiği sonucunu çıkarmamızı sağlamıştır. Özellik

seçimi yapılmadan elde edilen test sonuçlarının, özellik seçim algoritmalarının kullanılması neticesinde elde edilen test sonuçları ile kıyaslanması Çizelge 3.29'da sunulmuştur.

Çizelge 3.29. Tüm test sonuçları.



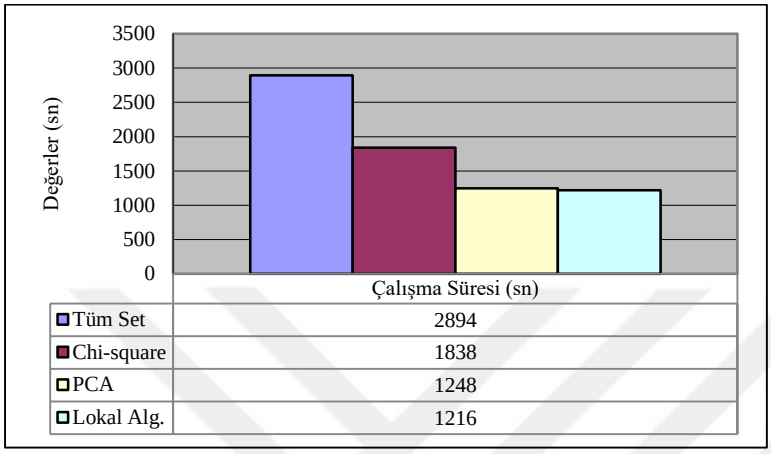
	Kesinlik (%)	Hassasiyet (%)	Duyarlılık (%)
Chi-Square	80.44	92.5	71.96
PCA	79.12	91.86	70.02
Lokal Alg.	77.4	91.49	67.08
Tüm Set	83.18	94.58	75.77

Manuel özellik seçim işleminin derin öğrenme metotlarının kullanımı durumunda tercih edilmediği, derin öğrenme modellerinin veriden özellik çıkarımı için daha fazla veriye ihtiyaç duydukları bilinmektedir. Ancak ağ trafik verisi her geçen gün artmaktadır ve tüm verinin kullanımı derin öğrenme algoritmalarının işlem verimliliğini azaltmaktadır. W. Yue ve arkadaşları bu hususta, derin öğrenme algoritmalarının etkin özellikleri öğrenme kabiliyetlerinin olmasına karşı saldırı tespit alanında hala başarı ile seçilmiş özelliklere ihtiyaç duyulduğunu vurgulamaktadırlar. Bazı derin öğrenme uygulamalarının manuel özellik çıkarım işleminden kaçınmak için işlenmemiş ağ trafik verisi kullanımı yoluna gittikleri belirtilmekte, ancak bunun ilk olarak eksponansiyel olarak artan ağ trafik verisinin işlenmesi için uygun olmadığı, işlem veriminin oldukça düşmesine neden olduğu ifade edilmektedir. Bununla birlikte, işlenmemiş ağ trafik verisinin kullanımının uygulanacak ön işlem adımlarında veri kaybına neden olabileceği ve bunun da model eğitimini olumsuz yönde etkileyebileceği vurgulanmaktadır. İşlenecek örnek verinin birçok paket içereceği, bu nedenle tespit işleminin yapılabilmesi için tüm akışın yakalanması gerektiğinden atak tespitinde sorunlar yaşanacağı vurgulanmaktadır [88]. Çalışmada bu problemin çözümü için verinin sıkıştırılması yoluna gidildiği ifade edilmektedir.

Tez kapsamında gerçekleştirilen çalışmada, manuel özellik çıkarım işleminin, veri boyutunun çok yüksek olduğu ve derin öğrenme yöntemlerinin işlem gücünden yararlanmak istenen durumlarda uygulanmasının, işlem verimine etkisi değerlendirilmiştir. Bu amaçla,

özellik çıkarım algoritmaları uygulanarak elde edilen verisetleri ile orijinal veriseti kullanılarak yapılan eğitim ve test faaliyetlerinin çalışma süreleri gözlemlenmiştir. Bu kapsamda elde edilen sonuçlar Çizelge 3.30'da sunulmuştur.

Çizelge 3.30. Çalışma sürelerinin kıyaslanması.



Manuel özellik azaltımı neticesinde elde edilen sonuçlar genel olarak değerlendirildiğinde; orjinal NSL-KDD veriseti ile yapılan testler neticesinde elde edilen sonuçların özellik çıkarımı neticesinde elde elde edilen verisetleri ile yapılan testler neticesinde elde edilen sonuçlara oranla daha başarılı olduğu gözlemlenmiştir. Böylece, LSTM modelinin, NSL-KDD veriseti özelinde, orjinal verisetinden özellik çıkarma işleminden olumsuz biçimde etkilendiği belirlenmiştir. Bu nedenle orjinal NSL-KDD veriseti içerisinde yer alan özelliklerin doğru ve yeterli olduğu sonucu çıkarılmıştır.

Chi-Square yöntemi ile elde edilen veriseti kullanılarak yapılan testler neticesinde elde edilen sonuçların, diğer özellik çıkarım algoritmaları ile elde edilen sonuçlara oranla düşük seviyede daha iyi olduğu görülmüştür. Bu gözlem neticesinde, Chi-Square algoritmasının, diğer algoritmalara oranla, atak tespitinde kullanılan özellikleri daha iyi seçtiği sonucu çıkarılmıştır.

Özellik seçimi ile elde edilen verisetleri ile yapılan testlerin, veriseti içerisindeki veri miktarındaki azalış nedeniyle, orjinal veriseti ile yapılan testlere oranla daha hızlı sonuç verdiği görülmüştür. Bu nedenle, eğitim zamanının kısaltılması gereken durumlarda, özellik çıkarımı ile elde edilen verisetlerinin, kesinlik değerinden bir miktar feragat edilerek kullanılabileceği değerlendirilmektedir.

Derin öğrenme yöntemleri kullanılarak geliştirilen bir modelin başarısının, eğitim için kullanılan veri miktarı ile ilişkili olduğu göz önünde bulundurulduğunda, yeterli işlemci gücü bulunması durumunda, eğitim ve test işlemlerinde orjinal verisetlerinin kullanılmasının faydalı olduğu gözlemlenmiştir. Diğer taraftan, günümüzde üretilen verisetlerinin boyutları göz önünde bulundurulduğunda, verisetlerinin özellik çıkarımına tabi tutulmak zorunda olabileceği değerlendirilmektedir. Bu durumda, süre ve kesinlik ile ilgili beklenti ve analizlerin iyi yapılarak optimum bir yöntemin belirlenmesi gerekmektedir.

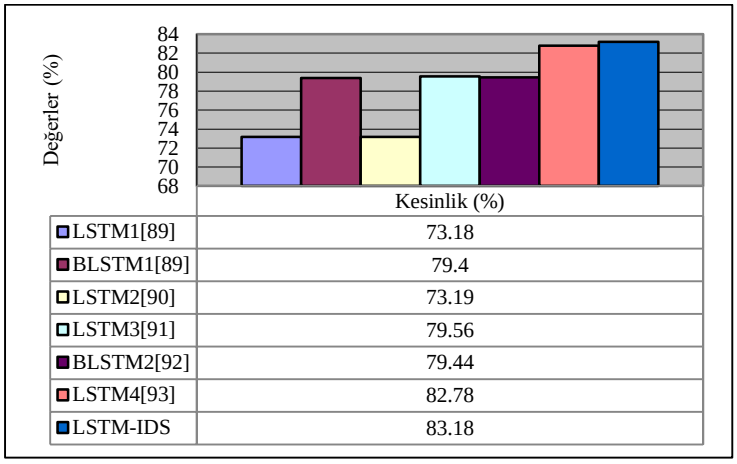
Bir verisetine özellik çıkarım algortimalarının uygulanması hususunda değerlendirme yaparken; model karmaşıklığı, işlemci gücü ve işlem süresi, içerdiği alanlar ve düzgün etiketleme açılarından veriseti kalitesi, eğitim süresi kısıtı, sistem donanım kapasitesi gibi hususların değerlendirmede göz önünde bulundurulması gerekmektedir.

3.8. Geliştirilen Modelin Literatürdeki Diğer LSTM Modelleri ile Kıyaslanması

Tez çalışmasında geliştirilen model başarısının değerlendirilmesi amacıyla, literatürdeki diğer LSTM uygulamalarının NSL-KDD verisetine uygulanması ile elde edilen sonuçlar kıyaslanmıştır. Bu kıyaslama neticesinde, KDDTest+ verisetinin test işleminde kullanıldığı durumlarda elde edilen sonuçlar değerlendirildiğinde, tez kapsamında geliştirilen LSTM modelinin daha başarılı olduğu görülmüştür. Model başarısının en önemli nedeninin, yapılan düzenlileştirme işlemleri ile aşırı öğrenme probleminin giderilmesi olduğu değerlendirilmiştir. Tez kapsamında geliştirilen model LSTM-IDS olarak isimlendirilmiş ve literatürdeki diğer modeller ile kıyaslanmasını içeren grafik Çizelge 3.31'de sunulmuştur.

Kaynak [90]'da belirtilen iki yönlü LSTM modeli BAT ((BLSTM (Bidirectional Long Short-term memory) Attention Mechanism) dışında, Çizelge 3.31'de görüleceği üzere LSTM-IDS modelinin KDDTest+ verisetine uygulanması neticesinde elde edilen kesinlik değerinin %83,18 ile literatürdeki diğer LSTM modellerine oranla daha iyi sonuç verdiği görülmektedir. Bu nedenle, çalışma kapsamında geliştirilen LSTM-IDS modelinin IDS geliştirme çalışmalarında etkin bir şekilde kullanılabileceği değerlendirilmektedir.

Çizelge 3.31. LSTM-IDS modelinin diğer LSTM çalışmaları ile kıyaslanması.



	Kesinlik (%)
LSTM1[89]	73.18
BLSTM1[89]	79.4
LSTM2[90]	73.19
LSTM3[91]	79.56
BLSTM2[92]	79.44
LSTM4[93]	82.78
LSTM-IDS	83.18

Çalışma kapsamında geliştirilen model başarısının, düzenlileştirme işlemlerinin uygun bir şekilde uygulanması ile ilişkili olduğu, model parametreleri ile ilgili çalışmaların devam ettirilmesi durumunda test sonuçlarının daha iyi seviyeye gelebileceği düşünülmektedir. Düzenlileştirme işlemleri kapsamında, parametre norm cezası, sınırlı optimizasyon için norm cezaları, gürültü giderme, çoklu görev öğrenimi, erken durdurma, karşıt öğrenme ve tanjant mesafesi gibi metotlar incelenmiş, bu metotlardan en fazla kullanılan üçü (ağ kapasitesi azaltımı, ağırlık düzenlileştirme ve dropout) modele uygulanmıştır.

Bununla birlikte, incelenen birçok çalışmada, KDDTrain+ verisetinin eğitim ve doğrulama için bölümlenmesi neticesinde elde edilen eğitim ve doğrulama sonuçlarının paylaşıldığı, bu durumda elde edilen kesinlik seviyesinin %95 üzerinde olduğu görülmüştür [94-100]. KDDTest+ veriseti ile yapılan test sonuçlarının paylaşıldığı, bir başka ifadeyle modelin genelleştirme başarısının değerlendirildiği az sayıda çalışma ile karşılaşılmıştır. NSL-KDD veriseti ile yapılan IDS çalışmalarında, modelin KDDTest+ veriseti üzerine uygulanması neticesinde elde edilen test sonuçlarının paylaşılması gerektiği değerlendirilmektedir.

3.9. Bellek Kullanımının Ağ Sızma Tespitine Etkisi

Daha önce vurgulandığı gibi, RNN tabanlı modellerde yaşanan uzun süreli bellek problemi, hücrelerde kullanılan durum ve kapılar ile giderilmektedir. Tez kapsamında, LSTM hücrelerinde kullanılan durum ve kapıların model performansına etkilerinin gözlemlenmesi amacıyla, basit tekrarlayan sinir ağları (Recurrent Neural Network –RNN) ve geçitli tekrarlayan sinir ağları (Gated Recurrent Units –GRU) yöntemleri kullanılarak modeller

oluşturulmuş ve NSL-KDD veriseti üzerinde test edilmiştir. Üretilen modeller ve bu modellerin testleri sonucunda elde edilen sonuçlara ilişkin bilgi bu bölümde paylaşılmıştır. RNN, zaman serileri ve doğal dil gibi sıralı verilerin modellenmesinde kullanılan güçlü bir sinir ağı sınıfıdır [101]. Daha önce vurgulandığı gibi, LSTM algoritması çeşitli özellikler kazandırılmış bir RNN modelidir. LSTM model performansının RNN tabanlı diğer modeller ile kıyaslanması için basit RNN ve GRU modelleri üretilerek test edilmiştir.

3.9.1. RNN-IDS

İlk olarak 1990 yılında Jeff Elman tarafından sunulan RNN [102], ileri beslemeli YSA'dan farklı olarak, girdi dizisi süresince tutulan gizli bir durum (h) ve opsiyonel bir çıktı (y) tutar. Her işlem adımında RNN'in gizli durumu, önceki gizli durum ve o anki girdiyi birlikte işleyen non-lineer bir aktivasyon fonksiyonu ile güncellenir. Bu işlem girdi dizisi boyunca yapılır. Böylece, verisetinin işlenmesi süresince yapılan hesaplama işlemlerinde ağda üretilen önceki hesaplama sonuçları kullanarak kısa süreli hatırlama problemi başarı ile çözülür.

Bu model mimarisinde, katmanlardaki LSTM hücreleri yerine RNN hücreleri kullanılmıştır. RNN-IDS modelinin NSL-KDD verisetine, K-fold kullanılmadan uygulanması neticesinde elde edilen sonuçlar Çizelge 3.32'de sunulmuştur. Yapılan test işlemlerinin LSTM-NIDS modeli ile kıyaslanması açısından Çizelge 3.15'te yer alan aynı hiperparametre değerleri kullanılmıştır. Bununla birlikte, LSTM-NIDS modeline uygulanan düzenlileştirme işlemleri aynı şekilde RNN-IDS modeline de uygulanmıştır.

Çizelge 3.32. RNN-IDS modelinin testi neticesinde elde edilen sonuçlar.

RNN-IDS Modeli Test Sonuçları			
Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
Eğitim: 99,64 Doğrulama: 99,68 Test: **80,15**	Eğitim: 99,66 Doğrulama: 99,60 Test: 92,80	Eğitim: 99,53 Doğrulama: 99,66 Test: 71,22	

Model performansının arttırılması amacıyla K-fold yöntemi uygulanmış ve neticede elde edilen test sonuçları Çizelge 3.33'te sunulmuştur.

Çizelge 3.33. SimpleRNN-IDS modelinin K-fold yöntemi ile testi neticesinde elde edilen sonuçlar.

Değerler	Katman				Ort.
	1	2	3	4	
Kayıp (Loss)	1,06	1,20	1,05	1,32	1,16
Kesinlik (Accuracy)	%80,06	%79,62	%80,66	%77,87	%79,55
Hassasiyet (Precision)	%92,64	%92,68	%92,75	%92,53	%92,65
Duyarlılık (Recall)	%70,83	%70,01	%71,97	%67,07	%69,97

RNN-IDS modeli ile gerçekleştirilen sonuçlar incelendiğinde, LSTM-NIDS modeline oranla daha düşük değerler elde edildiği görülmektedir. Ayrıca K-fold yönteminin model başarısına olumlu yönde bir etkisinin olmadığı gözlemlenmiştir.

3.9.2. GRU-IDS

GRU, Cho ve diğerleri tarafından 2014 yılında LSTM'den ilham alınarak öne sürülen bir gizli katman yapısıdır [103]. LSTM'den farkı, daha basit uygulanabilir ve hesaplanabilir olmasıdır. Yapı içerisinde bir reset kapısı bulunmaktadır ve bu kapı değeri eşitlik 3.20 ile hesaplanmaktadır.

$$r_j = \sigma\left([W_r x]_j + [U_r h_{(t-1)}]_j \right) \tag{3.20}$$

Bu eşitlikte σ lojistik sigmoid fonksiyonu, $[.]_j$ bir vektörün j'inci elemanı, x girdi değeri, $h_{(t-1)}$ önceki gizli durum, W_r ve U_r öğrenilen ağırlık matrislerini göstermektedir. GRU içerisinde bir de güncelleme kapısı yer almaktadır. Bu kapı değeri öncekine benzer şekilde eşitlik 3.21 ile hesaplanmaktadır.

$$z_j = \sigma\left([W_z x]_j + [U_z h_{(t-1)}]_j \right) \tag{3.21}$$

Bu durumda önerilen gizli katmanın aktivasyonu eşitlik 3.22 ile hesaplanmaktadır.

$$h_j^{(t)} = z_j\, h_j^{(t-1)} + (1 - z_j)\tilde{h}_j^{(t)} \tag{3.22}$$

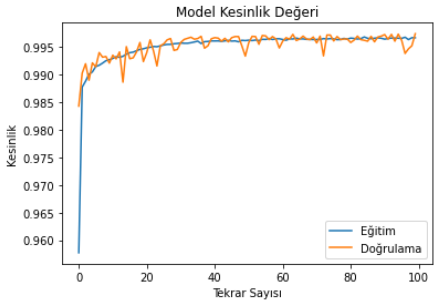
Bu eşitlikte yer alan $\tilde{h}_j^{(t)}$ değeri ise eşitlik 3.23 ile hesaplanmaktadır.

$$\tilde{h}_j^{(t)} = \phi\left([Wx]_j + [U\,(r \,.\, h_{(t-1)})]_j \right) \tag{3.23}$$

Bu formülde reset kapısı 0'a yakın olması durumunda, gizli durum önceki gizli durumu göz ardı edecek şekilde zorlanmakta ve sadece mevcut girdi değeri ile resetlenmektedir. Bu sayede gizli durum ilerde kullanılması gerekmeyen bilgiyi atmayı sağlamaktadır. Diğer taraftan güncelleme kapısı önceki gizli durumdan ne kadar bilginin mevcut gizli duruma aktarılacağını kontrol etmektedir. Bu, LSTM'deki hafıza kapısı gibi çalışmakta ve RNN'in bilgiyi bir süre hatırlamasını sağlamaktadır.

Tez kapsamında üretilen GRU-IDS model mimarisinde, SimpleRNN-IDS mimarisine benzer şekilde katmanlardaki LSTM hücreleri yerine GRU hücreleri eklenmiştir. GRU-IDS modelinin NSL-KDD verisetine, K-fold kullanılmadan uygulanması neticesinde elde edilen sonuçlar Çizelge 3.34'te sunulmuştur. Yapılan test işlemlerinin LSTM-NIDS modeli ile kıyaslanması açısından Çizelge 3.15'te yer alan aynı hiperparametre değerleri kullanılmıştır. Bununla birlikte, LSTM-NIDS modeline uygulanan düzenlileştirme işlemleri aynı şekilde GRU-IDS modeline de uygulanmıştır.

Çizelge 3.34. GRU-IDS modelinin testi neticesinde elde edilen sonuçlar.

GRU-IDS Modeli Test Sonuçları			
Kesinlik (Accuracy) %	Hassasiyet (Precision) %	Duyarlılık (Recall) %	Model Kesinlik Grafiği
Eğitim: 99,66 Doğrulama: 99,74 Test: **79,48**	Eğitim: 99,70 Doğrulama: 99,74 Test: 92,59	Eğitim: 99,60 Doğrulama: 99,66 Test: 69,70	

Model performansının arttırılması amacıyla K-fold yöntemi uygulanmış ve neticede elde edilen test sonuçları Çizelge 3.35'te sunulmuştur.

Çizelge 3.35.GRU-IDS modelinin K-fold ile testi sonucu elde edilen sonuçlar.

Değerler	Katman				Ort.
	1	2	3	4	
Kayıp (Loss)	1,12	0,90	0,93	0,89	0,96
Kesinlik (Accuracy)	%80,31	%80,63	%79,14	%80,21	%80,07
Hassasiyet (Precision)	%96,14	%92,61	%92,49	%92,42	**%93,15**
Duyarlılık (Recall)	%68,72	%72,12	%69,43	%71,32	**%70,39**

3.9.3. RNN, GRU ve LSTM modellerinin kıyaslanması

Simple RNN, GRU ve LSTM metotları kullanılarak oluşturulan modellerin orijinal NSL-KDD veriseti üzerinde aynı hiper-parametre değerleri kullanılarak testi neticesinde elde edilen sonuçlar, Çizelge 3.36'da sunulmuştur.

Çizelge 3.36. RNN tabanlı .

Metot	Kesinlik (%)	Hassasiyet (%)	Duyarlılık (%)
LSTM	**83.18**	**94.58**	**75.77**
GRU	80.07	93.15	70.39
simpleRNN	79.55	92.65	69.97

Elde edilen sonuçlar değerlendirildiğinde;

LSTM metodunun diğer RNN tabanlı modellere oranla oldukça başarılı sonuç verdiği gözlemlenmiştir. Bu gözleme dayalı olarak, uzun dönem hafıza kullanımının IDS performansına olumlu etki ettiği sonucu çıkarılmıştır.

GRU modelinin daha iyi bir sonuç vermesini beklememize karşı, simpleRNN modeli ile kıyaslandığında belirgin bir fark olmadığı görülmüştür. Bu nedenle LSTM'in IDS probleminin çözümü için daha iyi bir mekanizmaya sahip olduğu değerlendirilmiştir.

Modellerin çalışma sürelerine yönelik gerçekleştirdiğimiz gözlem neticesinde, LSTM modelinin en uzun çalışma süresine ihtiyaç duyduğu, bunu sırasıyla GRU ve simpleRNN modelinin takip ettiği belirlenmiştir. SimpleRNN modelinin diğer modellere kıyasla oldukça kısa sürede sonuç verdiği belirlenmiştir. Bu nedenle eğitim süresinin kısaltılmasına ihtiyaç duyulan durumlarda, SimpleRNN modelinin tercih edilebileceği sonucu çıkarılmıştır.

Veriseti açısından yapılan değerlendirmede, LSTM modeli ile alınan sonuçların başarılı olmasının NSL-KDD verisetinin içerdiği özellikler ile uzun dönem unutma problemine sahip olduğu sonucu çıkarılmıştır.

Literatür incelemesinde, LSTM modelinin sıklıkla görüntü işleme ve ses tanıma gibi problemlerde kullanıldığı görülmüş, çalışmamız neticesinde IDS geliştirme işlemlerinde de başarı ile kullanılabileceği belirlenmiştir.

4. VERİ GÖRSELLEŞTİRME VE CNN İLE SIZMA TESPİTİ

Derin öğrenme metotlarının görüntü işleme ve tanımada etkin olarak kullanıldığı bilinmektedir. Derin öğrenme yöntemlerinin görüntü işleme kabiliyetinin ağ sızma tespitinde kullanımına ilişkin değerlendirme yapılabilmesi amacıyla tez çalışmasının bu bölümünde, nümerik verinin işlenebilir formatta görselleştirilmesi, bu verinin evrişimli sinir ağı tarafından kullanılabilir şekilde sınıflandırılması, evrişimli sinir ağı modelinin oluşturulması, model iyileştirme çalışmalarının yapılması ve test işlemleri neticesinde elde edilen sonuçlar paylaşılmaktadır. Bununla birlikte bu bölümde, tez çalışmasının literatüre bir diğer katkısı olarak değerlendirilen, görüntü verisinin işlenmesine ilişkin teknik hakkında bilgi yer almaktadır.

Evrişimli sinir ağları derin ileri beslemeli sinir ağlarının bir sınıfı olup seyrek bağlantı ve ağırlık paylaşımı karakteristiklerine sahiptir. Tipik bir evrişimli sinir ağının yapısında giriş katmanı, evrişim katmanı, ortaklama katmanı, tam bağlantılı katman ve çıkış katmanı yer almaktadır. Görüntünün tamamı yerine bir bölümünün işlenmesi ile hesaplama hız ve gücünün arttırılması hedeflenmektedir. Evrişimli sinir ağları tarafından kullanılan görüntüler genelde yükseklik (Y), genişlik (G) ve kanal sayısını (K) içermekte ve *Y x G x K* formatında tutulmaktadır. Kanal sayısının üç olması durumunda renkli, bir olması durumunda ise gri skaladaki görüntüler kullanılmaktadır.

Bir evrişimsel ağ, girdi ve özellik haritasından istifade ile görüntüye ait lokal bir bölgenin özelliklerini kavrayabilmektedir. Özellik haritası, evrişimsel çekirdekler (convolution kernels) olarak isimlendirilen filtreler aracılığıyla, bir görüntüden elde edilen özellikleri içermektedir. Bu durumda farklı evrişimsel çekirdeklerin, farklı özellik çıkarıcılar ile elde edildikleri söylenebilir. Özellik haritası *M (yükseklik) x N (genişlik)* olacak şekilde iki boyutludur.

4.1. Evrişimli Sinir Ağı Modellerine İlişkin Literatür İncelemesi

Tezin önceki bölümünde LSTM, GRU ve RNN modelleri geliştirilerek testler gerçekleştirilmiş ve elde edilen sonuçlar göz önünde bulundurularak uzun kısa-dönem bellek etkisi araştırılmıştır. Bununla birlikte üç farklı özellik çıkarım algoritması ile azaltılmış veri

setleri elde edilmiş, bu setler ile yapılan testler neticesinde kaynak ve zaman kısıtı olan durumlarda özellik çıkarım algoritmalarının, performanstan yapılacak feragat ile ağ sızma tespiti geliştirme çalışmalarında kullanılabileceği sonucu paylaşılmıştır.

Tezin bu bölümünde, derin öğrenmenin başarı ile kullanıldığı görüntü tanıma alanına odaklanılmıştır. Bu kapsamda, literatürde yaygın olarak kullanılan evrişimli sinir ağları ile ilgili inceleme yapılmış, görüntü işleme ve tanıma özelliklerinin tez kapsamında ele alınan probleme nasıl uyarlanabileceği araştırılmıştır. Yapılan araştırma neticesinde, sayısal verinin görüntüye dönüştürülmesi ile ilgili çalışmalara ulaşılmıştır. İncelenen çalışmalardan öne çıkanlara ilişkin bilgi aşağıda sunulmuştur.

Wu ve diğerleri tarafından gerçekleştirilen çalışmada [104], ham veri setinden trafik özelliklerini otomatik olarak seçmek için CNN kullanıldığı ve dengesiz veri seti problemini çözmek için her sınıfın maliyet fonksiyonu ağırlık katsayısına bağlı olarak kayıt sayılarını belirledikleri ifade edilmektedir. Modelin yanlış alarm oranını (False Alarm Rate –FAR) düşürmekle kalmayıp, verisetinde düşük sayıda yer alan kayıtlara yönelik sınıflandırma kesinliğini de arttırdığı belirtilmektedir. Hesaplama maliyetinin düşürülmesi için nümerik verinin imaj formatına dönüştürüldüğü, model performansının değerlendirilmesi için standart NSL-KDD verisetinin kullanıldığı, deney sonuçlarının, kesinlik, FAR ve hesaplama maliyeti açılarından diğer standart algoritmalara oranla daha iyi sonuç verdiği ifade edilmektedir. Kullanılan CNN mimarisi incelendiğinde bir giriş katmanı, iki evrişim, iki ortaklama katmanı, bir tam bağlantılı katman ve bir çıkış katmanı içerdiği görülmüştür. Nümerikleştirme ve normalizasyon işlemleri neticesinde toplam 122 özellikli bir veriseti elde ettikleri, bunun 11*11 boyutlu bir imaja dönüştürüldüğü, bu nedenle 122 özellikten birinin çıkarılması gerektiği, özellik çıkarma işleminin coefficient of varriance (CV) yöntemi ile gerçekleştirildiği belirtilmektedir. Bir başka ifade ile, en düşük CV değerine sahip özelliğin verisetinden çıkarıldığı vurgulanmaktadır. Eğitim ve test işlemlerini GPU içermeyen, linux Red HAT işletim sistemine sahip bir bilgisayar üzerinde yaptıkları, 200 epoch ve 50 batch size parametrelerini kullandıkları, KDDTest+ veriseti ile yapılan test neticesinde %79,48 oranında bir kesinlikle sınıflandırma yapıldığı belirtilmektedir. Bu oranın RNN hariç diğer modellerden daha iyi olduğu, ayrıca modelin FAR değerinin 0.2790 ile RNN FAR değeri olan 0.2689'dan daha iyi olduğu vurgulanmaktadır.

Jiang ve diğerleri tarafından yapılan çalışmada [105], verinin dengesiz dağılımı problemini gidermek için tek taraflı seçim (One-side selection –OSS) yöntemi ile, verisetinde fazla sayıda kategorideki verinin azaltıldığı, az sayıda veri olan kategorideki veri sayısının arttırılması için ise Synthetic Minority Over-sampling Technique (SMOTE) yönteminin kullanıldığı vurgulanmaktadır. Böylece dengeli bir verisetinin üretildiği, az sayıda örnek olan atak kategorilerinin öğrenildiği ve model eğitim süresinin belirgin seviyede azaltıldığı vurgulanmaktadır. CNN modelinde kullanabilmek amacıyla 11*11 boyutlu, RGB = 1 olarak belirlenmiş imajların kullanıldığı belirtilmektedir. Burada da önceki çalışmada olduğu gibi NSL-KDD verisetinin 121 özellik içeren formata dönüştürüldüğü ifade edilmektedir. Çalışma kapsamında CNN ve iki yönlü LSTM (BiLSTM) modelleri kullanılarak hiyerarşik bir derin ağ modelinin oluşturulduğu, modelin NSL-KDD ile testi neticesinde %83,58 oranında kesinlik değerinin elde edildiği ifade edilmektedir. Modelin Tensorflow ortamında Python programlama dili kullanılarak geliştirildiği, öğrenme oranının 0.001, dropout oranının 0.5, epoch değerinin 100 ve batch size değerinin 128 olarak belirlendiği vurgulanmaktadır. Eğitim ve test işlemlerinin GPU içermeyen, Windows 10 işletim sistemine sahip bir bilgisayar üzerinde gerçekleştirildiği vurgulanmaktadır. Test sonuçları incelendiğinde, geliştirilen model CNN-BiLSTM'in, Random Forest, AlexNet, LeNet-5, CNN ve BiLSTM modellerine oranla daha iyi sonuç verdiği savunulmaktadır. Veriseti dengeleme işlemi neticesinin de etkisi ile NSL-KDD verisetinde yer alan normal, U2R ve R2L sınıfındaki kayıtların diğer modellere oranla çok daha iyi sınıflandırıldıkları vurgulanmaktadır. Çalışma süresi açısından değerlendirildiğinde CNN-BiLSTM'in LeNet-5 hariç diğer modellere oranla oldukça hızlı sonuç ürettiği, LeNet-5'ten ise kesinlik değeri açısından oldukça üstün olduğu ifade edilmektedir.

Hu ve diğerleri tarafından yapılan çalışmada [106], adaptive synthetic sampling (ADASYN) ve geliştirilmiş CNN tabanlı bir ağ sızma tespit sistemi önerilmektedir. NSL-KDD verisetinde bulunan dengesiz veri dağılımı probleminin çözümü için ADASYN metodunun kullanıldığı, daha sonra, dağıtılmış evrişim modüllerine dayalı geliştirilmiş CNN ile özellik çeşitliliğinin arttırıldığı ve ayrıca eğitimde kullanılan gereksiz verinin elimine edildiği ifade edilmektedir. Elde edilen sonuçlar değerlendirildiğinde, geleneksel CNN ve RNN modellerine oranla sırasıyla %4,6 ve %2,79 oranlarında iyileşme elde edildiği vurgulanmaktadır. Model geliştirme, eğitim ve test işlemlerinin Ubuntu 16.04 işletim sisteminde, 4*GTX 1080Ti GPU içeren bir bilgisayarda gerçekleştirildiği, öğrenme oranının 0.1, dropout oranının 0.5, epoch değerinin 200 ve batch size değerinin 50 olarak belirlendiği

belirtilmektedir. SPC-CNN yöntemi kullanılarak KDDTest+ veriseti ile yapılan test işlemleri neticesinde %83.83 oranında bir kesinlik değeri elde edildiği, bu oranın normal CNN ile elde edilen %79.48 kesinlik değerinden iyi olduğu vurgulanmaktadır. AS-CNN için ise bu sonucun %84.08 olduğu ifade edilmektedir. Verisetinde yer alan kayıtların dengeli hale getirilmesi neticesinde özellikle az sayıda kayda sahip R2L kotegorisindeki atakların, RNN, CNN, CL-CNN ve MS-CNN algoritmalarına oranla belirgin derecede daha iyi sınıflandırıldığı vurgulanmaktadır.

Lin ve diğerleri tarafından gerçekleştirilen çalışmada [107], CNN tabanlı bir model önerilmekte, NSL-KDD veriseti üzerinde yapılan test işlemleri neticesinde diğer modellere oranla daha iyi bir sonuç elde edildiği savunulmaktadır. Modelin başarısı özellik seviyesinde yapılan ön işlemlere dayandırılmaktadır. Bu çalışmada diğer CNN tabanlı modellerin üretildiği çalışmalara oranla verisetinde farklılık bulunmaktadır. Yazarlar verisetindeki sembolik değerleri nümerikleştirmeden her kaydı bir satır ve dolayısıyla CNN için bir imaj olarak işleme tabi tutmaktadır. Eğitim ve test işlemlerinin Core i7-2600 işlemcili, GTX 1080 GPU'ya sahip, Linux işletim sistemi olan bir bilgisayarda gerçekleştirildiği ifade edilmektedir. Eğitim ve test işlemlerinin Weka ortamında gerçekleştirildiği vurgulanmaktadır. Batch size değerinin 128, epoch değerinin 100 olarak belirlendiği, optimizer olarak adam, maliyet fonksiyonu olarak ise cross entropy kullanıldığı vurgulanmaktadır. İkili sınıflandırma için 0.0005, çoklu sınıflandırma için ise 0.0001 öğrenme oranlarının kullanıldığı ifade edilmektedir. KDDTest+ veriseti ile yapılan test işlemleri neticesinde %85.07 oranında kesinlik değerine ulaşıldığı ifade edilmektedir. Bu sonucun J48, Naive Bayes, NB Tree, Random Forest, Multilayer Perceptron ve SVM sınıflandırıcılara oranla daha iyi olduğu ifade edilmektedir.

Hsu ve diğerleri tarafından gerçekleştirilen çalışmada [108], LSTM tabanlı bir CNN modeli önerilmektedir. Model incelendiğinde, iki evrişim katmanı ile bir ortaklama katmanının ardından iki LSTM katmanından oluşan bir mimari görülmektedir. Eğitim ve test işlemlerinin GPU içeren Ubuntu işletim sistemine sahip bir bilgisayar üzerinde gerçekleştirildiği, ağırlık değerlerinin ayarlanmasında stochastic gradient descent (SDG), maliyet fonksiyonu olarak ise cross-entropy kullanıldığı vurgulanmaktadır. Batch size 64, epoch 100, öğrenme oranı 0.01, olarak belirlendiği ifade edilmektedir. KDDTest+ veriseti ile gerçekleştirilen test işlemleri neticesinde %89.23 oranında bir kesinlik değeri elde edildiği vurgulanmaktadır. Bununla birlikte, CNN-LSTM modelinin LSTM modeline oranla

daha kısa sürede sonuç verdiği ifade edilmektedir. Bu çalışmada CNN kullanımına özgü bir imaj verisi üretimi gerçekleştirilmediği, sadece nümerikleştirme ve normalizasyon işlemlerinin yapıldığı görülmektedir.

Bunların yanısıra CNN kullanılan başka çalışmalar da incelenmiştir [109-116]. Çalışmaların genelinde kullanılan verisetlerinin çeşitli yöntemlerle imaj verisetlerine dönüştürülerek CNN tarafından kullanılabilir uygun formata dönüştürüldüğü görülmektedir. Bununla birlikte, CNN tabanlı modeller ile KDDTest+ veriseti üzerinde yapılan testler neticesinde en yüksek %89.23 oranında bir kesinlik değeri elde edildiği belirlenmiştir [108]. İncelenen çalışmaların çoğunda veriseti üzerinde K-fold cross validation ve model üzerinde düzenlileştirme işlemlerinin uygulanmadığı görülmektedir. Genelde sayısal verinin imaj verisetine dönüştürülmesi aşamasında NSL-KDD veriseti özellik sayısına bağlı olarak, numerik veriden 11*11 boyutlu imajların üretildiği belirlenmiştir.

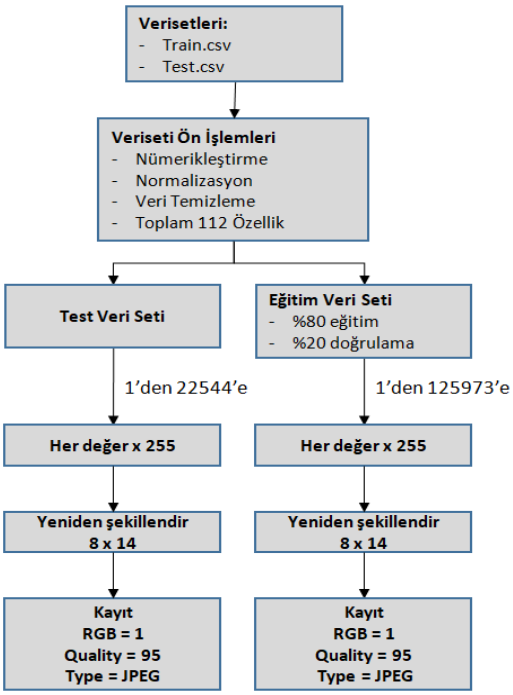
4.2. İmaj Veriseti Üretimi

Yapılan literatür taraması neticesinde, sayısal verinin imaj verisine dönüştürülmesi ve model düzenlileştirme ile yapılacak geliştirmelerin model performansına pozitif etki edeceği değerlendirilmiş ve bu doğrultuda çalışma yürütülmüştür. Tezin bu bölümünde NSL-KDD verisetinin nümerik içeriğinin imajlardan oluşan verisetlerine dönüştürülmesi sürecine ilişkin bilgi sunulmaktadır.

Daha önce vurugulandığı gibi bu çalışmada, NSL-KDD veriseti model başarısı karşılaştırma veriseti olarak kulanılmıştır. NSL-KDD verisetinde beş grup veri bulunmaktadır. Verisetinde yer alan veri miktarına ilişkin bilgi Çizelge 2.8'de görülmektedir. Çizelge incelendiğinde, özellikle U2R kategorisindeki atak sayısının eğitim ve test verisetlerinde içerik olarak oldukça az olduğu görülmektedir.Yapılan eğitim ve test işlemlerinde bu kategorideki veri miktarının az olması model genel performansına olumsuz etki etmekte ve kesinlik değerini düşürmektedir. Yapılan literatür incelemesinde, bu düzensiz veri dağılımını düzenli hale getirmek için çeşitli yöntemlerle veri sentezleme işlemlerinin gerçekleştirildiği ve model performansının suni bir şekilde arttırıldığı belirlenmiştir. Sentetik veri üretiminin bazı problemlerin çözümüne katkı sağladığı bilinmekle beraber, ağ sızma tespitinde kullanılmasının doğru bir yaklaşım olmadığı değerlendirilmektedir. Çünkü sentetik veri üretimi ile veriseti orijinalliğinin bozulduğu, böylece model performansının

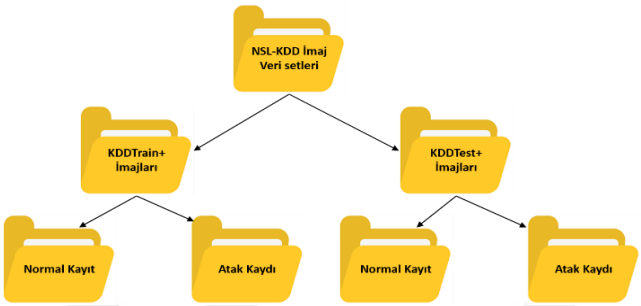
arttığı, ancak modelin gerçek ağ trafik ortamında uygulanabilecek seyrek ataklara karşı zayıf hale geleceği değerlendirilmiştir. Bu nedenle, bu tez çalışmasında sentetik veri üretimi ile model performansının arttırılması yoluna başvurmak yerine, veri gösterimi ve model iyileştirmeye ilişkin geliştirme çalışmalarına önem verilmiştir.

NSL-KDD veriseti ön işlemlerinin CNN için kullanılabilir uygun formata dönüştürülmesine ilişkin verisetindeki sembolik verinin nümerikleştirilmesi ve tüm değerlerin [0,1] aralığına haritalanması amacıyla veriseti ön işlemleri gerçekleştirilmiştir. Yapılan işlemler neticesinde, 112 özellik alanı ve bir etiket içeren veriseti elde edilmiştir. Elde edilen NSL-KDD eğitim ve test verisetlerinin CNN modelinde kullanılabilmeleri için görüntü verisetlerine dönüştürülmeleri gerekmektedir. Bu kapsamda yukarıda incelenen çalışmalardan farklı olarak elde edilen 112 özellik alanına sahip kayıtlar, 8x14 boyutlu görüntülere dönüştürülmüştür. Bu kapsamda CSV dosyasındaki her bir kayıt ayrı ayrı alınarak işlenmiştir. Her kaydın her alanı, görüntüye dönüştürme işleminden önce, toplam 255 piksel değeri olması nedeniyle 255 ile çarpılmıştır. RGB ve kalite değerleri sırasıyla 1 ve 95 olarak belirlenmiş, elde edilen imajlar JPEG uzantılı olarak kaydedilmiştir. Daha sonra kayıtlar, etiket değerlerine bağlı olarak normal ve atak klasörlerine taşınmıştır. Bu sayede CNN modelinin kullanabileceği dosya formatında yeni verisetleri elde edilmiştir. Yapılan işlemlere ilişkin akış diyagramı Şekil 4.1'de sunulmuştur.



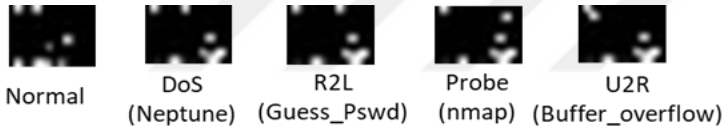
Şekil 4.1. İmaj verisetlerinin üretimi.

Üretilen imajların CNN modelinde kullanılabilmesi için uygun şekilde dosyalanması gerekmektedir. Bu nedenle eğitim ve test verisetleri, normal ve atak kayıtlarını içerecek şekilde hiyerarşik yapıda dosyalanmıştır. Yapılan dosyalama işlemine ilişkin şema Şekil 4.2'de sunulmuştur.



Şekil 4.2. İmaj veri setlerinin dosya hiyerarşisi.

Çalışma kapsamında normal, DoS, R2L, Probe ve U2R kategorilerinde imaj dosyaları üretilmiştir. Üretilen bu imajlara ilişkin yakınlaştırılmış örnekler Şekil 4.3'te sunulmuştur.

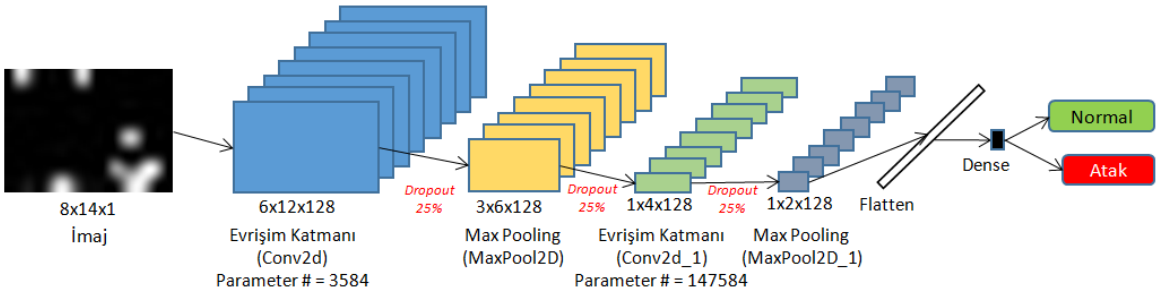


Şekil 4.3. Her kayıt kategorisine ilişkin imaj örnekleri.

İmaj dosyalarının üretilmesini takiben model geliştirme çalışmalarına başlanmıştır. Devam eden bölümde model geliştirme çalışmalarına ilişkin detaylı bilgi sunulmaktadır.

4.3. Geliştirilen Evrişimli Sinir Ağı Modeli

Bu çalışmada geliştirilen CNN modeli Şekil 4.4'te sunulmuştur. Model, iki evrişim katmanı, iki max pooling katmanı, bir flattened katmanı ve bir dense katmanı içermektedir. Ayrıca, convolution ve max pooling katmanları arasına, gereksiz ağırlıkların kontrol edilebilmesi için %25 dropout eklenmiştir. Şekil 4.4'te de görüldüğü gibi, model mimarisi basit ve anlaşılır bir yapıya sahiptir. Bu yapı sayesinde, modelin eğitimi ve sonuçlarının yorumlanması kolaylaşmıştır. Dropout katmanları da modelin performansını artırmakta ve ağırlıkların düzenlenmesine yardımcı olmaktadır. Bu şekilde, model daha düzgün ve tutarlı sonuçlar üretmektedir.



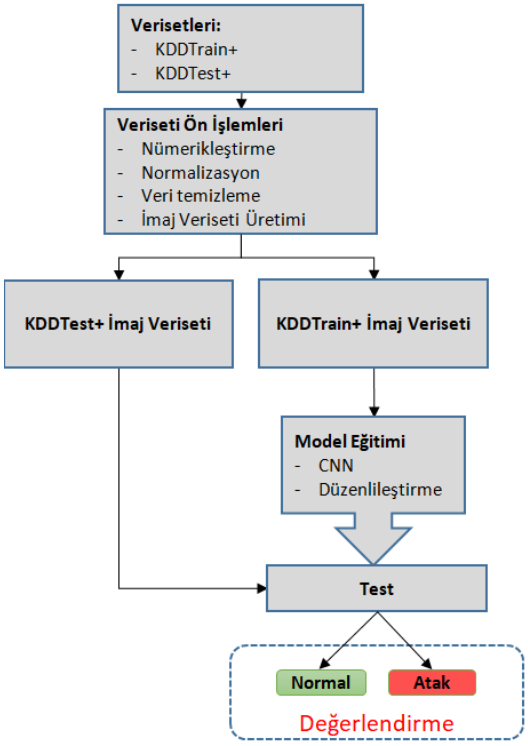
Şekil 4.4. Geliştirilen CNN modeli.

Modelde kullanılan hiperparametre değerleri ise Çizelge 4.1'de sunulmuştur.

Çizelge 4.1. CNN modelinde kullanılan hiperparametre değerleri.

Hiperparametre	Değer	Hiperparametre	Değer
Target size	*8x14*	*Regulizer*	*L2 (0.001)*
Batch size	*4*	*Dense activation*	*Sigmoid*
Class mode	*Binary*	*Optimizer*	*Rmsprop*
Input shape	*(4,8,14,3)*	*Learning rate*	*0.001*
Conv2D filters	*128*	*Epoch number*	*200*
Conv2D kernel size	*(3,3)*	*Steps per epochs*	*50*
Activation function	*Relu*	*Dropout rate*	*0.25*

Optimum hiperparametre değerleri belirlendikten sonra, model performansını daha da geliştirmek için çeşitli regularizasyon teknikleri incelenmiş ve çalışma kapsamına en yaygın kullanılan üç teknik dahil edilmiştir. Bu teknikler; model kapasitesinin azaltılması, ağırlıkların düzenlileştirilmesi ve dropout değerinin uygulanmasıdır. Model kapasitesi, iki katmanlı bir yapı ile en optimum haline getirilerek, aşırı öğrenmenin önüne geçilmiştir. Ayrıca, %25 dropout uygulanarak, modelin daha sağlam ve tutarlı bir performans sergilemesi sağlanmıştır. Bununla birlikte, kernel düzenleyici olarak L2 kullanılmış ve değeri 0.001 olarak belirlenmiştir. Bu düzenlileştirme işlemleri neticesinde, model RegCNN olarak adlandırılmıştır. Önerilen çalışma mimarisi, Şekil 4.5'te detaylı bir şekilde gösterilmektedir. Yapılan düzenlileştirme işlemleri neticesinde model, daha güvenilir sonuçlar elde etmek için kullanılabilir ve uygulanabilir bir yapıya kavuşturulmuştur.



Şekil 4.5. Geliştirilen CNN modeli.

4.4. Evrişimli Sinir Ağı Modeli Test Sonuçları

Model ile yapılan testler neticesinde elde edilen sonuçlar Çizelge 4.2'de sunulmuştur. Bu tabloda, tezin önceki bölümlerinde geliştirilen LSTM, GRU ve basit RNN modelleriyle elde edilen sonuçlar, CNN ile elde edilen sonuçlarla kıyaslanmaktadır. Tabloda, farklı modellerin başarı oranları yan yana gösterilerek, karşılaştırmalı bir analiz sunulmuştur.

Çizelge 4.2. CNN model sonuçlarının RNN tabanlı model sonuçları ile kıyaslanması.

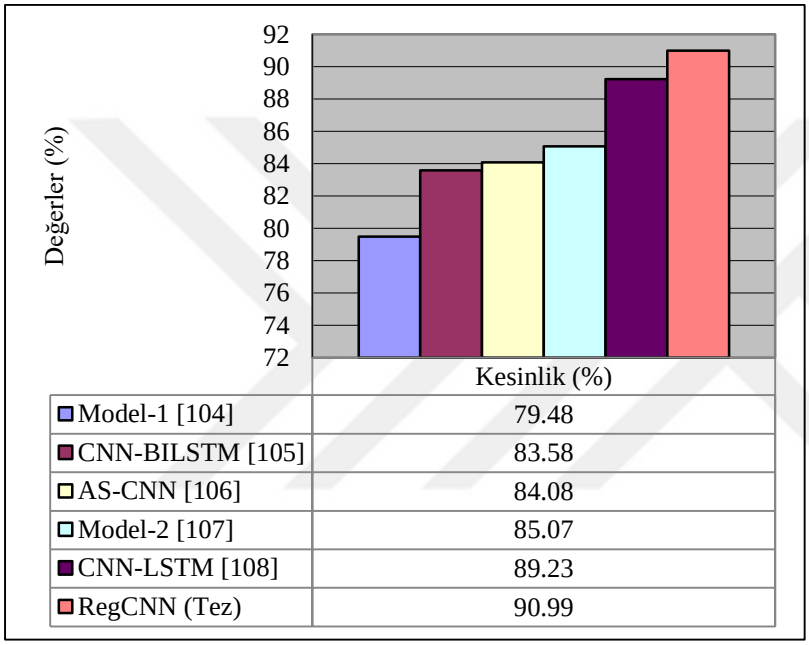
Metot	Kesinlik (%)	Hassasiyet (%)	Duyarlılık (%)	Süre (sn)
RegCNN	**90.99**	**95.94**	**89.55**	**5444**
LSTM	83.18	94.58	75.77	2894
GRU	80.07	93.15	70.39	2165
simpleRNN	79.55	92.65	69.97	1654

RegCNN modelinin test işlemleri sonucunda, Çizelge 4.2'de görüldüğü gibi, diğer modellere kıyasla oldukça yüksek bir kesinlik değeri elde edilmiştir. Bununla birlikte, bu modelin çalışma süresinin diğer modellere göre daha uzun olduğu belirlenmiştir. Görüntü işleme sürecinin daha fazla zaman alması nedeniyle ortaya çıkan bu farklılık, kaynak kapasitesinin

sınırlı olduğu ve büyük verisetleri ile çalışılması gereken durumlarda göz önünde bulundurulması gereken bir kısıttır.

Geliştirilen RegCNN modeli sonuçları, literatürdeki diğer CNN model sonuçları ile karşılaştırılmak üzere, Çizelge 4.3'te sunulmuştur. Çizelgeden de görüleceği üzere RegCNN modeli incelenen diğer CNN modellerine göre daha başarılı sonuç sağlamaktadır.

Çizelge 4.3. RegCNN modelinin incelenen diğer CNN modelleri ile kıyaslanması.



	Kesinlik (%)
Model-1 [104]	79.48
CNN-BILSTM [105]	83.58
AS-CNN [106]	84.08
Model-2 [107]	85.07
CNN-LSTM [108]	89.23
RegCNN (Tez)	90.99

Sonuç olarak tez çalışması kapsamında geliştirilen RegCNN modelini diğer CNN modellerinden daha başarılı olmasını sağlayan üç özellik olduğu belirlenmiştir. İlk özellik, veriseti ön işlemlerinin diğer çalışmalara göre farklılık arz etmesidir. Çalışmamızda, nümerikleştirme, normalizasyon ve veri temizleme işlemlerinin ardında toplam 112 özellik elde edilmiş ve bunların 8x14 boyutlu görüntülere dönüştürülmesini sağlanmıştır.Diğer çalışmalarda ise özellik sayısı 122 olarak elde edilmiştir.Bunlardan biri, manuel özellik çıkarım algoritmaları kullanılarak çıkarılmış vesonuçta 11x11 boyutlu görüntülerden yararlanılmıştır. Çalışmanın başarılı olmasını sağlayan ikinci özellik, RegCNN modeline, diğer modellere kıyasla çeşitli düzenlileştirme işlemleri uygulanarak performans artışının sağlanmasıdır.Üçüncü ve son özellik ise uzun süren test çalışmaları neticesinde elde edilen optimum hiperparametre değerlerinin model performansına olumlu etki etmesidir.

5. VERİSETLERİNE YÖNELİK İNCELEME VE ÖZGÜN VERİSETİ ÜRETİMİ

Derin öğrenme, donanımsal gelişmelerin yanı sıra güçlü algoritmaları sayesinde, klasik makine öğrenme yöntemlerine oranla daha iyi sınıflandırma sonuçları elde edilmesini sağlamıştır. Günümüzde derin öğrenme modellerinin hazır platform ve kütüphaneler yardımıyla rahatlıkla üretilebildiği, birçok yöntemin bağımsız ya da hibrit şekilde kullanıldığı, yapılan çalışmalarda genelde model mimarisi ve yöntemlere yoğunlaşıldığı, uygun model mimarisi ve hiperparametre optimizasyonu ile istenilen seviyede güçlü uygulamalar geliştirilebildiği görülmektedir. IDS geliştirme çalışmalarında, uygulama geliştirme tarafında yaşanan bu gelişmelere karşı, uygun nitelik ve nicelikte veri üretimi ve kullanımı halen bir problem olarak durmaktadır [117]. Ağ sızma tespit sistemlerinin gerçek anlamda değerlendirilmesi, kıyaslanması ve uygulanmasında sorunlar yaşanmakta olup bunun temel nedeni yeterli sayıda özgün veriseti olmamasıdır. Kurumlar tarafından üretilen verisetlerinin birçoğu güvenlik gerekçeleri ile paylaşılmamakta, erişilebilenlerin çoğu ise güncel saldırı trendlerini yansıtmamaktadır. DARPA'nın talebiyle MIT Lincoln Lab tarafından üretilen verisetleri ve Kaliforniya Üniversitesi tarafından üretilen KDD verisetlerinin ağ sızma tespit çalışmalarına büyük katkılarının olduğu, ancak kesinlik sorunu ve gerçek durumları yansıtmamaları nedeniyle eleştirildiği görülmektedir [118, 119]. Ağ davranış paternleri değiştikçe ve sızma yöntemleri geliştikçe, statik ve tek zamanlı verisetlerinin kullanımından; gücel trafik kompozisyonları ve sızma yöntemlerini yansıtan, değiştirilebilir, genişletilebilir ve yeniden üretilebilir dinamik verisetlerine geçmek gerektiği görülmektedir [120].

Ağ sızma tespitinde kullanılan belli başlı verisetleri mevcuttur ve yukarıda da belirtildiği gibi bunların sızma tespit sistemi geliştirme çalışmalarına önemli katkıları olmuştur. CAIDA (CAIDA, 2011), PREDICT (RTI International, 2011), The Internet Traffic Archive (Lawrence Berkeley National Lab, 2010), LNBL Traces (Lawrence Berkeley National Lab ve ICSI), DARPA verisetleri (MIT Lincoln Lab, 2011), KDD'99 verisetleri (Kaliforniya Üniversitesi, 2011), DEFCON (The Shmoo Group, 2011) [85], UNSW-NB15 (Avustralya Siber Güvenlik Merkezi, 2015) ve CIDDS-001 (Coburg Üniversitesi, 2017) [120] bu verisetlerinden bazılarıdır. Bu bölümde öncelikle, ağ sızma tespitinde kullanılan verisetlerinin özelliklerine yönelik bilgi sunulmakta, devamında özgün ağ sızma veriseti üretimine yönelik literatür inceleme sonuçları paylaşılmaktadır. Son olarak özgün veriseti

üretimi için oluşturulan mimarinin altyapı bileşenleri, ağ trafiğinden işlenmemiş veri elde etme yöntemleri ve elde edilen ham veriden model eğitiminde kullanılabilecek nitelikli veriseti üretimi için yapılan işlemler hakkında bilgi sunulmaktadır.

5.1. Ağ Sızma Tespit Verisetlerine Yönelik Literatür İncelemesi

Ağ sızma tespitinde kullanılan verisetleri, ağ trafiğinin izlenmesi ve bu trafikten elde edilen verinin işlenmesi ile üretilmektedir. Ağ trafiği paket tabanlı veya akış tabanlı olarak yakalanır. Paket tabanlı yakalama, portların ağ cihazları üzerinde aynalanması ile gerçekleştirilir. Yakalanan veri tüm yük bilgisini içerir. Paket tabanlı veri genelde PCAP formatındadır. Elde edilen veri, kullanılan ağ ve iletim protokolüne bağlıdır. Birçok iletim protokolü vardır ve bunların başlıcaları TCP, UDP, ICMP ve IP'dir.

Akış tabanlı yakalama ise daha kümelenmiş bir verinin elde edilmesini sağlar ve ağ bağlantılarındaki metadatayı içerir. Ağ üzerindeki host makineler arasındaki akış tek yönlü veya çift yönlü olabilir. Tek yönlü akışta, A hostundan B hostuna giden tüm paketlerin belirli özellikleri (zaman, süre, iletim protokolü, kaynak IP adresi, kaynak portu vb.) toplanır. Host B'den host A'ya giden paket bilgileri için bir başka tek yönlü akış oluşturulur. Buna karşı iki yönlü akışta, tüm paketler akış yönüne bakılmaksızın toplanır. Belli başlı akış tabanlı formatlar, NetFlow, IPFIX, sFlow ve OpenFlow'dur. Paket tabanlı verinin akış tabanlı veriye dönüştürülmesi mümkün olup, nfdump ve YAF gibi uygulamalarla bu işlem gerçekleştirilebilmektedir. Paket tabanlı ve akış tabanlı üretilen verisetlerinin dışında, akış tabanlı verisetlerinin, paket tabanlı veya host tabanlı veri ile zenginleştirilmesi ile elde edilen KDDCUP 99 veriseti gibi verisetleri de mevcuttur [86].

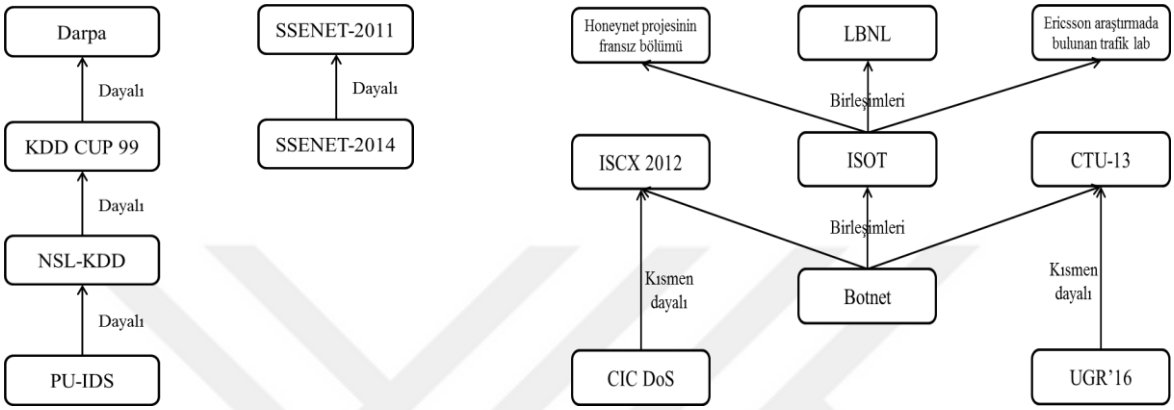
Hâlihazırda kullanılmakta olan birçok ağ sızma tespit veriseti vardır ve bunların üretim şekli ve özellikleri birbirinden farklı olabilir. Hatta bazı verisetleri ağ sızma tespitinde spesifik bir alan için geliştirilmiş olabilir. Bu nedenle kullanılan her verisetinin bağımsız değerledirilmesi gerekmektedir. Ağ sızma tespit verisetlerine yönelik yapılan bir çalışmada [120], veriseti özellikleri ve bunların değer aralıklarını içeren çizelge aşağıda sunulmuştur.

Çizelge 5.1. Veriseti özellikleri ve bunların değer aralıkları [120].

Veriseti Özelliklerinin Değer Aralıkları		
Genel Bilgi	Trafik oluşturma tarihi	Yıl (1998-2017)
	Genel Erişim	Hayır / Bilgi yok / Talep edilirse / Evet
	Normal trafik	Hayır / Evet
	Atak trafiği	Hayır / Tanımlı değil / Evet
Verisetinin Doğası	Metadata	Hayır / Biraz / Evet
	Format	İki yönlü akış / Loglar / Diğer / Paket / Tek yönlü akış
	Anonimlik	Yok / Tanımlı değil / Evet / Bazı özellikler için Evet
Veri Miktarı	Miktar	GB olarak data boyutu veya akış/paket/nokta sayısı
	Süre	Verisetinin kaydedilme süresi
Kayıt Ortamı	Trafik çeşidi	Emüle / Gerçek / Sentetik
	Ağ çeşidi	Farklı ağlar / Kurumsal ağ / Honeypotlar / İnternet Servis Sağlayıcı, Tanımlı değil / Küçük ağ / Kullanılan ağ / Üniversite ağı
	Tam ağ	Hayır / Tanımlı değil / Evet
Değerlendirme	Önceden tanınlı bölümler	Hayır / Tanımlı değil / Evet
	Dengeli	Hayır / Tanımlı değil / Evet
	Etiketli	Dolaylı / Hayır / Evet / Evet (IDS) / Arkaplan ile Evet

Çizelge incelendiğinde, bir veriseti ile ilgili bilinmesi gereken özelliklerin kapsandığı görülmektedir. Çalışmada tüm alanlarla ilgili gerekli açıklamalar ve bu alanlardaki bilgiye neden ihtiyaç duyulacağına dair bilgi sunulmaktadır. Örneğin, "veriseti oluşturma tarihi" bilgisinin, güncel atakların veriseti içerisinde kapsanması konusunda bir fikir verdiği, bir verisetinde "normal kullanıcı davranışı" bulunmaması durumunda ağ sızma tespit sisteminin değerlendirilmesinin mümkün olmadığı, bu durumda verisetinin tamamen kullanışsız hale gelmediği ancak bir başka veriseti veya gerçek bir ağ trafiği ile birleştirilmesi gerektiği ifade edilmektedir. Bununla birlikte "format bilgisi" ile verisetinin yukarıda belirtilen paket tabanlı, akış tabanlı veya diğer formatlardan hangisine ait olduğu,"veri boyutu" bilgisinin, içerilen paket, akış ve nokta sayısı ile birlikte veri setinin fiziksel boyutunu (GB) da içerdiği vurgulanmaktadır.

Çalışmada, ağ sızma tespit verisetleri ile ilgili önemli olarak değerlendirilen bir şema ve iki çizelge sunulmaktadır. Ağ sızma tespit verisetlerinin birbiri ile ilişkilerini gösteren şema Şekil 5.1'de yer almaktadır. Şekil incelendiğinde, verisetlerinin birbirine dayalı olarak üretildiği, bazı verisetlerinin ise birden çok verisetinin birleşimi ile elde edildiği görülmektedir.



Şekil 5.1. Ağ sızma tespit veri setlerinin birbirleri ile ilişkisi [120].

Ağ sızma tespiti amacıyla çeşitli verisetleri üretilmekte ve kullanılmaktadır. Bunların bir bölümü paket tabanlı, bir bölümü akış tabanlı, bazısı ikisini de içerecek şekilde olabilmektedir. Veri formatının yanısıra bir verisetinin sınıflandırılması ve değerlendirilmesini sağlayan verinin doğası, boyutu, veri kayıt ortamı, dengeli veya etiketli olma durumları gibi diğer paramatreler de mevcuttur. Bu kapsamda ağ sızma tespitinde kullanılan 34 adet verisetinin özelliklerini içeren bilgi, Çizelge 5.2'de sunulmaktadır.

Çizelge 5.2 incelendiğinde, üretilen en güncel verisetinin 2018 yılında üretilen PUF veriseti olduğu, verisetlerinin büyük bölümünün genel erişime açık olduğu, genelinin normal ve saldırı trafiği içerdiği, yarısına yakının metadata içermediği, paket ve akış formatında üretilen veriseti sayılarının birbirine yakın olduğu, genelde anonimlik bulunmadığı, bulunanlarda ise anonimliğin IP'ler ile sağlandığı, en büyük paket boyutunun 14 GB, en büyük akış boyutunun ise 150 GB olduğu, en fazla üç yıl, en az bir saat süreyle izlenen trafikten elde edilen veri olduğu, trafiğin genelde gerçek ve emüle ağlardan elde edildiği, genelde küçük ölçekli ağların dinlendiği, genelde tüm ağın dinlendiği, genelde önceden tanımlı bölümlerin olmadığı, genelde dengeli bir ağ olmadığı, genelde etiketli veri bulunduğu görülmektedir.

Çizelge 5.2 Ağ sızma verisetleri ve özellikleri [120].

	Genel Bilgi				Verinin Doğası			Veri Boyutu		Kayıt Ortamı			Değerlendirme		
Veriseti	**Trafik Yaratım Tarihi**	**Genel Erişim**	**Normal Trafik**	**Atak Trafiği**	**Metadata**	**Format**	**Anonimlik**	**Miktar**	**Süre**	**Trafik Çeşidi**	**Ağ Çeşidi**	**Tüm Ağ**	**Önceden Tanımlı Bölümler**	**Dengeli**	**Etiketli**
AWID	2015	Telep edilirse	Evet	Evet	Evet	Diğer	Yok	37M paketler	1 saat	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet
Booters	2013	Evet	Hayır	Evet	Hayır	Paket	Evet	250 GB paketler	2 gün	Gerçek	Küçük ağ	Hayır	Hayır	Hayır	Hayır
Botnet	2010/2014	Evet	Evet	Evet	Evet	Paket	Yok	14 GB paketler	Belirsiz	Emüle	Farklı ağlar	Evet	Evet	Hayır	Evet
CIC DoS	2012/2017	Evet	Evet	Evet	Hayır	Paket	Yok	4.6 GB paketler	1 gün	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
CICIDS 2017	2017	Evet	Evet	Evet	Evet	Paket, iki yönlü akış	Yok	3.1 M akışlar	5 gün	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
CIDDS-001	2017	Evet	Evet	Evet	Evet	Tek yönlü akış	Evet (IPler)	32 M akışlar	28 gün	Emüle ve gerçek	Küçük ağ	Evet	Hayır	Hayır	Evet
CIDDS-002	2017	Evet	Evet	Evet	Evet	Tek yönlü akış	Evet (IPler)	15 M akışlar	14 gün	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
CDX	2009	Evet	Evet	Evet	Evet	Paket	Yok	14 GB paketler	4 gün	Gerçek	Küçük ağ	Evet	Hayır	Hayır	Hayır
CTU-13	213	Evet	Evet	Evet	Evet	Tek ve iki yönlü akış, paket	Evet (Yük)	81 M akışlar	125 saat	Gerçek	Üniversite ağı	Evet	Hayır	Hayır	Evet (arka plan etiketleri ile)
DARPA	1998/99	Evet	Evet	Evet	Evet	Paket ve loglar	Yok	Tanımlı değil	7,5 hafta	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet
DDoS 2016	2016	Evet	Evet	Evet	Hayır	Paket	Evet (IPler)	2.1 M paketler	Belirsiz	Sentetik	Belirsiz	Belirsiz	Hayır	Hayır	Evet
IRSC	2015	Hayır	Evet	Evet	Hayır	Paket, akış	Belirsiz	Belirsiz	Belirsiz	Gerçek	Üretim ağı	Evet	Belirsiz	Belirsiz	Evet
ISCX 2012	2012	Evet	Evet	Evet	Evet	Paket, iki yönlü akış	Yok	2 M akışlar	7 gün	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
ISOT	2010	Evet	Evet	Evet	Evet	Paket	Yok	11 GB paketler	Belirsiz	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
KDD CUP 99	1998	Evet	Evet	Evet	Hayır	Diğer	Yok	5 M noktalar	Belirsiz	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet
Kent 2016	2016	Evet	Evet	Belirsiz	Hayır	İki yönlü akış ve loglar	Evet (Ipler, portlar ve tarih)	130 M akışlar	58 gün	Gerçek	Kurumsal Ağ	Evet	Hayır	Hayır	Hayır
Kyoto 2006+	2006 - 2009	Evet	Evet	Evet	Hayır	Diğer	Evet (IPler)	93 M noktalar	3 yıl	Gerçek	Bal küpleri	Hayır	Hayır	Hayır	Evet
LBNL	2004 - 2005	Evet	Evet	Evet	Hayır	Paket	Evet	160 M paketler	5 saat	Gerçek	Kurumsal Ağ	Evet	Hayır	Hayır	Hayır
NDSec-1	2016	Telep edilirse	Hayır	Evet	Hayır	Paket ve loglar	Yok	3.5 M paketler	Belirsiz	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
NGIDS-DS	2016	Evet	Evet	Evet	Hayır	Paket ve loglar	Yok	1 M paketler	5 gün	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
NSL-KDD	1998	Evet	Evet	Evet	Hayır	Diğer	Yok	150 K noktalar	Belirsiz	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet
PU-IDS	1998	Veri yok	Evet	Evet	Hayır	Diğer	Yok	200 K noktalar	Belirsiz	Sentetik	Küçük ağ	Evet	Hayır	Hayır	Evet
PUF	2018	Veri yok	Evet	Evet	Hayır	Tek yönlü akış	Evet (IPler)	300 K akışlar	3 gün	Gerçek	Üniversite ağı	Hayır	Hayır	Hayır	Evet (IDS)
SANTA	2014	Hayır	Evet	Evet	Hayır	Diğer	Evet (Yük)	Belirsiz	Belirsiz	Gerçek	ISP	Evet	Belirsiz	Hayır	Evet
SSENET-2011	2011	Veri yok	Evet	Evet	Hayır	Diğer	Yok	Belirsiz	4 Saat	Emüle	Küçük ağ	Evet	Hayır	Hayır	Evet
SSENET-2014	2011	Veri yok	Evet	Evet	Hayır	Diğer	Yok	200 K noktalar	4 Saat	Emüle	Küçük ağ	Evet	Evet	Evet	Evet
SSHCure	2013 - 2014	Evet	Evet	Evet	Hayır	Tek ve iki yönlü akış, loglar	Evet (IPler)	2.4 GB akışlar (sıkıştırılmış)	2 ay	Gerçek	Üniversite ağı	Evet	Hayır	Hayır	Dolaylı
TRAbID	2017	Evet	Evet	Evet	Hayır	Paket	Evet (IPler)	460 M paketler	8 saat	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet
TUIDS	2011 - 2012	Telep edilirse	Evet	Evet	Hayır	Paket ve iki yönlü akış	Yok	250 K akışlar	21 gün	Emüle	Orta ağ	Evet	Evet	Hayır	Evet
Twente	2008	Evet	Hayır	Evet	Evet	İki yönlü akış	Evet (IPler)	14 M akışlar	6 gün	Gerçek	Bal küpleri	Hayır	Hayır	Hayır	Evet
UGR'16	2016	Evet	Evet	Evet	Biraz	İki yönlü akış	Evet (IPler)	16900 M akışlar	4 ay	Gerçek	ISP	Evet	Evet	Hayır	Evet (arka plan etiketleri ile)
UNIBS	2009	Telep edilirse	Evet	Hayır	Hayır	Akış	Evet (IPler)	79 K akışlar	3 gün	Gerçek	Üniversite ağı	Evet	Hayır	Hayır	Hayır
Unified Host and Network	2017	Evet	Evet	Belirsiz	Hayır	İki yönlü akış ve loglar	Evet (Ipler ve tarihler)	150 GB akışlar (sıkıştırılmış)	90 gün	Gerçek	Kurumsal Ağ	Evet	Hayır	Hayır	Hayır
UNSW-NB15	2015	Evet	Evet	Evet	Evet	Paket ve diğer	Yok	2 M noktalar	31 saat	Emüle	Küçük ağ	Evet	Evet	Hayır	Evet

Yukarıda belirtilen ağ sızma tespit verisetlerinin kullanıldığı belirli uygulamalar mevcuttur. Çizelge 5.3'te, ağ sızma tespit verisetlerinin kullanıldığı spesifik saldırılara dair bilgi ve kullanılan araçlar sunulmaktadır. Herhangi bir saldırı tespit sistemi geliştirilmesi durumunda uygun veriseti seçimi için bu tablodan yararlanılabileceği değerlendirilmektedir.

Çizelge 5.3. Ağ sızma verisetlerinin kullanıldığı uygulamalar [120].

Veriseti	Saldırılar
AWID	802.11 protokolüne yönelik yetki talebi, ARP baskını, injection, inceleme talebi gibi popüler saldırılar.
Booters	Dokuz farklı DDoS saldırısı.
Botnet	Botnetler (Menti, Murlo, Neris, NSIS, Rbot, Sogou, Strom, Virut, Zeus)
CIC DoS	Uygulama katmanı DoS saldırıları (ddossim, Goldeneye, hulk, RUDY, Slowhttptest veSlowloris ile uygulanan)
CICIDS 2017	Botnet (Ares), cross-site-scripting, DoS (Hulk, GoldenEye, Slowloris, ve Slowhttptest ile uygulanan), DDoS (LOIC ile uygulanan), heartbleed, infiltration, SSH brute force, SQL injection
CIDDS-001	DoS, port taramaları (ping-scan, SYN-Scan), SSH brute force
CIDDS-002	Port taramaları (ACK-Taraması, FIN-Taraması, ping-Taraması, UDP-Taraması, SYN-Taraması)
CDX	Tanımlı değil
CTU-13	Botnetler (Menti, Murlo, Neris, NSIS, Rbot, Sogou, Virut)
DARPA	DoS, Yetki artırma (uzaktan lokale ve kullanıcıdan root'a), inceleme
DDoS 2016	DDoS (HTTP baskını, SIDDOS, smurf ICMP baskını, UDP baskını)
IRSC	Tanımlı değil
ISCX 2012	Dört atak senaryosu (1: Ağa içten sızma; 2: HTTP DoS; 3: IRC botnet kullanarak DDoS; 4: SSH kaba kuvvet)
ISOT	Botnet (Storm, Waledac)
KDDCUP 99	DoS, yetki artırma(uzaktan lokale ve kullanıcıdan root'a), inceleme
Kent 2016	Tanımlı değil
Kyoto 2006+	Bal küplerine karşı çeşitli saldırılar (Örneğin; backscatter,DoS, exploits, malware, port taramaları, shellcode)
LBNL	Port taramaları
NDSec-1	Botnet (Citadel), kaba kuvvet (FTP, HTTP ve SSH'e karşı), DDoS (HTTP baskınları, SYN baskınları ve UDP baskınları), istismar, inceleme, sızdırma, SSLproxy, XSS/SQL injection
NGIDS-DS	Arka kapılar, DoS, istismar, jenerik, keşif,shellcode, solucanlar
NSL-KDD	DoS, yetki artırma (uzaktan lokale ve kullanıcıdan root'a), inceleme
PU-IDS	DoS, yetki artırma (uzaktan lokale ve kullanıcıdan root'a), inceleme
PUF	DNS saldırıları
SANTA	DoS (ICMP baskını, RUDY, SYN baskını), DNS amplifikasyonu, heartbleed, port taramaları
SSENET-2011	DoS (LOIC ile yapılan), port taramaları (Angry IP Scanner, Nessus ve Nmap ile yapılan), çeşitli saldırı araçları (örneğin metasploit)
SSENET-2014	Botnet, baskılama, yetkiartırma, port taramaları
SSHCure	SSH saldırıları

Çizelge 5.3. (Devamı) Ağ sızma verisetlerinin kullanıldığı uygulamalar [120].

Veriseti	Saldırılar
TRAbID	DoS (HTTP baskını, ICMP baskını, SMTP baskını,SYN baskını, TCP tutma), port taramaları (ACK-taraması,FIN- taraması, NULL- taraması, OS Fingerprinting,Service Fingerprinting, UDP- taraması, XMAS- taraması)
TUIDS	Botnet (IRC), DDoS (Fraggle baskını, Ping baskını,RST baskını, smurf ICMP baskını, SYN baskını, UDP baskını), port taramaları (e.g. FIN- taraması, NULL- taraması,UDP- taraması, XMAS- taraması), coordinated port taraması,SSH kaba kuvvet
Twente	Bal küplerine karşı üç açık servis ile saldırılar (FTP, HTTP, SSH)
UGR'16	Botnet (Neris), DoS, port taramaları, SSH kaba kuvvet,spam
UNIBS	Yok
Unified Host and Network	Tanımlı değil
UNSW-NB15	Arka kapılar, DoS, istismarlar, fuzzer'lar, jenerik, porttaramaları, keşif, shellcode, spam, solucanlar

Çalışmada, sızma tespiti için bir diğer ağ trafik kaynağının trafik üreticiler olduğu ifade edilmekte ve bunların sentetik ağ trafiği oluşturduğu vurgulanmaktadır. Birçok durumda trafik üreticilerin kullanıcı tarafından tanımlanmış parametreleri kullandığı veya gerçek ağ trafiğinden temel özellikler çıkarılarak yeni sentetik ağ trafiği üretildiği belirtilmektedir. Verisetlerinin sadece sabit veri sunduğu, buna karşı trafik üreticilerin istenilen ağ yapısına adapte edilebilir ağ trafiği üretim kabiliyeti sağladığı vurgulanmaktadır. Örnek olarak FLAME ve IDT2 [121] trafik üreticileri gerçek ağ trafiğini girdi olarak kullanmaktadır. Bu girdi trafiği, normal kullanıcı davranışlarına yönelik bir altyapı oluşturmak için kullanılır. Daha sonra FLAME ve IDT2 vasıtasıyla girdi trafiğindeki değerler değiştirilerek veya bazı tipik atak durumlarında görülen akışların sentetik olarak enjekte edilmesiyle zararlı ağ trafiği eklenir.

Bunlara ek olarak bir başka çalışmada, sentetik zararlı ağ trafiği üretimi için yapılan literatür inceleme sonuçları paylaşılmaktadır [122]. Bu kapsamda;

1. Graf tabanlı ağ akış üreticisi ile gerçek ağ trafiğinden örneklerin çıkarıldığı ve üreticinin bu trafik örneklerini kullanarak yeni sentetik akış tabanlı ağ trafiği elde ettiği [123],
2. Generative Adversial Network (GAN) uygulamalarının sentetik ağ trafiği üretimine uygulandığı, ilk aşamada GAN'ın gerçek ağ trafiği ile eğitildiği ve trafik karakteristiklerini öğrendiği, daha sonra benzer karakteristiğe sahip sentetik akış tabanlı ağ trafiği oluşturduğu [124],

3. GENESIDS isimli bir trafik üreticisi ile kullanıcı tanımlamalarına dayalı http atak trafiğinin üretildiği [125] belirtilmektedir.

Bir diğer çalışmada [126], üretilen verisetlerinin gizlilik gerekçeleri ile paylaşılamaması problemine çözüm olarak, kullanıcılar tarafından veriseti yerine saldırı senaryoları paylaşımına imkân sunan Moirai isimli bir çerçeve önerildiği belirtilmektedir. Bu çerçeve sayesinde, kullanıcılar atak senaryolarını sunulan sanal makineler üzerinde çalıştırarak kendi verilerini üretebilmektedir.

Bir diğer yaklaşımda ise kullanıcıların gerçek ağ trafiğini etiketlendirmesi prensibine dayanan çerçevelerin kullanımı gerçekleştirilmektedir. Bu kapsamda;

1. INSecS-DCS isimli bir çerçevenin önerildiği çalışmadan bahsedilmekte [120], bu çerçeve sayesinde ağ trafiğinin, ağ cihazları üzerinden yakalanması veya hâlihazırda yakalanmış PCAP formatlı ağ trafik dosyalarının girdi olarak kullanılmasının mümkün olduğu, INSecS-DCS tarafından veri akışının zaman pencerelerine bölündüğü, veri noktalarının uygun özellikler ile çıkarıldığı ve ağ trafiğinin kullanıcı tarafından tanımlanmış saldırgan IP adres listesine dayalı olarak etiketlendiği,
2. Eğiticisiz anomali tabanlı ağ sızma tespit sistemi (IDS) kullanılarak otomatik veriseti etiketlemesi yapan bir uygulamanın bulunduğu [127], hiçbir IDS'in her kaydı doğru sınıflandırmasının mümkün olmadığı, bu nedenle false pozitif ve true negatif sayısının azaltılmaya çalışıldığı, IDS'in her veri noktasına normal ve atak olacak şekilde bir inaç değeri atadığı, bu iki sınıf için belirlenen inanç değerleri arasındaki fark daha önceden tanımlanmış eşik değerinden küçükse veri noktasının verisetinden çıkarıldığı, bu yaklaşımın etiket kalitesini arttırdığı, ancak en ilgi çekici veri noktalarının atılmasına neden olabileceği ifade edilmektedir.

Ağ sızma tespit sistemlerinin eğitim, doğrulama ve testi için kullanılabilecek 34 farklı verisetinin özellikleri ile birlikte açıklandığı, kullanım amaçlarının tanımlandığı, sınıflandırmaları yapılarak doğru verisetinin doğru uygulamada kullanımına yönlendirildiği bu çalışmanın son bölümünde, mükemmel verisetinin tanımı da yapılmaktadır. Buna göre mükemmel veri seti, güncel olmalı, doğru etiketlendirilmeli, herkes tarafından erişilebilir olmalı, tüm saldırı tipleri ile normal kullanıcı davranışlarını içeren gerçek ağ trafiğini içermeli ve uzun bir zaman diliminde elde edilmelidir. Hâlihazırda böyle bir veriseti

olmadığı, ancak erişilebilir birçok verisetinin mükemmel verisetinin biçok özelliğini kaşıladığı vurgulanmaktadır. Geliştirdikleri ağ sızma tespit sistemlerinin overfit olmaması, belirli bir verisetinin yapay hatalarından kaçınılması ve geliştirilen uygulamanın daha genel kapsamda değerlendirilmesi için kullanıcılara uygulamalarını farklı verisetleri ile değerlendirmeleri önerilmektedir. Ayrıca özellikle CICIDS 2017, CICIDS-001, UGR'16 ve UNSW-NB15 verisetlerinin kullanımı tavsiye edilmekte, bunların genel değerlendirme ayarlamalarına uygun olduğu, daha derin inceleme için detaylı metadata ve büyük miktarda akış içerdikleri ifade edilmektedir. Diğer verisetlerinin de belirli değerlendirme senaryoları için kullanılabileceği vurgulanmakta, ağ sızma tespit uygulamasının mutlaka yaygın bir veriseti ile de değerlendirilmesi önerilmektedir.

Bir diğer çalışmada [128], bir veritabanı üretim yaklaşımının,

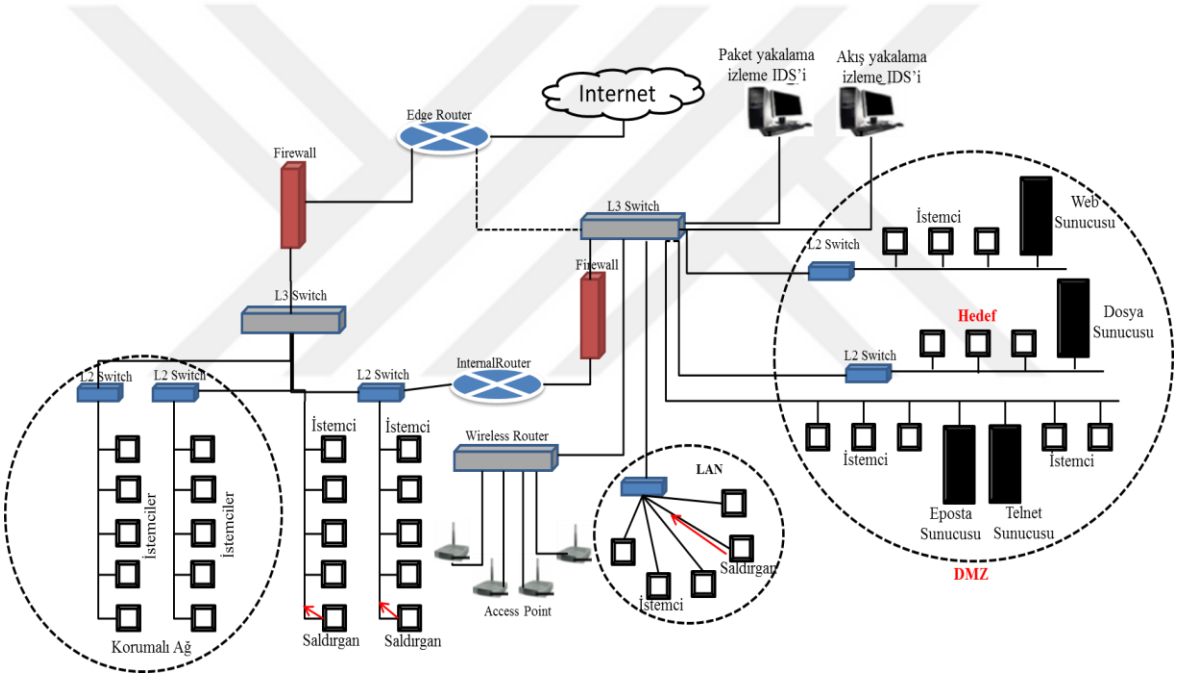
- Gerçek bir ağ trafiğinin günlük durumunun izlenmesini sağlayacak şekilde gerçekçi olması,
- Saldırı veya normal olarak belirlenen her trafik etiketlemesinin tüm kayıtlar için kanıta dayalı olması,
- Verisetindeki her kaydın etiketinin doğru olması,
- Saldırı veya normal kayıtların miktar olarak yeterli izlenebilirlik seviyesinde olması,
- Optimal somut özellik setinin elde edilebilir olması,
- Saldırıların artan sıklık, boyut, çeşitlilik ve karmaşıklığına bağlı olarak sızma tehditleri daha kompleks hale geldiğinden, güncel atakları içeren çok adımlı ve farklı senaryoları içermesi,
- Saldırı ve normal kayıt oranlarının iyi bir şekilde belirlemesi gerektiği ifade edilmektedir.

Çalışmada, önceki çalışmaya benzer şekilde birçok veriseti ile ilgili bilgi verilmekte, TUIDS (Tezpur University Intrusion Detection System) sızma veriseti, TUIDS tarama veriseti ve TUIDS DDoS veri setinin gerçek ağ trafiğinden paket ve akış seviyesinde üretildiği vurgulanmaktadır.

Çalışmanın devam eden bölümünde, veriseti üretimi için kurulan ağ mimarisinden bahsedilmekte, ağın Tezpur Üniversitesi kampüsü içinde 250 host bilgisayar, 15 L2 anahtar, 3 kablosuz kontrolcü ve 4 yönlendiriciden oluşan 5 farklı ağdan oluştuğunu ifade etmektedirler. Host bilgisayarların çeşitli VLAN'lara dağıtıldığı, her VLAN içerisinde L3

ve L2 switchler bulunduğu, tüm sunucuların ilave bir koruma sağlamak için DMZ içerisine kurulduğu vurgulanmaktadır.

Gerçek zamanlı ağ trafiği üretimi için hostların, iş istasyonlarının ve sunucuların konfigüre edildiği, 6 adet Ubuntu 10.10 iş istasyonunun birbiri ile bağlandığı, her iş istasyonuna bir ağ dosya sunucusu (Samba), bir mail sunucusu (Dovecot), bir telnet sunucusu, bir FTP sunucusu, bir Web sunucusu ve PHP ile uyumlu bir SQL sunucusunun kurulduğu belirtilmektedir. Ayrıca bilinen çeşitli zafiyetlere saldırı amacıyla 4 Windows Server 2003 kurulumu yapıldığı ifade edilmektedir. TUIDS veriseti üretimi için kurulan ağ mimarisi Şekil 5.2'de görülmektedir.



Şekil 5.2. TUIDS veriseti üretimi için kurulan ağ mimarisi [128].

Çeşitli tiplerde normal trafik yakalamanın önemli olduğu vurgulanmakta, bunun günlük ağ trafik aktivitelerinden ve özellikle konfigüre edilmiş sunucular üzerinde üretilen trafikten elde edildiği belirtilmektedir. Saldırı trafiğinin ise TUIDS sızma veriseti, tarama veriseti ve DDoS veriseti üretimi için yapılan ataklar vasıtasıyla elde edildiği ifade edilmektedir. TUIDS sızma veriseti elde edilmesi için toplam 22 farklı saldırı tipinin, tarama veriseti için 6 farklı saldırı tipinin, TCP, UDP ve ICMP protokollerinin kombinasyonunu içeren DDoS veriseti için ise yine 6 farklı saldırı tipinin kullanıldığı ifade edilmektedir. Kullanılan saldırı tipleri Çizelge 5.4'te görülmektedir.

Çizelge 5.4. Gerçek zamanlı ataklar ve üretim araçları [128].

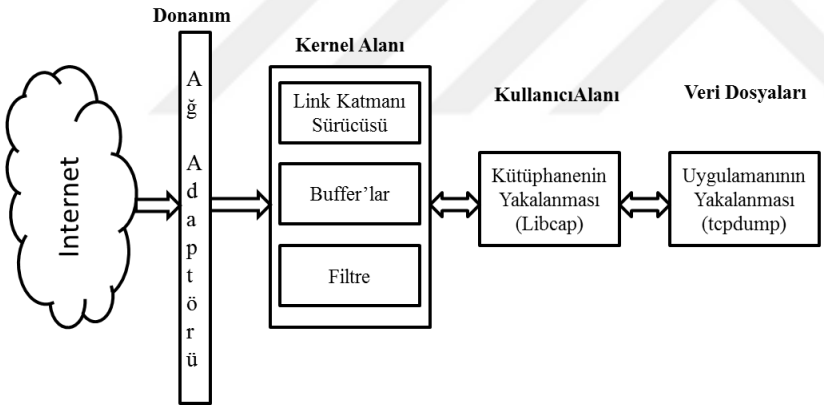
Saldırı Adı	Üretim Aracı	Saldırı Adı	Üretim Aracı
1. bonk	targa2.c	15. linux-icmp	linux-icmp.c
2. jolt	targa2.c	16. syn-flood	syn-flood.c
3. land	targa2.c	17. window-scan	nmap/rnmap
4. saihyoussen	targa2.c	18. syn-scan	nmap/rnmap
5. teardrop	targa2.c	19. xmasstree-scan	nmap/rnmap
6. newtear	targa2.c	20. fin-scan	nmap/rnmap
7. 1234	targa2.c	21. null-scan	nmap/rnmap
8. winnuke	targa2.c	22. udp-scan	nmap/rnmap
9. oshare	targa2.c	23. syn-flood (DDoS)	LOIC
10. nestea	targa2.c	24. rst-flood (DDoS)	Trinity v3
11. syndrop	targa2.c	25. udp-flood (DDoS)	LOIC
12. smurf	smurf4.c	26. ping-flood (DDoS)	DDoS ping v2.0
13. opentear	opentear.c	27. fraggle-flood (DDoS)	Trinoo
14. fraggle	fraggle.c	28. smurf icmp-flood (DDoS)	TFN2K

Veriseti üretimi amacıyla gerçek zamanlı ağ tarafiği elde edilmesi için toplam altı faklı saldırı senaryosu uygulandığı ifade edilmektedir. Bu atak senaryolarına ait bilgi aşağıda yer almaktadır.

1. Targa ile servis engelleme (Denial of Service) : Bir organizasyona ait ağın hızlıca zarara uğratılmasını sağlayacak kadar güçlü olan Targa aracının, IP aralığı, yapılacak saldırılar ve saldırı tekrar sayısı değerleri tanımlanarak kullanıldığı senaryo.
2. nmap ile sondalama (Probing) : Hedef host bilgisayara ait bilgilerin toplanması amacıyla girişimde bulunulan ve daha sonra bulunan zafiyetlere nmap aracı ile saldırılan senaryo.
3. rnmap aracı ile koordineli tarama (Coordinated Scan) : Bir veya birden çok hedefe yönelik koordineli port taraması ile başlayan, daha sonra rnmap aracı ile trafik toplanırken ağ üzerinde koordineli tarama yapılan senaryo.
4. Kaba Kuvvet (Brute Force) ssh ile kullanıcıdan root'a (User to Root): Merkezi sunucu üzerine brute force dictionary saldırısı yapılarak bir SSH hesabının ele geçirilmesinin hedeflendiği, brutessh aracı ve düzenlenmiş sözlük listesinin kullanıldığı, sözlüğün farklı boyutlarda 6100 alfanümerik giriş içerdiği, saldırının 60 dakika uygulandığı, neticede süper kullanıcı yetkilerinin elde edildiği ve bu yetki ile diğer kullanıcıların yekilerinin ele geçirildiği senaryo.
5. Agent-handler ağ ile dağıtık servis engelleme : TUIDS test ağı üzerinde DDoS saldırıları yapılarak bir agent-handler ağın istismarının hedeflendiği, atak yapmak için Trinity v3, TFN2K, Trinoo ve DDoS ping 2.0 araçlarının kullanıldığı senaryo.

6. IRC Botnet ile dağıtık servis engelleme :Organizasyonlara ait bilginin çalınmasına, ağın ele geçirilmesine veya zararlı yazılımın yayılmasına neden olan botnetlerin bazı host bilgisayarlara enfekte edildiği, bu amaçla kullanıcıların genel, özel veya gizli kanallar oluşturmasına imkân sunan IRC (Internet Relay Chat) üzerinde DDoS atakların gerçekleştirilmesini sağlayan LOIC aracının kullanıldığı senaryo.

Çalışmada, ağ trafiğinin gözlemlenmesinde kayıpsız paket yakalama ve kesin zaman işaretlemenin (timestamping) önemli olduğu, bu amaçla birçok sistem mimarisinde link seviyesinde frame yakalamayı sağlayan açık kaynak C kütüphanesi Libpcap'in kullanıldığı ifade edilmektedir. Libpcap'in üst seviye API'si (Application Programming Interface) sayesinde farklı işletim sistemleri için farklı paket yakalama framework'leri sağladığı, programcılara kolayca taşınabilir uygulamalarını hızlıca geliştirme imkânı sunduğu ifade edilmektedir. Trafik yakalama bileşen hiyerarşisi Şekil 5.3'te görülmektedir.



Şekil 5.3. Ağ trafiği yakalama bileşenleri [128].

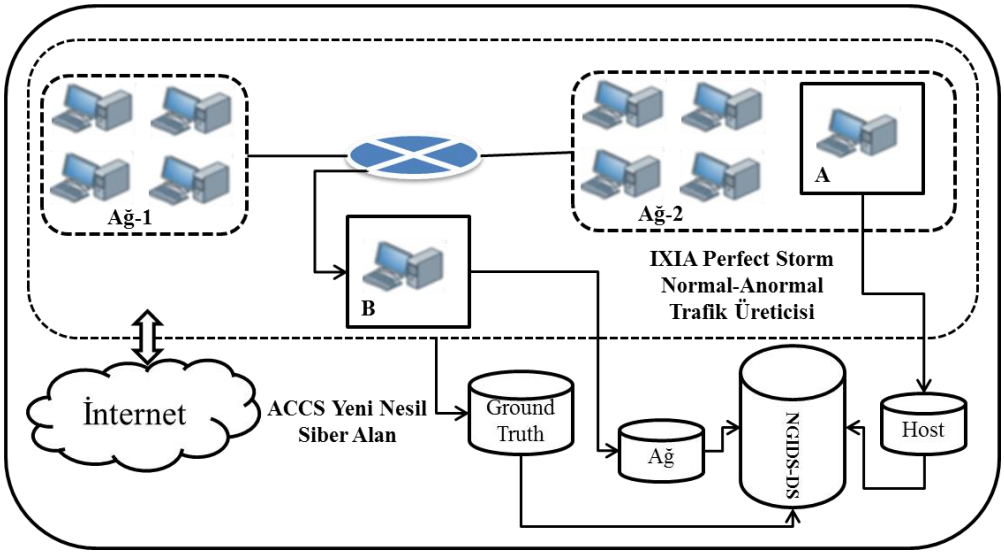
Paket ve akış seviyesi verisetlerinin üretimi için TUIDS ağının çeşitli lokasyonlarından hem paket hem de NetFlow trafiğinin yakalandığı, bu işlemin 7 gün boyunca devam ettiği, bu süreçte TUIDS sızma ve koordineli tarama verisetlerinin üretimi için saldırılar yapıldığı vurgulanmaktadır. DDoS saldırılarının da aynı süre boyunca toplandığı ifade edilmektedir. TUIDS ağ mimarisi içerisindeki aynalanmış port üzerindeki paket seviyesi trafiğin kayıpsız olarak yakalanması için Linux Gulp aracının kullanıldığı, Gulp'un paketleri direk olarak ağ kartından okuyarak diske kayıpsız olarak aktardığı ifade edilmektedir. NetFlow'un son dönemde en çok kullanılan IP ağ izleme yaklaşımı olduğu, Cisco gibi ağ aktif cihazı üreticilerin artık akış-erişilebilir cihazlar sunduğu, Cisco yönlendiricilerinin NetFlow

özelliklerine sahip olduğu, bunların NetFlow kayıtlarının üretimi için aktive edilebildiği belirtilmektedir. NetFlow trafiğinin alınması için, belirli bir UDP portunu dinleyebilen NetFlow toplayıcıya ihtiyaç olduğu, NetFlow toplayıcının birçok yönlendiricide oluşan trafiği toplayarak bir round robin verisetine (RRD) kaydettiği, bu amaçla kullanılan araçların, NFDUMP ve NFSEN olduğu vurgulanmaktadır.

Özellik çıkarma amacıyla, yönlendirme protokolleri ve uygulama seviyesi protokoller ile ilgili paketler gibi istenmeyen paketler ve yakalanan paketlerdeki ilgisiz verinin filtrelenmesi için wireshark ve Java rutinlerinin kullanıldığı, neticede tüm ilgili verinin Java rutinleri ile elde edilerek (,) ile ayrılmış bir tekst dosyasına kaydedildiği belirtilmektedir. NetFlow verisinden özellik çıkarımı için ise çeşitli C rutinleri geliştirildiği belirtilmektedir.

Trafik özelliklerinin birbirinden bağımsız olarak elde edildiği ve tüm özelliklerin bir zaman aralığına bağdaştırılmasının, ayrıca temel, içeriğe dayalı, zamana dayalı ve bağlantıya dayalı elde edilen özellik verilerinin normal veya anormal olarak etiketlenmesinin önemli olduğu ifade edilmektedir. Her özellik trafiğinin normal veya anormal olarak bağdaştırılması ve etiketlenmesinin geliştirilen bir algoritma ile sağlandığı, TUIDS sızma, TUIDS koordineli tarama ve TUIDS DDoS verisetlerinin üretildiği, her verisetinin paket ve akış seviyesi olacak şekilde hazırlandığı, eğitim ve test verisetleri olarak sunulduğu vurgulanmaktadır.

Bir başka çalışmada [129], mevcut verisetlerinin gerçekçilik kalitesinin değerlendirilmesi amacıyla Sugeno bulanık çıkarım modeline dayalı bir metrik önerilmekte, önerilen bu metrik sonuçlarına bağlı olarak yeni nesil bir ağ sızma tespit veriseti tasarlandığı ve geliştirildiği vurgulanmakta, son olarak üretilen veriseti önerilen metrik ile analiz edilerek gerçekçiliği değerlendirilmektedir. Ayrıca üretilen verisetinin mevcut verisetlerine oranla gerçekçilik kalitesi açısından daha iyi olduğu kıyaslı bir şekilde ortaya koyulmaktadır. Çalışma kapsamında üretilen IDS veriseti yeni nesil olarak tanımlanmakta NGIDS-DS olarak adlandırılmakta, veriseti üretim işlemlerinin Avustralya Siber Güvenlik Merkezi'nde oluşturulan Şekil 5.4'teki ağ mimarisi üzerinden trafiğin elde edilmesi ile üretildiği, bu mimarideki en önemli avantajın IXIA Perfect Storm donanımının kullanımına imkân sunması olduğu vurgulanmaktadır. Bu donanımın güncel normal ve anormal siber karışımı siber trafik üretebildiği, farklı dinamik davranışlara sahip maksimum sayıda atak oluşturabildiği, farklı ortamlar için siber trafik profili kurulumuna imkân sunduğu belirtilmektedir.



Şekil 5.4. NGIDS-DS veriseti üretimi için kurulan mimari [129].

Oluşturulan mimaride Ağ-1'in normal ve anormal trafik ile beş farklı ortam (e-ticaret, askeri, akademi, sosyal medya ve bankalar) için operasyonel zamanlama ve profil üretimi için tasarlandığı, Ağ-2'nin ise bu ortamların kritik altyapısını içeren kurban ağ olarak davrandığı belirtilmektedir. Anormal ağ trafiğinin istismarlar, DoS, solucanlar, jenerik, keşif, shellcode ve arka kapı atak ailelerinden olduğu, Ubuntu 14.04 kurulu A makinesinin hafıza, Web, FTP, email, NAT, SSH ve DNS gibi çeşitli servisleri içeren kritik ortam makinesi olarak tasarlandığı, Ubuntu 14.04 ve tcpdump kurulu B makinesinin Ağ-1'de üretilen ağ paketlerinin toplanması ve Ağ-2'ye aktarılması amacıyla kurulduğu belirtilmektedir. NGIDS-DS verisetinin üretimi için yaklaşık beş gün boyunca veri toplandığı ifade edilmektedir.

Bir başka çalışmada [130], normal trafik ile birlikte DoS, DDoS, Brute Force, XSS, SQL injection, infiltration, port tarama ve botnet ataklarını içeren, gerçek yaşam kriterlerini sağlayan ve CICIDS2017 olarak isimlendirilen güvenilir bir ağ sızma tespit veriseti önerilmektedir. Verisetindeki kayıtların tamamının etiketlendirildiği, 80 ağ özelliğinin tüm normal ve anormal akış için Kanada Siber Güvenlik Enstitüsü web sayfasında herkese açık olarak sunulan CICFlowMeter yazılımı ile hesaplandığı ve çıkarıldığı belirtilmektedir. Ayrıca üretilen verisetinin farklı atakların tespiti için gerekli en iyi özellikleri içermesine yönelik analiz gerçekleştirildiği, verisetinin değerlendirilmesi için yaygın olarak kullanılan yedi makine öğrenme algoritmasının uygulandığı vurgulanmaktadır.

Oluşturulan test ortamının kurban ve saldırı olacak şekilde tamamen ayrılmış iki farklı ağ içerdiği, diğer verisetlerinin aksine kurban ağ içerisinde Windows, Linux ve Macintosh işletim sistemlerine sahip makineler ile birlikte yaygın ve gerekli yönlendirici, firewall, ağ anahtarı gibi ekipmanın kapsandığı belirtilmektedir. Saldırı ağı içerisinde bir yönlendirici, bir ağ anahtarı ile Kali ve Windows 8.1 işletim sistemlerine sahip dört PC'nin bulunduğu, kurban ağ içerisinde ise üç sunucu, bir firewall, iki ağ anahtarı ve domain controller ve active directory ile bağlanmış 10 adet PC yer aldığı ifade edilmektedir. Ayrıca kurban ağ içerisindeki ana ağ anahtarının bir portu ayna port olarak konfigüre edilerek ağ üzerindeki giden ve gelen tüm trafiğin yakalandığı belirtilmektedir.

Çalışmanın en önemli amaçlarından birinin gerçekçi arka plan trafiği üretmek olduğu, bu amaçla B-Profile olarak isimlendirilen, görevi insan iletişimlerinin soyut davranışını profillemek ve doğal arka plan trafiği üretmek olan bir sistem önerilmektedir. B-Profile'in 25 kullanıcının HTTP, HTTPS, FTP, SSH ve email protokollerine dayalı soyut davranışlarını çıkardığı ifade edilmektedir. Daha sonra Java tabanlı geliştirilen bir ajan vasıtasıyla kurban ağ üzerinde daha önceden tanımlanan beş protokol için gerçekçi normal olayların üretildiği belirtilmektedir.

En güncel yaygın saldırı ailesinden toplam altı saldırı profilinin (Brute Force Attack, Heartbleed Attack, Botnet, DoS Attack, DDoS Attack, Web Attack ve Infiltration Attack) oluşturulduğu ve bunların ilgili araç ve Python kodları ile uygulandığı vurgulanmaktadır. Ağ trafiğinin beş gün süre ile dinlendiği, her atak tipinin belirli zaman dilimlerinde uygulandığı belirtilmektedir.

Brute Force atak için en çok kullanılan araçlardan biri olan Python tabanlı Patator isimli çoklu saldırı kabiliyetli aracından yararlanıldığı, her cevabı daha sonra gözden geçirilmek üzere farklı log dosyasına kaydetmesi ve FTP, SSH, Telnet ve SMPTP gibi 30'dan fazla metodu desteklemesi sayesinde daha güvenilir ve esnek olduğu belirtilmektedir. Bu senaryoda, saldırgan Kali Linux, kurban ise web sunucu olarak çalışan Ubuntu 16.04 makinesidir ve FTP-Patator sabah, SSH-Patator öğleden sonra çalışmaktadır.

DoS saldırısı için LOIC, HOIC, Hulk, GoldenEye, Slowloris ve Slowhttptest araçlarının yaygın olarak kullanıldığı, çalışmada ise Slowloris ve Slowhttptest'in kullanıldığı, bu araçların bir makinenin minimum bant genişliği ile bağlantıları açık tuttuğu, böylece web

sunucusu kaynaklarını tüketerek hızlıca çökmesini sağladığı belirtilmektedir. Bu senaryoda saldırganın Kali Linux, kurbanın ise Apache web sunucusu bulunan Ubuntu 16.04 makinesi olduğu ifade edilmektedir.

Heartbleed saldırısı için en iyi araçlardan birinin Heartleech olduğu, zafiyet tespiti için sistemi taradığı belirtilmekte, bu senaryoda OpenSSL'in zafiyete sahip versiyonu olan Open SSL version 1.0.1f'in Ubuntu 12.04 makinesine kurulduğu, daha sonra Heartleech ile web sunucunun bellek verisinin indirildiği belirtilmektedir.

Web saldırı senaryosu için zafiyete sahip PHP/MySQL web uygulaması olan Damn Vulnerable Web App (DVWA) kullanıldığı, XSS ve Brute Force bölümlerindeki saldırıların otomotikleştirilmesi için Selenium framework'ünde otomatize edilmiş bir kod geliştirildiği, saldırganın Kali Linux, kurbanın ise web sunucu olarak çalışan Ubuntu 16.04 sistemi olduğu ifade edilmektedir.

Infiltration saldırı senaryosu için en yaygın güvenlik sorunu ve zafiyet tanımlayıcısı olarak bilinen Metasploit aracının kullanıldığı, kurbanın enfekte dosyayı Windows makineler için Dropbox'tan, Macintosh makineler için USB üzerinden indirmesinden sonra saldırganın kurban ağ üzerinde Nmap ve port taraması başlattığı belirtilmektedir. Burada saldırganın Kali Linux, kurbanların ise kurban ağ üzerinde çalışan Windows, Ubuntu ve Macintosh sistemler olduğu ifade edilmektedir.

Botnet saldırıları için Grum, Windigo, Storm ve Ares gibi araçların kullanıldığı, bu çalışmada Python tabanlı Botnet olan Ares'in tercih edildiği, bu aracın remote shell, dosya indirme/yükleme, ekran görüntüsü alma ve klavye loglama kabiliyetlerinin olduğu, bu senaryoda saldırganın Kali Linux, kurbanların ise sırasıyla Vista, 7, 8.1, 10 (32 bit) ve 10 (64 bit) Windows işletim sistemine sahip makineler olduğu belirtilmektedir.

DDoS saldırıları için High Orbit Ion Canon (HOIC), Low Orbit Ion Canon (LOIC), DDoSIM gibi araçların kullanıldığı, çalışmada UDP, TCP ve HTTP taleplerini kurban sunucuya gönderen LOIC'in tercih edildiği, saldırganların bir grup Windows 8.1 makine, kurbanın ise web sunucu olarak hizmet veren Ubuntu 16 makinesi olduğu belirtilmektedir. Çalışmada ayrıca tüm Windows makineler üzerinden sS, sT, sF, sX, sN, sP, sV, sU, sO, sA, sW, sR, sL ve B ana NMap anahtarları ile Portscan yapıldığı ifade edilmektedir.

Üretilen verisetinin dört farklı açıdan değerlendirildiği, öncelikle verisetinden 80 adet trafik özelliğinin CICFlowMeter kullanılarak çıkarıldığı, daha sonra çıkarılan bu özelliklerin RandomForestRegressor ile test edilerek her saldırı için en iyi kısa özellik setinin seçildiği, daha sonra seçilen bu özelliklerin performans ve kesinliğinin en çok kullanılan yedi makine öğrenme algoritması ile değerlendirildiği, son olarak üretilen verisetinin kalitesinin üretilen diğer verisetlerine yönelik yaygın hata ve kritiklere bağlı olarak değerlendirildiği, bu amaçla Kanada Siber Güvenlik Enstitüsü (CIC) tarafından sunulan 11 kriterin kullanıldığı vurgulanmaktadır.

5.2. Siber Savunma Tatbikatları ve Özgün Veri Seti Üretimi

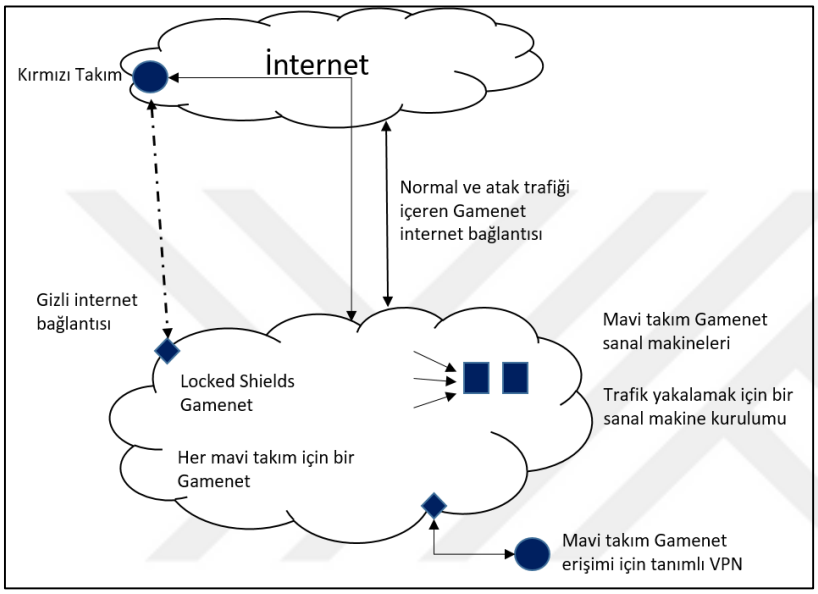
Siber savunma tatbikatları, ülkelerin siber savunma altyapılarının, raporlama kabiliyetlerinin, kullanılan güvenlik araçlarının verimliliklerinin, personel yetkinliğinin ve kurumsal yapının etkinliğinin test edilmesine imkan sunan önemli etkinliklerdir. Bu tatbikatlar, genellikle yılda bir kez düzenlenir ve ülkeler tarafından benzeri tatbikatlar da gerçekleştirilmektedir. Ayrıca, siber savunma alanında uluslararası düzenlenen tatbikatlar da bulunmaktadır.

Kilitli Kalkanlar (Locked Shields) tatbikatı, dünyanın en büyük ve karmaşık siber güvenlik tatbikatlarından biridir. Estonya'nın Tallinn şehrinde konuşlu NATO Siber Savunma Mükemmeliyet Merkezi (NATO CCDCOE) tarafından her yıl düzenlenmektedir ve yaklaşık 40 ülkeden 4000 siber güvenlik uzmanının katılımıyla gerçekleştirilmektedir. Bu tatbikat, siber savunma alanında uzmanlar arasında ciddi bir rekabet yaratmaktadır. Locked Shields, katılımcıların farklı senaryolara karşı hazırlıklı olmalarını ve güvenlik açıklarını tespit etmelerini sağlamaktadır. Bu sayede, katılımcılar siber saldırılara karşı mücadele konusunda deneyim kazanır ve siber savunma alanında daha dayanıklı hale gelirler.

Tatbikata katılımcı ülkeler tarafından mavi takım bünyesinde temsilciler sağlanmakta, bu temsilciler tarafından kritik sistemleri de içeren ağ altyapıları savunulmaktadır. Her mavi takım farklı izole bir ağ üzerinde çalışmakta, kırmızı takım tarafından bu ağları yok etme veya yıpratmaya yönelik saldırılar gerçekleştirilmektedir.

Tatbikat senaryo ve mimarisi her yıl değişmekte olup farklı askeri/sivil kritik altyapılara saldırı/savunma durumlarının çalışılmasına imkân sunacak şekilde güncellenmektedir. Mavi

takımlara ağ topolojisi ve bağlı cihazlarını içeren şemalar verilmektedir. Bu şemalar kırmızı takımın sistemlere atak için kullandıkları açıkları içermemektedir. Mavi takıma sistem savunmasını gerçekleştirmek için düşük bant genişliğine sahip bir ortamda, kısıtlı kaynaklarla tehdit analizi gerçekleştirebilecekleri sanal makineler sağlanmakta, bu nedenle mavi takımın etkin analiz araçları ve sızma tespit sistemleri kullanması veya geliştirmesi beklenmektedir. Locked Shields tatbikatının genel mimarisi Şekil 5.5'te sunulmuştur.



Şekil 5.5. Locked Shields tatbikatının genel mimarisi [131].

Kırmızı takım, saldırılarını belirli sıkı bir plan ve hedef doğrultusunda yapmaktadır. Kırmızı takımın saldırıları tatbikat süresince devam etmektedir. Kategorilerine göre bazı ataklar sadece belirli zamanlarda, bazıları ise sürekli gerçekleştirilmektedir. Kırmızı takım, tatbikat öncesi sistem konfigürasyonunu bildiği için uygun atak tipleri hazırlayabilmektedir. Bu kapsamda kullanılan araçlar sayesinde sistematik olarak sürekli otomatik enjeksiyonlar, zararlı kod yükleme ve veribağı yönetimi gibi zararlı aktiviteler icra edilmektedir.

N.Kanzing ve diğerleri tarafından yapılan çalışmada (2019), Locked Shields 2017 ve 2018 ağ trafikleri ile kırmızı takım loglarından elde edilen veriler ile yapay zekâ uygulamalarında kullanılabilecek uygun veriseti üretimi gerçekleştirdikleri ifade edilmektedir [131].

Veriseti üretimi için 2017 ve 2018 Locked Shields tatbikatı ağ trafik verisini içeren PCAP dosyalarının işlendiği ve veriseti özelliklerinin bu verinin işlenmesi neticesinde belirlendiği

vurgulanmaktadır. Kırmızı takım log dosyalarının ise verisetindeki kayıtların etiketlenmesinde kullanıldığı belirtilmektedir.

Veri ön işlemleri kapsamında, veri boyutunu azaltmak için truncate işlemi uyguladıkları, paketlerin akış ile eşleştirildiği, son olarak domain isimleri ile IP adresi eşleştirmelerinin yapıldığı ifade edilmektedir.

Truncate işlemi kapsamında, her paketin ilk 96 byte'ının kullanıldığı, bu bölümde taşıma katmanı başlık bilgisine kadar tüm bilginin yer aldığı, böylece verinin yaklaşık %75 seviyesinde azaltıldığı belirtilmektedir.

Paket ile akış eşleştirmesi kapsamında, kendi modellerinin akış seviyesinde çalıştığı, akışın, kaynak-IP, hedef-IP, kaynak portu, hedef portu ve taşıma katmanı protokol bilgilerinden oluştuğu ifade edilmektedir. Akışın TCP SYN paketi ile başladığı, ilk TCP-FIN paketinin gelmesi veya 15 saniyelik time-out neticesinde sonladığı ifade edilmektedir. Akış tabanlı özelliklerin çıkarılması için CICFlowMeter aracının kullanıldığı belirtilmektedir.

Domain ismi eşleştirmesi kapsamında Bro ağ analiz framework'ünün kullanıldığı, HTTP, TLS veya DNS bilgilerinden yararlanılarak paket izlerinden domain ismi IP-adresi eşleştirmesinin gerçekleştirildiği vurgulanmaktadır.

Veri etiketleme işlemi kapsamında, son noktası en az bir komuta kontrol sunucusu olan akışların saldırı, diğer akışın ise zararsız olarak etiketlendiği ifade edilmektedir. Bu yaklaşımın ardında, hiçbir zararsız cihazın komuta kontrol sunucusuna bağlanma ihtiyacı olmaması prensibinin bulunduğu belirtilmektedir. Bu bağlamda, komuta kontrol sunucularına erişmeye çalışan tüm cihazların zararlı bir amacının olduğu kabul edilmektedir.

Açık kaynak olarak kullanılan CICFlowMeter aracı ile toplam 77 özelliğin çıkarıldığı belirtilmektedir. Bu özelliklerin listesi Çizelge 5.5'te sunulmaktadır.

Çizelge 5.5. CICFlowMeter aracı ile belirlenen özellikler [131].

No	Özellik	Açıklama
1	Flow Duration	Akışın microsaniye seviyesinde gerçekleştiği süre
2-3	Tot Fwd/Bwd Pkts	Fwd/Bwd yönündeki toplam paket
4-5	TotLen Fwd/Bwd Pkts	Fwd/Bwd yönündeki paketlerin toplam boyutu
6-13	Fwd/Bwd Pkt Len	Fwd/Bwd yönündeki paketlerin min, maksimum, mean ve std değerleri
14-23	Fwd/Bwd IAT	Fwd/Bwd yönünde iletilen iki paket arasındaki toplam, min, maksimum, mean ve std zaman
24-35	Flag Counts	Fwd/Bwd için PSH, URG, SYN, FIN, RST, ACK, URG, CWE, ECE bayraklarının sayısı (UDP için 0)
36-37	Fwd/Bwd Header Len	Fwd/Bwd yönündeki header'lar için kullanılan toplam byte
38-40	Fwd/Bwd/Tot Pkts/s	Her saniyede oluşan Fwd/Bwd/Tot paket sayısı
41	Flow Byts/s	Her saniyede oluşan akış byte sayısı
42-45	Pkt Len	Bir akıştaki min, maksimum, mean ve std paket uzunluğu
46-49	Flow IAT	Fwd/Bwd yönündeki min, maksimum, mean, std paketlerin ulaşma zamanı
50	Down/Up Ratio	İndirme ve yükleme oranı
51	Pkt Size Avg	Ortalama paket boyutu
52-53	Fwd/Bwd Seg Size Avg	Fwd/Bwd yönünde gözlemlenen ortalama boyut
54-55	Fwd/Bwd Byts/b Avg	Fwd/Bwd yönündeki ortalama byte bulk sayısı
56-57	Fwd/Bwd Pkts/b Avg	Fwd/Bwd yönündeki ortalama paket bulk sayısı
58-59	Fwd/Bwd Blk Rate Avg	Fwd yönündeki ortalama bulk oranı sayısı
60-61	Subflow Fwd/Bwd Pkts	Fwd/Bwd yönündeki bir alt akıştaki ortalama paket sayısı
62-63	Subflow Fwd/Bwd Byts	Fwd yönündeki bir alt akıştaki ortalama byte sayısı
64-67	Active Time	Bir akışın idle olmadan önce min, maksimum, mean, std aktif süresi
68-71	Idle Time	Bir akışın aktif olmadan önce min, maksimum, mean, std idle süresi
72-73	Init Fwd/Bwd Win Byts	Fwd/Bwd yönündeki TCP pencere boyutu
74	Fwd Act Data Pkts	En az 1 byte TCP yükü olan fwd paket sayısı
75	Fwd Seg Size Min	Forward yönündeki minimum segment boyutu
76	Int/Ext Dst IP	Eğer Dst-IP akışı mavi takım subnetine dahilse 0, değilse 1
77	L3/L4 Protocol	TCP için 0, UDP için 1, ICMP için 2

Çıkarılan bu 77 özellikten en iyilerinin belirlenmesi amacıyla random forest gini önem skorlaması tabanlı recursif bir özellik eleme şeması uyguladıkları ve böylece ilgisiz özelliklerin atıldığı belirtilmektedir. Eleme işlemini tek tek gerçekleştirdikleri, birbiri ile ilişkili özelliklerin çıkarılması durumunda arzu edilen başarı seviyesinin sağlanamadığı, yapılan skorlama neticesinde 20 en iyi özelliğin belirlendiği ifade edilmektedir. Bu özellikler Çizelge 5.6'da görülmektedir.

Çizelge 5.6. Belirlenen en iyi 20 özellik.

Tot Fwd Pkts, Flow IAT Mean, Fwd IAT Max, Flow Pkts/s, Bwd Pkt Len Min, FIN Flag Cnt, Init Fwd Win Byts, Active Mean, Bwd IAT Mean, Bwd Pkt Len Std, Fwd Seg Size Min, Fwd Pkt Len Std, Tot Bwd Pkts, Bwd Header Len, Subflow Fwd Byts, Subflow Bwd Pkts, Fwd IAT Tot, Flow IAT Max, Int/Ext Dst IP, L3/L4 Protocol

Çalışmada ayrıca, geliştirilen model hakkında bilgi verilmekte, model eğitimi için Locked Shields 2017, testi için ise Locked Shields 2018 tatbikatlarından elde edilen verisetlerinin kullanıldığı vurgulanmaktadır.

Yukarıda detayları açıklanan çalışma neticesinde elde edilen verisetinin tez kapsamında kullanılması amacıyla çalışmayı gerçekleştiren yazarlar ile görüşme gerçekleştirilmiş ve veriseti talep edilmiştir. Ancak yazarlar tarafından üretilen verisetinin ülkeye özgü ağ trafiğinden üretildiği ve içeriğinin gizlilik derecesine sahip olduğu bilgisi alınmıştır. Bunun nedeni, Locked Shields tatbikatları neticesinde yayınlanan PCAP dosyalarında tatbikat süresince kırmızı takım tarafından gerçekleştirilen ataklara karşı ülkelerin verdikleri cevapları içeren bilgi bulunmasıdır. Bu bilgiden yararlanılarak ülkelerin siber savunma yetenek ve taktiklerine ilişkin çıkarım yapılabilmektedir.

Tez çalışması kapsamında, benzer bir verisetinin özgün bir şekilde üretilmesine yönelik çalışma gerçekleştirilmiştir. Bu kapsamda iki farklı yaklaşım ile veriseti üretim sürecine başlanmıştır. Öncelikle, gizlilik derecesine sahip bir PCAP dosyasından veriseti üretilmiş ve üretilen veriseti gizlilik dereceli olarak sınıflandırılmıştır.

Daha sonra Locked Shields tatbikat altyapısından istifade edilerek, akademik ve endüstride kullanılabilecek, gizililik derecesi olmayan, herkesin erişimine açık bir veriseti üretilmesi için araştırma grubu bünyesinde çalışma gerçekleştirilmiştir.

5.3. Locked Shields Ağ Trafiğinden Gizlilik Dereceli Veriseti Oluşturma Adımları

Tez çalışması kapsamında, Locked Shields tatbikatının özgün altyapısından istifade edilerek akademik çalışmalara oldukça katkısı olacağı değerlendirilen özgün bir veriseti üretmek için çalışma yapılması kararlaştırılmıştır. Ancak yapılan inceleme ve görüşmeler neticesinde, tatbikat altyapısına yeni ve bağımsız bir takım eklenmediği sürece, mevcut hali ile Locked Shields tatbikatından, ancak gizlilik derecesine sahip özgün veriseti oluşturulabileceği

belirlenmiştir. Bu kapsamda tatbikat altyapısına sanal mavi takım eklenmesine ilişkin bir makale yazılarak öneride bulunulmuştur. Yazılan makale International Conference on Cyber Conflict (CyCon) konferansına kabul edilerek IEEE explore üzerinde yayınlanmıştır. Bu sürece paralel olarak tatbikat verisinden istifade edilerek veriseti üretimine ilişkin inceleme ve çalışmalara devam edilmiştir. Gizlilik dereceli olarak üretilen ilk verisetine ilişkin üretim adımları ve yapılan çalışmalar neticesinde elde edilen bulgular aşağıda sunulmuştur:

1. Locked Shields tatbikatında, örün sayfası, kullanıcı bilgisayarı ve ağ olmak üzere üç farklı alanda saldırılar düzenlenmektedir. Örün sayfası saldırıları ATAW, kullanıcı bilgisayarı saldırıları ATAC ve ağ saldırıları ise ATAN etiketli olarak gerçekleştirilmektedir.
2. Saldırılar, kırmızı takım tarafından, mavi takımlara tanımlanmış domain'lere yapılmaktadır.
3. Saldırılar toplam beş fazda gerçekleştirilmekte, başlangıç, ilerleme ve tamamlandı olarak aşamalara ayrılmakta, saldırının belirlenen süre içerisinde tamamlanması durumunda atak başarı ile gerçekleştirilmiş olarak işaretlenmekte ve mavi takımın puan kaybetmesine neden olmaktadır.
4. Saldırılar tatbikat topolojisi içerisinde belirlenmiş bilgisayarlar üzerinden yapılmakta, her bölüm için belirlenmiş kırmızı takım personeli tarafından, önceden hazırlanmış bir plan dâhilinde icra edilmektedir. Bu nedenle, tatbikat neticesinde takımlara sağlanan PCAP dosyalarının uygun bir akıştan özellik çıkarım yöntemi ile işlenebilir hale dönüştürülmesini takiben, etiketleme işleminin kaynak ve hedef bilgisayarlara dayalı olarak, kullanılan protokol de göz önünde bulundurularak gerçekleştirilebileceği görülmektedir.
5. Burada önemli bir husus dikkati çekmektedir. Kaynak bilgisayar ne kadar saldırı amaçlı kullanılıyor olsa da, kaynak ve hedef arasında sadece atak trafiği olmadığı, normal trafik de olduğu göz önünde bulundurulmalıdır. Bu nedenle sadece kaynak, hedef ve protokol tipinin akışları etiketlemede gerçekçi bir sonuç sağlamayabileceği değerlendirilmektedir.
6. Tatbikat ortamının doğası gereği yüksek kapasitede ağ trafiği olduğundan, elde edilen PCAP sayısı da oldukça fazladır. Çalışma çerçevesinde, tatbikat trafiğinden üretilen

PCAP dosyalarının küçük bir bölümü, bilimsel amaçlı kullanılacağı ve üretilecek verisetinin paylaşılmayacağı taahhüt edilerek tatbikat düzenleyicilerinden alınmıştır.

7. Alınan örnek PCAP dosyalarından hangilerinde atak olabileceğinin tespiti için, yaygın olarak kullanılan bir ağ saldırı tespit sistemi olan, açık kaynak kodlu SURICATA yazılımı kullanılmıştır. Ancak, sadece ağ sızma tespit sistemi kullanımı tek başına yeterli değildir. Çünkü tatbikat kapsamında elde edilen log dosyaları ve kırmızı takım uzman desteği almadan yapılacak bir etiketlemenin eksik yanları olmaktadır.
8. SURICATA saldırı tespit sisteminin kullanımı için öncelikle Oracle VM Virtual Box Manager üzerinde bir sanal makine oluşturulmuştur. Bu sanal makineye, SURICATA yazılımını içeren imaj kurularak aktif edilmiştir. Daha sonra, indirilen PCAP dosyası, *suricata -c /etc/suricata/suricata.yaml -r /tmp/ls2021_00000.pcap -l /tmp/* komutu ile sisteme girdi olarak verilerek işlenmesi sağlanmıştır.
9. PCAP yani packet capture uzantılı dosyalar, adından da anlaşılacağı üzere trafikte iletilen paketlere ait bilgileri içermektedir. Bu paket içeriklerinin okunabilir veriseti formatına dönüştürülmesi için çeşitli dönüştürücü yazılımlar kullanılmaktadır. Bu dönüştürücü yazılımların amacı, paket içerisindeki alanları ayırmaktır. Yazılım özelliğine bağlı olarak veriseti içerisindeki alan isim ve içeriği değişmektedir.
10. Çalışma kapsamında, atak içeren PCAP dosyaları, Kanada Siber Güvenlik Enstitüsü tarafından açık kaynak olarak sağlanan, CICFlowMeter uygulaması kullanılarak toplam 84 özellik ve etkiket alanı içeren bir verisetine dönüştürülmüştür.
11. İşlenen dosya içerisinde yer alan paketler, içerdikleri atak tiplerine göre etiketlenmiştir. Böylece örnek ham veriseti elde edilmiştir. Veriseti içerisinde toplam 21633 kayıt olduğu belirlenmiş, bunların 7824'ü atak olarak etiketlenmiştir.
12. Elde edilen verisetine PCA özellik çıkarım algoritması uygulanarak sınıflandırmada en etkili 24 alan belirlenmiştir. Özellik çıkarım işlemi öncesi verisetinde yer alan alanlar Çizelge 5.7'de, özellik çıkarımı sonrası elde edilen alanlar ise aynı tablo üzerinde kırmızı renkte gösterilmiştir. Yapılan incelemede, tabloda mavi renkte gösterilen ve PCA algoritması ile belirlenen *Fwd_Pkts/b_Avg* ve *Fwd_Seg_Size_Min* alanlarında yer alan tüm verinin sıfır olduğu tespiti yapılarak bu alanlar verisetinden çıkarılmıştır. Bu durumda 22 özellik içeren yeni bir veriseti elde edilmiştir.
13. PCA uygulanarak belirlenen alanları içerecek şekilde veriseti tekrar üretilmiş, normalizasyon ön işlemleri yapılarak veriseti son haline getirilmiştir.

14. Elde edilen nihai veriseti, eğitim ve test verisetlerine dönüştürülmüştür. Bu kapsamda verisetinin %80'i eğitim, %20'si ise test amacıyla ayrılmıştır. Eğitim verisetinin de %80'i eğitim, %20'si ise doğrulama amacıyla kullanılmıştır.

Çizelge 5.7. CICFlowMeter ile Elde Edilen Alanlar

Flow ID, Src IP, Src Port, Dst IP, Dst Port, Protocol, Timestamp, Flow Duration, Tot Fwd Pkts, Tot Bwd Pkts, TotLen Fwd Pkts, TotLen Bwd Pkts, Fwd Pkt Len Max, Fwd Pkt Len Min, Fwd Pkt Len Mean, Fwd Pkt Len Std, Bwd Pkt Len Max, Bwd Pkt Len Min, Bwd Pkt Len Mean, Bwd Pkt Len Std, Flow Byts/s, Flow Pkts/s, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Flow IAT Min, Fwd IAT Tot, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Fwd IAT Min, Bwd IAT Tot, Bwd IAT Mean, Bwd IAT Std, Bwd IAT Max, Bwd IAT Min, Fwd PSH Flags, Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, Fwd Header Len, Bwd Header Len, Fwd Pkts/s, Bwd Pkts/s, Pkt Len Min, Pkt Len Max, Pkt Len Mean, Pkt Len Std, Pkt Len Var, FIN Flag Cnt, SYN Flag Cnt, RST Flag Cnt, PSH Flag Cnt, ACK Flag Cnt, URG Flag Cnt, CWE Flag Count, ECE Flag Cnt, Down/Up Ratio, Pkt Size Avg, Fwd Seg Size Avg, Bwd Seg Size Avg, Fwd Byts/b Avg, Fwd Pkts/b Avg, Fwd Blk Rate Avg, Bwd Byts/b Avg, Bwd Pkts/b Avg, Bwd Blk Rate Avg, Subflow Fwd Pkts, Subflow Fwd Byts, Subflow Bwd Pkts, Subflow Bwd Byts, Init Fwd Win Byts, Init Bwd Win Byts, Fwd Act Data Pkts, Fwd Seg Size Min, Active Mean, Active Std, Active Max, Active Min, Idle Mean, Idle Std, Idle Max, Idle Min

Daha önce de vurgulandığı üzere, üretilen verisetinin gizlilik derecesine sahip olması önemli bir sorun olarak karşımıza çıkmaktadır. IDS geliştirme çalışmalarında kaliteli veriseti kullanımı oldukça önemlidir. İnternet üzerinden erişilebilen verisetlerinin güncel olmama, yeterli miktarda veri içermeme, küçük ağlar üzerinden üretilme, yetersiz normal trafik içerme, sınıflandırma hataları barındırma gibi sorunları bulunmaktadır. Bu nedenle birçok kurum tarafından IDS veriseti üretim çalışmaları yürütülmektedir. Ancak güvenlik endişeleri nedeniyle tasnif dışı seviyeye indirilerek akademik amaçla kullanılacak da olsa veriseti paylaşımı yapılamamaktadır. Bu sorunun çözülmesi durumunda, Locked Shields gibi, oldukça büyük bir altyapı üzerinde gerçekleştirilen, yüze yakın atak tipi barındıran, iki günlük zaman zarfında atak trafiğinin yanısıra yeterli seviyede normal trafik üretimine imkan sunan, her yıl düzenlenmesi sayesinde güncel atak tiplerinin uygulanmasını sağlayan bir tatbikatın akademik seviyede kullanılabilir bir IDS veriseti üretimi için önemli bir fırsat olduğu değerlendirilmiştir. Tezin devam eden bölümünde gizlilik dereceli veriseti üretim probleminin nasıl çözüldüğü açıklanmaktadır.

5.4. Locked Shields Ağ Trafiğinden Tasnif Dışı Veriseti Üretimine Yönelik Öneri

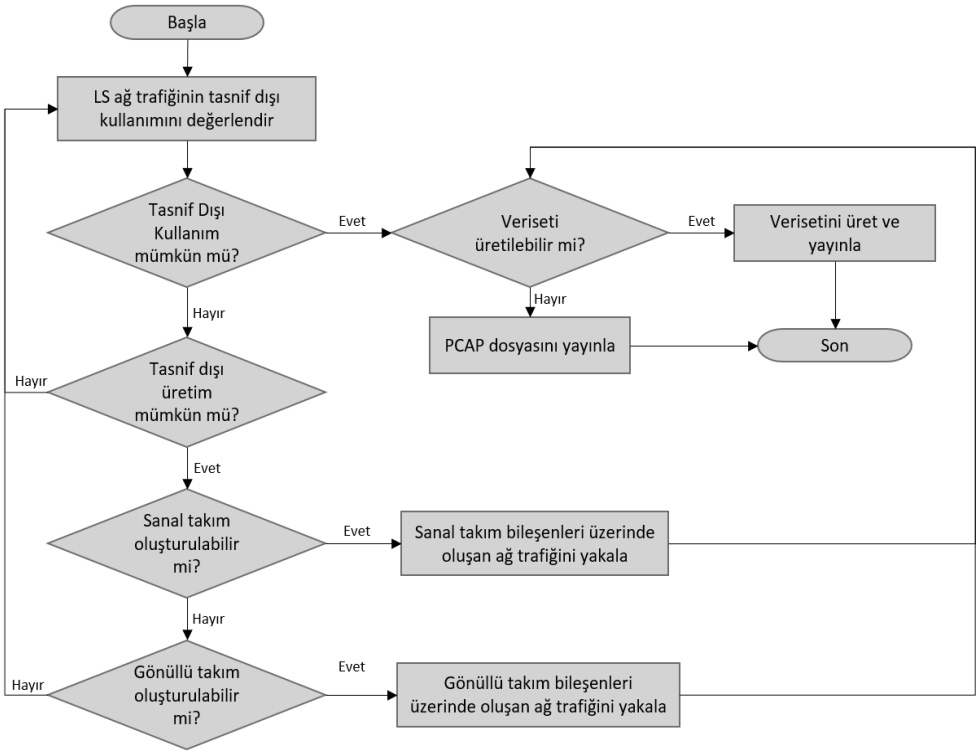
Tez çalışmasının en önemli hedeflerinden biri akademik alanda kullanılabilecek özgün ve kaliteli bir veriseti üretimi olarak belirlenmiştir. Locked Shields tatbikatı bu özelliklere sahip bir verisetinin üretimi için önemli bir altyapı sağladığı için bu altyapı kullanılarak tasnif dışı

veriseti üretimi için gerekli çalışmalara başlanmıştır. Bu kapsamda yapılan çalışmanın detayları aşağıda sunulmuştur:

1. İlk olarak, Locked Shields tatbikat altyapısı özelliklerinin uygun olmaması ve üretilen PCAP dosyalarının ülkelere özgü olması nedenleriyle, mevcut durumda tasnif dışı veriseti üretiminin mümkün olmadığı tespiti yapılmıştır.
2. Tatbikata 40'a yakın ülkeden 24 takım tarafından katılım sağlanmaktadır. Tatbikatta 24 takım bulunmasının nedeni, tatbikat senaryosunda savunulan ülkenin 24 alana bölünmesi ve her takımdan bu bölümlerde bulunan ve tamamen aynı özelliklere sahip enerji, su arıtma, haberleşme, intranet, hava savunma vb. sistemlerini aynı kaynaklarla savunmalarının beklenmesidir. Kırmızı takım tarafından gerçekleştirilen istemci, ağ ve örün sayfası ataklarına karşı her bir ülke/takım, sorumluğunda bulunan sistemleri kendi strateji ve taktikleri ile savunmaktadır. Bu durum, ülkelere özgü ağ trafiği ve buna bağlı olarak gizlilik dereceli PCAP dosyalarının üretilmesine neden olmaktadır.
3. Üretilen PCAP dosyaları tatbikat sonrası ülkelerin erişimine açılmakta, bir ülke bir başka ülkenin PCAP dosyasına erişim sağlayamamakta, ülkeler kendi PCAP dosyalarını istedikleri amaçlar için kullanabilmektedir. Ancak PCAP dosyaları tasnif dışı üzeri sınıflandırıldığından genel erişime açılamamakta ve bunlardan elde edilen ürünler paylaşılamamaktadır.
4. Bu değerlendirme neticesinde, tasnif dışı PCAP dosyası üretimi için tatbikata yönelik bir geliştirme yapılması gerektiği belirlenmiştir. Tatbikat altyapısı kurulumundan sorumlu birim ile yapılan görüşmeler neticesinde, tatbikata yeni bir takım dahil edilmesi ve bu takıma ait ağ trafiği yakalanarak PCAP dosyalarının oluşturulmasının en uygulanabilir çözüm olduğu tespit edilmiştir.
5. Oluşturulacak takımın idealde sanal bir takım olması, sanal bileşen ve oyuncuları içermesi ve bu takımın ağında yürütülen faaliyetlerin otonom şekilde gözlemlenerek ağ trafiğinin de otonom elde edilmesi gerektiği değerlendirilmiştir. Bu amaçla ihtiyaç duyulan altyapı ve takım elemanlarının geliştirilmesi ve kullanılabilir hale getirilmesi için inceleme çalışmalarına başlanmıştır.
6. Bununla birlikte, özellikle atak tiplerinin doğru belirlenmesi ve veriseti etiketleme işlemlerinin doğru gerçekleştirilmesi için kırmızı takıma sorumluluklar verilmesi,

gerekirse kırmızı takım bünyesinde sadece bu takıma saldırı düzenleyecek personel belirlenmesi gerektiği ortaya koyulmuştur.

7. Bu çalışmanın olgunlaştırılması amacıyla ilk adım olarak, literatürde sıklıkla kullanılan ve internet üzerinden erişilebilen verisetlerinin özelliklerini, hangi saldırı tiplerini içerdiklerini, artı ve eksi yönlerini, Locked Shields tatbikatı neticesinde elde edilecek tasnif dışı verisetinin diğer verisetleri ile kıyaslanmasını, yine bu verisetinin her yıl güncellenerek akademi ve endüstrinin kullanımına verilmesinin sağlayacağı avantajları, verisetinde bulunması muhtemel atak tiplerini, verisetinin veya PCAP dosyasının paylaşılması ile ilgili değerlendirmeyi, verisetinin üretimi için gerekli altyapı elemanlarını, veri yakalama ve işleme için kullanılacak araçları içeren "Data Quality Problem in AI-Based Network Intrusion Detection Systems Studies and a Solution Proposal" konulu makale CyCon (International Conference on Cyber Conflict) 2022 konferansında yayınlanmıştır.
8. Bu kapsamda, 2023 yılında icra edilen Locked Shields tatbikatında uygulamaya geçilmesi ve yukarıda bahsedilen işlem adımları takip edilerek verisetinin üretilmesi planlanmıştır. Locked Shields tatbikatı ağ trafiğinden tasnif dışı veriseti üretimi için oluşturulan iş akışı Şekil 5.6'da sunulmuştur.



Şekil 5.6. Locked Shields tatbikatı ağ trafiğinden tasnif dışı veriseti üretimi iş akışı.

5.5. Locked Shields Tatbikatına Sanal Mavi Takım Eklenmesi ve Veriseti Üretimi

Ağ sızma veriseti üretimi çeşitli zorluklar barındırmaktadır. Geniş ve gerçekçi bir altyapının olması, yeterli miktarda gerçekçi zararlı aktivitenin uygulanması, saldırı gerçekleştirenler hakkında detaylı bilgiye sahip olunması, gizlilik dereceli ve hassas bilgi içermemesi, zararsız ve zararlı kayıtların dengeli bir şekilde verisetinde yer alması gerekmektedir. Bu kapsamda, siber güvenlik alanında uluslararası olarak düzenlenen en önemli konferanslardan biri olarak değerledirilen CyCon (International Conference on Cyber Conflict) konferansında yayınlanan "Data Quality Problem in AI-based Network Intrusion Detection Systems Studies and a Solution Proposal" başlıklı makalemizde yapılan öneri doğrultusunda çalışmalar gerçekleştirilmiştir.

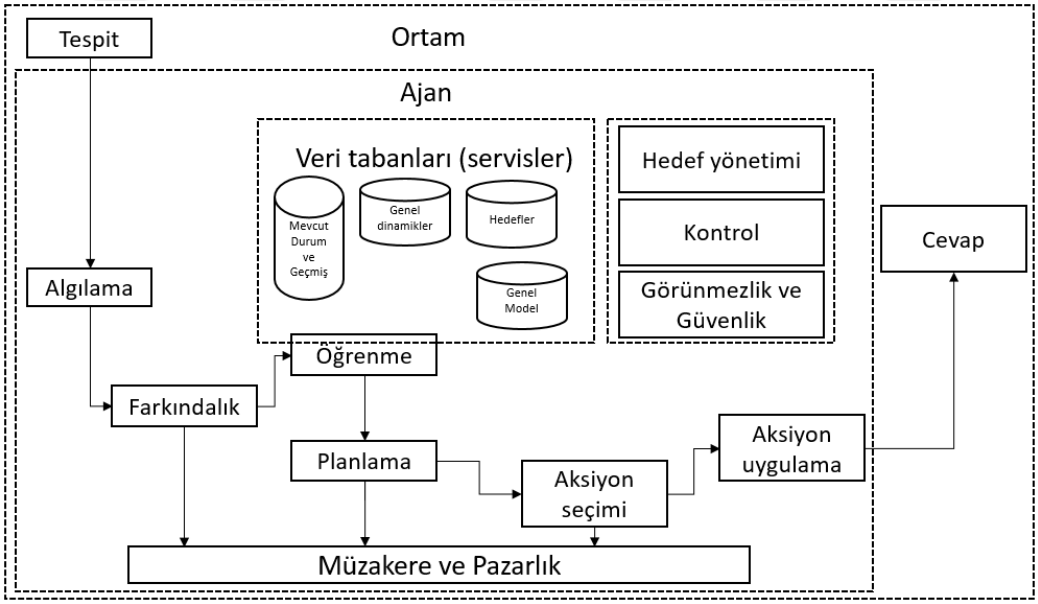
Üretilmesi hedeflenen verisetinin özgün olmasının yanı sıra, tatbikat altyapısının her yıl yeni cihazlarla güncellenmesi, gerçekleştirilen atak kampanyalarının güncel saldırı tiplerini yansıtması, normal ve zararlı ağ trafiğinin arzu edilen kriterler doğrultusunda ayarlanabilir olması, ağ trafiğinin elde edilmesi için yeterli süreye sahip olunması gibi IDS veriseti üretim sürecindeki gerekli özelliklerin karşılanıyor olması sayesinde çalışmanın literatüre önemli katkısının olacağı bilinciyle oluşturulan çalışma grubu tarafından çalışmalar sürdürülmüştür. Çalışmalar, veri toplama, veriseti üretimi ve veriseti analizi olmak üzere üç aşamada gerçekleştirilmiştir.

Veri toplama aşaması:

Akademi ve endüstri ihtiyaçlarına cevap verebilmesi amacıyla, verisetinin gizlilik derecesine haiz veri içermemesi gerekmektedir. Bu nedenle tatbikata katılım sağlayan ülkelerin ağ trafikleri veriseti üretimi için kullanılamamaktadır. Bu problemin aşılması amacıyla, Locked Shields 2023 tatbikatı test faaliyeti (Partners Run) altyapısına Autonomous Intelligent Agents (AICA) çerçevesi kurularak diğer tüm mavi takımlardan bağımsız bir sanal mavi takım (Virtual Blue Team –VBT) eklenmesi kararlaştırılmıştır. AICA çerçevesi birçok bileşeni barındırmaktadır. Şekil 5.7'de AICA çerçevesinin genel yapısı görülmektedir. Çerçeve incelendiğinde, saldırı algılama, mevcut bilgilerden öğrenme, durumsal farkındalık oluşturma, uygulanacak cevabı değerlendirme ve cevap üretim merkezlerinin olduğu görülmektedir [132]. Bu merkezlerin kurulumu ve etkin şekilde

çalışması için birçok alt bileşen ve uygulamanın kurulumu ve konfigürasyonu gerçekleştirilmiştir.

Açık kaynak ağ sızma tespit sistemi SURICATA, netflow, antivirüs, http log aracı, ağ takip ve gözlem araçları gibi çoklu ajan bileşenlerinin yanı sıra, sanal makinelere Ansible otomasyon yazılımı (kontrol merkezi), veri depolama ve sorgulama işlemlerinin JSON ile daha hızlı yapılmasını sağlayan Elasticsearch ve bunun yönetimi için Kibana arayüzü, olay yönetim aracı olarak Apache Kafka kurulumları gerçekleştirilmiştir. Ayrıca, kırmızı takım davranışlarının yakından takip edilmesi ve böylece doğru etiketlemenin yapılabilmesi amacıyla çalışma grubu tarafından Frankenstack çerçevesinden yararlanılmıştır.



Şekil 5.7. AICA çerçevesi [132].

Çalışma grubu tarafından, ağ trafiğinin yakalanması amacıyla tüm VBT makinelerinde oluşan ağ trafiği, bir Arkime bileşenine yansıtılmıştır. Bu Arkime bileşenin kullanımı ile ağ trafiği PCAP formatında elde edilmiş ve veri analizi yapılmıştır. Bununla birlikte, normalde sadece yeşil takımın erişebileceği VBT sanal bileşenlerinin MAC ve IP adresleri ile sanal internet servis sağlayıcısı gibi tüm destek altyapısına ilişkin bilgilere erişim sağlanmıştır. Ayrıca, kırmızı takımın saldırı düzenlenmeyi planladığı ağ segmentleri, IP adresleri ve domain isim bilgilerine de tatbikat öncesi erişim sağlanmıştır. Böylece kırmızı takım tarafından kullanılan atak makinelerine ait IP ve MAC adres bilgilerinden yararlanılarak dinamik ve lokal açılardan sorunlu trafik üreten IP adreslerinin gözlemlenmesi

hedeflenmiştir. Çalışmanın en önemli hedefi, içerdiği kayıtları doğru etiketlenmiş bir veriseti üretmek olduğundan, bahse konu bilgiler veri etiketleme işleminde etkinlikle kullanılmıştır.

Tatbikatta sanal mavi takım koruma faaliyetlerinin yürütülmesi amacıyla AICA bileşenlerinden istifade edilmiştir. Bunlardan en önemlilerinden biri olan SURICATA, ağ trafiğinin gözlemlenmesi ve uyarı üretmesi amacıyla sadece herkes tarafından erişilebilir Emreging Threats Open Ruleset, SURICATA Traffic ID Rule Set gibi kural setleri ile konfigüre edilmiştir. Böylece yeni bir kural seti geliştirme çalışmasına ihtiyaç olmamıştır. Üretilen alarmlar Kafka üzerine Syslog üzerinden aktarılmıştır. Kafka sayesinde birçok kaynakta üretilen sistem logları tek bir ortama aktarılmıştır. Böylece olaylara yönelik tespit ve cevap süreleri daha hızlı hale yapılmıştır. Alarmlar Extensible Event Format (EVE) kullanılarak JSON formatında üretildiğinden birçok güvenlik aracı ve ortamının daha rahat entegrasyonunu ve elde edilen verinin daha kolay analizini yapmayı sağlamıştır. Ayrıca false SURICATA üzerindeki pozitif alarm sayısının azaltılması ve sistemin daha iyi optimizasyonu amacıyla, tatbikat öncesi alışma periyodunda belirlenen imzalar devre dışı bırakılmıştır.

Veriseti üretim aşaması:

Locked Shields tatbikat altyapısı ile atak kampanyalarının testinin gerçekleştirildiği Partners Run faaliyeti, ilk günü alışma olmak üzere iki günde icra edilmektedir. Alışma gününde her mavi takım savunmasını yapacağı ağ üzerinde çalışma gerçekleştirmektedir. İkinci gün ise kırmızı takım tarafından, ağ servislerine zafiyet eklenmesi, ağ saldırıları, zararlı yazılım kullanımı, sistem arka kapılarının oluşturulması gibi birçok kategoride atak kampanyaları dost ülke silahlı kuvvetleri, bankası ve enerji grubu üzerine düzenlenmektedir. Her birimin altında birçok alt sistem mevcuttur. Bunradan bazıları, 5G altyapısı, intranet, genel servisler, sınır güvenliği, hava savunma, risk yönetim sistemi, SCADA olarak sayılabilir.

Partners Run faaliyeti ikinci gününde elde edilen ağ trafiği PCAP formatına dönüştürülmüştür. Daha sonra zaman-tabanlı özelliklerin çıkarılması amacıyla CICFlowmeter aracı kullanılmıştır. Faaliyet sonrası Arkime üzerinde gerçekleştirilen analiz neticesinde, tatbikatta kullanılan 26 uygulama protokolü belirlenmiş, CICFlowmeter üzerinde gerekli düzenlemeler yapılarak ağ paketlerinin bu protokollere göre ayrılması sağlanmıştır.

Yapılan çalışmalar neticesinde üretilen veriseti LSPR23 olarak isimlendirilmiştir. LSPR23 veriseti içerisinde aşağıdaki özellikler yer almaktadır.

- 84 adet CICFlowmeter özelliği [133],
- 5 adet SURICATA özelliği,
- Kanzig ve diğerleri tarafından yapılan çalışmada [131] belirlenen iki özellik,
- 1 adet Bro/Zeek bağlantı durum özelliği,
- Tespit edilen protokolü gösteren bir servis özelliği.

Çalışma grubu tarafından, üretilen verisetinin üretim süreci ve diğer özelliklerini içeren bir makale halihazırda hazırlanmaktadır. Makalenin yayımlanmasını takiben veriseti "https://zenodo.org/" üzerinden erişime açılacaktır.

Veriseti analiz aşaması:

LSPR23 veriseti, ağ konfigürasyonu, ağ trafiği, veri etiketleme, etkileşimler, veri yakalama, mevcut protokoller, atak çeşitliliği, heterojenlik, özellik kümesi ve metadata açılarından değerlendirilmiş ve değerlendirme sonuçları aşağıda sunulmuştur.

Ağ konfigürasyonu açısından bakıldığında, sadece Linux ve Windows sistemlerinin yanı sıra, IT/OT birleşimi sistemlerin yönetiminde kullanılan SCADA gibi diğer sistemlerin de yer aldığı tatbikat altyapısının detaylarından daha önceki bölümlerde bahsedilmiştir.

Etkileşim açısından bakıldığında, skorlama bölümü hariç tatbikat altyapısında yer alan tüm sistemlerin etkileşimleri kapsanmış, böylece olası tüm ataklar yakalanmıştır.

Veri yakalama açısından bakıldığında, tüm sistemlere ait ağ trafiği kaydedilmiştir. Atak gerçekleşmeyen bölümde veri yakalama yapılmamıştır.

Ağ trafiği açısından değerlendirildiğinde, yakalanan ağ trafiği içerisinde tatbikata katılım sağlayan tüm takımların trafikleri yer aldığından, tam bir veriseti üretimi için gerekli veri elde edilmiştir.

Mevcut protokollerin yanı sıra, birçok özel sistem protokolü de kapsanmıştır.

Veri etiketleme açısından bakıldığında, veriseti içerisindeki tüm kayıtlar saldırı veya zararsız olarak etiketlenmiştir. Etiketleme işleminde daha önce bahsedilen kurallardan, imzalardan ve Suricata tarafından üretilen anomali etiketlerinden yararlanılmıştır.

Atak çeşitliliği açısından bakıldığında, veriseti içerisinde, istemci, ağ ve örün sayfası ataklarının yanı sıra özel sistemlere yönelik gerçekleştirilen ataklara da yer verilmektedir. Bu, üretilen verisetini mevcut erişilebilir verisetlerinden ayıran önemli bir özelliktir. Kırmızı takım tarafından tatbikat altyapısında yer alan tüm teknolojilere çeşitli metotlarla saldırılar gerçekleştirilmiştir. Ataklar, daha az etkili olanlarla başlamış, yıkıcı olanlarla sonlanmıştır.

Heterojenlik açısından bakıldığında, atak ve zararsız kayıt dağılımının uygun şekilde yapıldığı görülmektedir. Bununla birlikte, ileride gerçekleştirilecek çalışmalarda kullanılmak üzere istemci tabanlı log kayıtları da elde edilmiştir.

Özellik kümesi açısından bakıldığında, yaklaşık 2 milyonu zararlı olmak üzere toplam 16 milyon kayıt, 93 adet özellik ile üretilmiştir. Veriseti CSV formatında hazırlanmış, endekslenmiş özellikler için tablolar ve ilave metadata sunulmuştur.

Metadata açısından bakıldığında, tatbikat destek sistemlerinden elde edilen tüm metadata yakalanmış ve kaydedilmiştir. Bu metadata içerisinde ağ segmenti, MAC ve IP adresleri, host ve domain isimleri gibi metadata bilgileri seçilmiştir.

Sonuç olarak, üretilen LSPR23 verisetinin, kaliteli ve özgün verisetinde olması gereken kriterleri karşıladığı, böylece önümüzdeki dönemde gerçekleştirilecek ağ sızma tespit sistemi geliştirme çalışmalarında etkinlikle kullanılabileceği değerlendirilmektedir.

5.6. Modelin Güncel ve Özgün Verisetine Uygulanması

Yapılan inceleme neticesinde, literatürdeki çalışmaların büyük bölümünde, geliştirilen modellerin NSL-KDD veya KDDCUP 99 verisetlerine uygulanması ile sonuçların elde edildiği görülmüştür. Ancak etkin bir ağ sızma tespit sistemi üretimi için gerçek zamanlı ve güncel verisetleri ile eğitim ve test işlemleri yapılması gerekmektedir. Bu nedenle tez kapsamında geliştirilen LSTM-NIDS modeli güncel veriseti UNSW-NB15 veriseti ile eğitilerek test edilmiş ve sonuçları paylaşılmıştır.

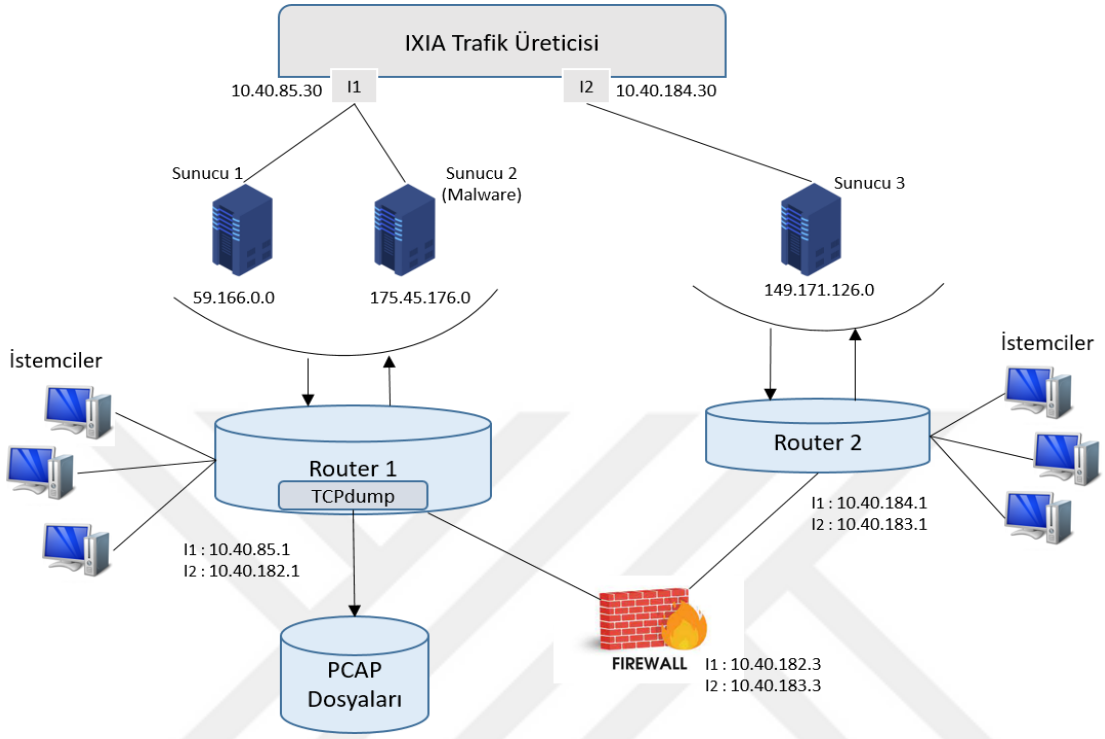
Bu bölümde LSTM-NIDS modeli UNSW-NB15 veriseti ile eğitilerek test edilmiş ve sonuçları paylaşılmıştır. Bu kapsamda öncelikle UNSW-NB15 veriseti hakkında bilgi verilmiş, daha sonra LSTM-NIDS modeli kullanılarak gerçekleştirilen eğitim ve testler neticesinde elde edilen sonuçlar paylaşılmıştır.

UNSW-NB15 veriseti

UNSW-NB15 Veriseti için gerekli ağ paketleri, modern normal aktiviteleri ve sentetik güncel saldırı davranışlarını içerecek şekilde Avustralya Siber Güvenlik Merkezi'nde bulunan IXIA PerfectStorm aracı kullanılarak üretilmiştir. Tcpdump aracı 100 GB trafic verisinin yakalanması için kullanılmıştır. Veriseti içerisinde, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms olmak üzere toplam dokuz farklı saldırı tipine ait veri yer almaktadır. Toplam kayıt sayısı 2540044'tür. Bu verisetinin 175341 kayıtlık bölümü eğitim, 82332 kayıtlık bölümü ise test için konfigüre edilmiştir. Veriseti içerisinde, kaydın saldırı veya normal olduğuna dair etiket ile birlikte toplam 49 alan bulunmaktadır. Bu alanlardan 6'sı nominal, 3'ü binary, 2'si timestamp ve kalanlar sayısal değerleri içermektedir.

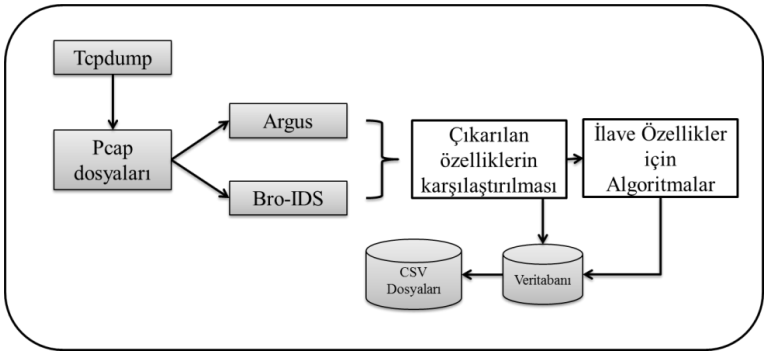
UNSW-NB15 veriseti üretimi için inşa edilen ağ mimarisi Şekil 6.1'de sunulmuştur [133]. Şekilden de görüleceği üzere IXIA trafik üreticisi, üç sanal sunucu ile konfigüre edilmiştir. İlk ve üçüncü sunucular normal trafik üretimi, ikinci sunucu ise anormal ve zararlı ağ trafiği üretimi için konfigüre edilmiştir. Sunucular host makinelere iki yönlendirici üzerinden bağlanmaktadır. Bu sunucular, normal ve anormal tüm ağ trafiğini geçirecek şekilde konfigüre edilen firewall'a bağlıdır. Tcpdump aracı, ilk yönlendirici üzerine kurularak ve simülasyon süresince Pcap dosyalarının yakalanması için kullanılmaktadır. IXIA trafik

üreticisi hem normal hem de saldırı trafiği üretimi sağlamaktadır. Saldırı davranışları gerçek bir modern tehdit ortamı gösterimi için CVE sitesinden beslenmektedir.



Şekil 6.1. UNSW-NB15 veriseti üretimi için kurulan ağ mimarisi [133].

49 özellik içeren UNSW-NB15 verisetinin son halinin üretimi için kullanılan framework mimarisi Şekil 6.2'de sunulmuştur [134]. Yukarıda da belirtildiği gibi ağ trafiği tcpdump aracı ile yakalanarak PCAP dosyaları üretilmiştir. UNSW-NB15 verisetine ait özellikler, Argus ve Bro-IDS araçları ile C# programlama dilinde geliştirilmiş toplam 12 algoritma ile çıkarılmıştır. Bu araçlar, Linux Ubuntu 14.0.4 işletim sistemine sahip bir makine üzerine kurulmuş ve konfigüre edilmiştir.



Şekil 6.2. UNSW-NB15 veriseti için kullanılan framework mimarisi [133].

Argus ve Bro-IDS araçları ile çıkarılan akış özellikler Çizelge 6.1'de görülmektedir.

Çizelge 6.1. Akış özellikleri [133].

No.	İsim	T.	Açıklama
1	scrip	N	Kaynak IP adresi
2	sport	I	Kaynak port numarası
3	dstip	N	Hedef IP adresi
4	dsport	I	Hedef port numarası
5	proto	N	Transaction protokolü

Argus ve Bro-IDS araçları tarafından üretilen ve çakışan özellikler üç grupta ele alınmaktadır. Bunlar, temel, içerik ve zaman özellikleridir. Bu araçlarla üretilen temel özellikler Çizelge 6.2'de görülmektedir.

Çizelge 6.2. Temel özellikler [133].

No	İsim	T.	Açıklama
6	state	N	Durum ve onun dayandığı protokol (ACC,CLO gibi)
7	dur	F	Toplam kayıt süresi
8	sbytes	I	Kaynaktan hedefe byte'ları
9	dbytes	I	Hedeften kaynağa byte'ları
10	sttl	I	Kaynaktan hedefe yaşam süresi
11	dttl	I	Hedeften kaynağa yaşam süresi
12	sloss	I	Tekrar gönderilen veya atılan kaynak paketleri
13	dloss	I	Tekrar gönderilen veya atılan hedef paketleri
14	service	N	http, ftp, ssh, dns vb.
15	sload	F	Saniyede gönderilen kaynak bitleri
16	dload	F	Saniyede gönderilen hedef bitleri
17	spkts	I	Kaynaktan hedefe paket sayısı
18	dpkts	I	Hedeften kaynağa paket sayısı

UNSW-NB15 verisetinde yer alan ilk 35 özellik, veri paketlerinden elde edilen bilgiyi temsil etmektedir. Bu özelliklerin büyük bölümü, header paketlerinden elde edilmiştir. Verisetinde yer alan içerik özellikleri Çizelge 6.3'te, zaman özellikleri ise Çizelge 6.4'te sunulmuştur.

Çizelge 6.3. İçerik özellikleri [133].

No	İsim	T.	Açıklama
19	swin	I	Kaynak TCP window advertisement
20	dwin	I	Hedef TCP window advertisement
21	stcpb	I	Kaynak TCP sıra numarası
22	dtcpb	I	Hedef TCP sıra numarası
23	smeansz	I	Kaynak tarafından iletilen akış paket boyutunun ortalaması
24	dmeansz	I	Hedef tarafından iletilen akış paket boyutunun ortalaması
25	trans_depth	I	http talebi/yanıtı işleminin bağlantı derinliği
26	res_bdy_len	I	Sunucunun http servisinden aktarılan verinin içerik boyutu

Çizelge 6.4. Zaman özellikleri [133].

No	İsim	T.	Açıklama
27	sjit	F	Kaynak titremesi (mSec)
28	djit	F	Hedef titremesi (mSec)
29	stime	T	Kayıt başlangıç zamanı
30	ltime	T	Kayıt bitiş zamanı
31	sintpkt	F	Kaynak paketler arası varış zamanı (mSec)
32	dintpkt	F	Hedef paketler arası varış zamanı (mSec)
33	tcpprtt	F	TCP'nin 'synack' ve 'ackdat' toplamı
34	synack	F	TCP'nin SYN ve SYN_ACK paketleri arasındaki geçen süre
35	ackdat	F	TCP'nin SYN_ACK ve SYN paketleri arasındaki geçen süre

UNSW-NB15 veriseti eğitim ve doğrulama işlemleri işlemleri 175341 kayıt içeren eğitim verisetinde gerçekleştirilmiştir. Normalizasyon ve nümerikleştirme işlemleri gerçekleştirilmiş haliyle temin edilen verisetinin LSTM-NIDS modelinde kullanımı amacıyla uygulamalarda gerekli düzenleme işlemleri yapılmıştır. Test amacıyla ise 82332 kayıt içeren test veriseti kullanılmıştır. Bu veriseti de normalizasyon ve nümerikleştirme işlemleri amacıyla ön işleme tabi tutulmuştur.

5.7. Modelin UNSW-NB15 Veriseti ile Testi Neticesinde Elde Edilen Sonuçlar

UNSW-NB15 veriseti kullanılarak gerçekleştirilen test işlemlerinde, uygulama başarı seviyesinin değerlendirilmesi amacıyla NSL-KDD eğitim ve testinde kullanılan hiperparametre değerleri kullanılmıştır. Model düzenlileştirme işlemlerini de içermektedir. Modelin UNSW-NB15 veriseti ile eğitimi ve testi neticesinde elde edilen sonuçlar Çizelge 6.5'te sunulmuştur.

Çizelge 6.5. LSTM-NIDS modelinin UNSW-NB15 veriseti ile eğitimi ve testi neticesinde elde edilen sonuçlar.

UNSW-NB15 Veriseti Test İşlem Sonuçları			
	Kesinlik (Accuracy)	Hassasiyet (Precision)	Geri Çağırma (Recall)
Katman Sayısı : 3 *Katmanlardaki Hücre Sayısı: 32* *LSTM Aktivasyon Fonk.: Relu* *Dropout1: 0.2* *Dense Aktivasyon Fonk.:sigmoid* *Optimizer: rmsprop* *Kayıp Fonk.: Binary_cross_ent* *Tekrar Sayısı: 100* *Batch_size: 256* *Dataset Size:* *Eğitim→175341 %80* *Doğrulama →175341 %20* *Test →82332*	Eğitim: % 93,74 Doğrulama: % 98,57 Test: **% 87,65**	Eğitim: % 94,00 Doğrulama: % 99,77 Test: % 89,92	Eğitim: % 95,68 Doğrulama: % 98,57 Test: % 87,36

Çizelge 6.5'te sunulan sonuçlar incelendiğinde, LSTM-nIDS modelinin güncel verisetlerinde başarı ile kullanılabileceği görülmüştür.

5.8. Locked Shileds Veriseti ile Yapılan Test Sonuçları

Önceki bölümlerde açıklandığı üzere Locked Shields tatbikatı test faaliyeti neticesinde, 1989484 adedi atak olmak üzere yaklaşık 16 milyon kayıt içeren bir veriseti üretilmiştir. Bu verisetinin geçerliliğinin gözlemlenebilmesi, makine öğrenmesi ve yapay zeka tabanlı modellerin geliştirilebilirliğine uygunluğunun belirlenmesi ve daha önce benzer çalışmalarda elde edilen sonuçlarla karşılaştırma yapılabilmesi amacıyla Random Forest yöntemi kullanılarak eğitim ve test işlemleri gerçekleştirilmiş, neticede Çizelge 6.8'de sunulan sonuçlar elde edilmiştir.

Çizelge 6.8. LSPR23 veriseti ile yapılan eğitim ve test sonuçları.

Parametre	Değer (%)
IDS imzaları ile elde edilen en yüksek precision değeri	91,1
Random forest yöntemi ile eğitim sonrası F1/Recall skoru	99,7
Random forest yöntemi ile eğitim sonrası kesinlik değeri	99,99

Elde edilen sonuçlar doğrultusunda, üretilen verisetinin IDS geliştirme çalışmalarında kullanılabileceği değerlendirilmektedir.

6. SONUÇ VE TARTIŞMA

6.1. Sonuç

Günümüzde yaşanan teknolojik gelişmeler, geçmişe kıyasla daha fazla siber güvenlik endişesi doğurmaktadır. Nesnelerin interneti, 5/6 G, yapay zekâ uygulamaları, blockchain gibi her yeni teknoloji birçok siber güvenlik riskini beraberinde getirmektedir. Bir başka ifadeyle, veri işleme ve iletişimini gerektiren her yeni teknoloji, yeni siber güvenlik sorunlarına neden olmaktadır. Örneğin, ana uygulamanın bulut mimari üzerinde kurulu sunucular üzerinde bulunduğu, istemci tarafında ise düşük kapasiteli uygulamaların yer aldığı çözümler gün geçtikçe yaygınlaşmaktadır. Bu altyapı, kullanım kolaylığı ve yönetilebilirlik açılarından birçok avantaj sağlamasına karşı, iki sistem arasında sürekli ve güvenilir bir ortamdan veri iletim gereksinimini doğurmaktadır. Bu gereksinim, güvenilir ağ altyapısı ve güvenli veri iletişimine olan ihtiyacı arttırmaktadır.

Yapılan literatür incelemesi neticesinde, bahse konu sistemlerin kurulumu için ağda oluşabilecek tehditlerin gerçek zamanlı tespiti ve önlenmesi amacıyla kullanılan ağ sızma tespit sistemlerine ihtiyaç olduğu belirlenmiştir. Güvenli ağ altyapısının önemli bir bileşeni olan ağ sızma tespit sistmelerinin performans ve etkinliklerinin geliştirilmesi amacıyla akademi ve endüstri tarafından yürütülen çalışmaların uzun süredir devam ettiği bilinmektedir. Ağ altyapılarına yönelik tehditlerin her geçen gün daha karmaşık hale gelmesi, ağ sızma tespit sistemlerine ilişkin çalışmaların önümüzdeki dönemde de aratarak devam edeceğini göstermektedir.

Ağ sızma tespit sistemlerinin kullanımı, internetin yaygınlaşmaya başladığı 1980'li yılların başına dayanmaktadır. İlk aşamada, önceden tanımlanan kuralları kullanarak tespit ve uyarı veren uzman sistemlerden yararlanılmıştır. Bu sistemler, basit saldırı tespitinde etkin şekilde kullanılmışlardır. Ancak zaman içeriside ağ saldırılarının, önceden tanımlanan kurallar çerçevesinde tespit edilemeyecek seviyede karmaşık hale gelmesi, araştırmacıları başka çözüm arayışlarına itmiştir. Neticede, çeşitli makine öğrenme yöntemlerinin ağ sızma tespit sistemlerinde kullanılabileceği görülmüş ve kural tabanlı yapıdan, eğitim tabanlı bir yapıya geçiş olmuştur. Yapılan literatür incelemesinde, birçok makine öğrenme yönteminin hem

yalnız hem de diğer yöntemlerle hibrit şekilde kullanımı neticesinde başarılı sonuçlar elde edildiği belirlenmiştir.

Günümüze yakın dönemde ise yapay zeka alanında devrim niteliğinde bir gelişme olarak değerlendirilen yapay sinir ağı tabanlı derin öğrenme yöntemleri yaygınlaşmaya başlamıştır. Bu yöntemler, donanım teknolojilerinde yaşanan gelişmeler ve üretilen yeni teknikler sayesinde oldukça yüksek seviyede işlem gücü sunar hale gelmiştir. Görüntü işleme ve ses tanıma gibi temel yapay zeka problemlerinin çözümünde gösterdikleri üstün başarılar, bu yöntemlerin otonom araçlar, robot teknolojileri, insansız hava araçları, medikal cihazlar, savunma sistemleri vb. birçok alanda kullanılmalarının yolunu açmıştır. Yapılan incelemede, son dönemde ağ sızma tespit sistemi geliştirme çalışmalarında derin öğrenme yöntemlerinin yaygın olarak kullanılmaya başlandığı görülmüştür.

Etkin ve yüksek performansa sahip bir ağ sızma tespit sistemi geliştirmek için doğru model ve yöntem seçiminin yanı sıra, üretilen modelin kaliteli ve yeterli veri ile eğitilerek test edilmesi de oldukça önemlidir. Bu nedenle ağ sızma tespit sistemi üretim çalışmalarında sadece model ve yönteme odaklanmak, arzu edilen performans seviyesine ulaşmayı engelleyecektir. Ağ sızma tespit sistemlerinde kullanılan bir verisetinin kalitesinin değerlendirilmesinde kullanılan bazı parametreler vardır. Örneğin, üretim tarihi, normal trafik içermesi, atak trafik oranı, veri miktarı, verinin ne kadar sürede toplandığı, etiketleme, verinin hangi tip ağ üzerinden elde edildiği, daha önceden tanımlanmış bölümleri içerip içermediği gibi özellikler bir ağ sızma tespit verisetinin kalitesinin değerlendirilmesine yönelik fikir vermektedir. Bir verisetinin güncel olması, normal ve atak trafiği arasında bir balans olması, aşırı fazla veya az veri içermemesi, mümkün olduğu oranda gerçek ağ trafiği içermesi, gerçeğe yakın büyük bir ağ üzerinden elde edilmiş olması, verinin elde edilmesi için ağın yeterli süre izlenmesi gibi hususlar, verisetinin kaliteli olduğunun ana göstergeleri olarak karşımıza çıkmaktadır.

Yukarıdaki hususların bir bölümü tez çalışması kapsamında belirlenmiş ve elde edilen bulgular ile literatür taraması sonuçları teze yansıtılmıştır. Elde edilen bilgiler doğrultusunda tez çalışması şekillenmiş ve çalışma iki öz temele dayandırılmıştır. İlk olarak modele odaklanılmış, sağladığı uzun süreli kısa dönem bellek sayesinde önemli bir hesaplama gücü sağlayan long short-term memory (LSTM) tabanlı bir model geliştirmeye karar verilmiştir. İkinci olarak verisetine odaklanılmış ve yapılan inceleme neticesinde, bir ağ sızma tespit

sisteminin başarısının doğru değerlendirilebilmesi için yaygın veriseti kullanımının önemli olduğu bulgusu doğrultusunda, incelenen çoğu akademik çalışmada kullanıldığı tespit edilen NSL-KDD verisetinin model eğitim ve testinde kullanımına karar verilmiştir.

Daha önce vurgulandığı gibi model geliştirme çalışmalarına LSTM modeli geliştirilerek başlanmıştır. Bu süreçte detaylı bir araştırma yapılarak LSTM modelinin inşası için gerekli bilgi elde edilmiştir. Katman sayısı, katmanlardaki hücre sayıları, optmizasyon algoritmaları, aktivasyon fonksiyonları, kayıp fonksiyonları gibi hiperparametre değerleri detaylı bir şekilde incelenmiş, birçok hiperparametre değeri ile test işlemleri yapılarak sonuçlar elde edilmiştir. Model performansının nasıl arttırılabileceği ile ilgili detaylı araştırmalar yapılarak düzenlileştirme teknikleri incelenmiş, bunlardan yaygın olarak kullanılanları modele yansıtılarak model performansının arttığı gözlemlenmiştir. Uzun süren testler neticesinde, LSTM'in sağladığı uzun süreli kısa dönem bellek yeteneğinin etkisinin araştırılmasına karar verilmiştir. LSTM modeli RNN yönteminin bir türevidir. LSTM yöntemini RNN'den ayıran en önemli özellik, barıdırdığı durum ve kapılar sayesinde sağladığı uzun süreli kısa bellektir. Bunların dışında bir başka RNN türevi yöntem ise GRU'dur ve bu yöntemin LSTM yönteminden farkı daha az sayıda kapı ile uzun süreli kısa bellek sağlamayı hedeflemesidir. Tez kapsamında yöntemlerin performansları arasındaki farkın, bir başka ifade ile uzun süreli kısa belleğin model performansına etkisinin belirlenmesi amacıyla, LSTM'in yanı sıra RNN ve GRU modelleri de geliştirilmiştir. Bu modeller orijinal NSL-KDD veriseti ile eğitilerek test edilmiş ve LSTM modelinin diğer modellere oranla daha başarılı sonuç verdiği, diğer yöntemlerin performans sonuçları arasında ise belirgin bir fark olmadığı belirlenmiştir. Bu konudaki değerlendirmeler ilgili bölümde paylaşılmıştır.

Model ile ilgili iki çalışma daha gerçekleştirilmiştir. Bunlardan ilki geliştirilen LSTM modelinin diğer model performansları ile kıyaslanması olmuş ve elde edilen sonuçlar göz önünde bulundurulduğunda, tez kapsamında geliştirilen modelin diğer modeller arasında oldukça iyi seviyede bir sonuç sağladığı gözlemlenmiştir. Bu bulgu neticesinde, tez kapsamında geliştirilen modelin, ağ sızma tespit sistemi geliştirme çalışmalarında kullanılabileceği değerlendirilmiştir. İkinci çalışma,modelin sadece NSL-KDD veriseti ile değerlendirilmesinin yeterli olmayacağı kabulüyle, güncel verisetlerinden UNSW-NB15 ve tez kapsamında üretilen örnek veriseti ile modelin eğitim ve testi gerçekleştirilmiş, elde edilen sonuçlar paylaşılmıştır. UNSW-NB15 veriseti ile yapılan testler neticesinde elde

edilen sonucun da literatürdeki diğer çalışma sonuçları ile kıyaslandığında başarılı olduğu gözlemlenmiştir.

Veriseti ile ilgili çalışmalar kapsamında öncelikle NSL-KDD verisetinin tüm özellikleri detaylı bir şekilde araştırılmış, literatürde bu veriseti kullanılarak gerçekleştirilen çalışmalar incelenmiştir. Veriseti ön işleme çalışmalarının önemi göz önünde bulundurularak NSL-KDD verisetinin nümerikleştirme ve normalizasyon işlemleri yapılmıştır. Model performansına etkisinin gözlemlenmesi amacıyla verisetinin farklı bölümlerinin eğitim ve test amaçları ile kullanımını sağlayan K-fold metodunun uygulanması sağlanmıştır.

NSL-KDD veriseti tez çalışması kapsamında kullanılan temel veriseti olmuştur. Verisetleri ile ilgili yapılan inceleme neticesinde, bazı verisetlerinde özellik çıkarım işleminin oldukça etkili sonuçlar verdiği gözlemlenmiştir. Bu kapsamda, özellik çıkarım işleminin NSL-KDD verisetine uygulanması neticesinde elde edilen sonuçların gözlemlenmesi amacıyla Chi-square, PCA ve lokal algoritma özellik çıkarım yöntemleri uygulanmış ve azaltılmış sayıda özellik içeren verisetleri elde edilmiştir. Bu verisetlerinin LSTM modeli ile testi neticesinde, performans açısından orijinal veriseti ile elde edilen sonuçların diğerlerine oranla belirgin şekilde iyi olduğu gözlemlenmiş, bu hususta gerekli değerlendirme ilgili bölümde sunulmuştur.

Tez kapsamında ayrıca derin öğrenme yöntemlerinin görüntü işleme alanındaki başarısı göz önünde bulundurularak bu kabiliyetin ağ sızma tespitinde kullanımına ilişkin inceleme gerçekleştirilmiştir. Yapılan inceleme neticesinde, nümerik verinin görsel veriye dönüştürülmesi ile elde edilen imaj verisetlerinin uygun bir model ile eğitilmesi ile başarılı sonuçlar elde edildiği tespit edilmiştir. Bu doğrultuda yapılan incelemeler neticesinde, evrişimsel sinir ağı (CNN) kullanılarak model eğitimi ve testinin yapılabileceği belirlenmiştir. Nümerik verinin imajlara dönüştürülmesi özgün bir yöntem ile gerçekleştirilmiş, üretilen CNN modeli ile yapılan test işlemleri neticesinde LSTM modeline oranla daha iyi sonuçlar elde edilmiştir.

Yapılan literatür incelemesi, internet üzerinden erişilebilen genel verisetlerinin avantaj ve dezavantajlarının daha iyi anlaşılmasını sağlamıştır. Bazı verisetlerinin oldukça eski olduğu, bu nedenle güncel atak tiplerini içermediği, bazı verisetlerinin yeterli miktarda normal trafik içermediği, bazılarının yanlış veya eksik etiketlenmiş kayıtlar içerdiği, bazı verisetlerinin

direk erişilebilir, bazılarının ise talep prosedürlerinin tamamlanması neticesinde elde edilebilir olduğu, bazı verisetlerinin PCAP dosyalarından dönüştürücü programlar ile veriseti formatına dönüştürüldüğü ve etiketleme işlemlerinin kaynak/hedef IP adresi, kaynak/hedef port numarası, protokol tipi, paket boyutu gibi değerlere bağlı kurallar çerçevesinde yapıldığı, bazılarının ise loglardan ve uzman görüşlerinden de yararlanılarak etiketlendikleri, bazı verisetlerinin aşırı büyük boyutlu olduğu ve çok zor işlenebildiği, bazılarının ise çok az sayıda veri içerdiği ve modelin yeterli seviyede eğitilememesine neden olduğu, bazı verisetlerinin profil tanımlaması yapılmış akışlar ile elde edildiği, böylece etiketleme işleminin kesin bir şekilde yapılabildiği, bazılarının sentetik, bazılarının ise gerçek ağ trafiğinden üretildiği, sentetik ağ üzerinden üretilenlerin pratik ağ davranışlarından ziyade teoriyi yansıttığı, bazı verisetlerinin üretiminde kullanılan altyapıda eski işletim sistemlerini içeren makinelerin kullanıldığı, bazılarının ise daha güncel işletim sistemleri ve yazılımları içeren ağlardan üretildikleri, bazı verisetlerinin küçük, bazılarının ise büyük ağlar üzerindeki trafiklerden elde edildiği gözlemlenmiştir.

İnternet üzerinden erişilebilen her verisetinin kendine özgü avantaj ve dezavantajları olduğu belirlenmiş, ağ sızma tespit sistemi geliştirme çalışmalarında kaliteli veriseti kullanımının oldukça önemli olduğu bilinciyle özgün bir veriseti üretimi için çalışmalara başlanmıştır. Geliştirilecek özgün verisetinin, yeterince geniş bir ağ altyapısı üzerinden elde edilmesi, kullanılan sistem ve yazılımların son teknolojiyi yansıtması, güncel atak tiplerini içermesi, atak-normal trafik balansının uygun olması, ne işlenemeyecek kadar büyük ne de modeli eğitemeyecek kadar küçük boyutlu olmaması, doğru etiketlenmiş kayıtlar içermesi, belirli bir frekansta yenilenebilir dinamik bir yapısının olması, gerçek zamanlı ağ trafiğinin özelliklerini yansıtması, ölçeklenebilir olması, genel erişime açık, akademik ve endüstri tarafından yürütülen ağ sızma tespit sistemi geliştirme çalışmalarına katkı sağlayacak seviyede kaliteli bir veriseti olması hedeflenmiştir. Yapılan inceleme ve araştırmalar neticesinde, siber savunma tatbikatlarının ağ sızma tespit sistemlerinin geliştirilmesinde kullanılabilecek kaliteli ve özgün bir veriseti üretimi için oldukça iyi bir ortam sunduğu, bu ortamın oldukça geniş bir altyapı üzerinden belirli bir süre boyunca hem normal hem de yeterli seviyede atak trafiğinin üretimine imkan sunduğu, her yıl düzenlenmelerinin güncel işletim sistemleri ile yazılımları içeren cihazların kullanımına ve güncel atak tiplerinin uygulanmasını sağladığı, üretilen PCAP dosyalarından yararlanılarak istenilen boyut ve içerikte ölçeklenebilir, dinamik bir verisetinin üretilebileceği belirlenmiştir. Bu kapsamda, dünyanın en büyük siber savunma tatbikatlarından biri olan Locked Shields tatbikatı

organizasyon ekibi ile birlikte çalışılmaya başlanmıştır. Böyle önemli bir tatbikat ve altyapıdan elde edilecek tasnif dışı bir verisetinin akademi ve endüstri tarafından yürütülen ağ sızma tespit sistemi geliştirme çalışmalarına yapacağı katkı ile ilgili görüşler tatbikat organizatörleri ile paylaşılmış, ilk adım olarak bu talebin bir öneri olarak sunumuna yönelik çalışma yapılmıştır. Çalışmada, en yaygın kullanılan erişilebilir tasnif dışı verisetlerinin genel özellikleri, avantaj ve dezavantajları, kullanılabilir oldukları atak tipleri, tasnif dışı Locked Shields verisetinin muhtemel özellikleri, bu verisetinin üretilmesi için gerekli sunucu, ağ ve istemci bileşenleri, elde edilecek verinin işlenmesi ve veriseti üretimi için uygulanması gereken metodoloji, üretilmesi muhtemel verisetinin avantaj ve dezavantajları ele alınmıştır. Yapılan çalışmada ayrıca bahse konu verisetinin üretilmesine yönelik bir yöntem önerilmiştir. Locked Shields tatbikat altyapısından üretilen bir verisetinin en önemli sorunu gizlilik endişesidir. Tatbikata katılan takımlar, geçerli güvenlik gerekçeleri nedeniyle verilerinin paylaşılmasını istememektedir. Bu nedenle tatbikat altyapısına, takımlardan bağımsız bir takım eklenmesi gerektiği değerlendirilmiş ve bu doğrultuda çalışmalara başlanmıştır. Kurulan çalışma grubu tarafından, tatbikat altyapısına sanal mavi takım eklemek amacıyla AICA (Autonomous Intelligent Cybersecurity Agents) çerçevesinin kurulumuna yönelik planlama yapılmış, çerçeveye ait tüm bileşenler tatbikat test faaliyeti esnasında aktif hale getirilmiş ve sanal mavi takım ağ trafiği tatbikat süresince takip edilerek yakalanmıştır. Yakalanan ağ trafiği üzerinde gerekli işlemler yapılarak neticede yaklaşık 2 milyonu atak olmak üzere toplam 16 milyon kayıt içeren güncel, kaliteli ve gizlilik derecesi içermeyen bir veriseti üretilmiştir. Üretilen verisetinin yayınlanmasına müteakip IDS geliştirme çalışmalarına ve literatüre önemli katkısının olacağı değerlendirilmektedir.

6.2. Tartışma ve Öneriler

Tez çalışması kapsamında ağ sızma tespit problemi model ve veriseti açılarından ele alınmıştır. Bu kapsamda en etkin derin öğrenme metotlarından biri olan LSTM seçilmiş, bununla birlikte RNN ve GRU metotları ile modeller de geliştirilmiştir. Çalışma kapsamında birçok diğer model incelenmiş, bunlardan bazılarının kurulumu yapılarak NSL-KDD veriseti ile testleri yapılmıştır. Ancak elde edilen sonuçlar tez kapsamına dahil edilebilecek seviyede bulunmamıştır. Tez çalışması kapsamında ağ sızma tespiti konusunda yapılan tüm güncel yayınlar incelenmiş, önemli olarak değerlendirilenleri referans olarak tez kapsamına dahil ediliştir.

Model açısından bakıldığında, LSTM yönteminin sağladığı performans göz önünde bulundurulduğunda ağ sızma tespit sistemlerinde kullanılabilir bir metot olduğu değerlendirilmiştir. Problemin çözümü için farklı yöntemlerin kullanılabileceği bilinmektedir. Nitekim derin öğrenme yöntemlerinin görüntü işleme alanındaki başarısının ağ sızma tespitinde değerlendirilmesi amacıyla CNN tabanlı model geliştirme ve nümerik verinin görsel verisetine dönüştürülmesine yönelik çalışmalar da gerçekleştirilmiştir. Bu problem özelinde sadece derin öğrenme yöntemlerinin değil, diğer yapay zekâ ve makine öğrenme yöntemlerinin de incelenmesinin faydalı olacağı düşünülmektedir. Model kurulumunda katman sayısı, hücre sayısı, optimizasyon algoritması, aktivasyon fonksiyonu gibi hiperparametrelerin oldukça önemli olduğu, bunlara atanacak değerlerin model performansını direk etkilediği tespit edilmiştir. Bu nedenle, model kurulum aşaması en fazla zaman alan, birçok test işleminin sabırla gerçekleştirilmesi gereken aşama olarak gözlemlenmiştir. Bu süreçte çeşitli tekniklerle hiperparametre optimizasyonunun gerçekleştirilmesi gerekmektedir. Kesin optimum hiperparametre değerlerinin elde edilmesi oldukça güçtür ve bu hususta çalışmalar devamlı bir faaliyet olarak yürütülmelidir. Bu parametrelerin belirlenmesinde, veriseti özellikleri göz ardı edilmemeli, veriye ilişkin çeşitli istatistiksel analizler yapılmalı ve bu analiz sonuçları göz önünde bulundurulmalıdır. Yapılan test işlemlerinin uzun sürmesi nedeniyle bu süreçte yüksek kapasiteli bilgisayar kullanımının zaman kaybının önlenmesi açısından önemli olduğu değerlendirilmektedir.

İlk model kurulumundan sonra platform ve metotlara hakim olunmakta, birçok farklı metot rahatlıkla inşa edilebilmektedir. Her problemin kendine özgü bir doğası olduğu ve belirli bir metodun belirli bir problemin çözümünde daha etkin olabileceği göz önünde bulundurularak, farklı modellerin kurulumuna devam edilmeli ve elde edilen test sonuçları not edilmelidir. Model performansının arttırılması için birçok düzenlileştirme işlemi kullanılabilmektedir. Düzenlileştirme işlemlerinin matematiksel temelleri incelenmeli, model performansına nasıl katkı sağlayacabilecekleri analiz edilmelidir. Düzenlileştirme işlemlerinin hiperparametre değerleri ile ilişkileri incelenmeli, aralarında bir uyumluluk oluşturulmalıdır. Bu işlem de yine devamlı faaliyetler kapsamında sürekli yapılacak testler ve analizler ile gerçekleştirilebilecektir.

Veriseti ile ilgili çalışmalar kapsamında, veriseti ön işlemlerinin oldukça uzun bir zaman aldığı gözlemlenmiştir. Nümerikleştirme işlemleri neticesinde mevcut veriseti boyutu artmakta, alanların doğru bir şekilde gösterildiğinden emin olunması gerekmektedir.

Verisetindeki alanlardan herhangi birinin eksik veya yanlış tanımlanması tüm sonuca etki etmekte, çalışmanın başka bir yöne gitmesine neden olmaktadır. Bu nedenle veriseti ön işlemlerinin oldukça titiz bir şekilde yapılması gerekmektedir. Özellik seçim algoritmalarının uygulanmaları neticesinde elde edilen yeni alanları içeren verisetlerinin kullanımında da aynı özen gösterilmelidir. Bununla birlikte, farklı özellik seçim algoritmaları kullanılarak yeni özellik kümeleri elde edilebileceği unutulmamalı devamlı faaliyetler kapsamına dahil edilmelidir.

Tez çalışması kapsamında, özgün veriseti üretimine ilişkin birçok çalışma incelenmiş, elde edilen bilgiler teze yansıtılmıştır. Kaliteli bir verisetinin sahip olması gereken özelliklere ilişkin yapılan inceleme sonuçları değerlendirilerek tez kapsamında özgün veriseti üretim çalışmaları esnasında yol gösterici olarak göz önünde bulundurulmuştur. Özgün veriseti üretimindeki ilk önemli problem verisetinin yayınlanabilirlik sorunudur. Oldukça kaliteli veriseti üretimine imkan sunabilecek birçok altyapı, haklı gizlilik gerekçeleri nedeniyle kullanılamamaktadır.

Tez çalışması kapsamında özgün veriseti üretimine yönelik öncelikle sanal makinelerden oluşan bir altyapı üzerinde, sentetik ağ trafiği ve kısıtlı sayıda saldırı tipi ile ağ trafiğinin yakalanması ve elde edilecek PCAP dosyası işlenerek veriseti üretimi hedeflenmiştir. Ancak yapılan inceleme neticesinde, literatürde bu tarz çalışmalar olduğu, üretilecek verisetinin akademik ve endüstri tarafından kullanılabilecek seviyede içerik ve özgünlüğe sahip olmayacağı, bu nedenle literatüre katkı sağlayabilecek bir çalışmanın yapılması gerektiği belirlenmiştir. Bu kapsamda siber savunma tatbikatlarına yönelik inceleme gerçekleştirilmiş, neticede siber savunma alanında icra edilen en önemli tatbikatlardan olan Locked Shields tatbikatı ağ trafiğinin, özgün ve kaliteli veriseti üretimi için mükemmel bir altyapı sağladığı belirlenmiştir. Locked Shields tatbikatı ağ trafiğinden yararlanılarak veriseti üretimi konusunda bugüne kadar sadece bir çalışma gerçekleştirildiği tespit edilmiş, çalışmayı gerçekleştiren ekip ile yapılan görüşme neticesinde, üretilen verisetinin gizlilik dereceli olduğu ve paylaşımının mümkün olmadığı bilgisi alınmıştır. Bu çalışma neticesinde yapılan yayına ilişkin detaylı bilgi teze yansıtılmıştır. Bu noktada, özgün veriseti ile ilgili yukarıda bahsedilen gizlilik sorununun Locked Shields tatbikatı için de geçerli olduğu tespit edilmiştir.

Ham PCAP dosyasından veriseti üretilebilirliğinin gösterilmesi amacıyla, Locked Shields 2021 tatbikatı neticesinde elde edilen örnek PCAP dosyası alınmış, işlenen dosya üzerinden düşük boyutlu bir veriseti üretilmiştir. Verisetine ilişkin temel bilgiler teze yansıtılmıştır. Tez kapsamında üretilen verisetinin gizlilik derecesine sahip olması, Locked Shields gibi oldukça geniş bir altyapıya sahip büyük bir tatbikatın ağ trafiğinden yararlanılarak tasnif dışı bir veriseti üretimi yönünde çalışma yapılması gerektiğini göstermiştir. Locked Shields tatbikatına birçok ülke tarafından katılım sağlanmakta, üretilen ağ trafiğinin temelde ülkelere ait olduğu kabul edilmektedir. Bu nedenle mevcut altyapı ile ağ trafiğinden veriseti elde etmenin mümkün olmadığı belirlenmiştir. Tasnif dışı veriseti üretimi için uygulanması gereken çözüm, tasnif dışı ağ trafiği üretebilecek bağımsız bir takım kurulması olarak tespit edilmiştir. Bu kapsamda sanal mavi takım kurulum çalışması gerçekleştirilmiştir. Sanal mavi takım AICA bileşenleri kullanılarak hazırlanmıştır. AICA nispeten yeni bir mimaridir ve bu mimarinin alt bileşenleri halen geliştirmeye açıktır. Gelecekte sanal siber güvenlik bileşenlerinin çok daha fazla yaygınlaşacağı ve zamanla insan faktörünün ortadan kalkacağı göz önünde bulundurulduğunda, AICA ve benzeri platformların geliştirilmesinin önemli olacağı değerlendirilmektedir.

Model performansının değerlendirilmesi için güncel verisetleri üzerinde uygulama yapılmasının faydalı olacağı değerlendirilerek çeşitli verisetleri incelenmiştir. Bu verisetlerinin bir bölümünün (CICIDS 2017, UGR'16) oldukça büyük boyutlu olduğu görülmüştür. Bu nedenle, daha düşük boyutlu UNSW-NB15 veriseti kullanılmıştır. Sonraki çalışmalarda, veriseti işleme amacıyla daha kapasiteli bilgisayarlar ile diğer güncel verisetlerinin de test amacıyla kullanılabileceği değerlendirilmektedir. Ancak veriseti seçiminde verisetlerinin hangi atak tiplerini içerdiklerinin incelenmesinin, arzu edilen özellikte ağ sızma tespit sitemi geliştirmeye katkı sağlayacağı değerlendirilmektedir. Bu doğrultuda tez çalışması kapsamında sunulan, "verisetleri ve içerdikleri atak tipleri" ile ilgili tablonun incelenmesi faydalı olacaktır.

Önümüzdeki dönemde ağ sızma tespit sistemlerine verilen önemin azalmadan devam edeceği göz önünde bulundurularak, yukarıda sıralanan bulgu ve öneriler doğrultusunda, model geliştirme ve özgün/kaliteli veriseti üretim çalışmalarına devam edilmesi gerektiği değerlendirilmektedir.

KAYNAKLAR

1. Leiner, B. M., Cerf V. G., Clark D. D., Kahn, R. E., Kleinrock, L., Lynch D. C., Postel J., Roberts, L. G. and Wolf, S. S. (1997). The Past and Future History of the Internet. *Communications Of The ACM*, 40 (2), 102-108.

2. Internet: Societies Online Trust Alliance. *Cyber Incident & Breach Trends Report*. URL: https://www.internetsociety.org/resources/ota/2019/2018-cyber-incident-breach-trends-report/. Son Erişim Tarihi: 21.05.2019

3. İnternet: Yoosofan, A, Ghajar F. G., Moghadasian, M., Babaee, R. (2019). A comparative study of using several artificial intelligence algorithims on intrusion detection system. *University of Kashan*, URL: https://www.researchgate.net/publication/332071797_A_comparative_study_of_using_several_artificial_intelligence_algorithims_on_intrusion_detection_system. Son Erişim Tarihi: 12.09.2020.

4. Pervez, M. S. and Farid, D. M. (18-20 Aralık 2014). *Feature Selection and Intrusion classification in NSL-KDD Cup 99 Dataset Employing SVMs.* 8. International Conference on Software, Knowledge, Information Management and Applications konferansında sunuldu, Dhaka/Bangladeş.

5. Liao, H. J., Lin C. H. R., Lin Y .C. and Tung K.Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications* 36, 16-23.

6. Gupta, A., Pandey, O. J., Shukla, M., Dadhich, A., Mathur and S., Ingle, A. (26-28 Aralık 2013). *Computational Intelligence based Intrusion Detection Systems for Wireless Communication and Pervasive Computing Networks*. IEEE International Conference on Computational Intelligence and Computing Research konferansında sunuldu, Madurai, Hindistan.

7. Bhuyan, M. H., Bhattacharyya D. K. and Kalita, J. K. (2014). Network Anomaly Detection: Methods, Systems and Tools. *IEEE Communications Surveys & Tutorials*, 16, 1-5.

8. Thing, V.L.L. (19-22 Mart 2017*). IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach*. IEEE Conference on Wireless Communications and Networking konferansında sunuldu, San Fransisko/ABD.

9. Kim, J., Shin, N., Jo S. Y. and Kim S. H. (13-16 Şubat 2017). *Method of Intrusion Detection using Deep Neural Network.* IEEE International Conference on Big Data and Smart Computing (BigComp) konferansında sunuldu, Jeju Adası/Güney Kore.

10. Zhang, J., Qin, G., Cui, Y., Dong, J. and Guo, L. (18-20 Ekim 2017). *SVM-FastICA Based Detection Ensemble System of EEG*. International Conference on Convergence Information Technology konferansında sunuldu, Jeju Adası/Güney Kore.

11. Yuan, J., Li, H., Ding, S. and Cao, L. (2-4 Nisan 2010). *Intrusion Detection Model based on Improved Support Vector Machine.* 3. International Symposium on Intelligent Information Technology and Security Informatics sempozyumunda sunuldu, Jian/Çin.

12. Jiang, J., Li, R., Zheng, T., Su, F. and Li, H. (18-20 Nisan 2011*). A new intrusion detection system using Class and Sample Weighted C-Support Vector Machine*. 3. International Conference on Communications and Mobile Computing konferansında sunuldu, Qingdao/Çin.

13. Zheng, K., Qian, X. and An, N. (23-24 Ekim 2010). *Supervised Non-Linear Dimensionality Reduction Techniques for Classification in Intrusion Detection*. International Conference on Artificial Intelligence and Computational Intelligence konferansında sunuldu, Sanya/Çin.

14. Townsend, K. R., Sun, S., Johnson, T., Attia, O. G., Jones, P.H. and Zambreno, J. (21-23 Mayıs 2015). *k-NN Text Classification using an FPGA-Based Sparse Matrix Vector Multiplication Accelerator*. IEEE International Conference on Electro-Information Technology konferansında sunuldu, Dekalb/ABD.

15. Yang, J., Chen, X., Xiang, X. and Wan, J. (12-14 Nisan 2010). *HIDS-DT: An Effective Hybrid Intrusion Detection System Based on Decision Tree.* International Conference on Communications and Mobile Computing konferansında sunuldu, Shenzhen/Çin.

16. Thaseen, S. and Kumar, C.A. (21-22 Şubat 2013*). An Analysis of Supervised Tree Based Classifiers for Intrusion Detection System*. International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME) konferansında sunuldu, Tamilnadu/Hindistan.

17. Relan, N. and Patil, D.R. (9-10 Ocak 2015). *Implementation of Network Intrusion Detection System using Variant of Decision Tree Algorithm.* International Conference on Nascent Technologies in the Engineering Field (ICNTE) konferansında sunuldu, Navi Mumbai/Hindistan.

18. Khor K. C., Ting C. Y. and Amnuaisuk S. P. (13-14 Mart 2020). *Comparing Single and Multiple Bayesian Classifiers Approaches for Network Intrusion Detection*. Second International Conference on Computer Engineering and Applications konferansında sunuldu, Gunupur/Hindistan.

19. Han, X., Xu, L., Ren, M. and Gu, W. (13-15 Kasım 2015). *A Naive Bayesian network intrusion detection algorithm based on Principal Component Analysis*. 7. International Conference on Information Technology in Medicine and Education, konferansında sunuldu, Huangshan/Çin.

20. Klassen, M. and Yang, N. (18-20 Ekim 2012*). Anomaly Based Intrusion Detection in Wireless Networks Using Bayesian Classifier*. IEEE fifth International Conference on Advanced Computational Intelligence (ICACI) konferansında sunuldu, Nanjing/Çin.

21. McCulloch, W.S. and Pitts, W. (1943). A Logical Calculus of The Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5, 115-133.

22. Akcayol M. A. (2017). *Zeki Optimizasyon Teknikleri Ders Notları*. Gazi Üniversitesi Bilgisayar Mühendisliği, Ankara.

23. Internet: Yapay sinir ağları tanıtımı. URL: http://kod5.org/yapay-sinir-aglari-ysa-nedir/ Son Erişim Tarihi: 15.10.2020.

24. Internet: Tekil yapay sinir ağı. URL: https://upload.wikimedia.org/wikipedia/commons/9/96/SingleLayerNeuralNetwork.png. Son Erişim Tarihi: 12.11.2020.

25. Kalkan, S. (2016), *Derin Öğrenme Ders Notları.* Ortadoğu Teknik Üniversitesi Bilgisayar Mühendisliği, Ankara.

26. Le Cun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning. *Nature*, 521, 436– 444.

27. Internet: Wang, H. and Raj, B. (2015). A Survey: Time Travel in Deep Learning Space: An Introduction to Deep Learning Models and How Deep Learning Models Evolved from the Initial Ideas. URL: https://arxiv.org/pdf/1510.04781v2.pdf. Son Erişim Tarihi: 15.11.2020.

28. Goodfellow, I , Bengio, Y. and Courville, A. (2015). *Deep Learning.* Cambridge/ABD: MIT Press, 213-290.

29. Chollet, F. (2017*). Deep Learning with Python.* New York/ABD: Manning Press, 21-132.

30. Batı, E. (2014). *Deep Convolutional Neural Networks With An Applicatıon Towards Geospatial Object Recognition*. Yüksek Lisans Tezi, Ortadoğu Teknik Üniversitesi Elektrik Elektronik Mühendisliği Bölümü, Ankara.

31. Taşhan, B. (2017). *Road Lane Detection System With Convolutional Neural Network.* Yüksek Lisans Tezi, Bahçeşehir Üniversitesi Bilgisayar Mühendisliği Bölümü, İstanbul.

32. Karagöz, B. B. (2015). *On The Analysis Of Deep Convolutional Neural Networks Applied To Building Detection In Satellite Images*. Yüksek Lisans Tezi, Ortadoğu Teknik Üniversitesi Elektrik Elektronik Mühendisliği Bölümü, Ankara.

33. Ergün H. (2016*). Video Concept Classificatıon And Retrieval*. Yüksek Lisans Tezi, Başkent Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

34. Mahsereci Karabulut, E. (2016). *Investigation Of Deep Learning Approaches For Biomedical Data Classification*. Doktora Tezi, Çukurova Üniversitesi Elektrik Elektronik Mühendisliği Bölümü, Adana.

35. Geçer, B. (2016). *Detection And Classification Of Breast Cancer In Whole Slide Histopathology Images Using Deep Convolutional Networks*. Yüksek Lisans Tezi, Bilkent Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

36. Öner, M. Ü. (2016). *Metastasis Detection And Localization In Lypmh Nodes By Using Convolutional Neural Networks.* Yüksek Lisans Tezi, Ortadoğu Teknik Üniversitesi Elektrik Elektronik Mühendisliği Bölümü, Ankara.

37. Velioğlu, B. (2016*). Multi-Subject Brain Decoding Using Deep Learning Techniques*. Yüksek Lisans Tezi, Ortadoğu Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

38. Bak, E. Ç. (2016). *Deep Learning Based Saliency Prediction In Videos*. Yüksek Lisans Tezi, Hacettepe Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

39. Bilgiç, A. (2017*). Derin Öğrenmeye Dayalı Yüz ve Vücut Biyometrilerinin Tümleştirilmesi*. Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği Bölümü, İstanbul.

40. Hacıoğlu, C. (2014). *Derinlikli Öğrenme Kullanılarak Konuşmadan Uykululuk / Uykusuzluk Tespiti*. Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Elektronik ve Haberleşme Mühendisliği Bölümü, İstanbul.

41. Aygün, R. C. (2017). *Derin Öğrenme Yöntemleri İle Bilgisayar Ağlarında Güvenliğe Yönelik Anormallik Tespiti*. Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü, İstanbul.

42. Elmasry, W. (2019). *Derin Öğrenme Yaklaşımı Kullanarak Bulut Ortamları İçin Saldırı Tespit Hizmet Tasarımı.* Doktora Tezi, İstanbul Ticaret Üniversitesi Bilgisayar Mühendisliği Bölümü, İstanbul.

43. Faker O. M. F. (2019). *Intrusion Detection Using Big Data And Deep Learning Techniques*. Yüksek Lisans Tezi, Çankaya Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

44. Çakır, B. (2019). *Zero-Day Attack Detection With Deep Learning.* Yüksek Lisans Tezi, Ortadoğu Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü, Ankara.

45. Zhao, G., Zhang, C. and Zheng, L. (21-24 Temmuz 2017*). Intrusion Detection using Deep Belief Network and Probabilistic Neural Network*. IEEE International Conference on Computational Science and Engineering (CSE) konferansında sunuldu, Guangzhou, Çin.

46. Van N. T., Thinh T. N. and Sach L. T. (21-23 Temmuz 2017). *An anomaly-based Network Intrusion Detection System using Deep learning*. International Conference on System Science and Engineering (ICSSE) konferansında sunuldu, Ho Chi Minh City/Vietnam.

47. Jing L. and Bin W. (17-18 Aralık 2016*). Network Intrusion Detection Method Based on Relevance Deep Learning.* International Conference on Intelligent Transportation, Big Data & Smart City konferansında sunuldu, Xiamen/Çin.

48. Yin C., Zhu Y., Fei J. and He X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access* 5, 21954 - 21961.

49. Seo, S., Park, S. and Kim, J. (23-25 Aralık 2016). *Improvement of Network Intrusion Detection Accuracy by using Restricted Boltzmann Machine*. 8. International Conference on Computational Intelligence and Communication Networks konferansında sunuldu, Tehri/Hindistan.

50. Jiang, F., Fu, Y., Gupta, B. B., Liang, Y., Rho, S., Lou, F., Meng, F. and Tian, Z. (2018). Deep Learning Based Multi-Channel Intelligent Attack Detection For Data Security. *IEEE Transactions on Sustainable Computing*, 5 (2), 204-212.

51. Shone, N., Ngoc, T. N., Phai, V. D. and Shi, Q. (2018). A Deep Learning Approach to Network Intrusion Detection. *IEEE Transactions On Emerging Topics In Computational Intelligence*, 2 (6), 41-50.

52. Wang, X. and Wang, L. (9-10 Aralık 2017). *Research on Intrusion Detection Based on Feature Extraction of Autoencoder and The Improved K-means Algorithm.* 10th International Symposium on Computational Intelligence and Design sempozyumunda sunuldu, Hangzhou/Çin.

53. Karataş, G. and Şahingöz, Ö. K. (22-25 Mart 2018*). Neural Network Based Intrusion Detection Systems with Different Training Functions.* 6. International Symposium on Digital Forensic and Security (ISDFS) konferansında sunuldu, Antalya.

54. Kim, J., Kim, J., Thu, H. L. T., and Kim, H. (15-17 Şubat 2016). *Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection*. International Conference on Platform Technology and Service (PlatCon) konferansında sunuldu, Jeju Adası/Güney Kore.

55. Le T. T. H., Kim J. and Kim H. (13-15 Şubat 2017). *An Effective Intrusion Detection Classifier Using Long Short-Term Memory with Gradient Descent Optimization*. International Conference on Platform Technology and Service konferansında sunuldu, Busan/Güney Kore.

56. Maimo L. F., Gomez, A. L. P., Clemente, F. J. G, Perez, M. G. and Perez, G. M. (2018). A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks. *IEEE Access*, 6, 7700-7712.

57. Farahnakian, F. and Heikkonen, J. (11-14 Şubat 2018). *A Deep Auto-Encoder based Approach for Intrusion Detection System.* International Conference on Advanced Communications Technology (ICACT) konferansında sunuldu, Chuncheon/Güney Kore.

58. Alrawashdeh, K. and Purdy, C. (23-25 Mart 2018). *Fast Hardware Assisted Online Learning Using Unsupervised Deep Learning Structure for Anomaly Detection*. International Conference on Information and Computer Technologies konferansında sunuldu, Dekalb/ABD.

59. Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H. and Wang, C. (2018*).* Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access*, 6, 35365-35381.

60. Zhang, X. Y., Yin, F., Zhang, Y. M., Liu, C. L. and Bengio, Y. (2018). Drawing and Recognizing Chinese Characters with Recurrent Neural Network. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 40 (4), 849-862.

61. Lotfidereshgi, R. and Gournay, P. (15-20 Nisan 2018). *Speech Prediction Using An Adaptive Recurrent Neural Network With Application To Packet Loss Concealment.* International Conference on Acoustics, Speech and Signal Processing konferansında sunuldu, Calgary/Kanada.

62. Xiao, A., Liu, J., Li, Y., Song, Q. and Ge, N. (2018). Two-Phase Rate Adaptation Strategy for Improving Real-Time Video QoE in Mobile Networks. *China Communications,* 5 (10), 12-24.

63. İnternet: LSTM genel yapısı ile ilgili bilgilendirme (2019). URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/, Son Erişim Tarihi: 21.06.2021.

64. Hochreiter, S. and Schmidhuber, J. (1997), Long Short-Term Memory. *Neural Computation*, 9 (8), 1735-1780.

65. İnternet: Silver D. L. and Frey C. (Nisan 2017*). Recurrent Neural Networks*, URL: https://slideplayer.com/slide/14885464/ Son Erişim Tarihi: 22.09.2021.

66. Wang, Z., Liu, K., Li, J., Zhu, Y. and Zhang, Y. (2019), Various Frameworks and Libraries of Machine Learning and Deep Learning: A Survey, *Archives of Computational Methods in Engineering*, 8 (2), 34-43.

67. Nguyen, G., Dlugolinsky, S., Bobak, M., Tran, V., Garcia, A. L., Heredia, I., Malik P. and Hluch, L. (2019*).* Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review* 52 (1), 77–124.

68. Deuschle, V., Alexandrov, A., Januschowski, T. and Markl, V. (26 Ağustos 2020), *End-to-End Benchmarking of Deep Learning Platforms*, 11. TPC Technology Konferansında sunuldu. Los Angeles/ABD.

69. Shi, S., Wang, Q., Xu, P. and Chu, X. (16-18 Kasım 2016*). Benchmarking State-of-the-Art Deep Learning Software Tools*. 7. International Conference on Cloud Computing and Big Data konferansında sunuldu, Macau/Çin.

70. İnternet: Guo, Q., Xie, X., Ma, Lei and Hu, Q. (2018), An Orchestrated Empirical Study on Deep Learning Frameworks and Platforms, URL: https://arxiv.org/pdf/1811.05187v1.pdf Son Erişim Tarihi: 11.10.2021.

71. İnternet: Bahrampour S., Ramakrishnan N., Schott L. and Shah M. (2016), Comparative Study of Deep Learning Software Frameworks, URL: https://arxiv.org/pdf/1511.06435v3.pdf Son Erişim Tarihi: 15.10.2021.

72. Tavallaee, M., Bagheri, E and Ghorbani, A. A. (8-10 Temmuz 2009), *A Detailed Analysis of the KDD CUP 99 Data Set*, IEEE Symposium on Computational Intelligencein Security and Defense Applications sempozyumunda sunuldu, Ottawa/Kanada.

73. Lopez-Martin, M., Carro, B. and Sanchez-Esguevillas, A. (2020), Application of deep reinforcement learning to intrusion detection for supervised problems. *Expert Systems With Applications* 141, 112963.

74. Kasongo, S. M. and Sun, Y. (2020), A Deep Long Short-Term Memory based classifier forWireless Intrusion Detection System, *ICT Express*, 6 (2), 98-103.

75. Zhang, C., Ruan, F., Yin, L., Chen, X., Zhai, L. and Liu, F. (25-27 Ekim 2019), *A Deep Learning Approach for Network Intrusion Detection Based on NSL-KDD Dataset.* IEEE 13. International Conference on Anti-counterfeiting, Security, and Identification (ASID) konferansında sunuldu, Xiamen/Çin.

76. Su, T., Sun, H., Zhu, J., Wang, S. and Li, Y. (2020), BAT: Deep Learning Methods on Network Intrusion Detection Using NSL-KDD Dataset. *IEEE Access*, 8, 29575-29585.

77. Gurung, S., Ghose M. K. and Subedi A. (2019), Deep Learning Approach on Network Intrusion Detection System using NSL-KDD Dataset. *I. J. Computer Network and Information Security*, 3, 8-14.

78. Alazab, A., Hobbs, M., Abawajy, J. and Alazab, M. (2-5 Ekim 2012), *Using Feature selection for intrusion detection system*, International Symposium on Communications and Information Technologies sempozyumunda sunuldu, Gold Coast/Avustralya.

79. İnternet: Çarkacı, N. (2018), Sık kullanılan Hiper-parametreler, URL: https://medium.com/, Son Erişim Tarihi: 27.10.2021.

80. Wilson, A. C., Roelofs, R., Stern, M., Srebro, N. and Recht, B. (4-9 Aralık 2017), *The Marginal Value of Adaptive Gradient Methods in Machine Learning*, 31st Conference on Neural Information Processing Systems konferansında sunuldu, Long Beach/ABD.

81. İnternet: Brownlee, J. (2020), Understand the Impact of Learning Rate on Neural Network Performance. URL: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/ Son Erişim Tarihi: 03.02.2022

82. İnternet: Keras layer activation functions. URL: https://keras.io/api/layers/activations/ Son Erişim Tarihi: 10.03.2022

83. Ikram, S. T. and Cherukuri A. K. (2017). Intrusion detection model using fusion of chi-square feature selection and multi class SVM. *Journal of King Saud University – Computer and Information Sciences*, 29 (4), 462–472.

84. School, M. N. A., Rahardjo, B. and Bambang, R. T. (24-27 Kasım 2014). *Improving Performance of Network Scanning Detection Through PCA-Based Feature Selection.* International Conference on Information Technology Systems and Innovation (ICITSI) konferansında sunuldu, Bandung/Endonezya.

85. Biswas, N. A., Shah, F. M., Tammi, W. M. and Chakraborty, S. (21-23 Aralık 2015). *FP-ANK: An Improvised Intrusion Detection Systemwith Hybridization of Neural Network and K-Means Clustering over Feature Selection by PCA.* 18. International Conference on Computer and Information Technology (ICCIT) konferansında sunuldu, Dhaka/Bangladeş.

86. Ikram, S. T. and Cherukuri, A. K. (2016). Improving Accuracy of Intrusion Detection Model Using PCA and Optimized SVM. *Journal of Computing and Information Technology*, 24 (2), 133–148.

87. Kang, S. and Kim K. J. (2016), A feature selection approach to find optimal feature subsets for the network intrusion detection system. *Cluster Computing*, 19 (1), 325–333.

88. Yue, W., Yiming, J. and Julong, L. (23-25 Ekim 2020), *A Fast Deep Learning Method for Network Intrusion Detection Without Manual Feature Extraction*, 2. International Conference on Electronics and Communication, Network and Computer Technology (ECNCT) konferansında sunuldu, Chengdu/Çin.

89. Su, T., Sun, H., Zhu, J., Wang, S. and Li, Y. (2020), BAT: Deep Learning Methods on Network Intrusion Detection Using NSL-KDD Dataset, *IEEE Access*, 8, 29575 - 29585.

90. Ding, Y. and Zhai, Y. (8-10 Aralık 2018), *Intrusion Detection System for NSL-KDD Dataset Using Convolutional Neural Networks*, 2. International Conference on Computer Science and Artificial Intelligence konferansında sunuldu, Shenzhen/Çin.

91. Li, Z., Batta, P. and Trajkovic, L. (7-10 Ekim 2018), *Comparison of Machine Learning Algorithms for Detection of Network Intrusions*, IEEE International Conference on Systems, Man and Cybernetics konferansında sunuldu, Miyazaki/Japonya.

92. Li, Z., Rios, A. L. G., Xu, G., and Trajkovic, L. (26-29 Mayıs 2019), *Machine Learning Techniques for Classifying Network Anomalies and Intrusions*, IEEE International Symposium on Circuits and Systems (ISCAS) sempozyumunda sunuldu, Sapporo/Japonya.

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

94. Mirza, A. H. and Coşan, S. (2-5 Mayıs 2018), *Computer Network Intrusion Detection Using Sequential LSTM Neural Networks Autoencoders*. 26. Signal Processing and Communications Applications Conference (SIU) konferansında sunuldu, İzmir/Türkiye.

95. Fu Y., Lou, F., Meng, F., Tian, Z., Zhang, H. and Ji, F. (18-21 Haziran 2018), *An Intelligent Network Attack Detection Method based on RNN*, 3. International Conference on Data Science in Cyberspace konferansında sunuldu, Guangzhou/Çin.

96. Xu, C., Shen, J., Du, X. and Zhang, F. (2018), An Intrusion Detection System Using a Deep Neural Network With Gated Recurrent Units. *IEEE Access*, 6, 48697-48707.

97. Shaikh R. A. and Shashikala, S. V. (25-27 Temmuz 2019), *An Autoencoder and LSTM based Intrusion Detection approach against Denial of service attacks*. 1. International Conference on Advances in Information Technology konferansında sunuldu, Chikmagalur/Hindistan.

98. Boukhalfa, A., Abdellaoui, A., Hmina, N. and Chaoui, H. (2020), LSTM deep learning method for network intrusion detection system, *International Journal of Electrical and Computer Engineering (IJECE)*, 10 (3), 3327~3334.

99. Muhuri, P. S., Chatterjee, P., Yuan, X., Roy, K. and Esterline, A. (2020), Using a Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) to Classify Network Attacks. *Information*, 11 (5), 243.

100. P.S. Muhuri (2019), *Using Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) to Classify Network Attacks*. Yüksek Lisand Tezi, A&T State Universitesi, Kuzey Karolina/ABD.

101. İnternet: Zhu, S. and Chollet, F. (2020), Working with RNNs. URL: https://www.tensorflow.org/guide/keras/rnn, Son Erişim Tarihi: 12.05.2022.

102. Elman, J. L. (1990), Finding Structure In Time. *Cognitive Science*, 14 (2), 179-211.

103. Cho, K., Merrienboer, B. v., Gülçehre, Ç., Bahdanau, D., Bougares, F., Shwenk, H. and Bengio Y. (25-29 Ekim 2014), *Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation*, Conference on Empirical Methods in Natural Language Processing (EMNLP) konferansında sunuldu, Doha/Katar.

104. Wu, K., Chen, Z. and Li, W. (2018), A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks. *IEEE Access*, 6, 50850 - 50859.

105. Jiang, K., Wang, W., Wang, A. and Wu, H. (2020), Network Intrusion Detection Combined Hybrid Sampling With Deep Hierarchical Network. *IEEE Access*, 8, 32464-32476.

106. Hu, Z., Wang, L., Qi, L., Li, Y. and Yang, W. (2020), A Novel Wireless Network Intrusion Detection Method Based on Adaptive Synthetic Sampling and an Improved Convolutional Neural Network. *IEEE Access*, 8, 195741-195751.

107. Lin, S. Z., Shi, Y. and Xue, Z. (8-13 Temmuz 2018), *Character-level Intrusion Detection Based onConvolutional Neural Networks*. International Joint Conference on Neural Networks (IJCNN) konferansında sunuldu, Rio de Janeiro/Brezilya.

108. Hsu, C. M., Azhari, M. Z., Hsieh, H. Y., Prakosa, S. W. and Leu, J. S. (2021), Robust Network Intrusion Detection Scheme Using Long-Short Term Memory Based Convolutional Neural Networks. *Mobile Networks and Applications*, 26, 1137–1144.

109. Mohammadpour, L., Ling, T. C., Liew, C. S. and Aryanfar, A. (2020), A mean convolutional layer for intrusion detection system. *Secure and Communication Networks*, 2020, 8891185.

110. Andresini, G., Appice, A., and Malerba, D. (2021), Nearest cluster-based intrusion detection through convolutional neural networks, *Knowledge-Based Systems*, 216, 106798.

111. Li, S. (28-29 Şubat 2020), *Network intrusion detection model based on improved convolutional neural network*. International Conference on Cyber Security Intelligence and Analytics konferansında sunuldu, Haikou/Çin.

112. Azizjon, M., Jumabek, A. and Kim, W. (19-21 Şubat 2020), *1D CNN based network intrusion detection with normalization on imbalanced data*. International Conference on Artificial Intelligence Information Community (ICAIIC) konferansında sunuldu, Fukuoka/Japonya.

113. Yu, L., Dong, J., Chen, L., Li, M., Xu, B., Li, Z., Qiao, L., Liu, L., Zhao, B. and Zhang, C. (2021), PBCNN: Packet bytes-based convolutional neural network for network intrusion detection. *Computational Network*, 194, 108117.

114. Ho, S., Jufout, S. A., Dajani, K. and Mozumdar, M. (2021), A novel intrusion detection model for detecting known and innovative cyberattacks using convolutional neural network. *IEEE Open Journal of the Computer Society*, 2, 14–25.

115. Wang, W., Zhu, M., Zeng, X., Ye, X. and Sheng, Y. (11-13 Ocak 2017), *Malware traffic classification using convolutional neural network for representation learning.* International Conference on Information Networks (ICOIN) konferansında sunuldu, Dan Nang/Vietnam.

116. Mendonca, R. V., Teodoro, A. A. M., Rosa, R. L., Saadi, M., Melgarejo, D. C., Nardelli, P. H. J. and Rodriguez, D. Z. (2021), Intrusion Detection system based on fast hierarchical deep convolutional neural network. *IEEE Access*, 9, 61024–61034.

117. McHugh, J. (2001). Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. *ACM Transactions on Information and System Security*, 3 (4), 262–294.

118. Brown, C., Cowperthwaite, A., Hijazi, A. and Somayaji, A. (8-10 Temmuz 2009), *Analysis of the 1999 DARPA/Lincoln Laboratory IDS Evaluation Data with NetADHICT*. IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA) sempozyumunda sunuldu, Ottawa/Kanada.

119. Shiravi, A., Shiravi, H., Tavallaee, M. and Ghorbani, A. A. (2012*).* Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers and Security*, 31, 357-374.

120. Ring, M., Wunderlich, S., Scheuring, D., Landes, D. and Hotho, A. (2019). A Survey of Network-based Intrusion Detection Data Sets. *Computers and Security*, 86, 147-167.

121. Brauckhoff, D., Wagner, A. and May, M. (28 Temmuz 2008). *FLAME: A Flow-Level Anomaly Modeling Engine.* Workshop on Cyber Security Experimentation and Test (CSET) çalıştayında sunuldu, San Jose, Kaliforniya/ABD.

122. Vasilomanolakis, E., Cordero, C. G., Milanov, N. and Muhlhauser, M. (25-29 Nisan 2016). *Towards the creation of synthetic, yet realistic, intrusion detection datasets*. IEEE Network Operations and Management Symposium (NOMS) sempozyumunda sunuldu, İstanbul.

123. Siska, P. (28 Haziran - 2 Temmuz 2010*). A Flow Trace Generator using Graph-based Traffic Classification Techniques*. International Wireless Communication and Mobile Computing Conference (IWCMC) konferansında sunuldu, Caen/Fransa.

124. M. Ring ve diğerleri (2019). Flow-based network traffic generation using Generative Adversial Networks. *Computer & Security*, 82, 156-172.

125. Erlacher, F. and Dressler, F. (20 Ağustos 2018). *How to Test an IDS? GENESIDS: An Automated System for Generating Attack Traffic.* Workshop on Traffic Measurements for Cybersecurity (WTMC) çalıştayında sunuldu, Budapeşte/Macaristan.

126. Brogi, G. and Tong, V. V. T. (17-19 Mayıs 2017). *Sharing and replaying attack scenarios with Moirai*. Rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI) konferansında sunuldu, Grenoble/Fransa.

127. Rajasinghe, N., Samarabandu, J. and Wang, X. (13-16 Mayıs 2018). *INSecS-DCS: A Highly Customizable Network Intrusion Dataset Creation Framework.* Canadian Conference on Electrical & Computer Engineering (CCECE) konferansında sunuldu, Quebec/Kanada.

128. Bhuyan, M. H., Bhattacharyya, D. K. and Kalita, J. K. (2015). Towards Generating Real-life Datasets for Network Intrusion Detection. *International Journal of Network Security*, 17 (6), 675-693.

129. Haidera, W., Hua, J., Slaya, J., Turnbulla, B. P. and Xieb, Y. (2017). Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. *Journal of Network and Computer Applications*, 87, 185–192.

130. Sharafaldin, I., Lashkari, A. H. and Ghorbani, A. A. (2018). *Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization*. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018), 108-116.

131. Känzig, N., Meier, R., Gambazzi, L., Lenders, V. and Vanbever, L. (28-31 Mayıs 2019), *Machine Learning-based Detection of C&C Channels with a Focus on the Locked Shields Cyber Defense Exercise*. 11. International Conference on Cyber Conflict (CyCon) konferansında sunuldu, Tallin/Estonya.

132. İnternet: Kott, A., Theron, P., Drasar, M., Dushku, E., LeBlanc, B., Losiewicz, P., Guarino, A., Mancini, L. V., Panico, A., Pihelgas, M. and Rzadca, K. (2019), Autonomous Intelligent Cyber-defense Agent (AICA) Reference Architecture. release 2.0, URL: https://apps.dtic.mil/sti/pdfs/AD1080471.pdf, Son Erişim Tarihi: 12.04.2023.

133. İnternet: Lashkari, A. H. (2018), Cicflowmeter-v4.0 Anomali tespiti için ağ trafik analiz aracı, URL: https://github.com/iscx/cicflowmeter, Son Erişim Tarihi: 17.04.2023.

134. Moustafa, N. and Slay, J. (10-12 Kasım 2015), *UNSW-NB15: A Comprehensive Data set for Network Intrusion Detection systems(UNSW-NB15 Network Data Set)*, Military Communications and Information Systems konferansında sunuldu, Kanberra/Avustralya.

Gazili olmak ayrıcalıktır