Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Araştırma Makalesi / Research Article

Web Uygulamalarına Yönelik Siber Saldırıların Tespitinde Derin Öğrenme Yöntemleri Kullanılarak Güvenlik Duvarı Uygulamasının Geliştirilmesi

The Development of a Firewall Application Using Deep Learning Methods for Detection of Cyber Attacks on Web Applications

Şengül Bayrak[1*], Alper Karaca[2], Ferhat Toson[3], Mehmet Emin Tayfur[4], Selçuk Yavaş[5]

[1] İstanbul Sabahattin Zaim Üniversitesi, Bilgisayar Mühendisliği Bölümü, bayraksengul@ieee.org
ORCID: https://orcid.org/0000-0002-4114-4305

[2] İstanbul Sabahattin Zaim Üniversitesi, Bilgisayar Mühendisliği Bölümü, alper.krc@outlook.com
ORCID: https://orcid.org/0009-0007-8167-4013

[3] İstanbul Sabahattin Zaim Üniversitesi, Bilgisayar Mühendisliği Bölümü, ferhattoson@hotmail.com
ORCID: https://orcid.org/0009-0009-1493-3702

[4] İstanbul Sabahattin Zaim Üniversitesi, Bilgisayar Mühendisliği Bölümü, mehmetemin.tayfur@outlook.com
ORCID: https://orcid.org/0009-0006-6647-223X

[5] İstanbul Sabahattin Zaim Üniversitesi, Bilgisayar Mühendisliği Bölümü, selcuk.yavas@outlook.com
ORCID: https://orcid.org/0009-0009-6649-6719

MAKALE BİLGİLERİ

Makale Geçmişi:

Geliş 17 Şubat 2025
Revizyon 22 Mart2025
Kabul 2 Mayıs 2025
Online 30 Haziran 2025

Anahtar Kelimeler:

Makine Öğrenmesi, Derin Öğrenme, Güvenlik Duvarı, Saldırı Tespiti, Siber Güvenlik, Web Uygulaması

ÖZ

Web uygulamalarının güvenliği, kullanıcı verilerinin korunması ve siber saldırılara karşı önlem alınması açısından kritik bir öneme sahiptir. İnternet kullanıcılarının hassas verilerini koruma, siber saldırılara karşı önlem alma ve kullanıcı deneyimini iyileştirme gibi amaçlarla güvenlik duvarı uygulamaları kullanılmaktadır. Geleneksel güvenlik duvarı yaklaşımları, belirli kurallar ve kalıplara dayanarak saldırıları tespit etmeye çalışsa da gelişen ve karmaşık hale gelen saldırı türlerine karşı yetersiz kalabilmektedir. Bu çalışmada, web uygulamalarına yönelik siber saldırıları tespit etmek için derin öğrenme tabanlı bir yaklaşım önerilmektedir. Önerilen yöntem, "FWAF" veri seti kullanılarak geliştirilmiş ve veri ön işleme, özellik çıkarımı ve veri ölçekleme aşamalarından geçirilmiştir. Altı farklı derin öğrenme modeli değerlendirilerek en yüksek doğruluk ve güvenilirliği sağlayan yöntem belirlenmiştir. Elde edilen sonuçlar, model performanslarını karşılaştırmalı olarak inceleyerek optimum yöntemin seçiminde önemli bir rol oynamıştır. Seçilen model, gerçek zamanlı çalışabilirliğini test etmek amacıyla Jetson Nano platformu üzerinde canlıya alınarak performansı değerlendirilmiştir. Deneysel sonuçlar, derin öğrenme tabanlı yaklaşımların geleneksel yöntemlere kıyasla daha yüksek doğruluk oranları sunduğunu ve yeni saldırı türlerine adapte olabildiğini göstermektedir. Bu çalışma, web uygulamalarına yönelik saldırıların tespitinde etkili ve dinamik bir güvenlik çözümü sunarak, siber güvenlik alanında önemli bir katkı sağlamaktadır.

ARTICLE INFO

Article history:

Received 17 February 2025
Received in revised form 22 March 2025
Accepted 2 May 2025
Available online 30 June 2025

Keywords:

Machine Learning, Deep Learning, Firewall, Intrusion Detection, Cybersecurity, Web Application

ABSTRACT

The security of web applications is critical for protecting user data and taking precautions against cyber-attacks. Firewall applications are used to protect Internet users' sensitive data, take precautions against cyber-attacks and improve the user experience. Although traditional firewall approaches try to detect attacks based on certain rules and patterns, they may be insufficient against evolving and complex attack types. In this paper, we propose a deep learning based approach to detect cyber attacks on web applications. The proposed method is developed using the "FWAF" dataset and goes through stages that are data preprocessing, feature extraction, and data scaling. Six different deep learning models are evaluated to determine the method with the highest accuracy and reliability. The results obtained played an important role in the selection of the optimum method by comparing the model performances. The selected model was put live on the Jetson Nano platform to test its real-time operability and its performance was evaluated. Experimental results show that deep learning-based approaches offer higher accuracy rates compared to traditional methods and can adapt to new types of attacks. This study makes an important contribution to the field of cyber security by providing an effective and dynamic security solution for detecting attacks against web applications.

DOI: 10.24012/dumf.1634525

* Sorumlu Yazar

Giriş

Web saldırıları, web sitelerine veya web uygulamalarına yapılan kötü niyetli girişimlerdir. Günümüzde web uygulamaları, bireylerden büyük ölçekli kuruluşlara kadar geniş bir kullanıcı kitlesi tarafından yaygın bir şekilde kullanılmaktadır. Ancak, bu yaygın kullanım, siber saldırılara karşı açık hedef haline gelmelerine de neden olmaktadır. Web tabanlı saldırılar, kullanıcıların hassas verilerini ele geçirmek, hizmetleri kesintiye uğratmak veya sistemlerin güvenliğini tehlikeye atmak amacıyla gerçekleştirilmektedir. Bilgi çalmak ve hizmet kesintisi yaratmak bu saldırıların temel amaçlarıdır. Yapısal Sorgu Dili (Structured Query Language – SQL) Enjeksiyonu, Siteler Arası Komut Dosyası Çalıştırma (Cross-Site Scripting - XSS), Dağıtılmış Hizmet Reddi Saldırısı (Distributed Denial of Service - DDoS), Brute Force Saldırıları, Phishing bu saldırılardan bazılarıdır. Bu saldırıları engellemek için Makine Öğrenimi ve Yapay Zeka, AI Destekli Bot Engelleme ve Web Uygulamaları Güvenlik Duvarı (Web Application Firewall - WAF) gibi yöntemler kullanılmaktadır. WAF'ın temel işlevi, web uygulamalarına gelen istekleri denetlemek, filtrelemek ve kötü niyetli trafikleri engellemektir. Bu sayede potansiyel saldırıları, kötü amaçlı yazılımları, veri sızıntılarını ve diğer web uygulaması güvenlik açıklarını sınırlamaya yardımcı olmaktadır. WAF, HTTP isteklerini denetler ve belirlenmiş kurallara göre filtreler, kötü amaçlı trafikleri engeller, kullanıcıların kimlik doğrulama süreçlerini izler ve yetkilendirme kontrolleri uygulamaktadır. Ancak, bu yöntemler önceden tanımlanmış saldırı imzalarına dayanmakta ve yeni veya karmaşık saldırı türlerine karşı yetersiz kalmaktadır. Son yıllarda, yapay zeka ve özellikle derin öğrenme tabanlı yaklaşımlar, daha adaptif ve öğrenme kabiliyetine sahip güvenlik çözümleri sunma potansiyeli taşımaktadır. Bu çalışma, web uygulamalarına yönelik siber saldırıları tespit etmek amacıyla derin öğrenme modelleri kullanarak optimum yöntemi belirlemeyi amaçlamaktadır. FWAF veri seti üzerinde gerçekleştirilen çalışmada, altı farklı derin öğrenme modeli değerlendirilmiş ve en yüksek doğruluk oranına sahip model seçilmiştir. Modelin gerçek zamanlı ortamda uygulanabilirliğini test etmek amacıyla Jetson Nano platformunda performans analizleri gerçekleştirilmiştir.

Bu çalışma, web uygulamalarına yönelik saldırıları tespit etmek için kullanılan geleneksel yöntemlerin eksikliklerini ele almakta ve derin öğrenme tabanlı bir yaklaşım sunarak literatürdeki boşluğu doldurmayı hedeflemektedir. Yapılan deneysel analizler, önerilen yöntemin saldırı tespitinde yüksek doğruluk oranlarına ulaştığını ve mevcut sistemlere kıyasla daha dinamik bir yapı sunduğunu göstermektedir.

Bu makalenin geri kalanı şu şekilde düzenlenmiştir. Materyal ve Yöntem bölümünde ilgili bu çalışmada kullanışmış olan DL modelleri detaylandırılmıştır ve kullanılan performans ölçütleri ve geliştirilen DL modelin entegrasyonu ile ilgili detaylar verilmiştir. Deneysel Çalışmalar bölümünde DL tabanlı modeller ile elde edilen sonuçlar verilmiştir. Daha sonra çalışmanın sonuçları ve gelecek hedefleri ile ilgili detaylar verilmiştir.

Literatür Taraması

Literatürde web uygulamalarına yönelik siber saldırıların tespit edilmesinde geliştirilen makine öğrenmesi, derin öğrenme tabanlı yaklaşımlar vardır. Brown ve arkadaşları [1], statik, dinamik ve çevrimiçi kötü amaçlı yazılım tespiti için AutoML kullanımı üzerine kapsamlı bir analiz ve içgörüler sunmuştur. Statik analiz için büyük veri kümeleri üzerindeki etkinliği göstermek için SOREL-20M ve makine öğrenimi modellerinin performansını zorlaştırmak amacıyla özel olarak hazırlanan daha küçük EMBER-2018 veri kümesini kullanmışlardır. Otomatik Makine Öğrenimi (AutoML) ile çevrimiçi kötü amaçlı yazılım tespiti senaryolarında, bulut IaaS için Hücresel Sinir Ağları (Convolutional Neural Networks – CNN) kullanmışlardır. SOREL-20M veri seti ile %99,8, EMBER 2018 veri seti ile %98,4 başarım elde etmişlerdir. Çevrim içi kötü amaçlı yazılım yakalamada Baseline'da en başarılı model Darts AutoML ile %98,9 başarım elde etmişlerdir. Uygulamada yine 7 katmanlı Darts AutoML ile %98,6 doğruluk başarımı elde etmişlerdir. Web uygulamaları, kullanıcıların bilgiye anında erişimini sağlayarak ve işletmelerin erişim alanlarını genişletmelerine olanak tanıyarak dijital ortamın merkezi haline gelmiştir. SQL enjeksiyonu (SQLi) gibi enjeksiyon saldırıları, çoğu web uygulamasının bir veri tabanı sistemini entegre ettiği göz önüne alındığında, web uygulamalarına yönelik öne çıkan saldırılardır. Paul ve ekibi [2], SQLi saldırısını çok sınıflı, çoklu saldırı vektörü, önceliklendirme ve önleme problemi olarak formüle etmektedirler. Saldırı tespiti ve sınıflandırması için 457,233 iyi huylu ve kötü huylu ağ trafiği örneğinin yanı sıra SQLi ve iyi huylu yüklere sahip 70,023 örnek toplamışlardır. Birkaç makine öğrenimi tabanlı algoritmanın değerlendirilmesinin ardından, hibrit CNN-LSTM modelleri web ve ağda ortalama 0,97 F1-skoruna ulaşmıştır. Çalışmanın sonuçlarını Vulnerable Web Application (DVWA) ve Vulnado gibi bilinen savunmasız web uygulamalarının boru hattına entegre ederek ve gerçek zamanlı tespit için SQLMap ile yürütülen SQLi DNS sızıntısından Wireshark kullanılarak yakalanan ağ trafiği aracılığıyla öne çıkarmışlardır. Dawadi vd. [3] DDoS, SQL enjeksiyonu ve XSS gibi tehditlerden korunmak için normal HTTP trafiği ile saldırı trafiği arasında karşılaştırmalı bir analiz yapmışlardır. ISCX, CISC ve CICDDoS standart veri setlerinden çeşitli özellikleri incelemiş ve normal ile saldırı trafiği arasındaki farkları belirlemişlerdir. Çalışmada, DDoS, XSS ve SQL enjeksiyonu saldırılarını tespit etmek üzere bir katman mimari modeli geliştirilmiştir. Simülasyon ortamından toplanan verilerle oluşturulan LSTM tabanlı bu mimarinin ilk katmanı, %97,57 doğruluk oranıyla DDoS tespit modeli olarak tasarlamışlardır. İkinci katman ise XSS ve SQL enjeksiyonu tespiti için %89,34 doğruluk oranı elde etmişlerdir. Yüksek HTTP trafiği önce incelenmiş, filtrelenmiş ve ardından ikinci katmana iletilmiştir. WAF, geleneksel ağ güvenlik duvarlarının sağlayamadığı uygulama düzeyinde filtreleme yaparak web uygulamalarına ek bir güvenlik katmanı ekleyerek

literatüre katkı sağlamışlardır. Köksal vd. [4], Kanada Siber Güvenlik Enstitüsü'nün (ISCX-URL-2016) Tekdüzen Kaynak Bulucu (Uniform Resource Locator - URL) veri seti kullanılarak iyi huylu ve kötü huylu URL'lerin sınıflandırılması için toplulukla öğrenme yöntemleri ve Rastgele Orman algoritması kullanılmıştır. Sonuçlar, ortalama %99,42 doğruluk oranıyla ikili sınıflandırma ve %95,68 doğruluk oranıyla çoklu sınıflandırma başarısı göstermektedir. Uçar vd. [5], ISCX-URL-2016 veri kümesi üzerinde zararlı URL'lerin tespiti için bir Derin Öğrenme (Deep Learning – DL) modeli geliştirilmiştir. Bu model, iyi huylu ve kötü niyetli URL'leri tespit etmek için DL algoritmaları olan Uzun Kısa Süreli Bellek (Long Short Term Memory – LSTM) ve CNN modellerini kullanmışlardır. Deneysel sonuçlara göre, LSTM'in %91 doğruluk oranı ve CNN'in %95 doğruluk oranı ile zararlı web sayfalarının tespitinde önemli bir başarı sağlamıştır. Abdi ve Wenjuan [6] 344,821 iyi niyetli URL ve 75,643 kötü niyetli URL üzerinde yapılan deneylerde, basit bir algoritma kullanılarak kötü niyetli URL'leri tespit etme başarısı %96'dan yüksek bir doğruluk oranı elde etmiştir. Bu algoritmada, geleneksel kara liste kullanımının yerine makine öğrenimi teknikleri tercih edilmiştir. Tiryaki vd. [7], 7 katmanlı Tekrarlayan Sinir Ağları (Recurrent Neural Networks – RNN) modeli kullanılarak ulusal ve uluslararası benzer iki veri setinin birleştirilmesiyle devasa bir veri seti oluşturulmuş ve zararlı URL adreslerinin tespitinde %91'in üzerinde başarı elde edilmiştir. Moghimi ve Varjani [8] phishing özellikleri taşıyan web sitelerinin sınıflandırılması için Yapay Sinir Ağı (Artificial Neural Network - ANN) modeli, özellikle de Çok Katmanlı Algılayıcı (Multi Layer Perceptron - MLP) tipi kullanılmıştır. Elde edilen sonuçlar, bu modelin %99,14 başarı oranıyla bu tür web sitelerini sınıflandırmada etkili olduğunu göstermektedir. Ismail ve ekibi [9], web uygulama saldırılarını tespit etmek için 19 geleneksel makine öğrenimi tekniğinin derinlemesine bir değerlendirmesini sunmaktadır. Değerlendirme, HTTPCSIC 2010 veri setinden türetilen rafine veri setleri üzerinde Sinir Ağları ile %90 başarım ile model geliştirmişlerdir. Günümüz güvenlik duvarları önceden verilmiş veriler dışında bir saldırıyla karşılaştığında cevap verememekte ve onu kullanan sistemi yeni saldırılardan koruyamamaktadır. 2001 yılında web uygulamalarındaki güvenlik açıkları ile birlikte Open Web Application Security Project (OWASP) başlatılmıştır [10] Literatürde son yıllarda hazır veri setleri ile geliştirilen siber saldırı tespit sistemleri incelendiğinde Baysal [11], bir web sitesinde alkol görüntüsü var ise "mitmproxy" aracı ile modeli birleştirerek alkol fotoğrafının yüklenmesini engelleyerek içerik filtrelemesi yapan bir güvenlik duvarı oluşturmuştur. Sezer ve Ali [12] çalışmalarında CSIC web uygulamaları veri setini LSTM + Word2Vec ile modelleyerek web sayfasına yapılan siber atakları %99,4 F1-skoru ile tespit edebilmişlerdir. Demir ve Aslan [13] çalışmalarında K-en yakın komşuluk yöntemi ile Fırat Üniversitesi Firewall Cihazını %98,40 doğrulukla modellemişlerdir. Demir [14] çalışmasında ISCX-2012 veri seti ile çalışmıştır. Karar ağacı ile %100 doğruluk başarımı elde etmiştir. Takaoğlu ve Çağdaş [15] elle oluşturdukları veri seti ile Destek Vektör Makinesi ve Naif Bayes yöntemleri ile modelleme sonucunu sırasıyla %79,00 ve %71,00 olarak elde etmişlerdir. Einy [16], hem yüz sahtekarlığı hem de ağ anomalisi tespiti için Çok Amaçlı Parçacık Sürüsü Algoritmasına dayalı Özellik Seçimi ve Hızlı Öğrenme Ağının Kombinasyonu ile %99,60 doğruluk oranı elde etmiştir. Seyyar vd. [17], web uygulamalarının tüm alanlarda kullanımının artması ve web uygulamalarına yönelik saldırıların artmasıyla birlikte ortaya çıkan güvenlik tehditlerini ele alan bir web saldırı tespit sistemi modellemiştir. Web saldırı tespit sistemi normal ve anormal URL ayırt edebilen bir modelden oluşmaktadır. URL analizi aşamasında BERT modelini kullanmışlardır. Sınıflandırma aşamasında CNN modeli kullanmışlardır. Eğitim ve test için CSIC 2010, FWAF ve http Params veri %96'nın üzerinde bir doğruluk elde etmişlerdir. Baş ve Süzen [18] çalışmalarında web uygulamalarındaki güvenlik açıklarını tespit etmede açıklanabilir yapay zekanın katkısını inceleyerek, geliştirme ve uygulama süreçlerinde nasıl kullanılabileceğini işlemişlerdir. Turgut ve Daş [19], akıllı şebeke güvenliğini artırmaya yönelik yeni IDS tekniklerinin geliştirilmesine katkıda bulunmak için hem bileşene özgü hem de genel sistem değerlendirmelerini analiz etmişlerdir. Arslan [20] gerçek zamanlı tespit gerekliliği nedeniyl ön işleme yöntemi ile internet trafiği daha hızlı işlenebilir hale getirmiş ve makine öğrenimi ile sınıflandırma yapmıştır. CSE-CIC-IDS2018 veri setinde test edilen model, hem yüksek başarı oranı hem de işlem süresinde ciddi iyileşmeler sağlamıştır. İkili sınıflandırmada ExtraTree algoritmasıyla %99,00 doğruluk ve %82,96 işlem süresi azaltımı; çok sınıflı (15 sınıf) tespitte ise Rasgele Orman ile %98.5 doğruluk ve %64.43 işlem süresi azaltımı elde edilmiştir. Xu ve ekibi [21] düşük kaynaklı ortamlarda büyük dil modelleri (LLM) kullanarak anormal kullanıcı davranışlarını tespit etmeye yönelik yeni bir yaklaşım sunmuşlardır. Çalışmalarında, sistem günlüklerinden örtük nedensel ilişkiler çıkararak davranış grafikleri oluşturup ve etiket gerektirmeyen grafik karşıt öğrenimi ile nedensel özellik vektörleri üretmişlerdir. Çok ajanlı yapıdaki modelde, anlatıcı ve karar verici ajanlar açıklayıcı metin üretimini geliştirirken, çevirici ajan bu vektörleri anlamlı metinlere dönüştürmüştür. WAB veri setinde yapılan deneysel çalışmalarda kullanılan yöntemin geleneksel yöntemlere kıyasla daha az kaynakla daha etkili analiz sağladığını ve güvenlik uzmanları için anlaşılır çıktılar sunduğunu göstermişlerdir. Salahat ve ekibi [22] SQLi ve XSS saldırılarına karşı makine öğrenimi tabanlı bir çözüm sunmuşlardır. Gerçek saldırı verilerinden oluşan büyük bir veri kümesi kullanılarak TF-IDF yöntemiyle anlamlı özellikler çıkarılmış ve Lojistik Regresyon modeli ile istekler zararlı ya da güvenli olarak sınıflandırılmıştır. Sistem, pfSense güvenlik duvarıyla entegre edilerek tehditlerin tespitinin yanı sıra zararlı kullanıcıların gerçek zamanlı engellenmesi sağlanmıştır. Test sonuçları, modelin %97 doğrulukla saldırıları tespit ettiğini göstermektedir. Wang ve ekibi [23], paket düzeyinde çalışan ve istatistiksel ile zamansal öznitelikleri içeren kısmi akış (partial-flow) temelli yeni bir öznitelik çıkarım yöntemi önermişlerdir. Bu yöntem, CSE-CIC-

IDS 2018 veri setinin oluşturulmasında kullanılan PCAP dosyalarına uygulanarak web saldırı tespiti için özel olarak geliştirilmiş WKLIN-WEB-2023 veri seti oluşturulmuştur. Her iki veri seti üzerinde dokuz farklı sınıflandırma modeli eğitilerek karşılaştırmalı bir değerlendirme yapılmıştır. WKLIN-WEB-2023 veri setiyle eğitilen modellerin; kesinlik, duyarlılık, F1 skoru ve tespit gecikmesi gibi ölçütlerde CSE-CIC-IDS 2018'e göre üstün performans sergilediğini göstermişlerdir. Gallego ve ekibi [24], 100.000'den fazla isteği içeren sentetik bir veri kümesi üzerinde, Naïve Bayes, k-en yakın komşu, destek vektör makineleri ve doğrusal regresyon gibi yapay zeka tabanlı sınıflandırma modellerinin kötü amaçlı istekleri tespit etme performansı incelemişlerdir. Uygulanan modeller, %92 ile %99 arasında değişen başarı oranları ile yüksek doğrulukta kötü niyetli istekleri sınıflandırarak tespit performansının optimize edilebileceğini göstermiştir.

Web uygulamalarına yönelik siber saldırı tespitinde kullanılan geleneksel ve modern yaklaşımlar çeşitli makine öğrenimi ve derin öğrenme modelleriyle geliştirilmektedir. Geleneksel makine öğrenme modelleri ile geliştirilen çalışmaların daha az hesaplama gücü gerektirdiği ve çeşitli veri setleri ile uygulanabilir oldukları tespit edilmiştir. Fakat bu çalışmaların karmaşık saldırı türlerinin tespitinde yeterince esnek olmadığı, veriye bağımlı oldukları ve veri boyutu arttıkça hesaplama maliyetinin de arttığı görülmektedir [9, 13]. Topluluk öğrenmesi modeli ile geliştirilen çalışmanın çoklu öğrenme tekniklerini birleştirerek daha yüksek doğruluk sağladığı görülmektedir. Fakat daha yavaş hesaplama süresi ve büyük veri setleri ile optimize edilmesi zor olduğu görülmektedir [4]. CNN ve LSTM tabanlı derin öğrenme ile geliştirilen modellerin sürekli öğrenme kapasitesi ve sekans bazlı veriler için uygun olması avantajdır. Fakat bu çalışmaların yüksek hesaplama maliyeti gerektirmesi ve aşırı uyum göstermesi risk olarak değerlendirilmektedir. Ayrıca kelime gömme modellerinin optimize edilmesi zorluğu da dezavantajdır [2, 12]. RNN ve Transformer tabanlı geliştirilen çalışmaların zaman serilerinde başarılı olduğu ve BERT tabanlı modelin bağlamsal anlam çıkarma konusunda güçlü olması avantajdır. Fakat hesaplama maliyetinin yüksek olması, uzun bağlılıklarda unutma probleminin olması, yüksek hesaplama maliyeti ve optimizasyon zorluğu dezavantajlardır [7, 17]. Otomatik makine öğrenimi ile geliştirilen çalışmanın model optimizasyonu ve hiperparametre ayarlamaları otomatize edilerek insan hatasından arındırılması yönüyle güçlüdür. Fakat, bu çalışmadaki gibi gerçek zamanlı uygulamalar için maliyetli ve yüksek hesaplama gücü gereksinimi eksikliktir [1]. Hibrit ve alternatif yaklaşımla geliştirilen çalışmanın hızlı öğrenme kapasitesiyle çok yönlü bir model sağlarken, optimizasyon için karmaşık hesaplama gerektirmektedir [16].

Literatürde yapılmış çalışmalara göre, makine öğrenmesi ve derin öğrenme modelleri, web uygulamalarına yönelik siber saldırıları tespit etmek için güçlü araçlar sunmaktadır. CNN, LSTM ve Transformer tabanlı yaklaşımlar daha esnek ve dinamik olsalar da hesaplama maliyetleri yüksektir. Otomatik makine öğrenmesi ise otomatizasyon sağlasa da gerçek zamanlı uygulamalarda sınırlamaları olabilmektedir. Web uygulamaları için siber saldırı tespit sistemlerinde geliştirilen modellerin başarımlarının iyileştirilmesi ve etkin kullanılabilecek özgün bir güvenlik duvarının eksikliği mevcuttur. Literatürde daha iyi sonuçların elde edildiği yeni çalışmalara ihtiyaç vardır. Bu çalışma ile birlikte web uygulamalarının güvenlik zafiyetleri otomatik tespit edilebilecek ve saldırıları engelleme konusunda katkı sağlayacaktır. Derin öğrenme algoritmalarının kullanılmasıyla birlikte, daha geniş bir saldırı yelpazesini tanımlayan ve oluşabilecek bir güvenlik zafiyetinde etkili bir şekilde savunma sağlama potansiyeli sunacaktır.

Materyal ve Yöntem

Bu çalışmada uygulanan veri ön işleme, model eğitimi ve değerlendirme aşamaları Şekil 1'deki kaba kod ile verilmiştir.

Algoritma 1 Derin Öğrenme ile Web Saldırılarını Tespit Etme

Veri Ön İşleme:
$dataset \leftarrow$ FWAF veri setini yükle() url in $dataset$
$domain \leftarrow$ Alan adını çıkar(url)
$path \leftarrow$ URL decode($url \setminus domain$)
Özellik vektörü $\leftarrow$ 23 özellik çıkar($path$)
Ölçeklenmiş özellikler $\leftarrow$ Robust Scaling uygula(Özellik vektörü)

Model Eğitimi:
$models \leftarrow$ {CNN, LSTM, GRU, CNN-LSTM, RNN, DNN} $model$ in $models$
Eğitim verisi, Test verisi $\leftarrow$ Veriyi böl(Ölçeklenmiş özellikler)
$trained\_model \leftarrow model.fit$(Eğitim verisi)
$performance \leftarrow model.evaluate$(Test verisi)

$best\_model \leftarrow$ En yüksek performanslı modeli seç($models$)
Tarayıcı Eklentisi Uygulaması:
while Kullanıcı internette geziniyor **do**
    $current\_url \leftarrow$ Kullanıcının ziyaret ettiği URL'yi yakala()
    $processed\_url \leftarrow$ Ön işleme uygula($current\_url$)
    $prediction \leftarrow best\_model.predict(processed\_url)$
    if $prediction ==$ "zararlı" **then**
        Web sitesine erişimi engelle()
        Kullanıcıyı uyar()
    else
        Erişime izin ver()
    end if
end while

Şekil 1. Çalışmanın adımları.

Şekil 1'e göre, çalışmada, web ortamında gelen her verinin doğasına uygun ön işleme yöntemi uygulanmıştır. Veri setinden özellik çıkardıktan sonra bu sayısal değerleri aynı aralıklara getirmek için veri ölçekleme uygulanmıştır. Web saldırı tespitinde metin verilerini kullanarak DL mimarilerini eğitmek için bu verileri sayısal vektörlere çevrilmiştir.

Veri Ön İşleme

Metindeki özel karakterler, semboller ve biçimlendirme hataları temizlenme aşamasıdır. Metni sözcüklere veya cümlelere bölme işlemidir. Metin ön işleme tamamlandıktan sonra metin sayısal formata dönüştürülerek modellemeye hazır hale girilmesi aşamasıdır [25, 26]. Bu çalışmada kullanılan FWAF veri

setindeki URL'ler üzerinde ön işleme adımları uygulanmıştır. İlk aşamadaki URL'lerin alan adı (domain) kısımları çıkarılarak sadece yol (path) ve parametre kısımları üzerinde çalışılmıştır. Bu yaklaşımın temel nedeni, saldırı örüntülerinin çoğunlukla URL'nin bu bölümlerinde yer almasıdır. Alan adı çıkarıldıktan sonra kalan kısımda URL kodlanmış (encoding) karakterler çözümlenmiştir (decode). Bu işlemler sonucunda elde edilen temizlenmiş URL'ler, özellik çıkarımı aşamasına hazır hale gelmiştir.

Özellik Çıkarımı

Web saldırılarının tespiti için URL'lerden 23 adet anlamlı özellik çıkarılmıştır. Bu özellikler dört ana kategoriye ayrılmıştır:

Saldırı Örüntü Özellikleri

- '**is_xss**': XSS (Cross-Site Scripting) saldırısında kullanılan '*<script>*', '*alert()*', '*onerror=*' gibi örüntülerin varlığını kontrol eder.

- '**is_lfi**': LFI (Local File Inclusion) saldırılarında kullanılan '*../*', '*/etc/passwd*', '*../../*' gibi dizin geçiş örüntülerini tespit eder.

- '**is_oci**': OS Command Injection saldırılarında kullanılan '*whoami*', '*ls*' gibi komut enjeksiyon örüntülerini tespit eder.

- '**is_sqli**': SQL Injection saldırılarında kullanılan '*UNION*', '*SELECT*', '*DROP TABLE*' gibi SQL örüntülerini tespit eder.

Bu özellikler, ilgili saldırı türüne özgü karakteristik örüntüleri yakalayarak modelin saldırıları tanımasını kolaylaştırır. Bu özelliklerin sonuçları ikili (binary) halindedir, yani örüntü tespit edilirse 1, edilmezse 0 değerlerini alırlar.

Karakter Seviyesinde İstatistiksel Özellikler

- '**semicolon_count**': Noktalı virgül sayısı (*;*). Komut enjeksiyonu ve SQLi saldırılarında sık kullanılır.

- '**underscore_count**': Alt çizgi sayısı (_). Özel parametre isimlerinde ve kod karıştırma tekniklerinde kullanılır.

- '**equal_count**': Eşit işareti sayıs (=). Sorgu parametrelerinin ayrılmasında kullanılır, fazla sayıda olması saldırı göstergesi olabilir.

- '**questionmark_count**': Soru işareti sayısı (*?*). Sorgu başlangıcını gösterir, birden fazla olması anormal olabilir.

- '**and_count**': Ampersand sayısı (*&*). Sorgu parametrelerinin ayrılmasında kullanılır, fazla sayıda olması saldırı göstergesidir.

- '**or_count**': Boru işareti sayısı (|). Komut zincirlemede kullanılır, OS Command Injection saldırılarında sık görülür.

- '**dot_count**': Nokta sayısı (.). Dizin geçişlerinde ve dosya uzantılarında kullanılır, LFI saldırılarında fazla kullanılır.

- '**at_count**': At işareti sayisi (@). E-posta adreslerinde, SQLi saldırılarında ve bazı kod karıştırma tekniklerinde kullanılır.

- '**non_numeric**': Alfa-numerik olmayan karakter sayısı. Saldırı URL'lerinde normal kullanıcı trafiğine göre daha fazla özel karakter bulunur.

URL Yapısal Özellikleri

- '**subdir_count**': Alt dizin sayısı ('*/*' ile ayrılmış bölümler). Derin dizin yapıları LFI saldırılarında sık kullanılır.

- '**param_count**' Sorgu parametre sayısı ('*&*' ile ayrılmış). Fazla sayıda parametre kompleks saldırı göstergesi olabilir.

- '**has_extension**': Dosya uzantısı varlığı. Doğrudan dosya erişim girişimlerini gösterebilir.

- '**has_parameter**': Sorgu parametresi varlığı. Dinamik saldırı girişimlerinin çoğu parametre kullanır.

Bu özellikler, URL'nin organizasyonel yapısı hakkında bilgi verir ve normal kullanıcı trafiği ile saldırı trafiği arasındaki yapısal farkları ortaya çıkarır.

Karakter Dağılım Özellikleri

- '**total_digits_url**': URL'deki toplam rakam sayısı. Kod karıştırma tekniklerinde artar.

- '**total_letter_url**': URL'deki toplam harf sayısı. Normal ve saldırı trafiği arasında dağılım farklılıkları gösterir.

- '**total_digits_domain**': Alt dizin kısmındaki rakam sayısıdır.

- '**total_letter_domain**': Alt dizin kısmındaki harf sayısıdır.

- '**total_digits_path**': Yol kısmındaki rakam sayısıdır.

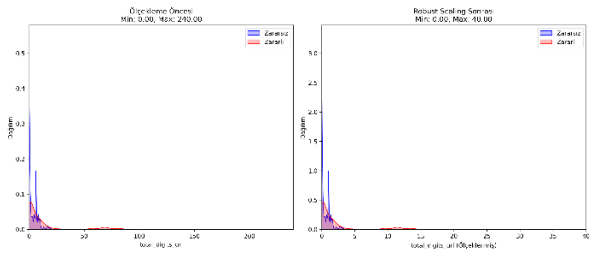
- '**total_letter_path**': Yol kısmındaki harf sayısıdır.

Bu özellikler, URL'nin farklı bölümlerindeki karakter dağılımını niceliksel olarak ölçerek anormal desenlerin tespitine yardımcı olmaktadır.

Veri Ölçekleme

Farklı özelliklerin farklı ölçeklerde olması sebebiyle bu tür verileri aynı ölçeğe getirme işlemi için

kullanılmaktadır [27, 28]. Veri ölçekleme işlemi ile elde edilen örnek bir sonuç Şekil 2'de verilmiştir.



Şekil 2. Veri ölçekleme işlemi ile elde edilen sonuç.

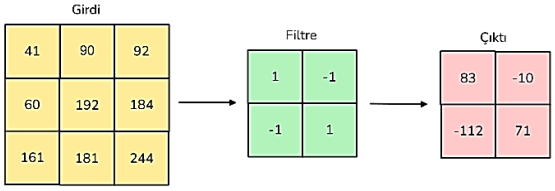
Çıkarılan özelliklerin değer aralıkları büyük farklılıklar göstermektedir. Örneğin, bazı özellikler [0 - 1] aralığında iken, bazı özellikler çok daha geniş aralıklara sahip olabilmektedir. Özelliklerin dağılımındaki bu değerler, derin öğrenme modellerinin eğitim sürecini olumsuz etkileyeceği için Robust Scaling yöntemi kullanılarak özellikler ölçeklendirilmiştir. Robust Scaling, medyan ve çeyrekler arası (interquartile - IQR) aralığını kullanarak verileri ölçekler ve uç değerlerin (outlier) etkisini azaltır. Bu yöntem, özellikle saldırı verilerinde sıkça karşılaşılan aşırı uç (anormal) değerlerin model performansını düşürmesini engeller. Bu çalışmada kullanılan Robust Scaling formülü Denklem (1)'de verilmiştir [28].

$$X_{ölçek} = X - \frac{Ortanca(X)}{IQR(X)} = \frac{X - Q_2}{Q_3 - Q_1} \quad (1)$$

Denklem 1'e göre, Q_1, 1. çeyrek değer (25. yüzdelik dilim), Q_3, 3. çeyrek değer (75. yüzdelik dilim), Q_2, ortanca değerini temsil etmektedir.
Ölçekleme sonrası değerler, orijinal dağılımın merkezi ve yayılımı hakkında bilgi korunarak istatistiksel ve örüntü tabanlı özelliklerin aynı ölçekte sunulması sağlanarak derin öğrenme modelinin daha kararlı ve doğru bir şekilde öğrenmesi mümkün kılınmıştır.

CNN

Verinin özelliklerini belirlemek için konvolüsyon ve havuzlama katmanlarını kullanmaktadır. Bu sayede özelliklerin hiyerarşik temsillerini verilerden otomatik olarak öğrenmektedir. Konvolüsyonel katman, özellik çıkarmak için kullanılmaktadır. Bu katmanda filtreler yer almaktadır. Filtreler kullanarak giriş verisini tarayarak belirli örüntüleri öğrenmektedir. Örneğin, bu çalışmada, web saldırılarında sık geçen kelime kombinasyonları veya karakter dizileri bu katmanda tanımlanmıştır. Kaydırma adımı belirlenerek verinin ne kadar detaylı işleneceği kontrol edilmiştir. Her filtre, veri üzerinden kayarak özellik haritaları oluşturmaktadır. Çekirdekteki değerler veri üzerinde kaydırılır ve değerler verinin değerleriyle çarpılmaktadır. Elde edilen değerler toplanmaktadır. Bu işlem, Şekil 3'teki gibi tüm veriye uygulanmaktadır.



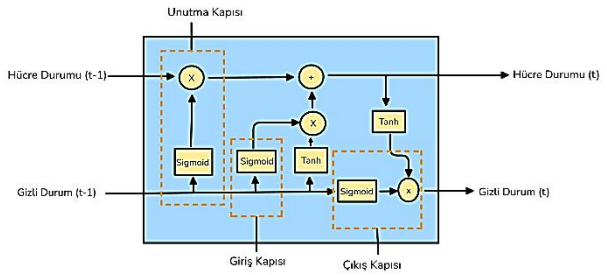
Şekil 3. Konvolüsyon katmanındaki filtreleme işlemi.

Aktivasyon fonksiyonu modelin doğrusal olmayan ilişkilerini öğrenmesini sağlar ve her konvolüsyon katmanından sonra aktivasyon işlemi uygulanmaktadır. Kullanılan aktivasyon fonksiyonu negatif değerleri sıfıra çekerek gradyan kaybolma problemini engellemektedir. Çıktı katmanında, saldırı tespitinde softmax/sigmoid fonksiyonu kullanılmaktadır. Bu, ağın daha karmaşık ve genel özellikler öğrenmesine yardımcı olmaktadır. Havuzlama katmanı, modelin daha genel özellikleri öğrenmesini sağlayarak aşırı öğrenmeyi önlemektedir. Bu çalışmada maksimum havuzlama(max pooling) yöntemi tercih edilmiştir. Bunun sebebi ise filtrelenen özellik haritalarından web saldırılarında en belirgin özniteliklerin seçilmesidir. Özellik haritalarının boyutunu küçültmek ve ağı daha da hızlandırmak için kullanılmaktadır. Böylece hesaplama maliyetini düşürerek, model genelleştirme kapasitesi arttırılmaktadır. Düzleştirme (flatten) katmanı, veriyi 1×1 haline getirmektedir. Böylece tam bağlı veri katmanının girişine hazırlanmış olur. Tam bağlı katmanlar, konvolüsyon katmanlarında çıkarılan özellikleri sınıflandırmak için işlem yapar. Düzleştirme işlemi ile özellik haritaları vektörleştirilir, dropout katmanı eklenerek aşırı öğrenme önlenmektedir. Son katmanda softmax aktivasyon fonksiyonu kullanılarak çıktılar saldırı veya norma istek olarak etiketlenmektedir. Evrişimli sinir ağlarının en önemli özelliği girdilerden otomatik olarak özellik çıkarabilmesidir. Fakat çok yoğun bir hesaplama işlemlerine sahiptir ve hiper-parametre ayarlamasına ihtiyaç duymaktadır [29].

LSTM

Yapay Sinir Ağları (YSA) yöntemi, ağdan hesaplanan hata değerinin gradyanlarını hesaplamaktadır, ardından ağırlıkları güncellemektedir. Geri yayılımlı bir YSA çıkış katmanından giriş katmanına kadar basit bir sinir ağında ağırlıkların güncellenmesinde sorun yaşanmayabilir, ancak derin bir sinir ağında gradyan kaybı veya gradyan patlaması ismi verilen sorunlarla karşılaşılabilmektedir [30]. Gradyan değerlerine geri dönüldüğünde, değerlerin üssel olarak küçülmesi ve bunun da gradyan kaybı sorununa neden olması veya üssel olarak daha büyük olması ve bunun da patlayan gradyan sorununa neden olması mümkündür. Bu sorunu önlemek için düzeltilmiş doğrusal birimini bir aktivasyon fonksiyonu olarak uygulamak çözümlerden birisidir. LSTM ağında zaman adımları mevcuttur ancak her adım için LSTM ağında "bellek" olarak adlandırılan ekstra bilgiler mevcuttur. Dolayısıyla LSTM hücresi sigmoid fonksiyonu içeren bir sinir ağında "unut kapısı (*f*)" ,"giriş kapısı (*I*)" ve "çıkış kapısı (*O*)", *tanh* fonksiyonlu bir sinir ağında "aday

katmanı ($\bar{C}$)", bir vektör olan "gizli durum (H)", bir vektör olan "Bellek durumu (C)" bileşenlerinden oluşmaktadır. Şekil 4'te, t zaman adımındaki LSTM ağı diyagramı verilmiştir.

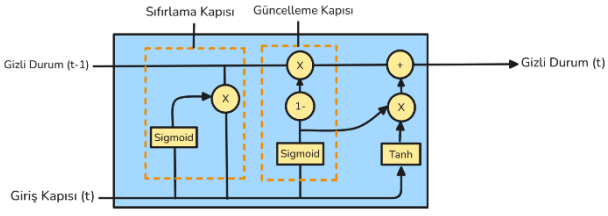


Şekil 4. t zaman adımındaki LSTM ağının genel yapısı.

Şekil 4'e göre, aday katman dışındaki ağlar sigmoid aktivasyon fonksiyonuna sahip tek katmanlı sinir ağlarıdır. Aday katman *tanh* fonksiyonuna sahip tek katmanlı sinir ağıdır. Bu kapılar önce U giriş vektörü ve önceki gizli durum vektörünü (W) alır, ardından bunları birleştirir ve aktivasyon fonksiyonunu uygular. Son olarak bu geçitler vektörler üretir.

GRU

Daha az tensör işlemine sahiptir, bu yüzden LSTM'den daha hızlı çalışmaktadır. LSTM'den farklı olarak, Şekil 5'te de görüldüğü gibi bilgi aktarmak için gizli katmanları kullanmaktadır. Güncelleme kapısı (update gate) ve resetleme kapısı (reset gate) olmak üzere iki kapıdan oluşmaktadır. Güncelleme kapısı, bilgilerin unutulup unutulmayacağına karar verir. Resetleme kapısı, ne kadar bilginin unutulacağına karar vermektedir [31].



Şekil 5. GRU çalışma mekanizması.

RNN

Zamana bağlı veya sıralı veri dizilerini işlemek için özel olarak tasarlanmış bir yapay sinir ağı türüdür. Bu ağlar, önceki girdilerden gelen bilgiyi hatırlayabilen döngüler oluşturarak bir tür hafızaya sahiptir. Bu özellik, dil modelleri, zaman serisi tahminleri ve metin üretimi gibi pek çok görev için idealdir. Ancak, RNN'lerin uzun süreli bağımlılıkları ele almakta zorlandığı bilinmektedir. Bu sorunlar genellikle "kaybolan gradyanlar" ve "patlayan gradyanlar" olarak adlandırılır ve eğitim sırasında karşılaşılabilmektedir. Kaybolan Gradyanlar Problemi, sinir ağlarının eğitimi sırasında gradyanların giderek küçülmesi veya kaybolması durumudur. RNN'ler, önceki zaman adımlarındaki bilgiyi koruyarak dizisel veriler üzerinde işlem yapmaktadır. Zaman İçinde Geriye Yayılım (Backpropagation Through Time - BPTT) algoritması ile eğitilmektedir. Uzun dizilerde, gradyanlar geriye doğru hesaplanırken giderek küçülmektedir. Küçülen gradyanlar, ağın erken katmanlarına ulaştığında nöron ağırlıklarını güncellemek için yetersiz hale gelmektedir. Sonuç olarak, ağ geçmiş bilgileri öğrenemez ve uzun süreli bağımlılıkları yakalayamamaktadır. Sigmoid, *tanh* gibi bazı aktivasyon fonksiyonları belirli aralıklarda gradyanları küçültmektedir. Bu tür aktivasyon fonksiyonları kaybolan gradyanlar problemine neden olmaktadır. Bu problemin üstesinden gelmek için çeşitli çözümler vardır. Doğrultulmuş Lineer Birim (Rectified Linear Unit - ReLU) aktivasyon fonksiyonu bu çözümlerden biridir ve bu çalışmada hesaplama açısından daha verimli olması, derin katmanlarda eğitim kolaylaşması sebepleriyle kullanılmıştır. RNN'lerde sigmoid veya tanh gibi klasik aktivasyon fonksiyonları kullanıldığında çıktı değerleri [-1, 1] veya [0, 1] aralığında kaldığı için gradyanlar küçülebilmektedir. ReLU, negatif değerleri sıfıra çeker ve pozitif değerleri olduğu gibi bırakır. Gradyanın sıfıra yaklaşmasını engelleyerek kaybolan gradyan problemini azaltmaktadır [32].

Geliştirilen DL Modellerinin Web Saldırı Tespitindeki Avantajları

Bu çalışmada geliştirilen 6 farklı DL modellerinin web saldırı tespitinde çeşitli avantajları vardır. CNN modelinin web sorgularında belirli karakter ve kelime öbeklerine dayalı kalıpları tespit etmedeki başarısı önemli bir avantajdır. SQLi, XSS gibi web saldırıları belirli karakter dizileriyle ortaya çıktığı için bu örüntüler CNN filtreleriyle kolaylıkla tespit edilebilmektedir. Bu çalışmada kullanılan diğer DL modeli olan LSTM modelinin uzun web sorgularındaki bağlam ilişkisini koruması ve zaman içinde gelen bilgiyi unutmaması kritik bir rol oynamaktadır. GRU, LSTM'e benzer şekilde sıralı verilerde başarılı tespit yapabilir, ancak daha az parametreye sahip olduğundan daha az hesaplama maliyeti ile daha hızlı eğitim süresi işlem yapmaktadır. RNN modeli temel sıralı model yapısında olması sebebiyle geçmiş girdileri dikkate almaktadır, fakat uzun dizilerde kaybolan gradyan problemi verebilmektedir. Bu çalışmada da karşılaştırmalı analiz yapabilmek için kullanılmıştır. CNN-LSTM hibrit modeli ile CNN ile özniteliklerin çıkarılması, LSTM ile sıralı analiz yaparak hem yerel saldırı örüntüleri hem de uzun vadeli bağlam tespit edilmiştir. DNN modeli ile yapay sinir ağı katmanları sayesinde yüksek doğrulukla web saldırı tespiti yapılabilmiştir.

Tarayıcı Eklentisi

Tarayıcı eklentisi, web tarayıcılarına ek işlevsellik kazandıran yazılım bileşenleridir. Kullanıcı deneyimini geliştirmek, güvenliği artırmak veya belirli görevleri otomatikleştirmek için geliştirilirler. Geliştirdiğimiz tarayıcı eklentisi, kullanıcıların internette daha güvenli

gezinmesini sağlamak amacıyla tasarlanmıştır. Bu eklenti, ziyaret edilen sayfanın URL'sini otomatik olarak alır ve değerlendirme için bir makine öğrenimi modeline iletir. Model, Jetson Nano üzerinde çalışan bir API'ye entegre edilmiştir ve aldığı URL'yi ön işleme sürecinden geçirerek belirli özellikleri çıkarır. Ardından, model bu URL'nin zararlı olup olmadığını sınıflandırır ve uygun bir yanıt üretir. Eğer model, sayfanın zararlı olduğunu tespit ederse, tarayıcı eklentisi otomatik olarak sayfanın içeriğini temizleyerek kullanıcıya bir uyarı mesajı gösterir. Böylece kötü amaçlı yazılımlar veya phishing saldırıları gibi tehditlerin önüne geçilir. Zararsız olarak sınıflandırılan sayfalar ise kullanıcı tarafından normal şekilde görüntülenebilir. Bu yöntem, gerçek zamanlı URL analizi yaparak dinamik ve etkili bir güvenlik katmanı sağlar.

Patlayan Gradyanlar Problemi, sinir ağlarının eğitimi sırasında gradyanlar belirli bir eşiğin üzerine çıkar ve bu da ağın aşırı derecede büyük güncellemeler yapmasına neden olur. Ağın öğrenme oranı çok yüksekse gradyanlar hızla büyüyebilmektedir. Bu sorunu çözmek için öğrenme oranı düşürülebilmektedir. Daha düşük bir öğrenme oranı, daha yavaş ve istikrarlı bir eğitim süreci sağlamaktadır. Bir diğer yöntem ise gradyanları belirli bir eşiğin altında veya üstünde tutmak için kısıtlamaktır. Bu, gradyanlar patlamasını önlemektedir.

Performans Değerlendirmesi

DL modelleme yöntemlerinden faydalanarak optimum matematiksel algoritma (lar)nın seçilebilmesi için geliştirilen her bir algoritmanın doğruluk ve güvenilirliği hesaplanacaktır. Bu çalışma kapsamında, sınıflandırma modelinin başarım değerlendirmesi için performans ölçütlerinden kesinlik, duyarlılık, F–ölçüsü, doğruluk metrikleri hesaplanacaktır. Bu metriklerin değerleri gerçek pozitif (TP), yanlış pozitif (FP), yanlış negatif (FN) ve karışıklık matrisindeki gerçek negatif (TN) değerleri kullanılarak Denklem 2–5'e göre hesaplanacaktır [33].

$$Kesinlik = \frac{TP}{TP+FP} \quad (2)$$

$$Duyarlılık = \frac{TP}{TP+FN} \quad (3)$$

$$F1 = \frac{2TP}{2TP+FP+FN} \quad (4)$$

$$Doğruluk = \frac{TP+TN}{TP+TN+FP+FN} \quad (5)$$

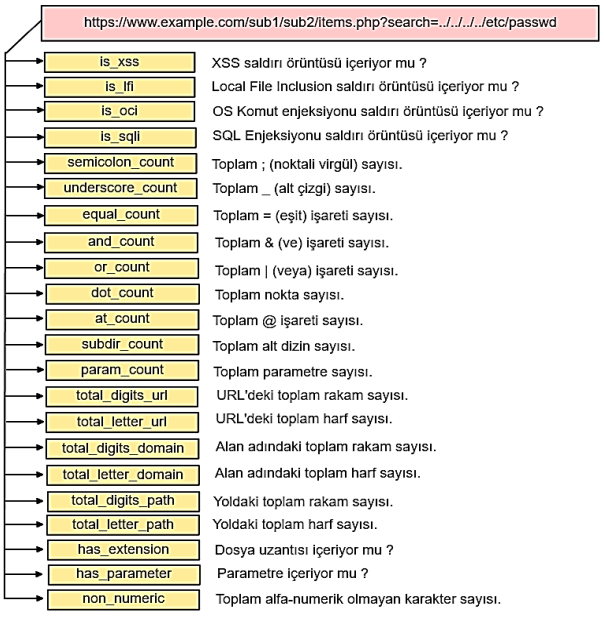
Çalışmanın sonucunda; DL algoritmaları, web uygulamalarında ortaya çıkabilecek çeşitli güvenlik tehditlerini tespit etme ve önleme konusunda kullanılabilecektir. Bu çalışma sayesinde, web uygulamasının normal çalışma davranışını analiz edebilir ve anormal aktiviteleri tespit edebilecektir. Bu, saldırı girişimlerini veya kullanıcı davranışında belirtilen potansiyel güvenlik sorunlarını tespit etmeye yardımcı olabilecektir. Ayrıca, web uygulamasına yönelik saldırıları otomatik olarak tespit edebilecek ve saldırıya karşı önlemler alabilecektir. Bu, güvenlik tehditlerinin hızlı bir şekilde tanımlanmasını ve müdahale edilmesini sağlayarak saldırıların etkisini azaltacaktır.

Web Uygulamasında Model Entegrasyonu

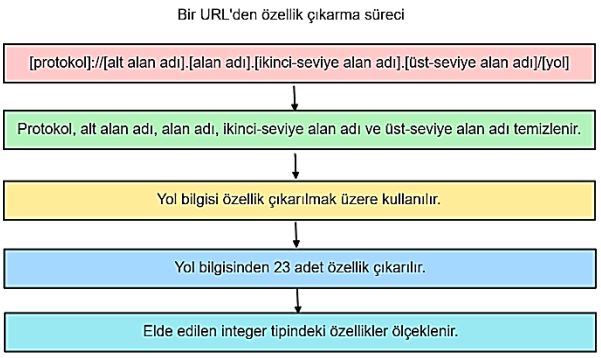
Bu çalışmada model geliştirmek için python dili ve scikit-learn, tensorflow ve keras kütüphaneleri kullanılmıştır. Grafikleri oluşturmak için matplotlib, seaborn ve plotly kütüphaneleri kullanılmıştır. Bu modeller NVIDIA Jetson Nano üzerinde çalıştırılmıştır. Jetson Nano, DLve yapay zeka uygulamalarını desteklemek üzere tasarlanmış güçlü bir cihazdır. Jetson Nano'nun küçük boyutu ve düşük güç tüketimi, yerleşik sistemlerde ve taşınabilir cihazlarda kullanımı kolaylaştırmaktadır. Flask ile bir API geliştirilmiş ve "AdminLTE" şeması kullanılarak oluşturulan web arayüzünde tahmin sonuçları gösterilmiştir [34]. Cihazlar üzerinde tarayıcı eklentisi ile URL bilgileri yakalanacak, sınıflandırılmak üzere ağdaki bulunan Jetson Nano üzerindeki API'ye gönderilecektir. Bu tahmin sonuçları da web arayüzünde gösterilecektir. Gelen her istek isteğe bağlı olarak bir bulut depolama biriminde veya jetson nano üzerinde tutulmuştur.

Deneysel Çalışmalar

FWAF veri seti "goodqueries.txt" ve "badqueries.txt" olmak üzere iki dosyadan oluşur. Bunların 44.662 adeti zararlı sorgu ve kalan 1.266,049 adeti zararsız sorgudur. Veri setlerini birleştirerek "label" isminde bir sütun ekliyoruz. Sorgu zararlı ise 1 zararsız ise 0 değerini veriyoruz. Daha sonra bu sorguları kullanarak özellik çıkarma işlemine geçiriyoruz. Çıkarılan özellikleri ve açıklamalarını Şekil 6'daki gibidir [35].



(a)



(b)

Şekil 6. Bir URL'den çıkarılan özellikler.

Tablo 1'e göre eğitim ve test veri setlerindeki Zararlı ve Zararsız etiketli sordu kayıtları verilmiştir.

Tablo 1. Veri seti.

	Zararsız	Zararlı
Eğitim Seti	1.139,389	40,196
Test Seti	126,600	4,466

Çalışmada kullanılan "goodqueries.txt" ve "badqueries.txt" dosyalarındaki veriler, sistematik bir şekilde seçilmiş ve etiketlenmiştir. Ancak, zararlı sorguların (badqueries) sayısının iyi niyetli sorgulara (goodqueries) kıyasla daha az olması veri dengesizliği sorununa yol açmıştır. Bu sorunu gidermek amacıyla sınıf ağırlıklandırması stratejisi uygulanarak kayıp fonksiyonuna sınıf ağırlıkları eklenmiştir. Bu çalışma için veri setindeki dengesizliği (zararlı/zararsız oranı 1:28.35) gidermek için karekök tabanlı ters frekans ağırlıklandırması uygulanmıştır [36]. Bu yöntem, standart ters frekans yönteminin çoğunluk sınıfını aşırı cezalandırma eğilimini yumuşatarak, azınlık sınıfını dengeli şekilde ön plana çıkarmaktadır. Ağırlıklar Denklem (6)'deki gibi yapılmıştır.

$$ağırlık_{sınıf}(i) = karekök\left(\frac{T}{2T_i}\right) \quad (6)$$

Denklem 6'ya göre T, eğitim setindeki toplam örnek sayısını bu çalışmadaki 1.179,585 değerini temsil etmektedir. T_i, i sınıfındaki örnek sayısını, ve karekök alma işlemi aşırı ağırlıklandırmayı önlemektedir. Bu çalışma için sınıflar için elde edilen ağırlıklar, Sınıf 0 (Zararsız): 0,72, Sınıf 1 (Zararlı): 3,83 olmuştur. Böylece, daha az sayıda verisi olan zararlı sorguların etkisini artırarak modelin bu sınıfı daha iyi öğrenmesi sağlanmıştır.

FWAF setindeki URL adreslerini zararlı ve zararsız olarak sınıflandırmak için bu adreslerden çeşitli özellikler çıkarılmıştır. Bu çalışmada web uygulamalarında zararlı sorguların tespit edilebilmesinde altı farklı DL modelleri için uygulanan optimum ince ayar değerleri Tablo 2'de verilmiştir.

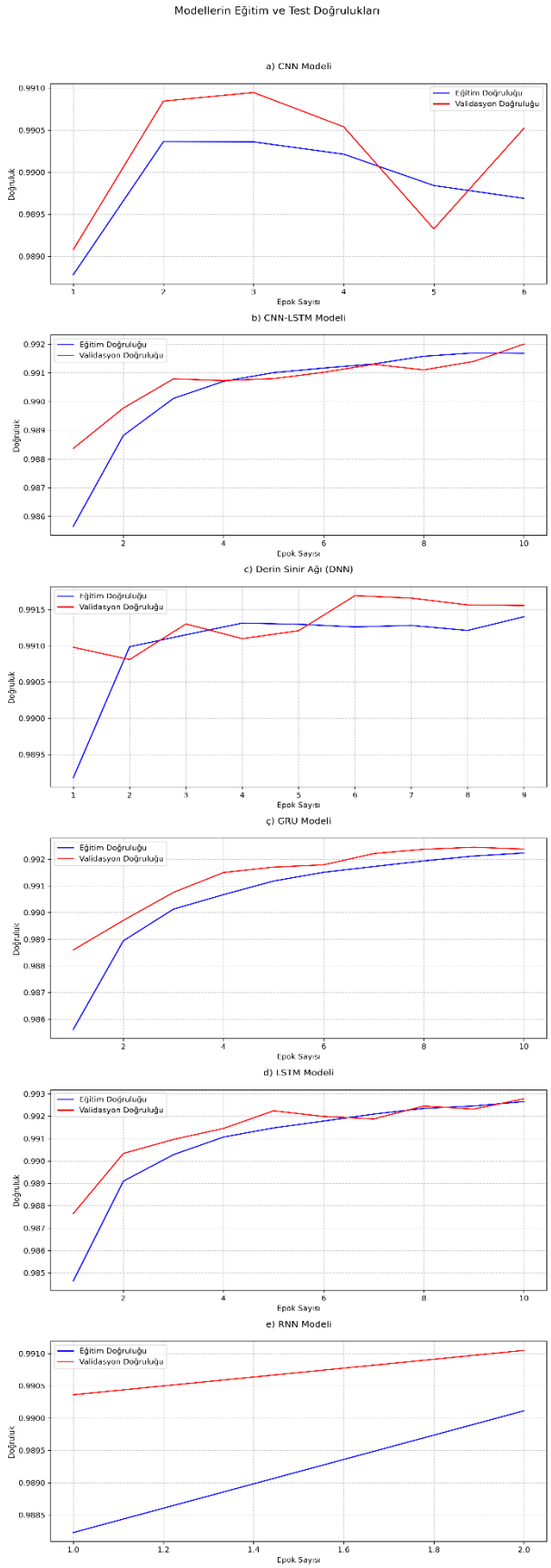
Tablo 2. Model ince ayar değerleri.

Model Adı	Batch Size	Optimizer	Öğrenme Katsayısı	Epok
CNN-LSTM	128	RMSprop	0,001	10
CNN	128	RMSprop	0,001	10
DNN	128	RMSprop	0,001	10
LSTM	128	Adam	0,001	10
GRU	128	RMSprop	0,001	10
RNN	128	RMSprop	0,001	2

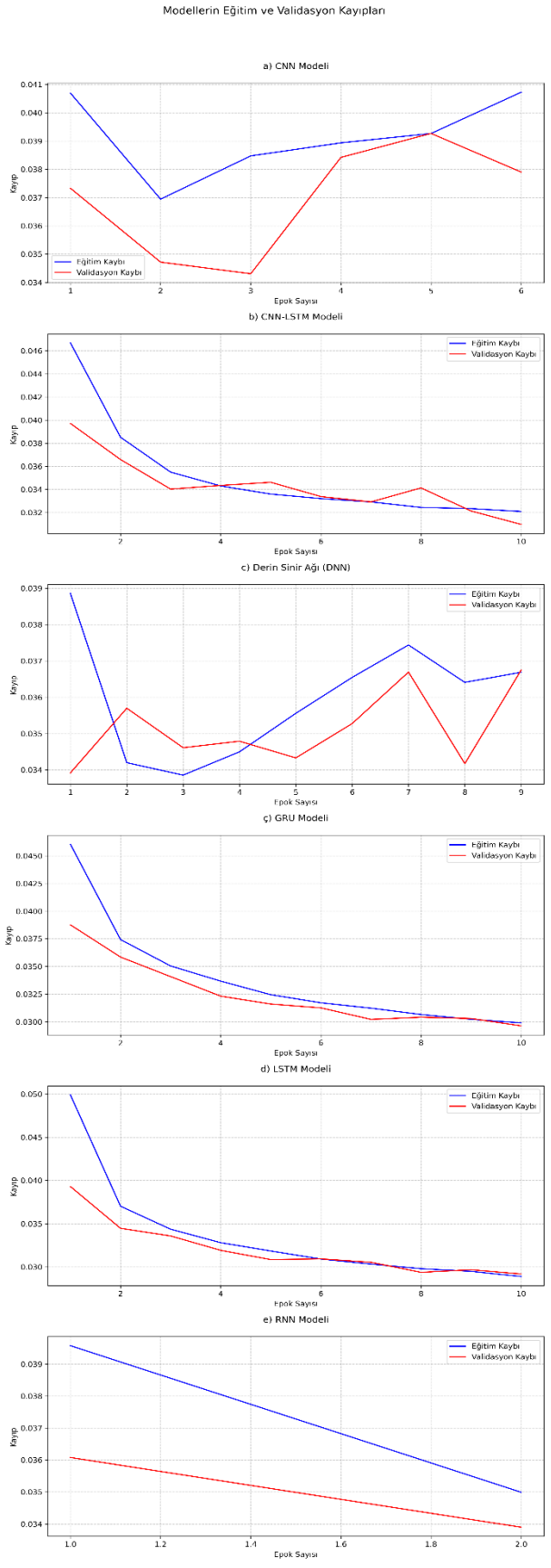
Çalışmada derin öğrenme modellerinin en uygun yapılandırmasını belirlemek amacıyla 'keras-tuner' kütüphanesi kullanılarak kapsamlı bir hiperparametre optimizasyonu gerçekleştirilmiştir. Modelin ağırlık güncelleme stratejisini belirleyen optimizasyon algoritması, öğrenme hızı, modelin tüm veri seti üzerinden kaç kez eğitileceği, her iterasyonda kullanılacak örnek sayısı olmak üzere dört krittik hiperparametre ile çalışılmıştır. Bu süreçte odak metrik olarak F1-skoru değerinin maksimizasyonu hedeflenmiştir. F1-skorunun tercih edilmesinin temel nedeni, dengesiz veri setlerinde model performansını daha bütüncül bir şekilde değerlendirebilmesidir. Çalışmada hiperparametre değerleri, modelin eğitim ve F1-skorları farklı deneylerle çeşitli değerlerle uygulanmıştır. Epok sayısı, erken durdurma göz önünde bulundurularak modelin aşırı öğrenmesini önleyecek biçimde bu değerler optimum değerler olarak seçilmiştir.

Model Değerlendirmesi

Bu çalışmada model eğitimi için Python dili ile birlikte "TensorFlow" kütüphanesi kullanılmıştır. Modeller, Ryzen 5 5600 işlemciye, 16 Gb RAM (Random Access Memory) ve RTX 3070 8 GB ekran kartına sahip bir bilgisayar üzerinde çalıştırılmıştır. Üç veri seti için de geliştirilen modeller Şekil 7'de verilmiştir. Veri seti dengesiz olduğu için başarı ölçütü olarak Doğruluk, Kesinlik, Duyarlılık, ve F1-skoru ölçütleri kullanılmıştır. Çalışmanın eğitim-validasyon kayıp grafikleri Şekil 8'de verilmiştir.

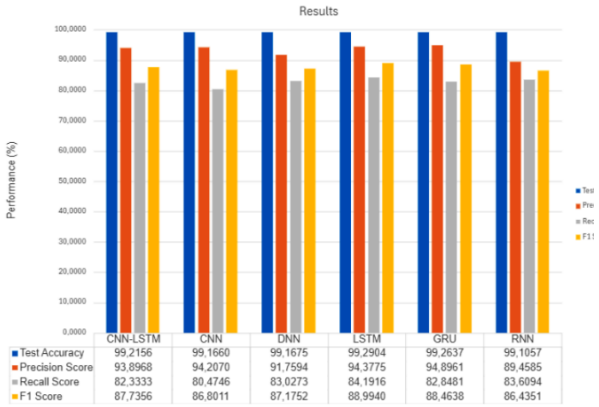


Şekil 7: DL modellerine ait eğitim - validasyon doğruluk eğrileri.



Şekil 8: DL modellerine ait eğitim - validasyon kayıp eğrileri

Şekil 8'de DL modellerine göre eğitim-validasyon kayıp eğrileri verilmiştir.



Şekil 9. Test Başarım Sonuçları

Tablo 3'te modellere göre eğitim ve test süreleri için geçen süre ile ilgili bilgi verilmiştir.

Tablo 3. Eğitim ve test sürelerinin karşılaştırılması.

Model Adı	Eğitim Süresi (sn)	Test Süresi (sn)
CNN	85,587	12,058
CNN-LSTM	586,473	13,455
DNN	454,114	10,654
GRU	1147,206	28,432
LSTM	1330,640	28,682
RNN	1732,097	45,619

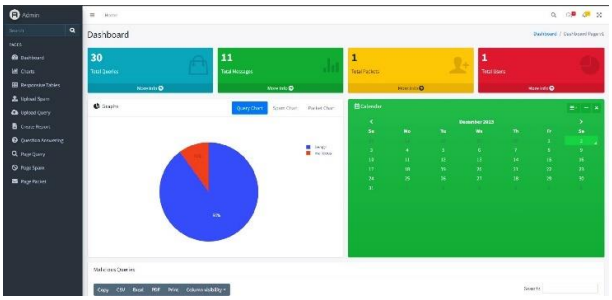
Şekil 7 ve Şekil 8'e göre, zararlı sorgu tespiti için eğitilen modellerin sonuçları verilmiştir. Şekil 9'a göre, LSTM modeli %99,29 test performansı, %94,7 hassasiyet performansı, %84,19 geri çağırma performansı, %88,99 F1-skoru elde edilmiştir. Tablo 3'e göre, 6 farklı model arasında en başarılı doğruluğu veren LSTM modelinin eğitim işlemi için 1331 saniye, test işlemi için ise 29 saniye gerekmiştir.

Web Uygulamasında Model Entegrasyonu

Zararlı web saldırı tespiti modelini gerçek zamanlı test etmek için bir tarayıcı eklentisi yazılmıştır. Kullanıcının girdiği her sayfayı yakalayıp eklentiye gönderilmektedir. Şekil 10'daki eklenti de API'ye adresi gönderip model tarafından sınıflandırılmaktadır.

Şekil 10. Modelin gerçek zamanlı çalıştırmak için yapılan işlemler.

Oluşturulan modelleri takip etmek ve sonuçları izlemek için "AdminLTE" şablonunu kullanılıştır. Şekil 11'e göre, AdminLTE açık kaynak kodlu yönetici panosu ve kontrol paneli temasıdır. Bu şablonu kullarak Flask kütüphanesi ile beraber bir web uygulaması geliştirilmiştir.



Şekil 11. Geliştirilen web uygulamasının ana sayfa ekranı.

Sonuç

Bu çalışmada, 6 farklı DL modeli ile web uygulamalarına yönelik siber saldırı tespit edilmesi modellenmiştir. Model seçimimiz Tablo 4'te verilen literatürdeki en başarılı yöntemlerle karşılaştırılmış ve test sonuçlarımız literatürdeki benzer çalışmalara kıyasla oldukça yüksek başarı oranları göstermiştir. Teknolojik ilerleme ve yapay zeka gelişimiyle birlikte artan tehdit ve saldırılar, bilgisayar sistemlerinin güvenliğini önemli bir şekilde etkilemektedir. Bu durum, güvelik duvarı uygulamalarının karmaşık ve dinamik tehditlere karşı yetersiz kalmasına neden olmaktadır. Bu duruma çözüm olarak DL yöntemleri kullanarak kural tabanlı güvenlik yöntemlerine kalmadan bir savunma yapılacaktır. Bu işlem için ilk olarak veri seti temizlenmiş, elde edilen veriler ölçeklendirilmiştir. Daha sonra bu veri kullanılarak DL modelleri ile eğitilmiştir. Bu çalışmada 6 farklı DL modelleri arasında en başarılı saldırı tespiti LSTM yöntemi ile elde edilmiştir.

Tablo 4. Literatür ile bu çalışmanın sonuçların karşılaştırılması.

Çalışma	Veri Seti	Model	Başarı Oranı	Başlıca Katkı
[1]	SOREL-20M, EMBER-2018	CNN, AutoML	SOREL-20M: %99.8, EMBER: %98.4	AutoM+F3:F15L ile çevrimiçi kötü amaçlı yazılım tespiti; CNN kullanımı
[2]	Özgün Veri Kümesi (SQLi)	CNN-LSTM	%0.97 F1-skor	SQL enjeksiyon tespiti için CNN-LSTM hibriti; Web ve ağ tespiti
[3]	ISCX, CISC, CICDDoS	LSTM	DDoS: %97.57, XSS: %89.34	DDoS, XSS, SQL enjeksiyon tespiti için LSTM tabanlı model
[4]	ISCX-URL-2016	Rastgele Orman	İkili: %99.42, Çoklu: %95.68	URL tespiti; Rastgele Orman kullanımı
[5]	ISCX-URL-2016	LSTM, CNN	LSTM: %91, CNN: %95	Zararlı URL tespiti; LSTM ve CNN kullanımının karşılaştırılması
[6]	Özgün Veri Kümesi (URL)	Makine Öğrenimi	%96+	Kötü niyetli URL tespiti için makine öğrenimi teknikleri
[7]	Ulusal & Uluslararası Veri Setleri	RNN	%91+	Zararlı URL tespiti; RNN modeli kullanımı
[8]	Özgün Veri Kümesi (Phishing)	ANN (MLP)	%99.14	Phishing web siteleri sınıflandırması; MLP kullanımının başarı oranı
[9]	HTTPCSIC 2010	Sinir Ağları	90%	Web uygulama saldırı tespiti; 19 geleneksel makine öğrenimi tekniği ile değerlendirme
[12]	CSIC	LSTM + Word2Vec	%99.4 F1-skor	Web uygulamalarına yapılan saldırıları tespit için LSTM + Word2Vec kombinasyonu
[13]	Özgün Veri Kümesi	K-NN	%98.40	Firewall cihazı tespiti; K-NN yöntemi ile modelleme
[18]	Özgün Veri Kümesi	LSTM	Test: %99.29	Zararlı sorgu tespiti; LSTM modelinin yüksek başarı oranı
Bu çalışma	**FWAF veri seti**	**LSTM**	**Test %99.29 F1-skor: %88.99**	**Zararlı URL tespitinde 6 farklı Derin Öğrenme Modeli geliştirilmiştir.**

Çeşitli saldırı türlerine karşı modelin genelleme yeteneği değerlendirilmiş ve farklı derin öğrenme modelleri ile kıyaslamalar yapılmıştır. Eğitilen modeller, tarayıcı eklentisi uygulaması ile yakalanan web adreslerini sınıflandırmak üzere kullanılmıştır.

Zararlı sorgu tespiti gibi konularda tehdit türleri zamanla değişebilme potansiyeline karşı modelin periyodik olarak yeniden eğitilmesi gerekmektedir. Derin öğrenme tabanlı modellerin eğitilmesi ve anlamlı verinin çıkarılması sırasında yüksek hesaplama gücü gerektirebilmektedir. Gelecekte, çalışmamızda farklı veri setleri veya modeller ile ek karşılaştırmalar yapılarak modelimizin genelleme yeteneği daha iyi değerlendirilecektir.

Etik kurul onayı ve çıkar çatışması beyanı

Hazırlanan makalede etik kurul izni alınmasına gerek yoktur.

Hazırlanan makalede herhangi bir kişi/kurum ile çıkar çatışması bulunmamaktadır.

Yazar Katkıları

1.ve 2. yazar DL algoritmaların tasarımına ve veri toplama sürecine katkıda bulunmuşlardır. 3, 4, ve 5. yazarlar metodoloji kısmını yazmış ve son düzeltmeleri gerçekleştirmiştir.

Teşekkür

Bu proje İstanbul Sabahattin Zaim Üniversitesi BAP-2023-36 numaralı proje ile desteklenmiştir.

Ayrıca bu çalışma TUBITAK 2209 - 1919B012322054 kodlu proje ile desteklenmiştir.

Kaynaklar

[1] A. Brown, M. Gupta, and M. Abdelsalam, "Automated machine learning for deep learning based malware detection," *Computers & Security*, vol. 137, p. 103582, Nov. 2023, doi: 10.1016/j.cose.2023.103582.

[2] A. Paul, V. Sharma, and O. Olukoya, "SQL injection attack: Detection, prioritization & prevention," *Journal of Information Security and Applications*, vol. 85, p. 103871, Aug. 2024, doi: 10.1016/j.jisa.2024.103871.

[3] B. Dawadi, B. Adhikari, and D. Srivastava, "Deep learning technique-enabled web application firewall for the detection of web attacks," *Sensors*, vol. 23, no. 4, p. 2073, Feb. 2023, doi: 10.3390/s23042073.

[4] K. Köksal, B. Doğan, and Z. A. Altıkardeş, "Topluluk sınıflandırıcı kullanarak zararlı url tespiti," *Veri Bilimi*, vol. *4, no.* 3, p. 113-122, 2021.

[5] E. Uçar, M. Ucar, and M. O. İncetaş, "A Deep learning approach for detection of malicious URLs," In *6th International Management Information Systems Conference*, p. 12-20, October, 2019.

[6] F. D. Abdi, and L. Wenjuan, "Malicious URL detection using convolutional neural network," *Journal*

International Journal of Computer Science, Engineering and Information Technology, vol. *7, no.* 6, p. 1-8, 2017.

[7] F. Tiryaki, Ü. Şentürk, and İ. Yücedağ, "Developing and evaluating an artificial intelligence model for malicious url detection," *Avrupa Bilim ve Teknoloji Dergisi*, vol.47, p.13-17, 2023.

[8] M. Moghimi and A. Y. Varjani, "New rule-based phishing detection method," *Expert Systems With Applications*, vol. 53, pp. 231–242, Jan. 2016, doi: 10.1016/j.eswa.2016.01.028.

[9] M. Ismail, S. Alrabaee, K.-K. R. Choo, L. Ali, and S. Harous, "A comprehensive evaluation of machine learning algorithms for web application attack detection with knowledge graph integration," *Mobile Networks and Applications*, Jul. 2024, doi: 10.1007/s11036-024-02367-z.

[10] Ş. Bayrak, and B. Ardanuç, "Web uygulamaları için derin öğrenme yöntemiyle güvenlik duvarı uygulaması firewall application with the deep learning method for the web applications," *Sinyal İşlemeleri Uygulamaları Konferansı*, 1-5, 2022.

[11] K. Baysal, "Derin öğrenme temelli yeni nesil güvenlik duvarının tasarlanması", *Trakya Üniversitesi, Fen Bilimleri Enstitüsü*, Doktora Tezi, 2022.

[12] S. Toprak, and A. Yavuz, "Web application firewall based on anomaly detection using deep learning," *Acta Infologica*, vol. 0, no. 0, p. 0, Oct. 2022, doi: 10.26650/acin.1039042.

[13] S. Demir, Z. Aslan, "K-NN, NN ve feature selection yöntemleri ile firewall verilerinin sınıflandırması", *Anadolu Bil Meslek Yüksekokulu Dergisi*, vol. 17, no.66, pp.139-148, 2023.

[14] F. Demir, "Siber saldırı tespiti için makine öğrenmesi yöntemlerinin performanslarının incelenmesi," *Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, vol. 23, no. 2, pp. 782–791, May 2021, doi: 10.25092/baunfbed.876338.

[15] M. Takaoğlu and Ç. Özer, "The effect of machine learning on intruson detectıon systems," *Uluslararası Yönetim Bilişim Sistemleri ve Bilgisayar Bilimleri Dergisi*, vol. 3, no. 1, pp. 11–22, Jun. 2019, doi: 10.33461/uybisbbd.558192.

[16] Einy, S. Makina öğrenmesile biyometrik sahtekarlığa ve ağ anormallik tespitine dayalı saldırı tespiti. *Sakarya Üniversitesi*, 2021.

[17] Y. E. Seyyar, A. G. Yavuz, and H. M. Unver, "Detection of web attacks using the BERT model," *2022 30th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, May 2022, doi: 10.1109/siu55565.2022.9864721.

[18] E. Baş, and A. A. Süzen, Web uygulama zafiyetlerinin keşfinde açıklanabilir yapay zekânın yeri. *All Science Proceeding*, 2023.

[19] I. Abasıkeleş-Turgut, and R. Daş, "Anomaly and intrusion detection systems for smart grids," *Elsevier*, 2024, pp. 231–270. doi: https://doi.org/10.1016/B978-0-443-14066-2.00005-0.

[20] R. S. Arslan, "FastTrafficAnalyzer: An efficient method for intrusion detection systems to analyze network traffic," *DÜMF Mühendislik Dergisi*, pp. 565–572, Sep. 2021, https://doi.org/10.24012/dumf.1001881.

[21] Y. Xu, Q. Zhang, H. Deng, Z. Liu, C. Yang, and Y. Fang, "Unknown web attack threat detection based on large language model," *Applied Soft Computing*, pp. 112905–112905,Feb.2025, https://doi.org/10.1016/j.asoc.2025.112905.

[22] Osama Salahat, Y. Assaf, and A. Younis, "Securing web applications using machine learning," *Najah.edu*, Feb.03,2025. https://repository.najah.edu/items/97bd9567-c9cd-4722-914e-4fba60c321c8

[23] T. Wang, Y. Xu, and Z. Tang, "Toward fast network intrusion detection for web services: partial-flow feature extraction and dataset construction," *International Journal of Web Information Systems*, Dec. 2024, doi: https://doi.org/10.1108/ijwis-09-2024-0261.

[24] R. Gallego, L. Pérez, Marcos Luengo Viñuela, and María Concepción Vega-Hernández, "Artificial intelligent web application firewall for advanced detection of web injection attacks," *Expert Systems*, Nov. 2023, doi: https://doi.org/10.1111/exsy.13505.

[25] L. Hickman, S. Thapa, L. Tay, M. Cao, and P. Srinivasan, "Text preprocessing for text mining in organizational research: review and recommendations," *Organizational Research Methods*, vol. 25, no. 1, pp. 114–146, Nov. 2020, doi: 10.1177/1094428120971683.

[26] C. P. Chai, "Comparison of text preprocessing methods," *Natural Language Engineering*, vol. 29, no. 3, pp. 509–553, Jun. 2022, doi: 10.1017/s1351324922000213.

[27] S. Feuerriegel *et al.*, "Using natural language processing to analyse text data in behavioural science," *Nature Reviews Psychology*, Jan. 2025, doi: 10.1038/s44159-024-00392-z.

[28] M. Ahsan, M. Mahmud, P. Saha, K. Gupta, and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," *Technologies*, vol. 9, no. 3, p. 52, Jul. 2021, doi: 10.3390/technologies9030052..

[29] J. Barbero-Gómez, R. P. M. Cruz, J. S. Cardoso, P. A. Gutiérrez, and C. Hervás-Martínez, "CNN explanation methods for ordinal regression tasks," *Neurocomputing*,

p. 128878, Nov. 2024, doi: 10.1016/j.neucom.2024.128878.

[30] S. Shao, "Enhancing sentiment analysis with a CNN-Stacked LSTM hybrid model," *ITM Web of Conferences*, vol. 70, p. 02002, Jan. 2025, doi: 10.1051/itmconf/20257002002.

[31] B. Ticona, F. Carranza, and V. Cotik, "Indigenous languages spoken in Argentina: a survey of NLP and speech resources," *arXiv (Cornell University)*, Jan. 2025, doi: 10.48550/arxiv.2501.09943.

[32] A. Dayan and A. Yilmaz, "Modelling the machines' language with natural language processing and machine learning algorithms," *DÜMF Mühendislik Dergisi*, Jul. 2022, doi: 10.24012/dumf.1131565.

[33] S. Bayrak, "Unveiling intrusions: explainable SVM approaches for addressing encrypted Wi-Fi traffic in UAV networks," *Knowledge and Information Systems*, vol. 66, no. 11, pp. 6675–6695, Jul. 2024, doi: https://doi.org/10.1007/s10115-024-02181-9.

[34] A. Almsaeed, (2018). AdminLTE control panel template. URl: https://almsaeedstudio. com/(visited on 17/4/2023).

[35] F. Ahmad, "GitHub - faizann24/Fwaf-Machine-Learning-driven-Web-Application-Firewall: Machine learning driven web application firewall to detect malicious queries with high accuracy.," *GitHub*. https://github.com/faizann24/Fwaf-Machine-Learning-driven-Web-Application Firewall?tab=readme-ov-file]

[36] P. Tufekci and E. Uzun, "Author detection by using different term weighting schemes," *2013 21st Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, Apr. 2013, doi: https://doi.org/10.1109/siu.2013.6531190.