School of Engineering
InIT Institute of Computer Science

Lisans tezi Bilgisayar Bilimleri

Web Erişim Günlükleri için Anomali Algılama Makine Öğrenmesi ile

Yazar	Matthias Koch Oliver Wiedler
Ana Süpervizör	Dr. Ariane Trammell
Dış Uzman	Pascal Imthurn
Endüstriyel ortak	Sonraki Adım AG
Tarih	07.06.2024

ÖZGÜNLÜK BEYANI

Mühendislik Fakültesi'nde Lisans Tezi

Bu Lisans tezini sunarak, imzası bulunan öğrenci bu tezin
kendisinin eseri olup, üçüncü bir şahsın yardımı olmaksızın yazılmıştır. (Grup çalışmaları:
(Diğer grup üyelerinin performansları üçüncü şahıs olarak değerlendirilmez).

Öğrenci, metindeki (İnternet sayfaları dahil) ve eklerdeki tüm kaynakların doğru bir şekilde belirtildiğini beyan eder. Bu, herhangi bir intihal olmadığı anlamına gelir.
yani Lisans tezinin hiçbir bölümü kısmen veya tamamen başka bir kaynaktan alınmamıştır
metinler ve öğrencinin kendi çalışması olarak temsil edilmiş veya doğru bir şekilde sunulmadan dahil edilmiştir referans alındı.

Herhangi bir uygunsuz davranış, Genel Yönetmeliğin 39 ve 40. paragraflarına göre işlem görecektir. Zürih Uygulamalı Bilimler Üniversitesi'ndeki (Rahmenprüfungsordnung ZHAW (RPO)) Lisans ve Yüksek Lisans Derecesi derslerine ilişkin Akademik Yönetmelikler ve aşağıdakilere tabidir: Üniversite yönetmeliklerinde yer alan disiplin cezalarına ilişkin hükümler uygulanır.

Şehir, Tarih:	Adı Öğrenci:
Winterthur, 07.06.2024	Matthias Koch
Winterthur, 07.06.2024	Oliver Wiedler

Zusammenfassung

Dijitalleştirme sırasında Web'de daha hassas veriler elde edildi. Siber Suçlar İçin Çekici Bir Şeyler Var. Um einen
Tüm İşlemler Web Sunucusunu Sürekli Olarak Çalıştırmak İçin Kullanılabilir
Gehostete Webanwendungen, Bankenbranche'de çok uygunsuzdu, mantıklı Veriler ve
Eingeschränkten Kaynaklar ile Anwendun-gen, von entscheiden-der Bedeutung ist. Düzenli
Kayıtlar ve Tipik Kayıtların Düzenlenmesi, Sisteme Uygun Olmayan Günlüklerin Etkinleştirilmesi

Veriler veya Schwachstellen veya Verfügbarkeit des'in Angriffen'i
Sistemler (DoS). Bir LSTM Otomatik Kodlayıcı için Bir Prototip Oluşturun
nutzen, um solche Anomalien zu erkennen. Kalıp Prototipi 14,5 ile birlikte
Milyonlarca Üretim Web Sunucusu Yeni Bir Zamanda Üretildi
Wochen antrenmanı. Bu, ekstra veriler ve kategorilere göre günlüklerin ayrıntılı bir kullanım kılavuzudur. Anschliessend wurden ölmek
Prototip ve Günlükleri bir gün içinde elde etmek, bir yandan da bir yandan da bir yandan göz atmanın bir yolu olarak görülüyor. Die Leistung wurde
Kennzahlen Recall, Precision ve F1-Score'un avantajlarından yararlanın. En iyi Prototip
%99,85'lik bir Geri Çağırma, %94,10'luk bir Hassasiyet ve %10'luk bir F1 Skoru
%96,89'u, Vergleich zu ähnlichen Publicationen konkurrenzfähige Werte sind'deydi.
Die hohe Recall-Rate zeigt, dass der Prototyp in der Lage ist, Webserver-Zugriffsprotokollen zuverlässig zuverlässig zuverlässig'de anormallik.

Soyut

Web sunucuları, barındırılan web uygulamalarına yapılan erişimleri sürekli olarak kaydeder; bu, kritik bir uygulamadır.
Bankacılık gibi hassas verileri ve kısıtlı kaynakları işleyen uygulamaların bulunduğu sektörler.
Düzenli erişim kalıpları tipiktir, ancak kötü amaçlı etkinlikler özellikle düzensiz günlükler oluşturur.
Sistematik veri aramaları, güvenlik açıkları veya DoS saldırıları sırasında. Bu makale, bu tür anormallikleri tespit etmek için bir LSTM otomatik kodlayıcı kullanan üç prototipi tanıtmaktadır.
Prototipler, üretimden yaklaşık 14,5 milyon günlük kaydı üzerinde eğitildi
dokuz hafta boyunca web sunucusunda ayrıntılı manuel ön işleme gerçekleştirildi.
verilerin çıkarılması ve kategorilendirilmesini içeren günlükler. Daha sonra bunlar test edildi
Daha sonraki bir döneme ait, denetlenen ve rızaya dayalı saldırıların gerçekleştirildiği kayıtlar.
Performans, hatırlama, kesinlik ve F1 puanı ölçümleri kullanılarak değerlendirildi.
En iyi prototip %99,85 geri çağırma, %94,10 hassasiyet ve F1 puanı elde etti
%96,89'u mevcut yöntemlere kıyasla rekabetçi performans gösteriyor.
Yüksek geri çağırma oranı, prototipin anomalileri güvenilir bir şekilde tespit etme konusundaki güçlü yeteneğini gösterir web sunucusu erişim kayıtlarında.

İçindekiler

I Giriş

Bir web sunucusunun karşılaştığı çok sayıda siber saldırı nedeniyle,
Bu tür saldırıları tespit edin. Bu amaçla, web sunucuları barındırdıkları herhangi bir uygulamaya yapılan her erişim için günlükler oluşturur. Bu günlüklerin oluşturduğu veri hacmi,
Verileri analiz etmek ve potansiyel saldırıları tespit etmek için makine öğrenme tekniklerinin kullanılması. Bu çalışmada, anormalliklerin tanınmasını otomatikleştirmek için üç prototip geliştirilmiştir.
LSTM otomatik kodlayıcıya sahip bir web sunucusu tarafından oluşturulan erişim günlüklerine de erişebilirsiniz. Ayrıca, bu Makale, otokodlayıcıların iç işleyişine, anomali tespitine ve veri ön işlemeye genel bir konu olarak ve siber güvenlik bağlamında içgörü sağlamayı amaçlamaktadır.
Sinir ağları, LSTM ve oto kodlayıcılara ilişkin kısa bir giriş II. bölümde sunulmaktadır.
Prototipler VII. bölümde değerlendirilir ve en iyi performansı gösteren prototip seçilir.
Kullanılabilir bir programa dönüştürüldü. %99'un üzerindeki geri çağırma oranı, prototipin
Web sunucusu erişim kayıtlarındaki anormallikleri güvenilir bir şekilde tespit eder. Bu giriş, daha fazla bilgi içerir Bu makalede ele alınan saldırıların ayrıntılı tanımı ve genel bir bakış
Kullanılan günlükler. Ayrıca, söz konusu günlükler bir web sunucusundan kaynaklanmaktadır.
Janereporting'i çalıştırıyor [1]. Bölüm II.a, bu uygulamanın kısa bir açıklamasını sağlar temel arayüzlere ve kullanım durumlarına odaklanıyoruz.

Ia Siber Tehditler

İşletmelerin ve bireylerin gizliliği üzerinde büyük bir etkiye sahip olabilecek birçok siber tehdit türü vardır. İş uygulamalarının kullanılabilir kalması da genellikle hayati önem taşır. Tipik tehditler, sunucu güvenliğini kesintiye uğratan Hizmet Reddi (DoS) saldırılarından,
veya uygulama kullanılabilirliğinden, saldırganların yükseltilmiş ayrıcalıklar elde etmek için kullandıkları güvenlik açığı taramalarına kadar [2, 3]. Gatling [4], nmap [5] ve Gobuster [6] gibi araçlar, saldırganların sistemlerdeki güvenlik açıklarını tespit edebilir ve istismarları başlatabilir [7]. Bu tür saldırılar, uygulamaların normal kullanımına kıyasla anormal davranışları temsil eder. Anormallikler tespit edilebilirse sistemler, saldırılar hakkında sonuçlar çıkarabilir. Anormallik tespiti yalnızca
Normal davranıştan sapmaları tespit ederek, bilinmeyenler de dahil olmak üzere çoğu saldırıyı ortaya çıkarabilir. Anormallik tespiti saldırıları önleyemese de,
Bir sistemin saldırı altında olup olmadığını belirlemek ve saldırının etkisini vurgulamak için bilgi toplamak Siber savunmanın önemi.

Ib Çok Büyük Miktarda Veri

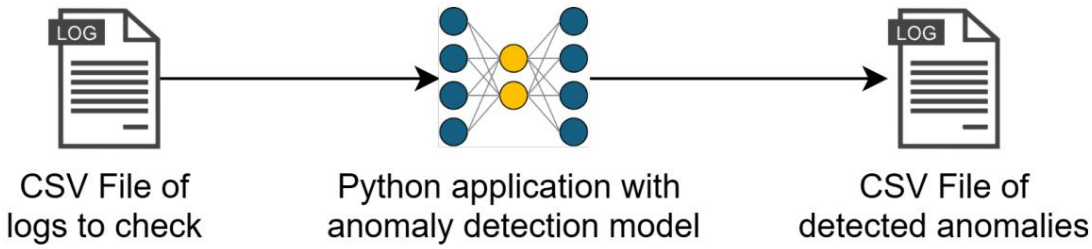
Bölüm II.a'da açıklanan Janereporting uygulamasının ortalama kullanımı şunu üretir:
Her gün yaklaşık 300.000 erişim kaydı tutulur. Erişim kayıtları, yapılan taleplerin kayıtlarıdır.
URL'ler, yanıt boyutları ve tablo 1'de gösterilen daha fazla özellik hakkında ayrıntılı bilgi içeren web sunucusu. Bu miktardaki veri, anormallikleri manuel olarak kontrol etme kapasitesini aşıyor ve anormallikleri belirlemek için otomatik kontrol gerekliliğiyle sonuçlanıyor.
Uygulama erişimlerindeki davranışlar.

Ic Varsayımları

Bu makale, Tablo 1'de açıklanan günlükleri analiz etmekte ve bir anomali tespit algoritması geliştirmektedir. Prototipin amacı, anomali tespit sorununu analiz etmektir.
Sabit bir eğitim ve test kümesindeki anomaliler. Mevcut eğitim ve test kümelerinde yeniden öğrenmenin otomasyonu

Verilerin güncellenmesi ve modelin periyodik olarak güncellenmesi bu makalenin kapsamı dışındadır. Daha fazla iyileştirmeye ihtiyaç duyulursa, eğitim gerekli değişikliklerle tekrar yapılabilir. Janereporting uygulaması ve web sunucusuna yalnızca dahili bir sunucudan erişilebilir ağ, eğitim ve doğrulama için kullanılan tüm günlüklerin güvenli ve ücretsiz olduğu varsayılmaktadır anomaliler. Bu varsayım, kayıtların tutulduğu dönemin bitiminden üç ay sonra herhangi bir saldırı olduğuna dair bir bulguya rastlanmaması gerçeğiyle desteklenmektedir.
Zararsız anomaliler sorunu VIII. bölümde daha ayrıntılı olarak ele alınmaktadır.
sınırlı veri alanlarına yönelik test vakaları DoS ve kaba kuvvet saldırılarıyla sınırlı olacaktır.
Daha geniş çeşitlilikte saldırılar için kullanıcı kimlikleri ve cihaz IP adresleri gibi alanlara ihtiyaç duyulmaktadır.
IP adresi, DDoS ve DoS saldırıları arasında ayrım yapmak için önemlidir.
LDAP gibi bir erişim rol modeliyle birleştirilmiş bir kullanıcı kimliği, ayrıcalık yükselmesinin tespit edilmesini sağlar.

Bu çalışmada geliştirilen program, günlük dosyalarını besleyerek manuel başlatma gerektirir CSV formatında, II.a bölümünde açıklanan özniteliklerle eşleştirilerek modele aktarılır.
Çıktı, anormal olarak tanımlanan tüm günlük kayıtlarını içeren bir CSV dosyası olacaktır. Bu akış Şekil 1'de gösterilmiştir.



Şekil 1: Anomali tespitinin giriş ve çıkış gereksinimleri.

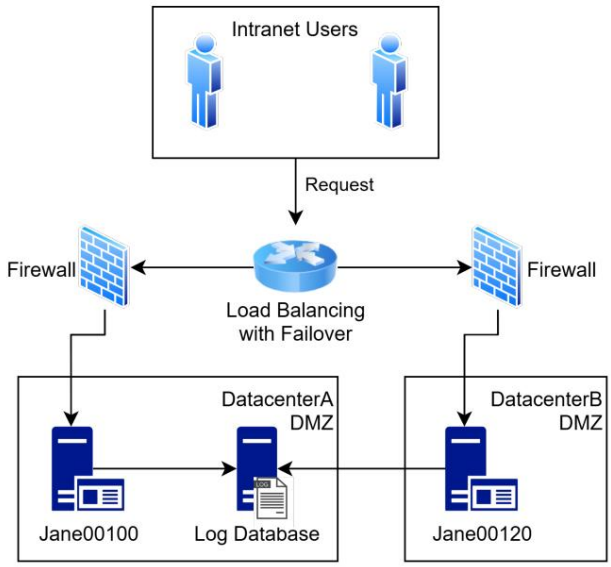
II Arka Plan

Bu bölüm, bu makalenin içeriğini anlamak için gerekli olan makine öğrenimi modelleri için uygulamaya özgü ve teknik temelleri özetlemektedir. Bölüm II.a, uygulamaya, uygulamayla ilişkili ve prototiplerin eğitimi için kullanılan erişim günlüklerine genel bir bakış sunmaktadır. Bölüm II.b, bu makalede geliştirilen anomali tespit modelleri için temel teknolojiyi anlamanıza yardımcı olan teknik bir derinlemesine incelemedir.

II.a Janereporting

Janereporting, Next Stride AG [8] tarafından geliştirilen bir iş uygulaması olan Jane'in [1] bir parçası olan bir uygulamadır . Bu makalede, gerçek, simüle edilmemiş uygulama kullanımına dayalı erişim günlüklerine örnek olarak Janereporting günlükleri kullanılmaktadır. Janereporting , çok çeşitli verilerin grafiksel olarak görselleştirilmesi için tasarlanmış bir araçtır. Toplanan günlükleri anlamak için uygulamaya ilişkin ayrıntılı bilgilere ihtiyaç duyulmaz, çünkü bunlar uygulamaya web erişimini temsil eder.

Şekil 2, bu makaledeki günlüklerin nasıl oluşturulduğunu görselleştirmektedir. Müşteri tarafında, çalışan bir Janereporting örneğine sahip iki fiziksel uygulama sunucusu bulunmaktadır. Bu sunucular iki farklı veri merkezinde bulunmaktadır. Sunucuya gelen istekler bir yük devretme işleyicisi tarafından işlenecektir. Yük devretme işleyicisi varsayılan olarak istekleri Jane00100 örneğine gönderir . Jane00100 örneğin sistem güncellemeleri veya yoğun trafik nedeniyle kullanılamıyorsa, istek Jane00120 örneğine gönderilir. Her iki sunucuya da yalnızca dahili bir ağ üzerinden erişilebilir. Bir sunucuyla ilişkili erişim günlükleri, sunucunun kendisinde saklanır ve düzenli olarak günlük veritabanına yazılır.



Şekil 2: Müşterinin uygulamasının mimarisi erişim.

Uygulama erişimi için kaydedilen öznitelikler tablo 1'de listelenmiştir. Bu günlüklerin tamamı web erişimine dayanmaktadır ve öncelikli olarak HTTP protokolünün ortak özniteliklerini içerir.

Bağlanmak	Açıklama Hangi	Örnekler •
AJAN	yazılımın janereporting'e istekte bulunduğuna dair bilgi.	Wget/1.21.2 • Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, Gecko gibi) Chrome/ 119.0.0.0 Safari/537.36 Edg/119.0.0.0

HTTP_ANA BİLGİSAYAR	Talep edilen sunucuya çoğunlukla alan adı üzerinden erişiliyor.	• jane-prod.intranet.customer.com • 10.10.100.101
YÖNTEM	Gerçekleştirilen HTTP Metodu.	• GET • POST • /
YOL	Parametreli istenen yol .	janereporting/janereporting/miscService • /janereporting/ janereporting/userServlet ?_operationType=getCurrentUser • janereporting
PATH_CONTEXT URL'nin ilk yolu, Janereporting	uygulamasının bağlamında her zaman 'janereporting'dir.	
KISA_YOL	Parametresiz kısaltılmış istenen yol .	• janereporting/janereporting/miscService • janereporting/ janereporting/userServlet
REFERANS	İsteği başlatan sitenin web adresi.	• https://jane-prod.intranet.customer.com/janereporting/ • https://jane-prod.intranet.customer.com/janereporting/outputServlet? 86F2B0CF693C491DBA9B28E0762403C3/ eoqcbMotIWTy8uv1yvy1.html •
REMOTE_HOST IP adresiyle erişilen istenen sunucu.		10.10.100.101 • 10.10.101.101
RESPONSE_CODE Sunucunun yanıt kodu.		• 200 • 300
SUNUCU	Hangi sunucuya hitap edildi? rica etmek.	• jane00100 • jane00120
KISA_LOG_ DOSYA ADI	Sunucudaki logların dosya adı kısaltılmış hali. https veya http olabilir.	• ns_proxy__https • ns_proxy__http
BOYUT	Yanıtın bayt cinsinden boyutu.	• 45 • 424
KAYNAK_DOSYASI	Sunucudaki logların dosya adı. https veya http.	• _proxy__https-access.log • _proxy__http-access.log
KAYNAK_TS	İsteğin zaman damgası. YYYYAAGG-SSHddssSSS biçimindedir. Son üç hane (ms) her zaman 0'dır.	• 20240225134834000 • 20240313090557000

Tablo 1: Janereporting uygulamasına erişim için toplanan ve anormallik tespiti için kullanılabilen Günlük Nitelikleri.

Aşağıdaki öznitelikler anomali tespit modeli için kullanılmadı ve bu nedenle doğrudan bırakıldı.

- PATH_CONTEXT: Her zaman aynı giriş janereporting çünkü bu, bu makalede ele alınan uygulamanın günlükleri için anahtardır.
- REMOTE_HOST: IP'ler, 'HTTP_HOST' özniteliğinin etki alanı adlarına birebir eşlenir . DNS, IP adreslerinden daha kararlı olduğundan, bu öznitelik kaldırıldı.
- SOURCE_FILE: Bu öznitelik, 'SHORT_LOG_' ile aynı bilgilere sahiptir DOSYA ADI'.

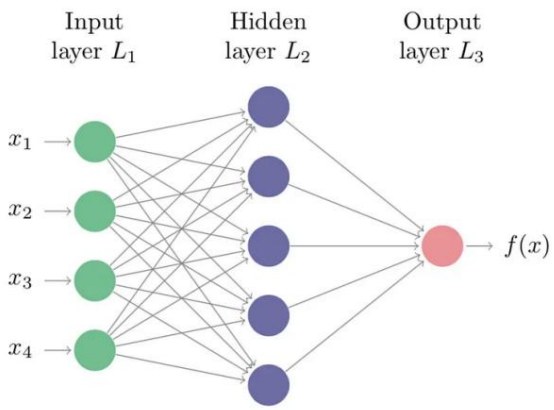
Listelenen diğer tüm öznitelikler, anomali tespit algoritmasına aktarılmak üzere farklı alanlara ayrılmıştır. Bu özniteliklerin nasıl yorumlandığı ve kullanıldığı Va bölümünde açıklanmıştır.

II.b Teknik Bilgi

Bu makalede, belirli günlük dosyalarındaki anormallikleri tespit etmek için bir LSTM otokodlayıcı kullanılmıştır. Otokodlayıcı, bir sinir ağının özel bir biçimidir; LSTM, Uzun Kısa Süreli Bellek anlamına gelir. Tüm bu terimler bu bölümde daha ayrıntılı olarak açıklanacaktır. Bu makalede kullanılan temel teori, kısmen Mark Cieliebak'ın 2022'deki makine öğrenimi ve veri madenciliği dersinden, özellikle L09 [9] ve L10 [10] derslerinden ve kısmen de Michael Nielsen'in Sinir Ağları ve Derin Öğrenme [11] kitabından alınmıştır.

II.b.1 Sinir Ağı

Bir sinir ağı, bir giriş katmanı, keyfi sayıda gizli katman ve bir çıkış katmanından oluşur. Her katman, belirli sayıda girişe, bir aktivasyon fonksiyonuna ve bir çıkışa sahip en az bir yapay nörondan oluşur. Basit bir örnek Şekil 3'te gösterilmiştir.



Şekil 3: Tek gizli katmana sahip sinir ağı diyagramı. [12]

"En basit yapay nöron, eşik mantık birimi veya TLU'dur. Temel işlevi, girdilerinin ağırlıklı bir toplamını gerçekleştirmek ve bu toplam bir eşiği aşarsa "1", aşarsa "0" çıktısını vermektir. TLU'nun, gerçek nöronların temel "entegre et ve ateşle" mekanizmasını modellemesi beklenir." [13]

Aktivasyon fonksiyonu, TLU'yu ikili bir yapıdan daha biyolojik bir yapıya dönüştürür ve teorik olarak herhangi bir reel değerli fonksiyon olabilir. Bu makalede, IV.b bölümünde açıklandığı gibi , kullanılan tek aktivasyon fonksiyonu tanh'dir. x1 ve x2 girişlerine, tanh aktivasyon fonksiyonuna , b önyargısına ve w1 ve w2 ağırlıklarına sahip bir nöronun çıktısı verilmiştir.
gibi

$$y = \tanh(w1 \quad x1 + w2 \quad x2 + b). \tag{1}$$

Daha genel olarak, N girişli nöron q için çıkış yq şudur:

$$yq = \tanh \sum_{i=0}^{N} (wi \quad xi) + bq \; . \tag{2}$$

Giriş katmanı için çıktı, doğrudan y = x olarak verilir , çünkü giriş katmanı harici verilere arayüz oluşturur ve dolayısıyla işlenecek verilerin boyutlarını tanımlar. Parametresi yoktur ve gerçekte uygulanmış olmaktan ziyade bir soyutlamadır. Gizli katmanların her biri ve çıkış katmanı, bir önceki katmanın çıktılarını girdi olarak alır. Basitlik açısından, burada tamamen bağlı bir ağ ele alınmıştır. Bu nedenle, Lp katmanının çıktısı şu şekilde verilir:

$$y \quad p \sum_{= q=0}^{Np} \tanh \sum_{i=0}^{N\,p\;\;1} (wi \quad xi) + bq \cdot eq \; , \tag{3}$$

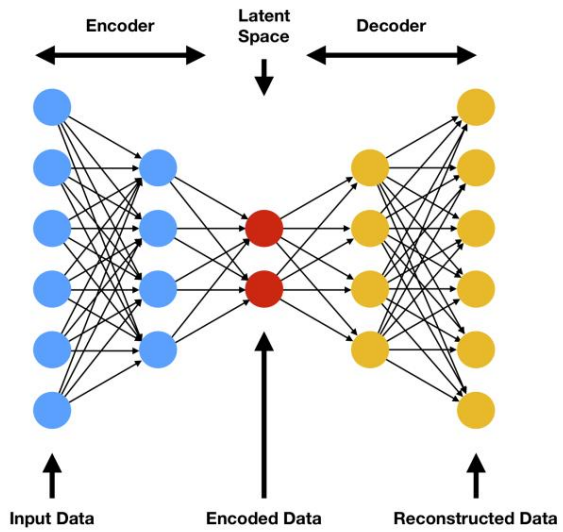
Burada x = yp 1 ve eq, doğru vektör uzayında skaler değerleri elde etmek için kullanılan birim vektörlerdir. Böyle bir ağı eğitmek için bir kayıp fonksiyonu tanımlanır ve ardından ağırlıkları ve önyargıları güncellemek için geri yayılım kullanılır [10].

> "Kayıp Fonksiyonu, bir sinir ağının eğitim süreci sırasında verilen örnekler için yaptığı hatayı ölçer." [9]

II.b.2 Otokoder

Oto kodlayıcı, çıkış katmanı Lp'nin giriş katmanı L1 ile aynı boyutta olduğu sinir ağının özel bir durumudur , bkz. Şekil 4. Adından da anlaşılacağı gibi, oto kodlayıcı bir giriş kaydını daha düşük boyutlu bir gösterime kodlar ve kullanım amacı, bu düşük boyutlu gösterimden çıkış katmanındaki giriş değerlerini yeniden oluşturmaktır [15]. Bu nedenle, bir oto kodlayıcı gözetimsiz olarak eğitilebilir. Anomali tespiti için, bir oto kodlayıcının diğer makine öğrenimi algoritmalarına kıyasla belirli bir avantajı vardır. Oto kodlayıcı, kendisine hangi girdinin verildiğini öğrenerek girdiyi kopyalamayı öğrenir.

Daha önce hiç görmediği bir girdi verilirse,



Şekil 4: Bir oto kodlayıcı, bir kodlayıcı ve bir kod çözücü parçadan oluşur, orta katman ikisi arasında bir arayüz görevi görür ve bu orta katmandaki 7 veri gizli bir alanda temsil edilir. [14]

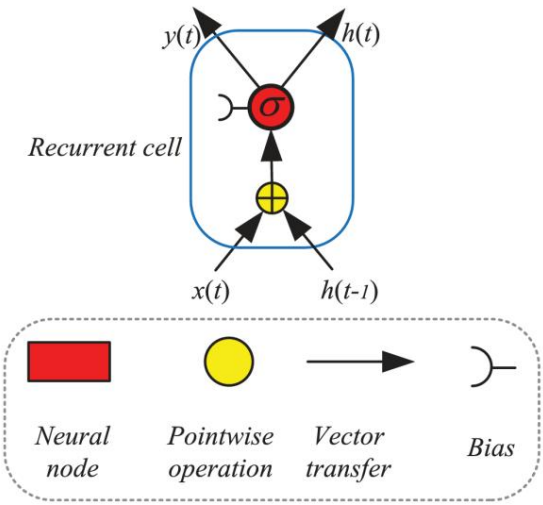
oto kodlayıcı bu girdiyi kopyalamayı başaramaz ve
bu da büyük değerlerle sonuçlanır

$$f(yp, y1) = y \quad p \quad y \quad 1 . \qquad (4)$$

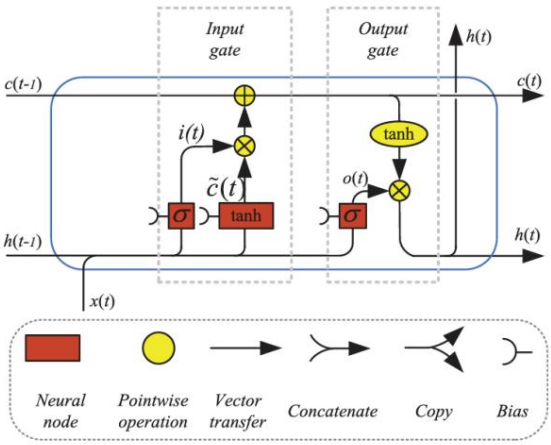
Denklem (4)'ü bir kayıp fonksiyonu olarak
kullanarak, bu özellik eğitilir ve dolayısıyla güçlendirilir. Dahası, otokodlayıcı herhangi bir anomali içermeyen bir veri kümesi üzerinde eğitilirse, tanım gereği, otokodlayıcının daha önce hiç görmediği bir anomali girdisi olur ve bu da f (yp, y1) için nispeten büyük bir değer verir.
Böylece, bir otokodlayıcı, neyin sıra dışı olduğunu tanımlama sorununu, bir kaydın ne kadar sıra dışı olduğunu bulmaya dönüştürür . Eğitimden sonra, otokodlayıcı, kaydın anormal kabul edilip edilmeyeceğine karar vermek için basit bir eşik parametresiyle karşılaştırılabilecek bir metrik tanımlar.

II.b.3 lstm

Günlük dosyaları incelenirken, tek bir günlük girişi genellikle yalnızca az miktarda bilgi içerir. Bu nedenle, bir mühendis hata ayıklama veya anormallik arama sırasında, sıranın veya herhangi bir sıralamanın anormallik içerip içermediğini görmek için ardışık birkaç günlük girişine bakar.



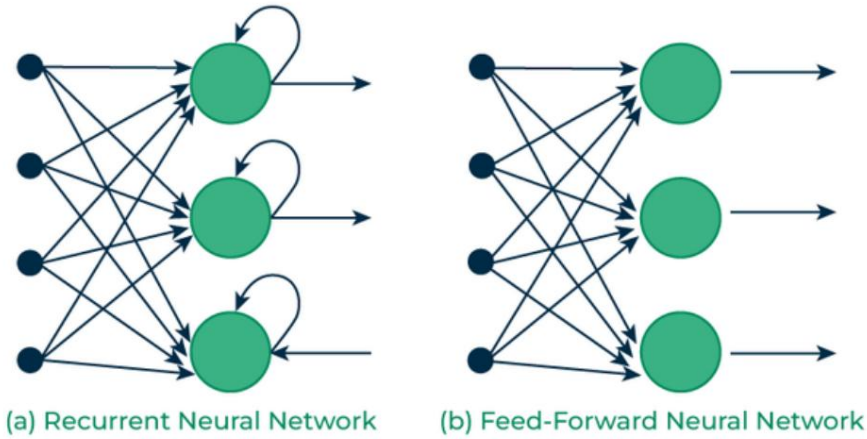
Şekil 5: Tekrarlayan bir hücre, iki ayrı çıktı gösterir: y(t), bir sonraki katmana giden çıktı ve h(t), aynı katmanın bir sonraki tekrarlayan hücresine giden çıktı. y(t) ve h(t) aynı bilgiyi içerir, ancak fark, temel yön değişikliğinde yatar. Daha sonraki bir zaman adımında aynı katmandaki başka bir hücreye devredilmesi, katmanın bilgiyi hatırlamasını sağlar [16].



Şekil 6: Hochreiter ve Schmidhuber[17] tarafından önerilen bir LSTM hücresi, Şekil 5'te [16] gösterilen standart tekrarlayan hücreden daha iyi performans göstermektedir.

Bu davranışı yansıtmak ve bir sinir ağının önceki kayıtları hatırlamasını sağlamak için uzun kısa süreli bellek veya LSTM sinir ağları geliştirilmiştir [18].
LSTM ağları, RNN'ler olarak adlandırılan tekrarlayan sinir ağlarının bir çeşididir ve şunlardan oluşur:

LSTM hücreleri, bkz. Şekil 6, yapay nöronlar yerine veya yapay nöronlara ek olarak kullanılır. LSTM hücresini anlamak için, öncelikle dayandığı temel prensip olan Şekil 5'te gösterilen yinelemeli hücreyi anlamak gerekir. Yinelemeli hücre, sinir ağını ileri beslemeli bir ağdan, geri bildirim döngüleri içeren yinelemeli bir ağa dönüştürür. Bu fark Şekil 7'de de gösterilmiştir. Geri bildirim döngüleri, RNN'nin ağ üzerinden geçirilen kayıtlar arasında bir durumu kaydetmesini sağlar ve bu da verilerdeki diziler hakkında bilgi içeren bir duruma yol açar. Şekil 6'da gösterilen LSTM hücresi, hücreye bir kapı ekleyerek standart yinelemeli hücreye kıyasla hatırlama kapasitesini artırır. [16]



Şekil 7: Tekrarlayan sinir ağı (RNN) ile ileri beslemeli sinir ağı (FFNN) arasındaki fark, RNN'nin geri besleme döngüleri içermesi ve FFNN'nin içermemesidir. [19]

III İlgili Çalışma

Makine öğrenmesi ile anomali tespiti konusunda çok sayıda yayın bulunmaktadır.
Birçoğu modellerinin performansını ölçmek için doğruluk ve F1 puanını kullanır.
Kwon ve diğerlerinin [20] deneysel çalışması, farklı evrişim sinir ağı (CNN) mimarilerini tam bağlı sinir ağları (FCN), varyasyonel otomatik kodlayıcı (VAE) ve LSTM'li diziden diziye yapı (Seq2Seq-LSTM) ile karşılaştırır.

Çalışmaları, bu modellerin üç farklı kamu trafiğinde karşılaştırılmasına odaklanıyor
Ancak bu makale, özel bir veri kümesindeki LSTM otokodlayıcılarına odaklanmaktadır.

> "Sığ CNN modelinin zaman zaman VAE'den daha iyi performans gösterdiğini gözlemledik modeller, ancak FCN ve Seq2Seq-LSTM'den daha iyi çalışmıyor."[20]

Le ve Zhang [21], NeuralLog'u ayrıştırmak için kullanarak günlüklerin ön işlenmesini otomatikleştirdi
Günlükler anlamsal vektörlere dönüştürülür. NeuralLog, %95'ten yüksek bir F1 puanına ulaşır. Günlükler
Söz konusu makalede ele alınanların, makine tarafından üretildikleri ve ifadeler veya benzer dilsel yapılar içermeleri amaçlanmadığı için anlamsal bir anlamı yoktur. He ve diğerlerinin [22] deneyim raporu, yaygın olarak kullanılan anomalilerin ayrıntılı bir incelemesini içerir.
Tespit yöntemleri arasında, gözetimsiz öğrenme yöntemlerinin gözetimli yöntemlere göre daha düşük performans gösterdiği yönünde genel bir eğilim bulunmaktadır. Yine, kamuya açık veri kümeleri
Farklı yöntemleri karşılaştırmak için kullanıldı. Çalışmalarının amacı, bu yöntemlerin benimsenmesi için kılavuzlar sağlamaktır. Bu çalışmada geliştirilen prototipin tasarımı için
Makalede, onların rehberleri dikkate alınmakta ve referans olarak kullanılmaktadır.

> "Denetimsiz yöntemler genellikle denetlenen yöntemlere göre daha düşük performans gösterir."[22]

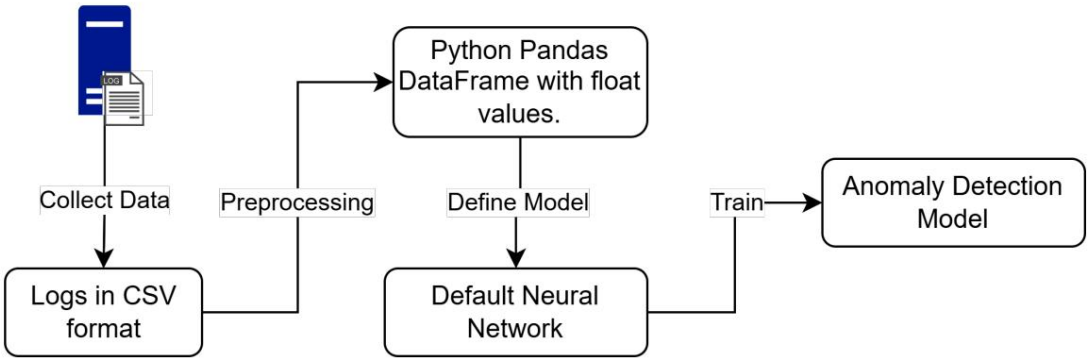
Lu ve diğerleri [23], bir CNN mimarisini çok katmanlı algılayıcı (MLP) ve bir
LSTM sinir ağı. CNN'leri için %99'un üzerinde doğruluk puanları ve sırasıyla %97,7 ve %99,3 hassasiyet ve geri çağırma değerleri buldular. Çalışmalarındaki MLP
%98,1 hassasiyet ve %98,0 geri çağırma oranına ulaşırken, kullandıkları LSTM %95 geri çağırma oranına ulaştı ve hassasiyet [24]. Bu, MLP ve LSTM'yi birleştirmek için ikna edici bir argüman sağlar
Bu çalışmada oto kodlayıcı kullanılarak uygulanan mimariler.
Oto kodlayıcı, özünde, bir MLP'nin geliştirilmiş bir versiyonudur. Lindemann ve diğerleri [18]
anomali tespitine yönelik çeşitli farklı yaklaşımların bir karşılaştırmasını gösterir, buna şunlar dahildir:
LSTM teknolojisini ve LSTM'nin oto kodlayıcılarla kombinasyonlarını kullanan ağlar ve
VAE. Ancak, TABLO 1'leri , LSTM oto kodlayıcısının karşılaştırmalardan yoksun olduğunu gösteriyor
Diğer yöntemlere kıyasla en yakın ilişkili yöntem LSTM'dir.
VAE, destek vektör makinelerinden (SVM) ve oto kodlayıcıdan daha iyi performans gösteriyor
LSTM. Catillo ve diğerleri tarafından geliştirilen oto kodlayıcı [15], aralıkta bir geri çağırma gerçekleştirir
%96 ila %99. Nassif ve diğerleri [25], araştırmaların güncel verileri kullanmasını ve özellik seçimi ve çıkarma türünü analiz ettikleri makalelerden daha ayrıntılı olarak tanımlamasını öneriyor
2000-2020 yılları arasındaki dönemi kapsamaktadır. Bu belge, onların önerilerini takip etmeyi ve özellik seçiminin daha ayrıntılı bir açıklamasını sunmayı amaçlamaktadır. Ayrıca, Patcha ve Park
[26] Saldırıları tespit etmenin bir yolu olarak anormallik tespitinin önemini vurgulamaktadır
Daha önce görülen, sıfırıncı gün saldırıları olarak adlandırılan.

IV Tasarım

Bu bölüm, anomali tespiti için gereklilikleri ele alıyor ve iyi performans gösteren bir algoritma için gerekli tüm ön koşulları ve kararları özetliyor.

IV.a Ham Kayıtlardan Anomali Algılama Modeline

Şekil 8, anomali tespit modelinin geliştirme sürecini göstermektedir. İlk olarak,
Tüm eğitim verileri gereklidir. Bunlar sunucudan toplanır ve bir csv dosyasında saklanır.
Tablo 1'de açıklanan günlük içeriğine sahip dosya. Sinir ağları bir giriş katmanına sahiptir
ve her düğümün girdi olarak bir kayan nokta değerine ihtiyacı vardır. Daha fazla teorik bilgi şurada bulunabilir:
Bölüm II.b.1.



Şekil 8: Anomali tespit modelinin oluşturulmasına ilişkin genel prosedür.

Metinsel logaritmik verilerden bu sayısal girdiyi elde etmek için ön işleme gereklidir.
Le ve Zhang [21] tarafından sunulan yaklaşım, diğer modellerle otomatik belirteçleme ve
sınıflandırmayı kullanır. Ancak, bu makale için manuel özellik çıkarma
Hangi alanların hangi kategorilere ön işleme tabi tutulacağı konusunda en fazla kontrolü elde etmek
için seçilmiştir. Ön işleme, Va bölümünde ayrıntılı olarak açıklanmıştır . Veriler artık temsil edildiğinden,
Sadece sayısal değerler halinde, bunları bir makine öğrenmesi algoritmasına aktarmak mümkündür.
Modelin giriş katmanının, önceden işlenmiş verilerin boyutuna uyması gerekir. Daha fazla ayrıntı
Sinir ağının yapısı IV.b ve Vb bölümlerinde verilmiştir. Modelin tanımlanmasından sonra
mimaride eğitim süreci başlar. Bu, mimarinin ağırlıklarını ve önyargılarını düzeltecektir.
nöronlar (bölüm II.b.1) ve belirli bir eğitim döngüsü (dönem) miktarından sonra,
model tamamlandı.

IV.b Makine Öğrenmesi Algoritması

Anormal kayıtları bulmak için taranması gereken veri miktarı, anormal olanın ne olduğunu öğrenme
göreviyle birleştiğinde, bir makine öğrenimi algoritmasının uygulanmasını gerektirir. Bu durumda
hangi tür algoritmanın kullanılacağına karar vermek için
prototip için çeşitli kaynaklar dikkate alındı [15, 18, 20, 21, 22, 23, 24, 25, 26], bkz. bölüm III. Analiz,
çeşitli makine öğrenimi algoritmalarının kullanılabileceğini gösterdi
Anormallikleri tespit etmek için Destek Vektör Makineleri (SVM) gibi iyi bilinen algoritmalar
ve karar ağaçları, tek sınıf problemlerini çözmek için özel olarak uyarlanmalıdır.
anomali tespiti olarak. SVM için bu, tek sınıflı SVM olarak adlandırılan bir sonuçla sonuçlanır. Ayrıca,
analiz, LSTM ile MLP'nin veya bir LSTM'nin bir kombinasyonunun

Otokodlayıcı daha ileri araştırmalar için umut verici bir yöndür.

Otokodlayıcılar, normal günlük kayıtlarının örüntülerini kendileri öğrenir. Bilinmeyen günlük kayıtlarının yeniden yapılandırılması daha büyük hatalara yol açtığından, bu günlükler aykırı değerler gibi davranır ve anomali olarak işaretlenebilir. Bu nedenle, etiketlenmiş verilere ihtiyaç duyulmaz ve model, gözetimsiz öğrenme kullanılarak eğitilebilir. Bu, herhangi bir değerlendirme sürecine dahil edilmeyen anomalilerin tespit edilmesini sağlayarak bilinmeyen saldırıların belirlenmesine yardımcı olur. Dahası, tek başına bir otokodlayıcı yalnızca tek bir günlük girişini analiz edebilir. Günlükler çoğunlukla birbirine bağımlı olduğundan ve toplu ve bağlamsal anomaliler oluşturduğundan, bunları sırayla değerlendirmek gerekir.

> "Yapılan çalışma, farklı LSTM ağ mimarilerinin mevcut olduğunu ve toplu ve bağlamsal anomaliler gibi çeşitli karmaşık anomalileri hassas bir şekilde tespit edebildiğini göstermektedir."[18]

LSTM hücrelerinin yardımıyla, bellekte birden fazla kaydı bir dizi halinde tutmak ve aralarında bağlantılar oluşturmak mümkündür. Yukarıdaki analizin bir sonucu olarak, bu makale bir LSTM otokodlayıcısı ile yaklaşıma odaklanmaktadır. Aktivasyon fonksiyonunun seçimi (bkz. bölüm II.b.1), genellikle prototiplerin eğitileceği donanıma bağlıdır. Bu çalışma sırasında kullanılan donanım, LSTM ağlarında aktivasyon fonksiyonu seçeneklerini kısıtlar ve hızlı yürütme için desteklenen tek donanım tanh'dir [27].

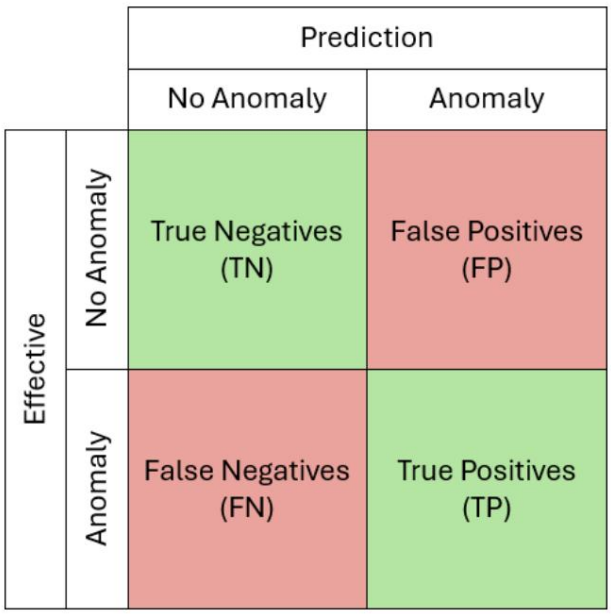
IV.c Kayıp Fonksiyonu

Bu makale, anomalileri bulmak için bir prototip oluşturmaya odaklandığından ve optimal kayıp fonksiyonlarına odaklanmadığından, en iyi uygulama ve en çok kullanılan kayıp fonksiyonları üzerine yapılan araştırmalar, en temel iki metrik olan ortalama karesel hata (MSE) ve ortalama mutlak hata (MAE) ile sınırlıdır. MSE, büyük hataları daha şiddetli cezalandırma avantajına sahipken, MAE daha esnektir. Öte yandan, MSE'nin karesel yapısı nedeniyle MAE'ye göre patlama ve yok olma riskine daha açıktır. MSE'nin aksine, MAE'nin esnek yapısı, eğitim sürecinin daha yavaş yakınsamasına ve daha fazla zaman alan deneylere yol açar. Eğitim süresi bu çalışmada bir sorun olmadığından, daha kararlı olması nedeniyle MAE seçilmiştir.

IV.d Değerlendirme

Bu makalede kullanılan test seti, ardışık altı günün günlük verilerinden oluşmaktadır. Bu günlerden ikisinde, gözetim altında birkaç saldırı gerçekleştirilmiştir (bkz. Bölüm VI) ve ilgili günlükler etiketlenebilir. Bir prototipin performansını değerlendirmek için, çıktısı Şekil 9'da gösterilen dört değişken kullanılarak karşılaştırılır. Gerçek pozitifler (TP), gerçek anomaliler olan ve model tarafından doğru bir şekilde tahmin edilen değerlerdir. Model tarafından tespit edilemeyen anomalilere ise yanlış negatifler (FN) denir.
Gerçek negatifler (TN), anomali olmadığı doğru bir şekilde etiketlenen günlük girdileridir; yanlış pozitifler (FP), anomali olarak yanlış bir şekilde etiketlenen meşru günlüklerdir.
Prototip için belirlenen birincil hedef, FN'yi en aza indirmektir. İkincil hedef ise FP'yi en aza indirmektir. Teknik terimlerle, bu hedeflere geri çağırma ve kesinlik denir.
Bu bağlamda geri çağırma , anormal olarak doğru şekilde etiketlenen günlüklerin oranını ölçer

Prototip (TP), etkili bir şekilde anormal olan toplam günlük sayısına oranını ölçer, bkz. denklem (5).
Öte yandan, hassasiyet , TP'nin tahmin edilen toplam günlük sayısına oranını ölçer
Prototipe göre anormal, bkz. denklem (6). F1 puanı, hassasiyeti esas alan bir ölçümdür
ve dikkate alınması gerekenleri göz önünde bulundurun, bkz. denklem (7). Genellikle modelleri karşılaştırmak için kullanılır
Çeşitli veri kümeleri, özellikle çok sınıflı kategorizasyon problemlerinde, hassasiyet ve geri çağırmayı
tek bir sayıda birleştirdiği için. Tüm bu metrikler 0 ile 0 arasında değerler üretir.
1, burada 1 istenen değerdir.



Şekil 9: Karışıklığın genel kavramı matris.

$$\text{hatırlama} = \frac{TP}{TP+FN} \quad (5)$$

$$\text{hassasiyet} = \frac{TP}{TP+FP} \quad (6)$$

$$\text{F1 puanı} = 2 \cdot \frac{\text{hassas geri çağırma}}{\text{hassasiyet+geri çağırma}} \quad (7)$$

Odaklanılacak metrik, şunlara bağlıdır:
Aranan tespit türü için
anomali tespiti, tanımlanması için çok önemlidir
mümkün olduğunca çok sayıda etkili anomali,
Bu da yüksek bir geri çağırma anlamına gelir.
Hassasiyet, normal günlüklerin kaçının
aslında anormallikler olarak işaretlendi.
Hassasiyeti artırmak da ikinci bir adım olarak önemlidir
hedef, çok fazla günlük işaretlemek olduğu için
anomaliler manuel çabayı artırır

F1 puanı , yüksek geri çağırmayı hedeflerken hassasiyetin de önemli olduğunu garanti eder .
Kabul edilemez seviyelere düşmez. Ayrıca, piyasadaki çeşitli modellerin performansını karşılaştırmak
için tek bir değer sağlar. Ancak, bu kullanım durumu için geri çağırma hala önemlidir.
en kritik metrik.

V Uygulaması

Bu bölüm, günlüklerin ön işlemesinin, atoen kodlayıcı prototiplerinin ve test kurulumunun uygulanmasını açıklamaktadır. Etkili uygulama için kullanılan dil, numpy [28], pandas [29], keras [30] ve tensor-flow [31] çerçeveleriyle birlikte Python'dur. Çizim için matplotlib.pyplot [32] kullanılmıştır.

İlk olarak, eğitim verileri Python çekirdeğine pandas.DataFrame olarak aktarılır .
Birden fazla CSV dosyası. Ön işleme başlamadan önce, dosyalar 'SOURCE_TS' alanına göre sıralanır. Ön işlemeden sonra, otomatik kodlayıcı aşağıda açıklandığı gibi tanımlanır ve eğitilir.
Bölüm Vb Eğitilmiş oto kodlayıcı modeli daha sonra anomalileri tespit etmek ve yerleştirmek için kullanılır İlgili günlük girişleri bir çıktı dosyasına aktarılır. Çıktı, bir test verisinden üretilir.
set, oto kodlayıcının performansı Vd bölümünde ayrıntılı olarak açıklandığı gibi değerlendirilebilir

Va Ön İşleme

Bu bölüm, günlüklerin her bir niteliğini ve bunların ayrıntılı olarak nasıl işlendiğini ele almaktadır.
Orijinal özniteliklerin bir veya daha fazla işlenmiş özniteliğe farklı şekilde dönüştürülmesi
ent isimleri.

Va1 KAYNAK_TS

'SOURCE_TS' niteliği, YYYYAAGGDHHddSSsss biçiminde bir zaman damgasını temsil eden on yedi basamaklı bir sayı içerir . Bunu doğrudan tam sayı olarak yorumlamak, 60 ile 99 arasındaki değerlerin eksik olması gibi çeşitli sorunlara yol açtığından,
Dakika ve saniye için rakamlar içeren bu orijinal öznitelik, on işlenmiş özniteliğe dönüştürülür. Bunlardan ilki "hafta içi" olarak adlandırılır ve 0 ile 100 arasında bir kayan nokta değeri içerir.
ve 1, burada 0 pazartesi sabahı saat 12'yi, 0,994 ise pazar akşamı saat 23:00'ü ifade eder.
Belirli bir tarihin hangi hafta gününü temsil ettiğinin hesaplanması veya aranması,
Python kütüphanesi datetime. İşlenen ikinci öznitelik 'is_holiday' olarak adlandırılır. Tarih, cumartesi, pazar veya ilgili bölgedeki bir tatili temsil ediyorsa 1 içerir.
Müşteri. Bu makale için tatil listesi elle derlenmiş ve sabit kodlanmıştır.
model. Üçüncü ila onuncu işlenmiş nitelikler, bir kova temsilidir
Günün saati, üst üste binen dört saatlik adımlar halinde. Örneğin, 'h0-3' niteliği 0 ile 1 arasında bir kayan nokta değeri içerir; burada 0, 00:00:00.000 anlamına gelir ve 1, 04:00:00:000 anlamına gelir.
Bu nedenle, 'SOURCE_TS' 20240327005959000 olan bir günlük girişi 'weekday' = değerini alır
0,2857, 'holiday' = 0, 'h0-3' = 0, 'h3-6' = 0,2499, 'h6-9' = 'h9-12' = = 'h18-21' = 0, ⋯
ve 'h21-0' = 0,9999.

Va2 SUNUCUSU

'SERVER' niteliği jane00100 veya jane00120 içerdiğinden , şu şekilde işlenir:
jane00100 veya jane00120 yerine sırasıyla 0 ve 1 içerir.

Va3 HTTP_HOST

'HTTP_HOST' niteliği altı farklı değerden birini içerebilir, bunlar şunlardır:
'10.10.100.101', '10.10.101.101', 'jane-prod.intranet.customer.com', 's100-jane-prod.intranet.
customer.com', 's102-jane-prod.intranet.customer.com' ve 'sv243554.zit.customer.com'. Bu

Bu, müşterinin Janereporting uygulamasına erişim biçiminden ve Şekil 2'de belirtilen devralma işleyicisini kullanan mimariden kaynaklanmaktadır . Bu değerlere sıralama koyma riskini azaltmak için tek sıcak kodlama kullanılır. Böylece, orijinal öznitelik, her biri olası değerlerden birini temsil eden altı işlenmiş özniteliğe dönüştürülür.

Va4 YÖNTEMİ

Web sunucusuna HTTP üzerinden erişildiğinden, istekler her zaman bir HTTP yöntemi kullanılarak yapılır. Bu yöntem 'METHOD' özniteliğinde bulunur. Janereporting uygulaması bir POST veya GET isteği gerektirdiğinden, 'GET' değeri 0'a, 'POST' değeri ise 1'e eşlenir.

Va5 AJANI

'AGENT' özniteliğinde, isteği oluşturmak için hangi tarayıcı veya aracın kullanıldığına dair bilgi bulunur. Bu, Mozilla Firefox, Microsoft Edge veya Google Chrome gibi herhangi bir tarayıcı veya wget gibi bir komut satırı aracı olabilir . Üreticilerin yayınladığı her güncellemeyle değişebileceği ve uygulamanın kullanımı üzerinde gerçek bir etkisi olmadığı için, kullanılan tarayıcıların tam sürümü hakkındaki bilgilerin kaldırılması yönünde bir tasarım tercihi yapılmıştır. Ayrıca, bir müşterinin Chrome, Safari veya başka bir tarayıcı kullanıp kullanmadığı göz ardı edilir ve 'AGENT' özniteliğinde geriye kalan tek bilgi, bir komut satırı aracının kullanılıp kullanılmadığıdır. Bu, bir komut satırı aracı kullanılmışsa 1, kullanılmamışsa 0 içeren işlenmiş 'IS_WGET' özniteliğiyle sonuçlanır.

Va6 YANIT_KODU

'RESPONSE_CODE' niteliği, HTTP yanıt kodunu ifade eder; örneğin 200 , OK anlamına gelir . Eğitim setinde bulunan değerler 200, 302, 304, 301, 500 ve 408'dir. Bu nedenle ve 301 ve 302 gibi benzer kodlarla ilgili karışıklıkları azaltmak için, işlenmiş nitelikler olan '2xx', '200', '3xx', '304', '4xx' ve '5xx' hesaplanır. 200 ve 304'ün özel durumları, sırasıyla 2xx ve 3xx'e göre önceliklidir .

Va7 LOG_DOSYASI

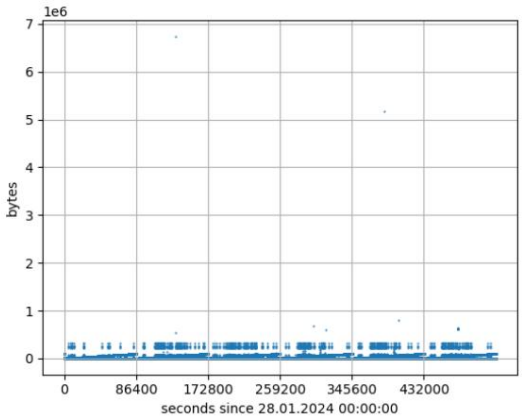
Kullanılan web sunucusu, isteğin HTTPS üzerinden yapılıp yapılmadığına bağlı olarak iki ayrı günlük dosyasına yazar. Bu nedenle, orijinal 'LOG_FILE' özniteliği, 1 veya 0 içeren işlenmiş 'IS_HTTPS' özniteliğine dönüştürülür.

Va8 BOYUTU

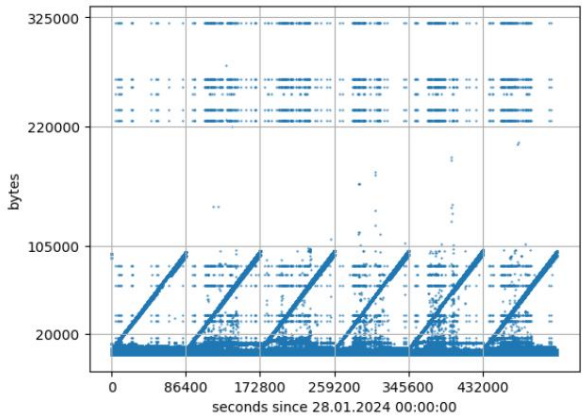
'SIZE' niteliği, isteğe yanıt olarak kaç bayt gönderildiğine dair bilgileri içerir. Eğitim veri kümesinde hangi boyutların yaygın olduğunu daha iyi anlamak için, 'SIZE' değerleri altı ardışık güne ait günlükleri içeren bir alt küme üzerine çizilmiştir. Bu çizimler Şekil 10 ve Şekil 11'de gösterilmiştir. Sol tarafta, Şekil 10'da, filtrelenmemiş boyutlar gösterilmiştir. Birkaç uç değer olduğundan, normal değerler neredeyse tek bir görünür boyuta sıkıştırılmıştır. Bu nedenle, Şekil 11'deki ikinci çizim yalnızca 400.000 bayttan küçük boyutlar içermektedir. Bu ikinci çizimde, 'SOURCE_TS' niteliğinin işlenmesinde karar vermede iki özellik önemlidir. İlk olarak, 220.000 ile 325.000 bayt arasındaki bölgedeki noktaların sıklığı, normal çalışmanın açık işaretlerini göstermektedir.

Saatler. İkinci olarak, zamanla büyüyen ve her gün sıfırlanan altı ayırt edilebilir çizgi vardır. Referans amacıyla, Şekil 11'de gösterilen bu tür bir grafik, iş günleri, çalışma saatleri ve tatiller hakkında benzer özellikler gösteren birkaç farklı haftalık veriyle oluşturulmuştur. Bu diğer grafikler, daha fazla alakalı olmadıkları için bu makalede gösterilmemiştir. Ayrıca, Şekil 11'de, y ekseninde 20.000, 105.000, 220.000 ve 325.000 bayt olmak üzere dört belirli değer bulunmaktadır. Bu değerler, değerleri beş bölgeye ayırır.

İşleme için bu beş bölge, değerlerin kovalara ayrılması amacıyla kova olarak kullanılır.
Bu nedenle, boyuta karşılık gelen işlenmiş nitelikler '<20k', '20-15k', '105- 220k', '220-325k' ve '>325k'dir. Bu kovalar içinde değerler 0 ile 1 arasında değişir; burada 0 alt sınır, 1 ise üst sınırdır.



Şekil 10: 28.01.2024 Pazar gününden itibaren altı güne ait günlüklerin boyut özelliğinin zaman damgalarına göre grafiği.



Şekil 11: Şekil 10'da gösterilen verilerle aynı olup, yalnızca 400000 bayttan küçük değerleri içerecek şekilde filtrelenmiştir.

Va9 YOLU

Web sunucusu, Janereporting kullanımı bağlamında erişilebilen birkaç farklı servlet sunar . Bunlara ilgili bağlam yolları üzerinden erişilir. Eğitim verileri üzerinde yapılan bir değer sayımı, bunlardan on tanesinin sıklıkla, diğerlerinin ise nispeten nadiren kullanıldığını ortaya koydu. Bu durum, günlükleri on bir kategoriye ayırma kararına yol açtı; en çok kullanılan on bağlam yolunun her biri için bir tane ve diğerleri için bir tane. Dolayısıyla, işlenen öznitelikler 'OUTPUT_SERVLET', 'REPORT_SERVICE', 'MISC_SERVICE', 'USER_SERVLET', 'NEWS_SERVLET', 'MENU_SERVLET', 'CONFIG_SERVLET', 'TIME_SERVICE', 'JANEREPORTING', 'JANEREPORTING/' ve 'PATH_ UNCATEGORIZED'dir ve değerleri atamak için tek sıcak kodlama kullanılır.

Va10 REFERANS

'REFERER' niteliği, kullanıcının bir kaynağı nereden talep ettiğine dair bilgi içerir. Bu, örneğin kullanıcının bir belge talep etmek için açılış sayfasında gösterilen bir bağlantıya tıkladığı anlamına gelebilir. 'PATH' niteliği belgenin yolunu gösterirken, 'REFERER' açılış sayfasına giden yolu gösterir. Hem 'PATH' hem de 'REFERER' bir jsessionId içerebilir. Bir jsessionId, uygulama düzeyinde oturumları tanımlamak için kullanılır. Oturum bir kullanıcıya bağlıdır, ancak bir kullanıcı mutlaka bir oturuma bağlı olmak zorunda değildir, çünkü kullanıcı istekte gerekli çerezleri göndermeden her yeniden bağlandığında yeni bir oturum oluşturulur. Bunu özetlemek gerekirse

Bilgileri kayan nokta değerlerine dönüştüren işlenmiş öznitelikler 'JID_IN_REF', 'JID_IN_PATH', 'JID_R_SEEN', 'JID_P_SEEN' ve 'JID_DIFFERENT', bu jses-sionId'ler hakkında bilgi içerir . İlk ikisi olan 'JID_IN_REF' ve 'JID_IN_PATH', sırasıyla 'REFERER' veya 'PATH' içinde bir jsessionId mevcutsa 1, aksi takdirde 0 olarak ayarlanır . Sonraki ikisi olan 'JID_R_SEEN' ve 'JID_P_SEEN', sırasıyla 'REFERER' veya 'PATH' içinde bir jsessionId mevcutsa ve daha önce karşılaşılmışsa 1, aksi takdirde 0 olarak ayarlanır. Son öznitelik olan 'JID_DIFFERENT', 'REFERER' içindeki jsessionId , 'PATH' içindeki jsessionId ile eşleşiyorsa 1'dir , aksi takdirde 0'dır.

Vb Otokoder

Katman sayısı veya katmanların büyüklüğü gibi hiperparametrelerin doğru seçimi, eğitim sürecinin birkaç dönem boyunca daha yakından gözlenmesiyle birden fazla deney yapılarak yapılır.

> "Herhangi bir makine öğrenimi çalışmasında olduğu gibi, hiperparametrelerin seçimi, doğrulama kümesine göre sonucun analiz edilmesiyle gerçekleştirilen deneysel testler tarafından yönlendirilir." [15]

Keras [30] kütüphanesiyle , bir LSTM'yi uygulamak için gereken kod aşağıdakine indirgenir:

```
1 keras'ı içe aktar 2
keras.layers'dan LSTM,Dense,RepeatVector,TimeDistributed'ı içe aktar 3 keras.models'dan
Sequential'ı içe aktar 4 autoencoder = Sequential() 5
autoencoder.add (LSTM(70, activation='
tanh ',     input_shape=(seq_size,n_features),
        return_sequences=True)) 6 autoencoder.add(LSTM(42, activation='tanh',
return_sequences=True)) 7 autoencoder.add(LSTM(6, activation='tanh', return_sequences=False))
8 autoencoder.add(RepeatVector(seq_size)) 9 autoencoder.add(LSTM(42, activation='tanh',
return_sequences=True)) 10 autoencoder.add(LSTM(70,
aktivasyon='tanh', dönüş_dizileri=True)) 11 autoencoder.add(TimeDistributed(Dense(n_features)))
12 autoencoder.compile(optimizer='adam', kayıp='ortalama_mutlak_hata',      kayıp_ağırlıkları=ağırlıklar)
```

12. satırda kullanılan değişken ağırlıklar şu şekilde tanımlanmıştır:

```
1 ağırlıklar = np.ones(n_features, dtype='float') 2 sütunAğırlıklı =
{'IS_WGET':1.1, "<20k":1.2 ,"20-105k":1.2,      "105-220k":1.2, "220-325k":1.2, ">325":1.2}

3 sütunlar için , Ağırlıklı.öğeler() içindeki sütunlar için: dizin =
4                   df.sütunlar.get_loc(sütun) ağırlıklar[indeks]
5                   = ağırlık
```

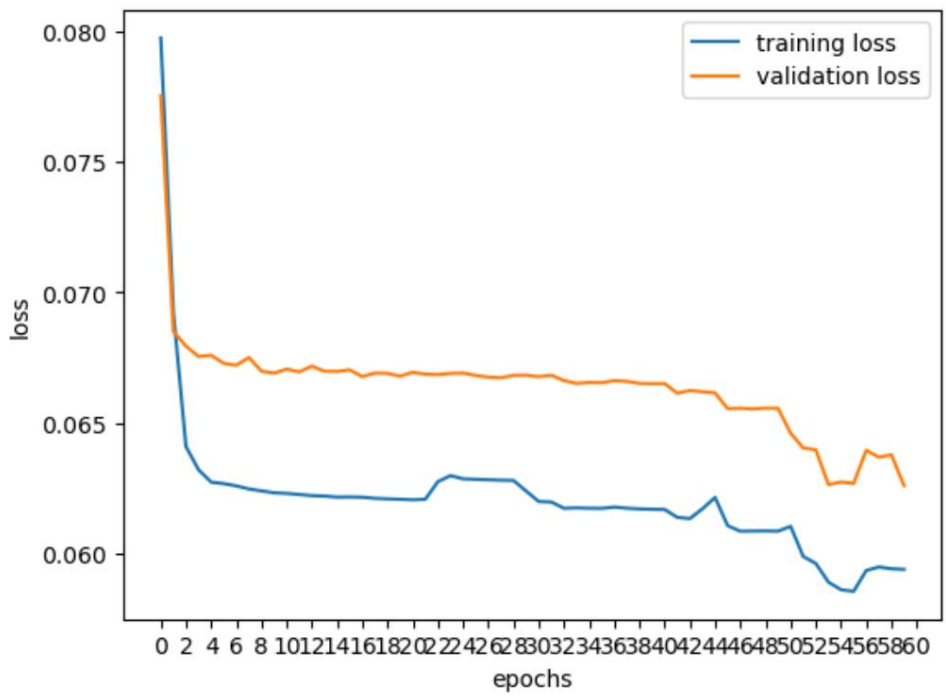
Burada n_features , ön işlemeden sonraki özellik sayısıdır. Ağırlıklar, kayıp fonksiyonundaki ilgili sütunlardaki sapmaların etkisini tartarak eğitim sırasında modeli yönlendirmek için kullanılır. LSTM katmanlarına sahip otokodlayıcının tanımı daha sonra tamamlanır ve eğitim süreci başlatılabilir. Her adımdan sonraki kaybın hesaplanması ve geri yayılım, çerçeve tarafından gerçekleştirilir. Eğitim sürecini başlatmak için, aşağıdaki kod parçacığında gösterildiği gibi, otokodlayıcı nesnesinde fit() yöntemi çağrılır:

```
autoencoder.fit( x=train_d,
        y=train_d,

        validation_data=(val_d,val_d),
        batch_size=batch_size, epochs=60

)
```

Otokodlayıcıyı eğitmek için dokuz hafta boyunca 14,5 milyondan fazla günlük kaydı toplandı. 1,7 milyon günlük kaydı içeren onuncu hafta, doğrulama kümesi olarak kullanıldı. Model, doğrulama kümesi üzerinde eğitilmedi; aşırı eğitimi önlemek veya fark etmek için referans olarak kullanıldı. Bu günlük kayıtları, Va bölümünde açıklandığı gibi ön işleme tabi tutuldu.
train_d değişkeni, önceden işlenmiş günlükleri içeren Veri Çerçevesini temsil eder ve hem girdi x hem de referans çıktısı y olarak kullanılır. Burada, oto kodlayıcının girdisini kopyalamaya çalıştığını ve bu nedenle çıktının orijinal girdiyle aynı sonucu vermesi gerektiğini unutmayın. validation_data parametresinin de iki Veri Çerçevesine ihtiyacı vardır . Yine, çıktı girdiyle aynı olmalıdır. Doğrulama kümesi, modelin aşırı eğitim yapmadığından emin olmak için kullanılır ve bu durumda eğitim kümesine dahil olmayan bir haftalık günlük verisinden oluşur. Ayrıca, eğitim verilerine genel bir bakış, en fazla 71 günlük kaydının aynı zaman damgalarına sahip olduğunu gösterdi, bu nedenle dizi boyutu en az iki farklı zaman damgası içeren 100 olarak ayarlandı. Sonuç olarak, batch_size 200 olarak ayarlandı, böylece tek bir partide iki dizi eğitilebilir. Ek olarak, modelin dizileme penceresini parti üzerinde kaydırmasına izin vermek için bir shift_window_size [33] tanımlanmıştır. Şekil 12'de görülebileceği gibi, kayıp birçok dönem boyunca durağanlaşmakta ve yaklaşık 50 dönem sonra tekrar düşmektedir. Bu durum, eğitim için 60 dönem kullanılmasına sebep olmuştur.



Şekil 12: Prototip PB'nin eğitiminin geçmiş grafiği . Eğitim seti üzerindeki kayıp mavi renkle, doğrulama seti üzerindeki kayıp ise her dönemin sonunda sarı renkle gösterilmiştir. Kaybın yaklaşık 55. döneme kadar azaldığı görülebilir.

Prototipin Vc Çıkışı

Vb bölümündeki eğitilmiş otokodlayıcı modeliyle , toplanan log verileri işlenebilir. İşleme için, veriler önce ön işleme tabi tutulmalı ve bu da otokodere girilebilecek bir x Veri Çerçevesi ile sonuçlanmalıdır. Otokoderin çıktısı, x ile aynı boyutlara sahip bir y Veri Çerçevesidir. x ve y'deki sütun sayısı, Vb bölümünde tanımlandığı gibi n_özelliktir . Ardından , her satır için ortalama mutlak hata hesaplanır. Bunu matematiksel olarak açıklamak için denklem (8)'e bakın.

$$\text{maei} = \frac{1}{\text{n\_özellikler}} \sum_{j=0}^{\text{n\_özellikler}} x_{i,j} \quad y_{i,j} \qquad (8)$$

Denklem (8), mae'yi bir vektör veya dizi olarak tanımlar ; burada her giriş, bir günlük girişini yeniden üretmedeki ortalama mutlak hataya karşılık gelir. Daha sonra, bir girişin anormal olarak etiketlenip etiketlenmeyeceğine karar vermek için eşik değeri olarak manuel olarak ayarlanan bir parametre kullanılır. Eşik değeri için seçilen değer, VII. bölümde açıklanmıştır. Bu eşik değeri ve mae dizisiyle, orijinal Veri Çerçevesi bir 'ERROR_ABOVE_THRESHOLD' sütunuyla genişletilebilir . Bu sütun, mae'deki karşılık gelen girişin değerini eşik değeri çıkarılarak içerir . Dolayısıyla prototipin çıktısı, yalnızca bu yeni sütunda pozitif değere sahip günlükleri içerecek şekilde filtrelenmiş 'ERROR_ABOVE_THRESHOLD' sütunuyla genişletilmiş orijinal CSV dosyasıdır .

Vd Değerlendirmesi

Vc bölümünde açıklanan çıktıyla , geri çağırma, kesinlik ve F1 puanı metrikleri, çıktının etkin değerlerle karşılaştırılmasıyla hesaplanabilir. Bu hesaplama, sklearn.metrics paketiyle, daha doğrusu classifica-tion_report ve confusion_matrix yöntemleriyle yapılır. Bu metrikleri elde etmek için, VI.a bölümündeki saldırılardan gelen tüm günlükler 'IS_ANOMALY' sütununda True (Doğru) olarak işaretlenir . Prototipleri optimize etmek için eşik parametresi değişken olarak tanımlanır ve farklı değerler için geri çağırma, kesinlik ve F1 puanını gösteren grafikler oluşturulur (bkz. Şekil 15-17). En iyi eşik değeri daha sonra bu değerler karşılaştırılarak belirlenir.

VI Test Kurulumu

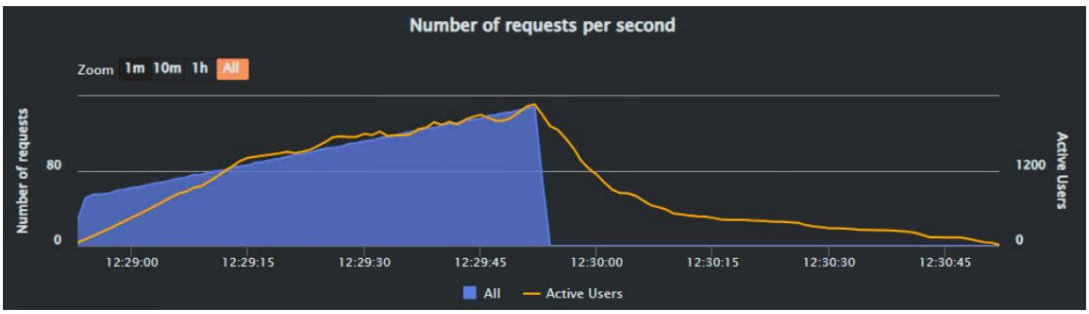
Anomali tespit modeli, yalnızca geçerli günlüklerden oluşan etiketlenmemiş veriler üzerinde eğitilerek, herhangi bir anomali olmadığından emin olunur. Modelin performansını değerlendirmek için ek testler gereklidir. Bu bölümde, testlerin nasıl yapıldığı ayrıntılı olarak açıklanmaktadır.

VI.a Saldırıları

Modeli test etmek için anormal verilere ihtiyaç vardır. Bu nedenle, anormallikleri toplamak için bazı saldırılar gerçekleştirilmiştir. Bu makalede ele alınan verilerin günlük öznitelikleri, kullanıcılar hakkında veri veya IP adresleri gibi bir tanımlayıcı aracılığıyla yapılan bir istekten hangi makinenin sorumlu olduğu hakkında bilgi içermez. Bu durum, kimliği doğrulanmamış ve yetkisiz erişim gibi saldırıları kullanım örneklerinin dışında tutar. DoS ve kaba kuvvet saldırıları öncelikli olarak kullanıcı verilerine bağlı değildir ve bu nedenle bir anormallik tespit modelinin performansını değerlendirmek için geçerli kullanım örnekleridir.

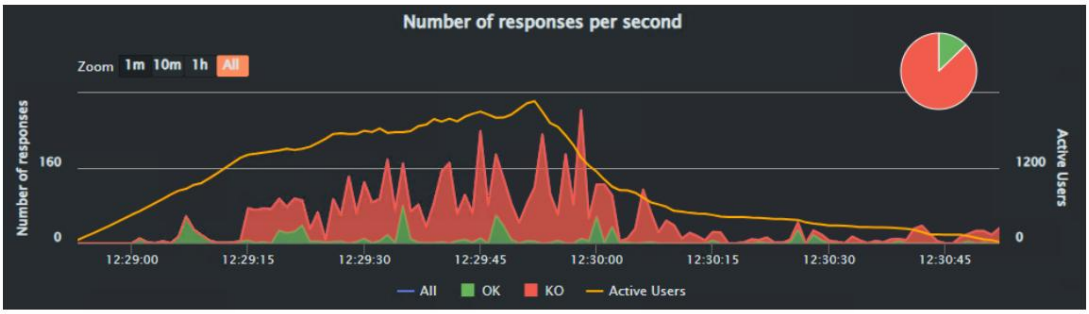
VI.a.1 DoS Saldırısı Yürütme

Bir DoS saldırısı gerçekleştirmek için, her istek için uygulama üzerinde 1,5 saniyelik bir yük oluşturan belirli bir URL talep edildi. İstekler Gatling [4] kullanılarak yapıldı.
Şekil 13'te görüldüğü gibi saldırı saniyede 50 kullanıcı ile başlatılmış ve bir dakikalık süre içerisinde saniyede 150 kullanıcıya çıkarılmıştır.



Şekil 13: Gatling'den [4] DoS saldırısına yönelik istek sayısının grafiği.

Şekil 14, kaç isteğin gerçekten yanıt aldığını göstermektedir. Yaklaşık 20 saniye sonra, isteklerin çoğunun yanıt almadığını görebilirsiniz. Genel olarak 6000 isteğin yalnızca 771'i yanıt aldı. Bu, hizmetin çoğunlukla kullanılamadığı başarılı bir DoS saldırısıdır.



Şekil 14: DoS saldırısı sırasında sunucudan gelen yanıt sayısının Gatling [4] grafiği.

VI.a.2 Kaba Kuvvet Saldırısı Gerçekleştirme

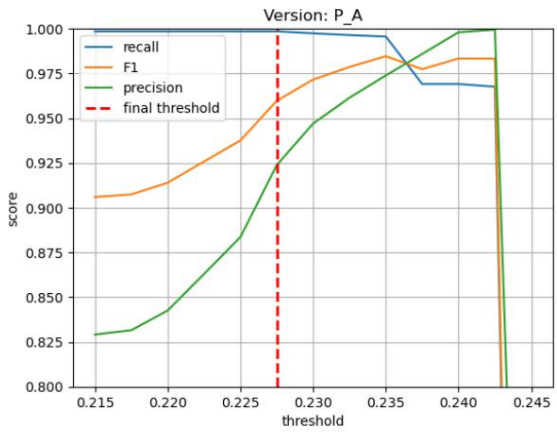
Sunucuya gerçekleştirilen ikinci saldırı, Janereporting'e yönelik bir kaba kuvvet saldırısıydı. Bu bağlamda kaba kuvvet, çok sayıda olası saldırının sistematik olarak denenmesini içerir.
Bir web sunucusundaki gizli dizinleri veya dosyaları ortaya çıkarmak için URL kombinasyonları. Bu saldırı
Gobuster [6] adlı bir araç kullanılarak gerçekleştirildi . Gobuster, kelime listesini içeren bir kelime listesi kullandı
Sunucuya sorgu göndermek için yaklaşık 45.000 potansiyel dizin ve dosya adı. Bu saldırı simülasyonunda,
daha derin yolları kaldıran ön işleme nedeniyle, bkz. Va9 bölümü, saldırı
yalnızca belirli bir kök dizini hedef alıyordu. Bu, çoğu yanıtın kaynakların bulunamadığını göstermesiyle
sonuçlandı. Her ne kadar bu işlem sırasında yararlı hiçbir şey bulunamamış olsa da
Saldırının ardından istekler yürütüldü ve anormalliğin test edilmesi için günlükler oluşturuldu
tespit modeli.

VI.b Test Verilerinin Etiketlenmesi

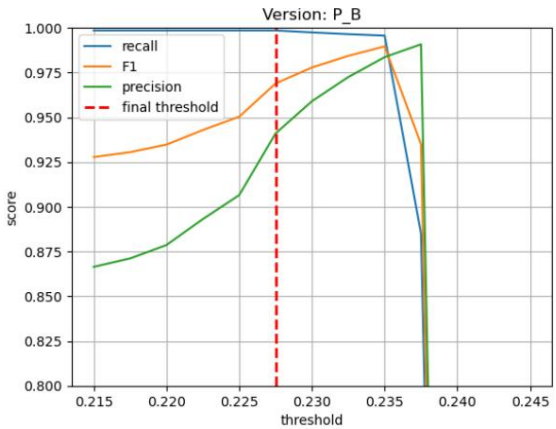
Otomatik kodlayıcının performansını değerlendirebilmek için, bir saldırıyı temsil eden tüm günlüklerin etkin
bir şekilde anormal olarak etiketlenmesi gerekir. Bu, şu şekilde yapıldı:
saldırı isteklerinde belirli bir yol, bu nedenle test sırasında tüm bu günlükler filtrelenebilir
ve gerçek bir anormallik olarak işaretlendi. Tanımlama , Va9 bölümündeki spesifikasyona göre yalnızca bilgi
içerecek şekilde önceden işlenmiş olan yolda yapılır.
Kullanılan servlet hakkında. Bu nedenle, tanımlama ön işlenmiş verileri etkilemez.
Veri Çerçevesi.

VII Sonuçlar

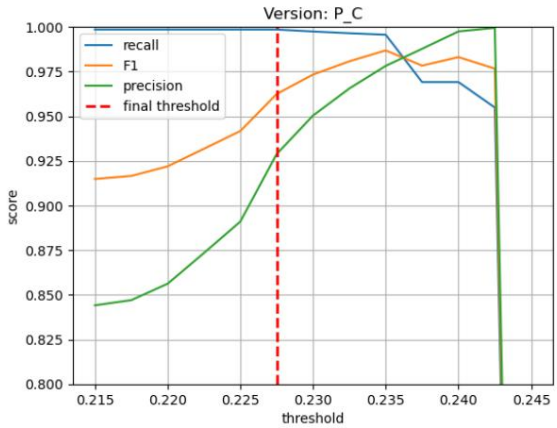
Bu çalışma kapsamında üç prototip oluşturuldu. Referans amaçlı olarak,
PA, PB olarak adlandırılırlar. Tüm prototipler, bir taneden oluşan oto kodlayıcılardır
47 boyutunda giriş katmanı, beş LSTM katmanı ve 47 boyutunda bir çıkış katmanı. Giriş boyutu
Vb bölümünde n_features olarak adlandırılan giriş DataFrame'inin özelliklerinin sayısıyla verilir ,
Ön işlemeden sonra. Çıkış katmanı, oto kodlayıcıların tanımı gereği, şu özelliklere sahip olmalıdır:
Girişle aynı boyutta. PA'nın beş LSTM katmanının boyutları 70, 42, 12, 42, 70'tir.
Bu sırayla, PB'nin beş LSTM katmanının boyutları 70, 42, 6, 42, 70'tir. Bu nedenle,
fark yalnızca sırasıyla 12 veya 6 büyüklüğündeki en iç katmanda bulunur ve bu da
Gizli alanın boyutu. PB'nin gizli alanının daha küçük olması, onu daha genelleştirmeye zorlar
Eğitim sırasında. Üçüncü prototip olan PC , 60, 35, 5, 35, 60 boyutlarında gizli katmanlara sahiptir.
PA ve PB ile karşılaştırıldığında eğitilebilir parametrelerin yalnızca yaklaşık %80'i ile sonuçlanır . Ayrıca,
gizli alan PB'den bile daha küçüktür , bu nedenle daha da genelleştirmeye zorlar.
En iyi eşik değerlerini bulmak için, her prototip için bir grafik oluşturulur ve bunların
Geri çağırma, kesinlik ve F1 puanı farklı değerlerle eşik olarak ayarlanmıştır.



Şekil 15: Prototip PA'nın performansı ,
geri çağırma, hassasiyet ve F1 ile ölçüldü
0,215'ten başlayan eşik değerlerine sahip puanlar
0,245'e kadar.



Şekil 16: Geri çağırma, hassasiyet ve F1 ile ölçülen
prototip PB'nin performansı
0,215'ten başlayan eşik değerlerine sahip puanlar
0,245'e kadar.



Şekil 17: Geri çağırma, hassasiyet ve F1 puanı ile ölçülen prototip PC'nin performansı
0,215 ile 0,245 arasında değişen eşiklerle.

Şekil 15-17'de görüldüğü gibi her üç prototip için de optimum eşik değeri 0,2275'tir.
Bu durumda, optimal, daha yüksek bir değerin daha küçük bir geri çağırma değeriyle sonuçlanacağı anlamına gelir,
bu da birincil amaca aykırı olacaktır ve daha düşük bir değer,
Daha düşük hassasiyet, ikincil hedefe karşı çalışır. Şekil 18 ila 20'de, 0,2275 eşik değerine sahip üç
prototipin karışıklık matrisleri gösterilmektedir. Değerler
Gerçek pozitifler, gerçek negatifler, yanlış pozitifler ve yanlış negatifler için geri çağırma,
hassasiyet ve F1 puanının hesaplanmasına olanak tanır. Tablo 2, 0,2275 eşik değerine , ve PC
sahip PA ve PB puanlarını daha ayrıntılı olarak göstermektedir. İkinci , vurgulanır
prototip olan PB , PA ve PC'den daha yüksek bir hassasiyet ve F1 puanı elde etmektedir .

P_A		Prediction	
		No Anomaly	Anomaly
Effective	No Anomaly	3360183	3868
	Anomaly	70	46890

Şekil 18: Prototip PA'nın karışıklık matrisi .

P_B		Prediction	
		No Anomaly	Anomaly
Effective	No Anomaly	3661109	2942
	Anomaly	70	46890

Şekil 19: Prototip PB'nin karışıklık matrisi .

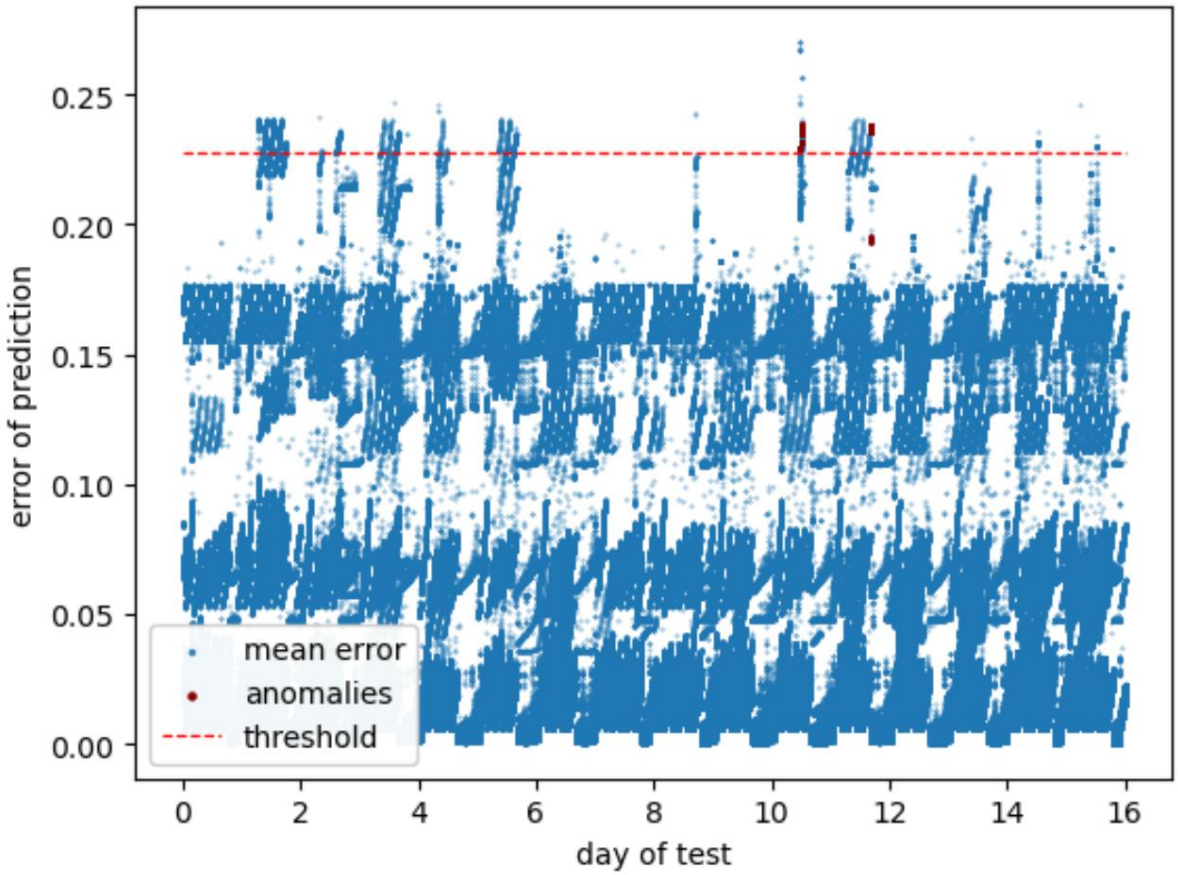
P_C		Prediction	
		No Anomaly	Anomaly
Effective	No Anomaly	3660457	3594
	Anomaly	70	46890

Şekil 20: Prototip PC'nin karışıklık matrisi .

Prototip	Geri Çağırma	Hassasiyeti	F1 Puanı
PA	0,9985	0,9238	0,9597
Fıstık ezmesi	0,9985	0,9410	0,9689
Bilgisayar	0,9985	0,9288	0,9624

Tablo 2: Üç prototipin performans metrikleri PA, PB eşiği 0,2275. , ve PC ile

Şekil 21, her bir test setinin yenilenmesinde üretilen ortalama mutlak hatayı göstermektedir
Günlükler. Normal kullanım günlükleri mavi renktedir, VI.a bölümünde açıklanan saldırıların
oluşturduğu bilinen anomaliler koyu kırmızı renkte işaretlenmiştir. Ek olarak, eşik değeri
0,2275 kesikli kırmızı çizgi olarak gösterilmiştir. Test setinin 10. gününde, DoS saldırısı bölüm VI.a.1
gerçekleştirildi ve işaretli veri noktaları bu saldırıya karşılık gelmektedir. Benzer şekilde,
11. günde, VI.a.2 bölümündeki kaba kuvvet saldırısı gerçekleştirildi ve veriler işaretlendi
Buna göre. Şekil 19 ve Şekil 21'i karşılaştırdığımızda, 70 günlük kaydının olduğunu görebiliriz.
Eşik değerinin altında olanlar yanlışlıkla negatif olarak etiketlendi. Öte yandan, eşik değerinin
üzerinde yaklaşık 3000 mavi nokta var ve bunlar anomali olarak yanlış sınıflandırıldı.



Şekil 21: Prototip PB'nin test setini yeniden oluştururken ürettiği hata. Koyu
Kırmızı noktalar, VI.a bölümünde açıklanan saldırılarla yaratılan anomalileri işaretler.
Kesikli kırmızı çizgi bu prototip için optimum eşiği temsil eder.

Geliştirme sonrasında, ortaya çıkan prototip, ZHAW GitHub deposunda [34] bir Python uygulaması olarak mevcuttur. Kullanım talimatları, ilgili README dosyasında [35] verilmiştir. Prototip, aynı anda birden fazla günlük dosyasını okuyabilir, bu bölümde daha önce açıklandığı gibi anomalileri tespit edebilir ve tek bir birleştirilmiş çıktı üretebilir.
Şüpheli günlük dosyası. Giriş dosyaları tablo 3'e benzeyebilirken, ilgili çıktı dosyaları
dosya tablo 4'e benziyor. Birleştirilen çıktı dosyası, tespit edilen anormal günlüklerden oluşuyor.
Anomali olarak tespit edilmeyen günlükler hariç. Ayrıca, yeni bir özellik sunuyor
Anomali tespit eşiğinin üstündeki hatayı gösteren sütun.

KAYNAK_TS	YÖNTEM	YANIT_KODU	...	
20240512063722000	ELDE ETMEK	200	...	
20240512064921000	POST	200	...	

KAYNAK_TS	YÖNTEM	YANIT_KODU	...	
20240512155515000	POST	404	...	
20240512155517000	POST	200	...	
20240513145114000	ELDE ETMEK	404	...	

Tablo 3: İki giriş CSV dosyasına bir örnek.

KAYNAK_TS	YÖNTEM	YANIT_KODU	...	ÜSTTEKİ_HATA		EŞİK
20240512063722000	AL			200	...	0,0142
20240512064921000	POST			200	...	0,0024
20240513145114000	AL			404	...	0,0156

Tablo 4: Sadece anormal olarak algılanan günlüklerden oluşan ER-ROR_ABOVE_THRESHOLD ek sütununa sahip bir çıktı CSV dosyası örneği .

VIII Tartışma

Prototip PB , Tablo 2'de gösterildiği gibi %99,85 ile umut verici performans değerleri göstermektedir.
Geri çağırma, %94,1 hassasiyet ve %96,89 F1 puanı. Bu sonuç, yaklaşık %96 ila %99 geri çağırma, yaklaşık
%93 ila %98 hassasiyet ve yaklaşık %99 ila %99 geri çağırma oranı elde eden benzer çalışmalarla tutarlıdır.
Yaklaşık %94 ila %98 F1 puanı için bkz. [15, 36, 37]. Karşılaştırma tamamen doğru değil,
Eğitim ve değerlendirme için kullanılan veri setlerinde önemli tutarsızlıklar bulunmaktadır.
Manuel ön işleme, farklı veri kümeleri üzerinde eğitim yapılmasına izin vermez. Daha fazla araştırma
yapılması ve prototipin daha büyük test kümelerinde değerlendirilmesi gerekir. Saldırılar
VI.a bölümünde açıklanan her ikisi de kısa süreler içinde çok sayıda günlük üretir ve bu da yanıltıcı puanlara
yol açabilir. Öte yandan, yanlış pozitifler anormaldir, hatta
zararsız olsalar da. Bu zararsız anomaliler kısmen uygulamadan kaynaklanabilir.
Nadiren kullanılan işlevler. Bu ayrımın daha ayrıntılı olarak araştırılması gerekiyor. Ayrıca,
Eşiği tanımladıktan sonra performansı görmek için başka bir test seti kullanılmalıdır.
İlk test seti. Şekil 15 ila 17'de gösterilen eşik değerinin optimize edilme şekli, onu yalnızca bu özel test seti
için optimize eder. Prototipi daha iyi test etmek için daha fazla çalışma yapılmalıdır.
kontrollü koşullar altında farklı anormal günlükler oluşturmaya da yöneliyor. Ayrıca,
Ön işlemede daha fazla optimizasyon yapılabilir. Birçok özellik gereksiz bilgi içerir ve tek sıcak kodlama
yerine ikili gösterim gibi tekniklerle yoğunlaştırılabilir. 'SIZE' niteliğinin , aşağıda açıklandığı gibi,
gruplandırılması
Va8 bölümü şu anda örtüşmüyor ve bu da 0 ve 0 boyutları için aynı değerlerin elde edilmesiyle sonuçlanıyor
örneğin 20000. Ek olarak, farklı niteliklerin ortalamaya göre ağırlıklandırılması
Mutlak hatanın da daha detaylı araştırılması gerekir. Ayrıca, kayıp fonksiyonunun kendisi
yeniden değerlendirilebilir. Daha az eğitilebilir parametrelere sahip bir oto kodlayıcı bundan faydalanabilir.
IV.c bölümünde belirtildiği gibi kayıp patlaması riski daha az olduğundan, ortalama karesel hata.

Bulunan sonuçların güvenilirliğini artırmak için eğitim veri kümesi yeniden kontrol edilmelidir.
Uygulama hakkında daha fazla bilgi edinin. Uygulamanın bazı işlevleri
Jane raporlamaları çok nadiren kullanılır ve bu nedenle bilgisi olmayan herkese anormal görünür.
uzmanlık gerektirir, ancak meşrudurlar. Nadiren kullanılsa da, böyle bir eylem
birkaç saniye içinde birden fazla günlük girişine yol açabilir. Ayrıca,
Daha kesin zaman damgaları prototipi iyileştirebilir, çünkü saniye cinsinden çözünürlük
Her zaman doğru sıralamaya izin vermez. Benzer şekilde, kesin bir kullanıcı kimliği veya etkili IP adresleri,
modelin daha fazla saldırı türünü tespit etmesini sağlar.
Değerli bilgiler arasında coğrafi konum veya sistemin zaman dilimi yer alır.

Bu çalışma, web erişim günlüğü kayıtlarının tam olarak ön işlenmesine ilişkin bir içgörü sunmaktadır.
Kayan nokta değerleri ve ikili gösterimler. Kurs boyunca eğitilen prototip
Bu çalışmanın anomali tespitinde başarılı bir şekilde kullanılabilecek bir programda kullanılabilmesi
mümkündür.

IX Sonuç

Bu çalışmada, web erişim kayıtlarındaki anormallikleri tespit etmek için bir LSTM otokodlayıcı kullanan bir prototip geliştirildi ve değerlendirildi. İş ortamlarında, birçok uygulama günde önemli miktarda kayıt oluşturur ve bu kayıtlar, olası ihlalleri gösterebilir.
Bu miktardaki günlük, manuel analizin pratik sınırlarını aşıyor. Prototip, manuel analizin iş yükünü azaltırken gerçek anomalilerin tespit edilmesine yardımcı oluyor.
yönetilebilir sınırlar içinde doğrulama.

Prototipin güvenilirliği, yüksek geri çağırma ve hassasiyet değerleriyle vurgulanmaktadır
Test verilerinde, benzer projelerdeki birçok modeli geride bırakarak başarıya ulaşıyor.
Prototipin, hiçbir zaman dahil edilmemiş olsalar bile saldırıları tespit edebildiğini gösteriyor
Bu yetenek, web uygulamalarının güvenliğini artırmak için hayati önem taşımaktadır.
Ayrıntılı manuel ön işleme, anormallik tespiti için sağlam bir temel sağlar.
Özellikle Janereporting uygulaması için , saldırıların erken tespitini mümkün kılmak ve anında karşı önlemler alınmasını sağlamak.

Gelecekteki çalışmalar ayrıntılı özellik seçimi ve ön işleme yöntemlerine dayanabilir
Sunuldu. Veri setinin çeşitli ortamlardan günlükleri içerecek şekilde genişletilmesi, modelin genelleme yeteneğini artırabilir. Ön işlemede daha fazla iyileştirme, örneğin
tek sıcak kodlama yerine ikili gösterimlerin kullanılması ve 'SIZE' gibi niteliklerin gruplandırılmasının iyileştirilmesi de modelin performansını artırabilir.
Kayıp fonksiyonunun eklenmesi ve daha hassas zaman damgalarının, kullanıcı tanımlayıcılarının ve coğrafi konum verilerinin dahil edilmesi, prototipi daha da geliştirerek ek iyileştirmeler sağlayabilir. anomalileri tespit etmede sağlam ve doğrudur.

Referanslar

[1] Next Stride AG, “Jane,” erişim tarihi: 2024-06-07. [Çevrimiçi]. Erişilebilirlik: https://www.shrimp.org/index.php?sub_confirmation=1 //nextstride.com/de/jane/

[2] Y. Li ve Q. Liu, “Siber saldırılar ve siber güvenliğe ilişkin kapsamlı bir inceleme çalışması; ortaya çıkan eğilimler ve son gelişmeler”, Enerji Raporları, cilt 7, s. 8176– 8186, 2021.

[3] X. Zhang, M. Xu, G. Da ve P. Zhao, “Siber tehditler altında bir ağ sistemi üzerinden hassas verilerin gizliliğinin ve erişilebilirliğinin sağlanması”, Güvenilirlik Mühendisliği ve Sistem Güvenliği, cilt 214, s. 107697, 2021.

[4] Gatling, “Gatling (3.11.2),” erişim tarihi: 2024-05-12. [Çevrimiçi]. Erişilebilirlik: https://www.gatling.com/gatling/3.11.2 //github.com/gatling/gatling

[5] Gordon Lyon, “nmap,” erişim tarihi 2024-06-06. [Çevrimiçi]. Erişilebilirlik: https: //nmap.org

[6] Gobuster, “Gobuster (3.6.0),” erişim tarihi 2024-05-12. [Çevrimiçi]. Erişilebilirlik: https://gobuster.gobuster.com/gobuster/3.6.0/ //github.com/OJ/gobuster

[7] ASB Singh, Y. Yusof ve Y. Nathan, “Eagle: Tarama ve sayım için GUI tabanlı penetrasyon testi aracı”, 2021 14. Uluslararası eSistem Mühendisliğinde Gelişmeler Konferansı (DeSE). IEEE, 2021, s. 97–101.

[8] Next Stride AG, “Next Stride AG,” 2024, erişim tarihi 2024-06-07. [Çevrimiçi]. Mevcut: https://nextstride.com/de/

[9] M. Cieliebak, “Makine öğrenimi ve veri madenciliği - sinir ağları - bölüm 1,” Üniversite Dersi, PDF, 2022.

[10] ——, “Makine öğrenimi ve veri madenciliği - sinir ağları - bölüm 2,” Üniversite Konferansı, PDF, 2022.

[11] MA Nielsen, Yapay sinir ağları ve derin öğrenme. Belirleme basını San Fran- Cisco, CA, ABD, 2015, cilt. 25.

[12] P. Kose, "Sinir ağlarının şafağı - açıklanabilir yapay zeka görselleştirmesi (bölüm 5)", 2022, erişim tarihi 2024-05-29. [Çevrimiçi]. Şurada mevcuttur: https://medium.com/deepviz/ explainable-ai-and-visual-interpretability-dawn-of-neural-networks-part-5-b302e7d85650

[13] K. Gurney, Sinir ağlarına giriş. CRC basımı, 2018.

[14] S. Flores. (2019) Varyasyonel oto kodlayıcılar güzeldir. Erişim tarihi: 2024-05-29. [Çevrimiçi]. Şuradan erişilebilir: https://www.compthree.com/blog/autoencoder/

[15] M. Catillo, A. Pecchia ve U. Villano, “Otolog: Sistem günlüklerinin derin otomatik kodlamasıyla anomali tespiti”, Uygulamalı Uzman Sistemler, cilt 191, s. 116-263, 2022.

[16] Y. Yu, X. Si, C. Hu ve J. Zhang, “Tekrarlayan sinir ağlarının bir incelemesi: Lstm hücreleri ve ağ mimarileri”, Sinirsel hesaplama, cilt 31, sayı 7, s. 1235–1270, 2019.

[17] S. Hochreiter ve J. Schmidhuber, “Uzun kısa süreli bellek”, Sinirsel hesaplama, cilt 9, sayı 8, s. 1735–1780, 1997.

[18] B. Lindemann, B. Maschler, N. Sahlab ve M. Weyrich, “lstm ağlarını kullanan teknik sistemler için anomali tespiti üzerine bir araştırma”, Endüstrideki Bilgisayarlar, cilt 131, s. 103498, 2021.

[19] aishwarya.27, “Tekrarlayan sinir ağlarına giriş”, 2023, erişim tarihi 2024-05-30. [Çevrimiçi]. Erişilebilirlik: https://www.geeksforgeeks.org/ introduction-to-recurrent-neural-network/

[20] D. Kwon, K. Natarajan, SC Suh, H. Kim ve J. Kim, “Konvolüsyonel sinir ağları kullanılarak ağ anomalisi tespiti üzerine ampirik bir çalışma”, 2018 IEEE 38. Dağıtık Hesaplama Sistemleri Uluslararası Konferansı'nda (ICDCS). IEEE, 2018, s. 1595–1598.

[21] V.-H. Le ve H. Zhang, “Günlük ayrıştırma olmadan günlük tabanlı anormallik tespiti”, 2021 36. IEEE/ACM Uluslararası Otomatik Yazılım Mühendisliği Konferansı'nda (ASE). IEEE, 2021, s. 492–504.

[22] S. He, J. Zhu, P. He ve MR Lyu, “Deneyim raporu: Anomali tespiti için sistem günlüğü analizi”, 2016 IEEE 27. Uluslararası Yazılım Güvenilirliği Mühendisliği Sempozyumu'nda (ISSRE). IEEE, 2016, s. 207–218.

[23] S. Lu, X. Wei, Y. Li ve L. Wang, “Evrişimsel sinir ağı kullanarak büyük veri sistemi kayıtlarındaki anormallikleri tespit etme”, 2018 IEEE 16. Uluslararası Güvenilir, Otonom ve Güvenli Bilgi İşlem Konferansı, 16. Uluslararası Yaygın Zeka ve Bilgi İşlem Konferansı, 4. Uluslararası Büyük Veri Zekası ve Bilgi İşlem Konferansı ve Siber Bilim ve Teknoloji Kongresi (DASC/PiCom/DataCom/CyberSciTech). IEEE, 2018, s. 151–158.

[24] M. Du, F. Li, G. Zheng ve V. Srikumar, “Derin günlük: Derin öğrenme yoluyla sistem kayıtlarından anomali tespiti ve teşhisi”, 2017 ACM SIGSAC bilgisayar ve iletişim güvenliği konferansının bildirilerinde, 2017, s. 1285–1298.

[25] AB Nassif, MA Talib, Q. Nasir ve FM Dakalbab, “Anormallik tespiti için makine öğrenimi: Sistematik bir inceleme”, Ieee Access, cilt 9, s. 78 658–78 700, 2021.

[26] A. Patcha ve J.-M. Park, “Anormallik tespit tekniklerine genel bakış: Mevcut çözümler ve en son teknolojik trendler”, Bilgisayar ağları, cilt 51, sayı 12, 2007-8.

[27] Tensorflow, “tf.keras.layers.lstm,” 2024, erişim tarihi 2024-06-05. [Çevrimiçi]. Erişilebilir: https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM#:~: text=Based%20on%20available,the%20outermost%20context.

[28] Numpy, “Numpy (1.23.4),” erişim tarihi 2024-06-06. [Çevrimiçi]. Erişilebilirlik: https://www.numpy.com/en/news/numpy/numpy/1.23.4/ //numpy.org/

[29] T. pandas geliştirme ekibi, “pandas-dev/pandas: Pandas”, Şubat 2020, erişim tarihi 2024-06-06. [Çevrimiçi]. Erişilebilirlik: https://zenodo.org/records/7037953

[30] F. Chollet ve diğerleri, “Keras (2.6.0),” Erişim tarihi: 2024-06-06. [Çevrimiçi]. Mevcut: https://keras.io

[31] Tensorflow, “Tensorflow (2.6.0),” 2024-06-06 tarihinde erişildi. [Çevrimiçi]. Mevcut: https://doi.org/10.5281/zenodo.5181671

[32] JD Hunter, “Matplotlib: 2 boyutlu bir grafik ortamı”, Bilim ve Bilgisayar Bilimlerinde Bilgi İşlem Mühendislik, cilt 9, sayı 3, s. 90–95, 2007.

[33] Tensorflow, “tf.data.dataset,” 2024, erişim tarihi 2024-06-06. [Çevrimiçi]. Erişilebilir: https://www.tensorflow.org/api_docs/python/tf/data/Dataset#shift

[34] M. Koch, O. Wiedler, “Anomali tespit modeli prototipi”, erişim tarihi : 2024-06-07. [Çevrimiçi]. Erişilebilir: https://github.zhaw.ch/student-theses-traa/BA-log-analyzis-Wiedler-Koch

[35] ——, “Anomali tespit modeli prototipi için beni oku”, erişim tarihi : 2024-06-07. [Çevrimiçi]. Şurada mevcuttur: https://github.zhaw.ch/student-theses-traa/BA-log-analyzis-Wiedler-Koch/blob/main/README.md

[36] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat ve V. Chandrasekhar, “Rakip tarafından öğrenilen anormallik tespiti”, 2018 IEEE Uluslararası Veri Madenciliği Konferansı (ICDM). IEEE, 2018, s. 727–736.

[37] J. Wang, Y. Tang, S. He, C. Zhao, PK Sharma, O. Alfarraj ve A. Tolba, “Lo-gevent2vec: Nesnelerin internetinde büyük ölçekli kayıtlar için logevent-vektör tabanlı anomali tespiti,” Sensors., cilt 20, sayı 9, 2020-5-26.