UNIVERZITET U NIŠU
PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA RAČUNARSKE NAUKE

Nikola Stevanović

VEŠTAČKE NEURONSKE MREŽE ZA DETEKCIJU VEB NAPADA

DOKTORSKA DISERTACIJA

Niš, 2025.

UNIVERSITY OF NIŠ
FACULTY OF SCIENCES AND MATHEMATICS
DEPARTMENT OF COMPUTER SCIENCE

Nikola Stevanović

ARTIFICIAL NEURAL NETWORKS FOR WEB ATTACK DETECTION

DOCTORAL DISSERTATION

Niš, 2025.

Подаци о докторској дисертацији

Ментор:	др Бранимир Тодоровић, ванредни професор, Природно-математички факултет, Универзитет у Нишу
Наслов:	Вештачке неуронске мреже за детекцију веб напада
Резиме:	Ова дисертација представља свеобухватни приступ детекцији веб напада помоћу вештачких неуронских мрежа. За прикупљање разноликог малициозног веб саобраћаја су коришћене замке, чиме је омогућено креирање робусног корпуса за тренирање модела за идентификацију сајбер претњи. Студија обрађује детекцију напада нултог дана помоћу различитих техника машинског учења које су у стању да идентификују претходно непознате рањивости у мрежном саобраћају. Да би се умањило катастрофално заборављање у динамичним окружењима напада, предложено је коришћење више стратегија инкременталног учења, које омогућавају непрекидну адаптацију модела уз минимални губитак претходно стеченог знања. Студија уводи метод одабира атрибута базиран на популацији, који побољшава ефикасност класификације фокусирањем на најрелевантније мрежне атрибуте. У дисертацији је представљен модел дубоког учења за детекцију фишинг мејлова, базиран на архитектурама рекурентних и конволуционих неуронских мрежа. Поред тога, примењене су напредне технике пондерисања и уградње атрибута како би се побољшала детекција фишинг веб сајтова. Интегрисањем ових метода, ово дисертација нуди скалабилно и адаптивно решење за детекцију веб претњи у реалном времену, пружајући значајан напредак у областима веб безбедности и машинског учења.
Научна област:	Рачунарске науке
Научна дисциплина:	Вештачка интелигенција
Кључне речи:	Вештачке неуронске мреже, рекурентне неуронске мреже, конволуционе неуронске мреже, одабир атрибута, пондерисање атрибута, инкрементално учење, сајбер безбедност, детекција веб напада, замке, детекција фишинга
УДК:	004.8(043.3)
CERIF класификација:	P 176 Вештачка интелигенција
Тип лиценце Креативне заједнице:	**CC BY-NC-ND**

Data on Doctoral Dissertation

Doctoral Supervisor:	Branimir Todorović, PhD, associate professor, Faculty of Sciences and Mathematics, University of Niš
Title:	Artificial neural networks for web attack detection
Abstract:	This dissertation presents a comprehensive approach to web attack detection using artificial neural networks. The collection of diverse malicious web traffic was carried out using honeypots, enabling the creation of a robust dataset for training models to identify cyber threats. The study addresses zero-day attack detection through various machine learning techniques capable of identifying previously unknown vulnerabilities in network traffic. To reduce catastrophic forgetting in dynamic attack environments, the use of multiple incremental learning strategies has been proposed, which enable continuous model adaptation with minimal loss of previously acquired knowledge. The study introduces a population-based feature selection method, which improves classification efficiency by focusing on the most relevant network features. The dissertation presents a deep learning model for phishing email detection, based on the architectures of recurrent and convolutional neural networks. Moreover, advanced feature weighting and embedding techniques are employed to enhance phishing website detection. By integrating these methods, this dissertation provides a scalable and adaptive solution for real-time detection of web-based threats, offering significant advancements in the fields of web security and machine learning.
Scientific Field:	Computer science
Scientific Discipline:	Artificial intelligence
Key Words:	Artificial neural networks, recurrent neural networks, convolutional neural networks, feature selection, feature weighting, incremental learning, cybersecurity, web attack detection, honeypots, phishing detection
UDC:	004.8(043.3)
CERIF Classification:	P 176 Artificial intelligence
Creative Commons License Type:	**CC BY-NC-ND**

Прилог 4/1

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:	
Идентификациони број, **ИБР**:	
Тип документације, **ТД**:	монографска
Тип записа, **ТЗ**:	текстуални / графички
Врста рада, **ВР**:	докторска дисертација
Аутор, **АУ**:	Никола М. Стевановић
Ментор, **МН**:	Бранимир Т. Тодоровић
Наслов рада, **НР**:	Вештачке неуронске мреже за детекцију веб напада
Језик публикације, **ЈП**:	српски
Језик извода, **ЈИ**:	српски / енглески
Земља публиковања, **ЗП**:	Србија
Уже географско подручје, **УГП**:	Србија
Година, **ГО**:	2025.
Издавач, **ИЗ**:	ауторски репринт
Место и адреса, **МА**:	Ниш, Вишеградска 33.
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)	96 стр., граф. прикази
Научна област, **НО**:	рачунарске науке
Научна дисциплина, **НД**:	вештачка интелигенција
Предметна одредница/Кључне речи, **ПО**:	вештачке неуронске мреже, рекурентне неуронске мреже, конволуционе неуронске мреже, одабир атрибута, пондерисање атрибута, инкрементално учење, сајбер безбедност, детекција веб напада, замке, детекција фишинга
УДК	004.8(043.3)
Чува се, **ЧУ**:	библиотека
Важна напомена, **ВН**:	

Извод, **ИЗ**:		Ова дисертација представља свеобухватни приступ детекцији веб напада помоћу вештачких неуронских мрежа. За прикупљање разноликог малициозног веб саобраћаја су коришћене замке, чиме је омогућено креирање робусног корпуса за тренирање модела за идентификацију сајбер претњи. Студија обрађује детекцију напада нултог дана помоћу различитих техника машинског учења које су у стању да идентификују претходно непознате рањивости у мрежном саобраћају. Да би се умањило катастрофално заборављање у динамичним окружењима напада, предложено је коришћење више стратегија инкременталног учења, које омогућавају непрекидну адаптацију модела уз минимални губитак претходно стеченог знања. Студија уводи метод одабира атрибута базиран на популацији, који побољшава ефикасност класификације фокусирањем на најрелевантније мрежне атрибуте. У дисертацији је представљен модел дубоког учења за детекцију фишинг мејлова, базиран на архитектурама рекурентних и конволуционих неуронских мрежа. Поред тога, примењене су напредне технике пондерисања и уградње атрибута како би се побољшала детекција фишинг веб сајтова. Интегрисањем ових метода, ово дисертација нуди скалабилно и адаптивно решење за детекцију веб претњи у реалном времену, пружајући значајан напредак у областима веб безбедности и машинског учења.
Датум прихватања теме, **ДП**:		05.12.2022.
Датум одбране, **ДО**:		
Чланови комисије, **КО**:	Председник:	
	Члан, ментор:	
	Члан:	
	Члан:	
	Члан:	

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

KEY WORDS DOCUMENTATION

Accession number, **ANO**:	
Identification number, **INO**:	
Document type, **DT**:	monograph
Type of record, **TR**:	textual / graphic
Contents code, **CC**:	doctoral dissertation
Author, **AU**:	Nikola M. Stevanović
Mentor, **MN**:	Branimir T. Todorović
Title, **TI**:	Artificial neural networks for web attack detection
Language of text, **LT**:	Serbian
Language of abstract, **LA**:	Serbian / English
Country of publication, **CP**:	Serbia
Locality of publication, **LP**:	Serbia
Publication year, **PY**:	2025
Publisher, **PB**:	author’s reprint
Publication place, **PP**:	Niš, Višegradska 33.
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)	96 p. ; graphic representations
Scientific field, **SF**:	computer science
Scientific discipline, **SD**:	artificial intelligence
Subject/Key words, **S/KW**:	artificial neural networks, recurrent neural networks, convolutional neural networks, feature selection, feature weighting, incremental learning, cybersecurity, web attack detection, honeypots, phishing detection
UC	004.8(043.3)
Holding data, **HD**:	library
Note, **N**:	

Abstract, **AB**:		This dissertation presents a comprehensive approach to web attack detection using artificial neural networks. The collection of diverse malicious web traffic was carried out using honeypots, enabling the creation of a robust dataset for training models to identify cyber threats. The study addresses zero-day attack detection through various machine learning techniques capable of identifying previously unknown vulnerabilities in network traffic. To reduce catastrophic forgetting in dynamic attack environments, the use of multiple incremental learning strategies has been proposed, which enable continuous model adaptation with minimal loss of previously acquired knowledge. The study introduces a population-based feature selection method, which improves classification efficiency by focusing on the most relevant network features. The dissertation presents a deep learning model for phishing email detection, based on the architectures of recurrent and convolutional neural networks. Moreover, advanced feature weighting and embedding techniques are employed to enhance phishing website detection. By integrating these methods, this dissertation provides a scalable and adaptive solution for real-time detection of web-based threats, offering significant advancements in the fields of web security and machine learning.
Accepted by the Scientific Board on, **ASB**:		05.12.2022.
Defended on, **DE**:		
Defended Board, **DB**:	President:	
	Member, Mentor:	
	Member:	
	Member:	
	Member:	

Sadržaj

Predgovor

Kako je rastao značaj računarskih mreža na svakodnevni život ljudi, otvorile su se i mogućnosti za ostvarivanje dobiti putem malicioznih aktivnosti. Ta dobit može imati materijalnu prirodu, ali može biti i u obliku informacija, znanja i slično. Kako bi ostvarili takvu dobit, sajber kriminalci izvršavaju napade na veb portale, sisteme pružanja usluga, državne sisteme, kao i na mašine običnih korisnika. Poslednjih godina svedoci smo napada na nuklearna postrojenja, napada koji onemogućavaju dostavljanje električne energije, kao i napada koji prekidaju rad pogona za industrijsku proizvodnju. Uprkos zakonima koji kažnjavaju ovakve aktivnosti, broj sajber napada je u konstantnom porastu.

Da bi iskustvo korišćenja računara i računarskih mreža bilo prijatno za krajnje korisnike, neophodno je u što većoj meri detektovati i sprečiti ovakve napade. Kako sajber kriminalci konstantno kreiraju nove napade, kao i nove varijacije već postojećih napada, tradicionalne metode detekcije često ne mogu na vreme da otkriju ovakve vrste napada. Veštačka inteligencija, a pre svega veštačke neuronske mreže, su zadnjih decenija doživele veliki razvoj. Pogotovo su efikasne prilikom učenja na osnovu obeleženih primera, i karakteriše ih dobra generalizacija na nove do sada neviđene primere. Kako je velika sličnost u problemu koji se javlja u detekciji napada i problemu koji neuronske mreže rešavaju, u ovoj disertaciji će biti istražena primena veštačkih neuronskih mreža na detekciju veb napada.

Rad se sastoji iz šest poglavlja. U **prvom poglavlju** su date teorijske osnove. Opisano je šta su veštačke neuronske mreže i kako je tekao njihov razvoj kroz istoriju. Nelinearnost omogućava neuronskim mrežama da modeliraju kompleksije procese, pa će u ovoj glavi biti opisano više načina kako se ona postiže. U glavi je opisano i više tipova slojeva neuronskih mreža, koji će kasnije biti korišćeni. Dat je i kratak pregled trenutnih metoda primene veštačke inteligencije na detekciju veb napada.

Svaki veb servis je u nekoj meri ranjiv. Slanjem malicioznih zahteva, napadači pokušavaju da ostvare pristup delovima sistema koji nisu predviđeni za njih. Saobraćaj prema nekom veb servisu ima veliki broj regularnih i mali broj malicioznih zahteva. Da bi prikupili maliciozne zahteve, koristićemo zamke (eng. honeypots), do kojih je teško doći regularnim ponašanjem na mreži. Ovakav pristup nam takođe omogućava da potencijalno prikupimo informacije o novim, do sada neviđenim tipovima napada, za razliku od standardnih pristupa u kojima se često korpus malicioznih zahteva kreira simulacijom već poznatih napada. Koristeći prikupljene zahteve iz regularnog saobraćaja i maliciozne zahteve iz zamki, u **drugom poglavlju** ćemo učiti modele veštačkih neuronskih mreža da detektuju maliciozne zahteve sa što većom tačnošću. Testiraćemo više pristupa ekstrakcije atributa (eng. feature extraction) i kreirati više modela, u kojima ćemo isprobati pristupe i sa i bez uzimanja u obzir sekvencijalne prirode sadržaja veb zahteva. Pored

standardnog načina testiranja, izvršićemo i testiranje u kome ćemo zahteve za treniranje i evaluaciju razdvojiti po vremenu kada su se desili, da bi testirali takozvane napade nultog-dana (eng. zero-day attacks).

Regularan saobraćaj i tipovi malicioznih zahteva napadača se menjaju vremenom. Kako neuronske mreže koristeći standardne tehnike treniranja imaju problem da prilikom dotreniravanja na novim primerima izgube znanje akumulirano do tada (eng. catastrophic forgetting), u **trećem poglavlju** će biti predstavljeno više tehnika inkrementalnog učenja. Ove tehnike imaju za cilj da se izbegne potpuno retreniranje modela, i da modeli inkrementalno akumuliraju znanje učeći samo na novim primerima, uz što manje zaboravljanje. Drugo i treće poglavlje temelje se na radu:

- Nikola Stevanović, Branimir Todorović, and Vladan Todorović. Web attack detection based on traps. Applied Intelligence, 52(11):12397–12421, 2022. `https://doi.org/10.1007/s10489-021-03077-9`. Reproduced with permission from Springer Nature.

Neke napade je teško detektovati samo analizom sadržaja veb zahteva. Radi sveobuhvatnije analize, u **četvrtom poglavlju** ćemo istražiti problem detekcije veb napada pomoću analize mrežnih atributa u toku regularnog saobraćaja i napada. S obzirom da ovde nije jednostavno odrediti korisne atribute mreže, analiziraćemo problem selekcije mrežnih atributa u cilju poboljšanja efikasnosti detekcije napada. Fokusiraćemo se na generalnijem pristupu selekciji atributa, koji je moguće primeniti na različite modele detekcije. Četvrto poglavlje je bazirano na radu:

- Nikola Stevanović. Population-based feature selection for intrusion detection. In First Serbian International Conference on Applied Artificial Intelligence (SICAAI). Kragujevac, Serbia, 2022.

U poslednje vreme jedan od najprisutnijih tipova napada su takozvani fišing (eng. phishing) napadi, u kojima se napadači lažno predstavljaju kao određene osobe ili institucije od poverenja, ne bi li dobili od napadnutog lica određene poverljive informacije ili materijalna sredstva. Da bi ublažili ovaj problem, u petom i šestom poglavlju ćemo istražiti primenu dubokih neuronskih mreža na detekciju ovih vrsta napada. U **petom poglavlju** ćemo analizirati problem detekcije fišing mejlova, od kojih ovi napadi najčešće počinju. Iz mejlova ćemo najpre izvući koristan tekstualni sadržaj, a potom kreirati efikasan klasifikator teksta, baziran na dubokoj neuronskoj mreži i ugradnji reči i karaktera u vektorski prostor. Peto poglavlje se temelji na radu:

- Nikola Stevanović. Character and word embeddings for phishing email detection. Computing and Informatics, 41(5):1337–1357, 2022.

Šesto poglavlje obrađuje problem detekcije fišing veb sajtova. Do njih se najčešće dolazi klikom na linkove iz fišing mejlova, ali i pomoću drugih kanala za njihovu distribuciju. Analiziraćemo uticaj različitih atributa sajta na to da li se radi o fišing sajtu, raditi ugradnju tih atributa u vektorski prostor i kreirati efikasan sistem za detekciju fišing veb sajtova. Šesto poglavlje je bazirano na radu:

- Nikola Stevanović. Embedding and weighting of website features for phishing detection. Facta Universitatis, Series: Mathematics and Informatics (prihvaćen 22.11.2024. godine).

Disertacija je urađena pod mentorstvom prof. dr Branimira Todorovića. Koristim ovu priliku da mu se zahvalim na velikoj pomoći koju mi je pružio tokom čitavih doktorskih studija.

Želeo bih da se zahvalim i Vladanu Todoroviću i Danijelu Sokoloviću, ekspertima iz oblasti sajber bezbednosti, koji su me uputili u ovu oblast i pomogli mi prilikom prikupljanja podataka o veb napadima neophodnim za analizu.

Takođe bih hteo da se zahvalim Nikoli Mihajloviću i svim kolegama sa posla, jer su uvek imali razumevanja za moje aktivnosti vezane za doktorske studije.

Na kraju bih se najtoplije zahvalio i svojoj porodici na razumevanju i podršci koju mi je pružila tokom mog školovanja.

Poglavlje 1

Teorijske osnove

1.1 Veštačke neuronske mreže

Prve ideje o veštačkim neuronskim mrežama datiraju još iz 40-ih godina 20. veka. McCulloch i Pitts su 1943. godine [74] predložili matematički model neurona i time postavili temelje za dalji razvoj neuronauke i veštačke inteligencije. U svojoj knjizi "Organizacija ponašanja"[41], Hebb je analizirao plastičnost neuronskih mreža, i sposobnost sinapsi da jačaju ili slabe tokom vremena.

Već u sledećoj deceniji, Rosenblatt [93] je inspirisan funkcionisanjem oka muve kreirao perceptron, uprošćeni matematički model funkcionisanja neurona. U ovom modelu se svaki od ulaznih signala množi određenom težinom. Ukoliko je suma tih proizvoda veća od granične vrednosti, izlaz neurona je 1, a u suprotnom je 0. On je predložio i način učenja ovakvog neurona na osnovu primera. Glavni nedostatak predloženog perceptrona je što je mogao da uči samo klase koje su linearno razdvojive. U narednim godinama se nastavlja istraživanje sa sličnim modelima, a pojavljuju se i prve praktične primene u detekciji govornog jezika, kreiranju vremenske prognoze, dijagnostici EKG-a (elektrokardiograma) i drugim oblastima [110].

Nakon perioda umanjenog istraživanja i ulaganja u ovakve sisteme, poznatijeg kao zima u veštačkoj inteligenciji (eng. the AI winter), 80-ih godina se ponovo obnavlja interesovanje. U srži tog interesovanja je bekpropagacija (eng. backpropagation), čiji začeci datiraju još iz 60-ih godina, ali je pravu popularnost doživela 80-ih, posebno nakon što su Rumelhart i dr. objavili poglavlje u knjizi pod nazivom "Učenje unutrašnjih reprezentacija propagiranjem greške"[94]. Bekpropagacija je u kombinaciji sa gradijentnim spustom omogućila treniranje višeslojnih veštačkih neuronskih mreža. Ovo je i danas preovlađujući način treniranja neuronskih mreža.

Krajem 80-ih i početkom 90-ih kreće primena bekpropagacije i u računarskom vidu (eng. computer vision). LeCun i dr. su 1989. godine objavili rad [68] u kome su pokazali da neuronska mreža može uspešno da prepoznaje rukom napisane cifre poštanskih brojeva. Ovaj rad je pokazao i efikasnost konvolucionog sloja, koji i danas čini osnovu mnogih modela u računarskom vidu.

Tih godina su se slične ideje pojavile i u obradi sekvencijalnih podataka. Waibel i dr. su [106] predložili neuronsku mrežu sa vremenskim kašnjenjem za prepoznavanje fonema, koja je idejno veoma slična LeCun-ovoj mreži. Ipak, u procesiranju sekvenci su

u narednom periodu primat preuzele rekurentne neuronske mreže. Za razliku od mreža u kojima izlaz nekog sloja može da bude ulaz samo narednim slojevima, u rekurentnim mrežama to nije slučaj. U rekurentnim mrežama se često dešava da se izlaz poslednjeg sloja koristi kao ulaz prvog (ili u ekstremnom slučaju izlaz jednog neurona kao njegov ulaz), a to elegantno rešava problem dodavanja memorije mreži, što je često dosta korisno u obradi sekvencijalnih podataka. Za treniranje ovakvih modela kreiran je i novi vid bekpropagacije nazvan bekpropagacija kroz vreme [109].

Naredni period usporenog razvoja i primene neuronskih mreža izazvali su problemi vezani za treniranje dubokih modela. Više algoritamskih tehnika je korišćeno kako bi se ovaj proces unapredio [42, 16, 17]. Ipak, ključni korak desio se u hardveru. Umesto dotadašnjih centralnih procesorskih jedinica, za treniranje dubokih neuronskih mreža počele su da se koriste grafičke procesorske jedinice, koje imaju značajno veći nivo paralelizacije, čime su omogućile znatno brže treniranje ovakvih modela. Kreiranje većih korpusa je omogućilo većim modelima da iskoriste svoj pun potencijal. Jedan od najpoznatijih iz tog vremena bio je ImageNet korpus, na kome su Krizhevsky i dr. dubokom konvolucionom mrežom ostravili ubedljivu pobedu 2012. godine, i time započeli period nadmoći veštačkih neuronskih mreža, koji i dalje traje.

1.2 Primeri slojeva neurona

Neuronske mreže sadrže veliki broj neurona. Neuroni su najčešće podeljeni po slojevima, koji predstavljaju celine unutar mreže. U nastavku će biti opisano više tipova slojeva koji se koriste u modelima koji će biti predloženi za detekciju napada. Za svaki od njih će biti objašnjeno koju funkciju obavlja, i kako procesira ulazni signal.

1.2.1 Sloj ugradnje

Slojevi ugradne se koriste kada postoji rečnik tokena (npr. slogova, reči, slova i slično), i postoji potreba da svaki token ima svoju vektorsku reprezentaciju. Svaki token ima svoj jedinstveni identifikacioni broj (ID) unutar rečnika (ceo broj između 0 i $N-1$, gde N predstavlja veličinu rečnika). Sloj ugradnje se sastoji iz jedne matrice sa parametrima $E \in \mathbb{R}^{N \times d}$, gde d predstavlja veličinu vektora ugradnje. Označimo sa e_i i-ti red u matrici $E = [e_0, \ldots, e_{N-1}]^\intercal$. Vektor e_i je vektorska reprezentacija tokena čiji je identifikacioni broj u rečniku i.

Na svom ulazu sloj ugradnje prima listu od L celih brojeva $i_0, \ldots, i_{L-1}$, koji predstavljaju ID-eve tokena iz ulazne sekvence. Sloj ugradnje uzima odgovarajuće redove iz matrice ugradnje E (na osnovu ID-eva), i kreira matricu $[e_{i_0}, \ldots, e_{i_{L-1}}]^\intercal$, koja predstavlja izlaz sloja ugradnje.

Jedna pogodnost sloja ugradnje je što njegovi parametri mogu da se uče zajedno sa svim ostalim parametrima modela. Kada se vrši bekpropagacija, računaju se gradijenti kriterijumske funkcije u ondosu na sve parametre modela, uključujući i parametre sloja ugradnje koji odgovaraju tokenima koji su bili prisutni na ulazu. Ovi gradijenti se kasnije koriste od strane optimizacionih algoritama baziranih na gradijentu kako bi se ažurirali parametri sloja ugradnje (kao i svi ostali parametri modela).

1.2.2 Linearni sloj i aktivacione funkcije neurona

Linearni sloj je jedan od najčešće korišćenih slojeva u arhitekturama neuronskih mreža, i sastoji se iz dva skupa parametara: matrice $W \in \mathbb{R}^{d_2 \times d_1}$ i vektora $b \in \mathbb{R}^{d_2}$. Ulaz je d_1-dimenzioni vektor x, a na izlazu kreira d_2-dimenzioni vektor $v = Wx + b$.

Da bi se kreirale kompleksnije reprezentacije iz ulaznih podataka, potrebno je koristiti nelinearnosti. Neke od najkorišćenijih nelinearnosti su sigmoidalna (σ), hiperbolički tangens (tanh) i funkcija ispravljača (eng. rectified linear unit ili ReLU). Formule ovih aktivacionih funkcija su date u jednačinama od (1.1) do (1.3). One se primenjuju po individualnim neuronima.

$$\sigma(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{1.1}$$

$$\tanh(x) = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}} \tag{1.2}$$

$$\mathrm{ReLU}(x) = \max(0, x) \tag{1.3}$$

1.2.3 LSTM

Duga kratkoročna memorija [44] (eng. long short-term memory ili LSTM) je arhitektura rekurentne ćelije koja je smanjila problem sa isčezavajućim gradijentima koji su tradicionalne rekurentne ćeilije imale. Rekurentne ćelije se koriste za procesiranje sekvencijalnih podataka.

Neka je data sekvenca od L d-dimenzionih ulaznih vektora $x_0, \ldots, x_{L-1}$. LSTM ćelija sekvencijalno prima ulazne vektore x_t (jedan po jedan od x_0 do x_{L-1}), koje koristi da bi ažurirala svoja dva d-dimenziona vektora stanja: vektor stanja ćelije c_t i vektor skrivenog stanja h_t. Početne vrednosti vektora stanja se najčešće inicijalizuju nulama. Formule po kojima se ažuriranju vektori stanja su date u jednačinama od (1.4) do (1.9).

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{1.4}$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{1.5}$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{1.6}$$

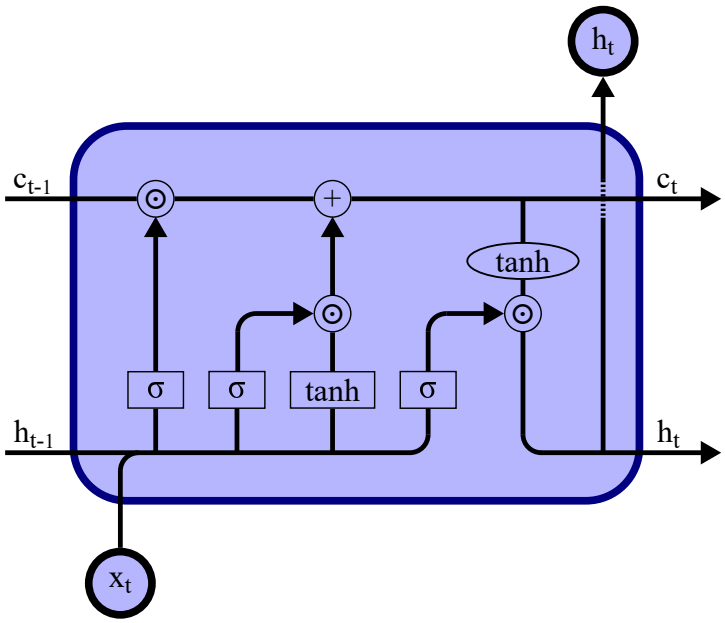
$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \tag{1.7}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{1.8}$$

$$h_t = o_t \odot \tanh(c_t) \tag{1.9}$$

Simbol $\odot$ predstavlja Hadamard-ovo množenje po elementima. Ćelija koristi mehanizme kapija (eng. gating mechanism) kao bi kontrolisala protok informacija. Ulazna kapija (eng. input gate) i_t kontroliše količinu informacija koje ćelija prima iz svog ulaza. Kapija zaboravljanja (eng. forget gate) f_t je odgovorna za resetovanje znanja koje već postoji u vektorima stanja ćelije. Izlazna kapija (eng. output gate) o_t kontroliše koji deo informacija koje ćelija ima treba da bude prosleđen na izlaz ćelije. Grafička ilustracija LSTM ćelije je data na slici 1.1.

Ćelija sadrži osam matrica ($W_{ii}, W_{if}, W_{io}, W_{ig}, W_{hi}, W_{hf}, W_{ho}, W_{hg} \in \mathbb{R}^{d \times d}$) i osam vektora ($b_{ii}, b_{if}, b_{io}, b_{ig}, b_{hi}, b_{hf}, b_{ho}, b_{hg} \in \mathbb{R}^{d}$) sa parametrima. Ovi parametri se ažuriraju u toku treniranja neuronskih mreža optimizacionim algoritmima baziranim na gradijentu.



Slika 1.1: LSTM ćelija

Poslednji vektor skrivenog stanja, h_{L-1}, se često koristi za reprezentaciju cele ulazne sekvence.

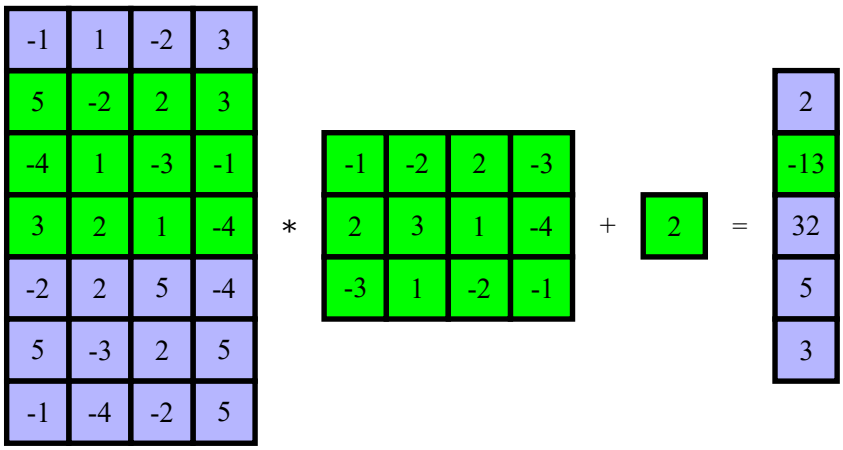
Postoji i dvosmerna (eng. bidirectional) varijanta LSTM arhitekture. Ona koristi dve odvojene LSTM ćelije. Jedna ćelija procesira ulazne vektore x_t u standardnom redosledu (od x_0 do x_{L-1}), dok ih druga procesira u obrnutom redosledu (od x_{L-1} do x_0). Izlaz arhitekture na poziciji t se dobija nadovezivanjem izlaza ove dve ćelije u trenutku nakon što procesiraju ulazni vektor x_t.

1.2.4 1-dimenziona konvolucija

U poslednjih par decenija, konvolucioni slojevi su bili uspešno primenjivani u različitim oblastima [64, 60, 3, 34]. Za ekstrakciju atributa u tekstualnim podacima, najčešće korišćeni konvolucioni sloj je 1-dimenzioni konvolucioni sloj. Neka je data ulazna matrica $U \in \mathbb{R}^{L \times d}$ formirana od sekvence L d-dimenzionih vektora. Sloj se sastoji iz više kernela (često nazivanim i konvolucionim matricama ili filterima). Svaki kernel sadrži svoju posebnu matricu parametara $W \in \mathbb{R}^{S \times d}$, i svoj poseban parametar pomeraja $b \in \mathbb{R}$. Hiperparametar S se zove veličina kernela. Izlaz konvolucije korišćenjem ovog kernela je vektor $v \in \mathbb{R}^{L-S+1}$, čije se vrednosti mogu izračunati korišćenjem formule (1.10). Vizuelni prikaz izračunavanja na jednom konkretnom primeru je dat na slici 1.2.

$$v_i = b + \sum_{j=0}^{S-1} \sum_{k=0}^{d-1} W_{j,k} U_{i+j,k} \tag{1.10}$$

Označimo sa N broj kernela u konvolucionom sloju. Svaki od njih će na izlazu proizvesti poseban $(L - S + 1)$-dimenzioni vektor. Konkatenacijom ovih vektora po no-



Slika 1.2: 1-dimensional convolution

voj dimenziji se dobija izlaz konvolucionog sloja, koji je u obliku matrice dimenzija $(L - S + 1) \times N$. Broj redova u izlaznoj matrici može dodatno biti povećan korišćenjem dopunjavanja ulazne matrice (eng. padding). Dopunjavanje se vrši tako što se određeni broj redova (najčešće popunjenih nulama) doda ulaznoj matrici U na njen početak i kraj, i kao rezultat toga se i veličina izlazne matrice poveća za isti broj redova. Jedan od čestih razloga zašto se vrši dopunjavanje je da bi prva dimenzija ulazne i izlazne matrice bila ista, ali postoje i drugi razlozi.

1.2.5 Selekcija globalnog maksimuma

Sloj selekcije globalnog maksimuma (eng. global max-pooling) se koristi da bi se izračunale najjače aktivacije ulaza duž ulazne sekvence. Neka je na ulazu data matrica $U \in \mathbb{R}^{L \times d}$. Izlaz sloja biće d-dimenzioni vektor v čiji elementi mogu da se izračunaju korišćenjem jednačine (1.11).

$$v_j = \max_i U_{i,j}. \tag{1.11}$$

Prilikom detekcije napada, nalaženje malicioznih delova negde u tekstu je često dobar indikator da se radi o nekoj malicioznoj aktivnosti. Kako konvolucioni sloj ekstraktuje atribute iz lokalnih delova teksta, primena sloja selekcije globalnog maksimuma nad njegovim izlazom može da pomogne u pronalaženju mesta u tekstu gde se dešavaju najjače aktivacije nekih malicioznih šablona, što ga čini posebno korisnim u ovom domenu.

1.2.6 Normalizacija sloja

Normalizacijom aktivacija neurona, moguće je postići brže i efikasnije treniranje. Za razliku od nekih ranijih normalizacionih tehnika, kao što je normalizacija grupe (eng. batch normalization) [46], normalizacija sloja (eng. layer normalization) [15] računa srednju vrednost i varijansu potrebu za normalizaciju koristeći samo jedan primer za treniranje. Takođe se izvršava identično izračunavanje tokom delova za treniranje i testiranje, što nije slučaj kod normalizacije grupe.

Neka je dat vektor $x \in \mathbb{R}^d$, koji predstavlja jedan ulazni primer. Sloj sadrži dva vektora parametara koja uči, $\gamma, \beta \in \mathbb{R}^d$. Pre početka treniranja, γ se inicijalizuje na vektor jedinica, a β na vektor nula. Izlaz sloja je d-dimenzioni vektor v, čije izračunavanje je dato u (1.12).

$$v = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} \odot \gamma + \beta \tag{1.12}$$

Kako je x vektor, a $\mathrm{E}[x]$ skalar, $\mathrm{E}[x]$ se oduzima od svakog elementa vektora x. Srednja vrednost i varijansa se računaju formulama (1.13) i (1.14). Za računanje varijanse će biti korišćen pristrasni estimator. Hiperparametar ϵ je neka mala konstanta (na primer 10^{-5}), i koristi se da bi se izbeglo potencijalno deljenje nulom.

$$\mathrm{E}[x] = \frac{1}{d}\sum_{i=0}^{d-1} x_i \tag{1.13}$$

$$\mathrm{Var}[x] = \frac{1}{d}\sum_{i=0}^{d-1} (x_i - \mathrm{E}[x])^2 \tag{1.14}$$

Pored toga što čini proces treniranja efikasnijim, ovaj sloj takođe ima pozitivan uticaj i na generalizaciju modela.

1.3 Primena mašinskog učenja na detekciju napada

Kako su računarske mreže vremenom rasle i postajale sve značajnije, rasla je i opasnost od napada. Kao posledica toga istraživanje u oblasti detekcije napada postaje sve značajnije. Veći deo literature iz oblasti detekcije veb napada je bazirana na nekim labeliranim korpusima. Od starijih korpusa, DARPA1998 [70] i KDDCup'99 [56] se i dalje aktivno analiziraju. NSL-KDD [100] korpus je kreiran od strane kanadskog instituta za sajber bezbednost na osnovu KDDCup'99 korpusa, eliminisanjem primera koji se ponavljaju i poboljšanjem balansa između broja primera različitog tipa. Na Univerzitetu u Nju Bransviku je kreiran ISCX2012 korpus [99], koji sadrži sirove podatke iz mrežnog saobraćaja prikupljene u periodu od 7 dana. Sadrži regularan saobraćaj i 4 tipa napada. AWID [62] korpus sadrži podatke iz 802.11 saobraćaja. Istraživači sa Univerziteta iz Novog Južnog Velsa su kreirali UNSW-NB15 [82] korpus sa ciljem da reše neke od problema ranijih korpusa. Među novijim korpusima treba svakako istaći korpuse CIC-IDS2017 i CSE-CIC-IDS2018 [97] kreirane od strane kanadskog instituta za sajber bezbednost, jer sadrže veliki broj napada koji su danas prisutni. Većina korpusa sadrži ili sirove podatke iz mrežnog saobraćaja, ili tabelarne podatke u CSV (eng. comma-separated values) formatu. Polja kod podataka u tabelarnom formatu najčešće imaju numeričke vrednosti ili predstavljaju određene kategorije. Među korpusima koji čuvaju veb zahteve u tekstualnom formatu, FWAF[1] korpus se izdvaja svojom veličinom.

Modeli u nastavku predstavljaju različite pristupe rešavanju problema detekcije veb napada pomoću mašinskog učenja. Dok su neki od njih jednostavniji i imaju veliku brzinu izvršavanja, drugi imaju složeniju strukturu i u stanju su da modeliraju kompleksnije

[1] `https://github.com/faizann24/Fwaf-Machine-Learning-driven-Web-Application-Firewall`

nelinearnosti. Modeli su najčešće bazirani na podacima saobraćaja u mreži, koji mogu da budu u sirovom ili tabelarnom formatu, ili na sadržajima samih zahteva, koji se često procesiraju kao tekst.

HAST-IDS [108] sistem detekcije koristi sirove podatke iz mrežnog saobraćaja. On analizira saobraćaj kao niz bajtova, koji su u HAST-II modelu dodatno grupisani u pakete. HAST-I model koristi konvolucionu neuronsku mrežu, dok HAST-II model pored nje koristi i LSTM [44] rekurentnu mrežu kako bi naučio vremenske atribute. Sistemi detekcije Lunet [111] i Pelican [112] su zasnovani na ovom modelu. Prvi ima različite kombinacije konvolucionih i LSTM slojeva, dok drugi sadrži i rezidualne slojeve, modelirane po ResNet [40] arhitekturi.

Kasongo i Sun [54] su koristili selekciju atributa i duboku LSTM mrežu kako bi kreirali sistem za detekciju upada. Ostvarili su tačnost od 86.99% na NSL-KDD korpusu. Isti autori su kasnije kreirali još jedan model [55], koji koristi GRU [22] arhitekturu rekurentne ćelije. Model je dostigao tačnost od 88.42%. Kako bi poboljšali ranu detekciju upada, Andalib i Vakili [14] su koristili ansambl od tri modela. Njega čine rekurentna neuronska mreža sa GRU ćelijom, konvoluciona neuronska mreža i klasifikator slučajne šume. Finalna klasifikacija se vrši ili metodom najviše glasova ili metodom bar jednog glasa. Agarap [4] je kreirao sistem detekcije napada koristeći rekurentnu neuronsku mrežu sa GRU ćelijom i SVM klasifikatorom u zadnjem sloju. U testiranju se ovaj sloj pokazao kao bolji u odnosu na softmax sloj, koji se najčešće koristi kao poslednji sloj.

Kanimozhi i Prem Jacob [52] su koristili višeslojnu potpuno povezanu neuronsku mrežu za detekciju napada. Oni su svoj pristup testirali na CSE-CIC-IDS2018 korpusu, i postigli su F_1-skor od 99.92%. Isti autori su kasnije poredili [53] preformanse veštačkih neuronskih mreža sa performansama drugih modela mašinskog učenja (naivni Bajesov klasifikator, model slučajne šume, k-najbližih suseda, metod potpornih vektora i AdaBoost). Tokom eksperimentalnog testiranja neuronske mreže su postigle bolje rezultate nego svi ostali klasifikatori na CSE-CIC-IDS2018 korpusu. Koristeći metod potpornih vektora, Rawat i Shrivastav [91] su rešavali problem detekcije SQL injekcije (eng. SQL injection).

Modifikujući generativnu suparničku mrežu (eng. generative adversarial network - GAN), Mohammadi i Sabokrou [80] su kreirali detektor napada baziran isključivo na regularnom saobraćaju. Zhang i dr. [116] su iskoristili autoenkoder za redukciju dimenzionalnosti ulaza, kao i rekurentnu neuronsku mrežu (GRU) za učenje reprezentacija i klasifikaciju. U zavisnosti od tipa napada, njihov pristup je ostvario tačnost između 94.7% i 97.6% na KDDCup'99 korpusu. Almseidin i dr. [12] su testirali veliki broj modela mašinskog učenja na NSL-KDD korpusu. Stablo odluke je imalo najmanju stopu lažno negativnih primera, dok je algoritam slučajne šume imao najveću tačnost i najmanju stopu lažno pozitivnih primera. Farnaaz i Jabbar [27] su takođe koristili algoritam slučajne šume na NSL-KDD korpusu. Oni su ostvarili tačnost od 99.67% bez korišćenja odabira atributa (eng. feature selection), i 99.69% koristeći odabir atributa.

Rong i dr. [92] su primenili konvolucionu neuronsku mrežu nad karakterima parametara zahteva kako bi otkrili da li se radi o malicioznim zahtevima. Da bi evaluirali svoj pristup, kreirali su sopstveni korpus sakupljanjem podataka sa interneta, i dodavanjem podataka iz već postojećih korpusa (uključujući i FWAF korpus). Još jedan pristup baziran na konvolucionoj neuronskoj mreži nad karakterima su predložili Ito i Iyatomi [48],

ali sa dva konvoluciona sloja i dva sloja udruživanja. Takvom arhitekturom su uspeli da postignu tačnost od 98.8%. Umesto ugradnje karaktera u vektorski prostor, Zhang i dr. [115] su najpre izvukli reči iz URL-ova zahteva. Zatim su vršili ugradnju reči u vektorski prostor, koje su dalje obrađivali konvolucionom neuronskom mrežom. Sve parametre mreže, uključujući i vektore ugradnje su adaptirali zajedno u procesu treniranja. Liang i dr. [69] su iskoristili dve rekurentne neuronske mreže kako bi naučili šablone u regularnim zahtevima, jednu za putanje zahteva i jednu za parametre zahteva. Obe su obučavali da predvide verovatnoće za sledeći token u sekvenci na osnovu prethodnih tokena. Verovatnoće koje ova dva modela generišu (kako za regularne, tako i za maliciozne zahteve) su kasnije korišćene kao ulaz potpuno povezane neuronske mreže koja je davala finalnu predikciju klasifikacije. Testirali su svoj model na dva korpusa, i postigli nivoe tačnosti od 98.42% i 98.56%.

Poglavlje 2

Detekcija malicioznih veb zahteva pomoću zamki

Milijarde veb sistema opslužuju svoje korisnike preko interneta, najveće računarske mreže, a mnogi od tih sistema su meta sajber kriminalaca. Njihov cilj je da nađu ranjivosti u veb sistemima i ostvare nelegalnu dobit. Tokom istorije veb-a, veliki broj različitih metoda je korišćen u cilju detekcije malicioznih veb zahteva poslatih veb sistemima. Ranije su sistemi detekcije uglavnom bili bazirani na potpisu (eng. signature-based). Oni rade na principu kreiranja unapred definisanih šablona, i kasnije traženja istih u zahtevima koje korisnici šalju sistemu. Ti šabloni mogu biti specifične sekvence reči ili bajtova u mrežnom protoku, pa čak i čitavi zahtevi poslati serveru. Ovaj pristup je dobar za detekciju već poznatih napada, ali ima problem prilikom generalizacije na nove tipove napada. Zahteva prikupljanje podataka o napadima, njihovu analizu i kreiranje potpisa.

Kako bi primenili modele mašinskog učenja na detekciju malicioznih veb zahteva, neophodno ih je prvo trenirati korišćenjem značajnog broja primera. Standardna komunikacija se sastoji od velikog broja regularnih zahteva i veoma malog broja malicioznih. Korpusi kreirani iz standardne komunikacije su veoma neuravnoteženi, i mogu da kreiraju pristrasne klasifikatore. Zbog toga je bitno kreirati način prikupljanja većeg broja malicioznih primera.

Jedan način koji se često koristi u literaturi je da kreatori korpusa izvrše veštačke napade koristeći dostupne alate za napade, i da onda sačuvaju informacije iz mrežnog saobraćaja. Iako ovim pristupom može da se prikupi značajan broj malicioznih primera, oni nisu od pravih napadača, već iz alata sa kojima su kreatori korpusa već upoznati. Ukoliko napadači koriste njihove sopstvene alate i generišu modifikacije postojećih ili nove napade, ti tipovi napada neće biti u korpusu. Samo u 2019. godini je detektovano više od 114 miliona[1] novokreiranih zlonamernih programa. Zbog toga je potrebno kreirati mehanizam prikupljanja primera novokreiranih napada. U ovom poglavlju biće prikazan jedan efikasan način kreniranja korpusa koristeći zamke za sakupljanje malicioznih zahteva. Posle toga će biti primenjeno više metoda veštačke inteligencije za učenje nad ovim zahtevima.

Zamke koje su korišćene su specifični veb servisi, bazirani na kloniranju realnih

[1] https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2019-2020.pdf

uređaja, postavljeni na javnim IP adresama, bez DNS (eng. Domain Name System) zapisa. Kako nema DNS zapisa, kao ni eksternih linkova ka IP adresama zamki, njima mogu da pristupe jedino ljudi ili alati koji rade skeniranje mreže, tražeći servise koji postoje na tim IP adresama. Oni pokušavaju da pokrenu interakciju sa veb servisima, a neki od njih i da izvrše napade nad njima. Koristeći primere zahteva upućenih zamkama, možemo konstantno da prikupljamo informacije o novim tipovima napada. Kombinujući ove zahteve sa zahtevima prikupljenim tokom standardne komunikacije, možemo da kontrolišemo odnos između regularnih i malicioznih zahteva u korpusu.

Još jedan problem koji ćemo u ovom poglavlju obraditi je ekstrakcija atributa koje će modeli mašinskog učenja koristiti u cilju prepoznavanja malicioznih zahteva. Primarni fokus biće na n-gramima (jednogramima, dvogramima i trigramima) karaktera izvučenih iz zahteva. Testiraćemo veliki broj plitkih i dubokih modela mašinskog učenja na problemu klasifikacije veb zahteva. Pod plitkim modelima podrazumevaćemo modele mašinskog učenja koji nemaju skrivene slojeve neurona, takozvane linearne ili log-linearne modele, kao i klasifikatore bazirane na stablima odlučivanja, kao što je klasifikator slučajne šume. Pod dubokim modelima smatraćemo veštačke neuronske mreže sa bar jednim skrivenim slojem. Kod plitkih modela će sekvencijalni redosled n-grama biti zanemaren, dok će kod dubokih modela biti uzet u obzir. Osim klasičnog testiranja u kome će primeri na slučajan način biti podeljeni na trening i test skup, modele ćemo testirati i na detekciju napada nultog-dana. To ćemo učiniti tako što ćemo sve primere prikupljene do nekog trenutka u vremenu koristiti za treniranje, a sve primere prikupljene nakon tog trenutka za testiranje.

Pored korpusa koji smo mi kreirali pomoću zamki, sva testiranja ćemo izvršiti i na velikom FWAF korpusu, koji ćemo koristiti za poređenje. FWAF korpus je kreiran od strane kompanije Fsecurify, koja se bavi kreiranjem inteligentnih zaštitnih zidova (eng. firewall) za veb aplikacije. Oni su koristili određene heuristike i ekspertsko znanje kako bi labelirali HTTP (eng. HyperText Transfer Protocol) zahteve iz ogromne kolekcije sa sajta SecRepo[2]. U korpusu se nalaze i dodatni primeri XSS (eng. Cross-Site Scripting), SQLi (eng. SQL injection) i drugih vrsta napada sakupljenih sa nekih poznatih GitHub[3] repozitorijuma.

2.1 Pregled literature

Zamke sadrže informacije i resurse koji mogu da budu od interesa napadačima. Saobraćaj ka njima se prati i analizira, kako bi se bolje razumele namere napadača. Veb zamke su zamke koje simuliraju veb korisnički interfejs. Veb korisnički interfejs je standardni interfejs mnogih uređaja, uključujući jednostavne kućne uređaje (IP kamere i sično), industrijske uređaje (eng. ICS devices) i korporativne sisteme.

Brown i dr. [19] su odradili studiju koristeći zamke unutar platformi za računarstvo u oblaku (eng. cloud computing), kao što su Amazon EC2, Microsoft Azure, IBM Smartcloud i ElasticHosts. Analizirali su odakle dolaze napadi i kog su tipa. Otkrili su da je najviše napada bilo preko SSH i HTTP protokola, i da su najčešće dolazili iz Sjedinjenih

[2] `http://www.secrepo.com/`
[3] `https://github.com/`

Američkih Država i Kine. Kombinovanjem sistema za detekciju upada i tri tipa zamki, Saadi i dr. [95] su dostigli stopu lažno pozitivnih od 1% i stopu lažno negativnih od 0.5% prilikom detekcije napada na OpenStack-u. OpenStack je softverski projekat računarstva u oblaku otvorenog koda, sličan Amazon-ovim veb servisima (eng. Amazon Web Services - AWS).

Kondra i dr. [63] su analizirali performanse različitih sistema detekcije upada baziranih na zamkama. Zaključili su da je većina napada bila na protokolima baziranim na TCP/IP-u (eng. Transmission Control Protocol/Internet Protocol), te da su HTTP portovi bili jedni od najpodložnijih napadima.

Ghourabi i dr. [35] su predložili implementaciju zamke za veb servise nazvanu VS Zamka (eng. WS Honeypot). Ekstraktovali su tri kategorije atributa iz prikupljenih podataka: potrošnju resursa za obradu poruka, sadržaje poruka i karakteristike sesije. Za detekciju DoS (eng. denial of service) napada su koristili SVR algoritam mašinskog učenja, SVM algoritam za detekciju novih napada, i više algoritama klasterovanja za karakterizaciju napada. SVM model je dostigao stopu detekcije od 94%, stopu lažno pozitivnih od 13%, i stopu lažno negativnih od 6%. SVR model je postigao stopu detekcije od 100%, uz stopu lažno pozitivnih od 4%, prilikom detekcije DoS napada. Matin i Rahardjo [73] su koristili SVM klasifikator i klasifikator stabla odluke nad paketima mrežnog saobraćaja prikupljenih od strane zamke. Primeri klasifikovani kao maliciozni su potom pregledani od strane ljudskih eksperata. Nakon toga je vršeno retreniranje sa ciljem povećanja tačnosti klasifikacije. Više informacija o različitim tipovima zamki i njihovoj upotrebi može se naći u studiji koju su sproveli Han i dr. [39].

2.2 Korpus

2.2.1 Zamke

Kako je HTTP jedan od najčešće napadanih protokola, za prikupljanje malicioznih veb zahteva su korišćene veb zamke. One simuliraju realne korisničke veb interfejse, i očekuje se da budu napadnute. Kreiranje zamki, njihovo raspoređivanje i prikupljanje podataka sa njih i sa regularnog sajta radili su eksperti sajber bezbednosti iz kompanije Advanced Security Technologies[4]. Zamke su kreirane kloniranjem ponašanja realnih uređaja (mrežni štampač, IP kamera) tokom regularnog korišćenja.

Za svaki zahtev ka njima pamti se relevantno zaglavlje (eng. header) i sadržaj odgovora (eng. payload), kako bi se ostvarila maksimalna sličnost sa veb interfejsima ovih realnih uređaja. Prikupljanjem svakog HTTP zahteva tokom regularnog korišćenja ovih servisa, definišu se liste "regularnihžahteva za svaki od njih. Bilo koji zahtev ka zamci koji nije ranije identifikovan kao "regularanše smatra malicioznim, osim nekih specijalnih slučajeva koji će biti objašnjeni u delu 2.2.2.

Da bi se povećala raznovrsnost, neke zamke su koristile osnovnu HTTP autentifikaciju, dok druge uopšte nisu imale autentifikaciju. Zamke su postavljene u različitim regionima radi raznolikosti jezika geolokacija.

[4] `https://www.ast.co.rs/`

Da bi se postigao visok nivo otkrivanja napada i otežalo otkrivanje zamki vršena je verifikacija. Verifikacija je vršena analiziranjem izveštaja servisa koji su sposobni da razlikuju realne uređaje od zamki vršeći interaktivno skeniranje mreže (npr. shodan.io, censys.io).

2.2.2 Kreiranje korpusa

Kreirani korpus se sastoji iz trening, validacionog i test dela. Nazivamo ga "Korpus za detekciju veb upada baziran na zamkama"(eng. "Trap-based web intrusion detection dataset"ili TBWIDD skraćeno). Prikupljena su dva neobrađena korpusa, koji će služiti za kreiranje delova za treniranje i validaciju. Prvi od njih je od zahteva koje korisnici šalju sajtu, i on je prikupljen u periodu od 15. do 17. maja 2020. godine. Drugi čine zahtevi koje su zamke primile, i prikupljen je u periodu od 18. februara do 24. jula iste godine.

Iako su oba neobrađena korpusa prilično čista, prvi sadrži neke zahteve koji su maliciozni, a drugi sadrži neke zahteve koji su regularni. Broj takvih zahteva je jako mali. Glavni deo čišćenja drugog je već odrađen automatski od strane zamki, proveravanjem da li URL-ovi (eng. Uniform Resource Locator) postoje u listi zahteva koji su prikupljeni tokom regularne interakcije sa zamkama. Sa ciljem da treniramo modele na čistim podacima, bez uticaja šuma, svi zahtevi su verifikovani od strane eksperata iz sajber bezbednosti.

I dalje postoji određeni mali broj specifičnih zahteva koji, u zavisnosti od konteksta, mogu biti i regularni i maliciozni. Neki od njih dati su u tabeli 2.1. Prva dva zahteva iz tabele mogu biti regularna ako su poslata od strane administratora sajta da bi se odradile izvesne promene na sajtu ili bazi podataka, ali se ne očekuje od običnih korisnika da šalju ovakve zahteve, pa su označeni kao maliciozni. Treći zahtev je potpuno regularan za veb sajtove rađene u programskom jeziku PHP. Kako sajt sa koga smo mi prikupili zahteve nije rađen u ovom programskom jeziku, i ovaj zahtev smo označili kao maliciozan. Napadači često šalju ovakve tipove zahteva da bi dobili informacije o tehnologijama koje su korišćene za izradu veb sajtova koje napadaju. Poslednja dva zahteva nisu zahtevi koje korisnici direktno šalju veb sajtu. Ti zahtevi su od strane pregledača i pretraživača veba. Kako oni ne vrše neke maliciozne aktivnosti, ovi zahtevi su označeni kao regularni.

Tabela 2.1: Primeri zahteva iz TBWIDD korpusa koji nisu jasno regularni ili maliciozni

GET /cpanel
GET /phpmyadmin
GET /index.php
GET /robots.txt
GET /manifest.json

Iz oba korpusa su eliminisani duplikati zahteva. Cilj toga je da se izbegne evaluacija modela na primerima koje je on već video u toku treniranja, što može da dovede do lažne slike o većoj tačnosti. Iz svakog od ova dva korpusa je selektovano na slučajan način 70% zahteva koji će biti korišćeni za treniranje, dok je ostatak zahteva korišćen za validaciju.

Za slučaj testiranja na detekciju napada nultog dana, umesto da se zahtevi dele na slučajan način, prvih 70% zahteva (sortiranih po trenutku njihovog prvog pojavljivanja) je selektovano za treniranje, dok je ostatak korišćen za validaciju.

Radi dodatnog testiranja, prikupljen je još jedan korpus sa zahtevima koje su korisnici slali regularnom sajtu. Prikupljanje se odvijalo u periodu od 7. do 12. avgusta 2020. godine. Kao i u prethodnim korpusima, eliminisani su dupli zahtevi. Takođe su eliminisani i svi zahtevi koji se već pojavljuju u skupovima zahteva za treniranje i validaciju. Ostatak zahteva je labeliran tako što su pronađeni svi maliciozni zahtevi među njima (nema puno takvih zahteva). Krajnje veličine kreiranih korpusa za treniranje, validaciju i testiranje date su u tabeli 2.2. U tabeli 2.3 su dati neki primeri zahteva iz kreiranog korpusa.

Tabela 2.2: Broj primera u kreiranom TBWIDD korpusu

Deo korpusa	**Broj regularnih**	**Broj malicioznih**
Treniranje	836	6465
Validacija	359	2772
Testiranje	11853	12

Tabela 2.3: Primeri zahteva iz kreiranog TBWIDD korpusa

Regularni zahtevi
GET /api/areu/v1/cadmuni?muni=Ponticelli%20Toromanides&cyr=true
GET /static/pictures/logo_footer.png
GET /loginopendata
GET /api/dcp/v1/doc
GET /static/swagger-ui-bundle.js
Maliciozni zahtevi
GET /bin/backdoor.html?cmd=cat%20/etc/passwd
GET /admin/config.php
GET /?index.action?redirect:%24%7B1337*1337*1337*1337%7D
GET /shell?cd%20/tmp;wget%20http:/%5C/142.11.215.203/armv7l;%20chmod%207 77%20armv7l;%20./armv7l
GET /api/v3/activities/(SELECT%20(CASE%20WHEN%20(7543=7543)%20THEN %2012%20ELSE%20(SELECT%201383%20UNION%20SELECT%205346)%20END))

Regularni zahtevi koji su prikupljeni sa sajta sadrže poverljive informacije, kao što su korisnička imena stvarnih ljudi i njihovi privatni podaci. Da bi se zaštitila njihova privatnost, svi regularni zahtevi su pseudonimizovani. Sakupljeni su svi termini razbijanjem zahteva karakterima zapete, znaka jednakosti, simbola "&"(eng. ampersand), znakom pitanja, simbolima kose crte (eng. slash) i razmacima. Termini koji su deo jezika veb sajta su prevedeni u odgovarajuće termine iz engleskog jezika. Imena fajlova su ostala nepromenjena. Ostali termini su zamenjeni slučajnim terminima iz engleskog jezika.

Pored kreiranog TBWIDD korpusa, kompletna analiza performansi biće odrađena i na javno dostupnom FWAF korpusu. Tabela 2.4 pokazuje neke primere zahteva iz ovog korpusa. Suprotno kreiranom TBWIDD korpusu, ovaj korpus sadrži i određenu količinu šuma. I ovde su delovi za treniranje i validaciju dobijeni slučajnom podelom regularnih i malicioznih zahteva, tako da 70% iz obe grupe ide u deo za treniranje, dok se ostatak koristi za validaciju. Za testiranje napada nultog dana se prvih 70% zahteva oba tipa koristi za treniranje, a ostatak za validaciju. Nasuprot kreiranom TBWIDD korpusu, u ovom korpusu ne postoji informacija o tome kada je koji zahtev registrovan. U toku izvršavanja eksperimenata, uzeta je pretpostavka da su zahtevi sortirani po vremenu kada su se dogodili, kako u datoteci koji sadrži regularne, tako i u datoteci koja sadrži maliciozne zahteve. Kako se maliciozni zahtevi sličnog tipa i strukture nalaze jedni pored drugih u ovom korpusu, detekcija napada nultog dana na njemu može da bude čak i teža nego na kreiranom TBWIDD korpusu. Broj primera za treniranje i evaluaciju u FWAF korpusu je dat u tabeli 2.5.

Tabela 2.4: Primeri zahteva iz FWAF korpusa

Regularni zahtevi
/javascript/banner.gif
/interface/login/login_frame.php?
/library/fonts/php_helvetica-boldoblique.afm
/docs/api/org/apache/catalina/tribes/cvs/entries
/search-jobs/
Maliciozni zahtevi
/cgi/ion-p?page=../../../../../etc/passwd
/webdav/phprun.php?cmd=c:\wce.exe -h
/phpmyadmin/popup.php?include_path=/etc/passwd\x00
/scripts/admin/file_manager.php?action=read&filename=../../../../../../../../etc/passwd
/showmail.pl?folder=<script>alert(document.cookie)</script>

Tabela 2.5: Broj primera u FWAF korpusu

Deo korpusa	**Broj regularnih**	**Broj malicioznih**
Treniranje	886195	31299
Validacija	379798	13414

2.3 Reprezentacija atributa

HTTP zahtevi koje će modeli klasifikovati su u tekstualnom formatu. Atributi koji će iz njih biti izvučeni su n-grami karaktera (jednogrami, dvogrami i trigrami). Nazivamo

ih tokenima, i biće korišćene tri različite strategije za njihovo izvlačenje iz veb zahteva. N-grami imaju dugu istoriju korišćenja u procesiranju govornog jezika. Kako je HTTP tekstualni protokol, a i Linuks (eng. Linux) komande koje se često koriste u veb napadima su bazirane na ljudskim rečima ili akronimima, jasno je zaključiti zašto su n-grami pogodni i u detekciji napada.

U prvoj strategiji tokenima ćemo smatrati sve uzastopne sekvence karaktera dužine $N = 3$. Za demonstraciju ekstrakcije atributa ćemo koristiti sledeći veb zahtev: "GET /html/..;/api/liferay". Koristeći prvu strategiju, tokeni koji će biti ekstraktovani iz ovog zahteva su: "GET", "ET ", "T /", "/h", "/ht", "htm", "tml", "ml/", "l/.", "/..", "..;", ".;/", ";/a", "/ap", "api", "pi/", "i/l", "/li", "lif", "ife", "fer", "era"i "ray".[5]

Drugom strategijom se ekstraktuju samo jednogrami, tj. svaki karakter predstavlja jedan token. Korišćenjem ove strategije na primeru koji koristimo za demonstraciju, ekstraktovani tokeni bi bili: "G", "E", "T", , "/", "h", "t", "m", "l", "/", ".", ".", ";", "/", "a", "p", "i", "/", "l", "i", "f", "e", "r", "a"i "y".

Treća strategija koristi n-grame različitih dužina. Korišćen je raspon od jednog do tri karaktera. Korišćenjem ove strategije, ekstraktovani tokeni bi bili svi jednogrami, dvogrami i trigrami karaktera iz veb zahteva.

Koristeći bilo koju od ove tri strategije, dobijamo listu tokene za svaki veb zahtev. Od liste tokena kreiramo dve vrste ulaza modela.

Za plitke modele je korišćen TF-IDF (eng. term frequency inverse document frequency) vektor atributa, u kome svakom elementu vektora odgovara tačno jedan n-gram. TF-IDF vektori su izabrani za korišćenje umesto tehnike normalizovanih vektora broja pojavljivanja tokena, zato što oni dodatno uzimaju u obzir i koliko se inače često svaki od tokena pojavljuje u veb zahtevima iz trening skupa. Kako druga strategija ekstraktuje samo individualne karaktere iz zahteva, koji sami po sebi ne sadrže dovoljno informacija za klasifikaciju bez njihovog sekvencijalnog redosleda, kod plitkih modela će biti korišćene samo prva i treća strategija ekstrakcije atributa. U dubokim modelima tokeni će biti obrađivani sekvencijalno. Kako treća strategija nema jasno definisan sekvencijalni redosled tokna, u ovom slučaju će biti korišćene samo prve dve strategije ekstrakcije atributa.

Tokom početnog eksperimentisanja bila je korišćena još jedna strategija za ekstrakciju atributa, kod koje se tokeni kreiraju razbijanjem zahteva na mestima gde se pojavljuju standardni specijalni karakteri HTTP zahteva ('<', '>', '/', '&', '=', ':', '?', '(' i ')'). Ovakav pristup se pokazao manje efikasnim, pa je izbačen iz daljeg razmatranja.

Napadači često koriste i neke enkodirane simbole u svojim napadima ("%2b", "%3A", "%22", "%5C"i slično). Pokušajem da se treniraju i testiraju modeli na zahtevima gde su ovi simboli prethodno dekodirani ustanovljeno je da to nema značajan uticaj na performanse modela, pa su zato zahtevi korišćeni u svojoj originalnoj formi. Ovaj eksperiment je pokazao da su modeli u stanju da sami razumeju ove enkodirane simbole koristeći n-grame.

[5]Zajedno sa ekspertima iz sajber bezbednosti obavljena je diskusija o tome da li standardne HTTP komande, definisane HTTP protokolom (GET, POST, PUT, OPTIONS, itd.), treba da učestvuju u tokenizaciji ili ne. Posle duboke analize malicioznih zahteva, otkriveno je da napadači koriste ručno kreirane komande, koje nisu definisane HTTP protokolom. Zbog toga je odlučeno da se i HTTP komande ostave kao deo zahteva u kreiranom TBWIDD korpusu.

2.3.1 Ulaz plitkih modela

Za kreiranje ulaza plitkih modela, ignorišemo sve tokene iz svih zahteva koji se ne pojavljuju u bar MBP[6] zahteva iz skupa za treniranje. Ovi tokeni neće uticati na kreiranje ulaza. Kada bi MBP bilo jednako jedan, nijedan token ne bi bio ignorisan. Što je veća vrednost MBP, vise tokena se ignoriše. Svrha ovog koraka je smanjivanje veličine ulaza (a samim tim i povećanje brzine procesiranja) izbacivanjem retkih tokena. Dodatno ovo može da pomogne modelima da bolje generalizuju.

Za ulazni vektor kod plitkih modela biće korišćeni TF-IDF atributi. Ulazni vektor imaće veličinu jednaku broju svih različitih tokena iz svih zahteva iz trening skupa, koji nisu ignorisani zbog njihove retkosti. Vrednost vektora na poziciji koja odgovara tokenu t jednaka je proizvodu dve vrednosti. Prva vrednost se računa kao $1 + \log(\text{tf})$, gde tf predstavlja broj koliko puta se ovaj token pojavljuje u veb zahtevu čiji vektor atributa kreiramo. Druga vrednost se koristi da bi ublažila uticaj tokena koji se pojavljuju u velikom broju zahteva, jer ovakvi tokeni obično ne sadrže važne informacije za klasifikaciju. Ovu vrednost računamo kao $\log \frac{1+n}{1+\text{df}(t)} + 1$. Parametar n predstavlja broj zahteva u trening skupu, dok $\text{df}(t)$ predstavlja broj zahteva iz trening skupa koji sadrže token t. Na kraju se ceo ulazni vektor normalizuje koristeći L_2 normalizaciju.

2.3.2 Ulaz dubokih modela

Da bi se obavila analiza veb zahteva dubokim modelima, prvo se svakom tokenu koji se pojavljuje u bar MBP veb zahteva iz trening skupa dodeli jedinstveni identifikacioni broj (ID). Ovi identifikacioni brojevi se kreiraju tako što se tokenima redom dodeljuju uzastopni prirodni brojevi počevši od broja dva. Identifikacioni broj 0 je rezervisan za dopunjavanje (biće obrazloženo kasnije), dok se identifikacioni broj 1 koristi za sve one tokene koji nemaju svoj identifikacioni broj. To uključuje tokene koji se pojavljuju u manje od MBP zahteva iz trening skupa, kao i tokene koji će se prvi put pojaviti u toku testiranja. Kao i u slučaju plitkih modela, MBP je hiperparametar koji može da se podešava.

Duboki modeli na svom ulazu dobijaju zahteve u obliku sekvenci ID-eva. Da bi se od jednog veb zahteva dobila ovakva sekvenca, najpre se iz njega izvuku tokeni. Svaki token koji ima svoj identifikacioni broj se zameni njime, dok se svi ostali tokeni iz veb zahteva zamene identifikacionim brojem 1. Kako duboki modeli procesiraju podatke u grupama (eng. batches) od više zahteva istovremeno, kraće sekvence se dopunjuju sa kraja identifikacionim brojem 0, kako bi sve sekvence iz jedne grupe imale istu dužinu. Nakon toga, ove sekvence ID-eva se koriste da bi se selektovali odgovarajući redovi iz sloja ugradnje, o čemu će više reči biti u delu 2.4.2.

[6]skraćenica za minimalni broj pojavljivanja

2.4 Modeli za detekciju malicioznih zahteva

2.4.1 Plitki modeli

U analizi su korišćeni sledeći plitki modeli mašinskog učenja: logistička regresija, logistička regresija sa L_1 regularizacijom, linearna metoda potpornih vektora (eng. support vector machine, ili skraćeno SVM), pasivno-agresivni klasifikator i klasifikator slučajne šume (eng. random forest classifier). Ovi modeli su često korišćeni u literaturi detekcije napada [35, 73, 91, 12, 27, 45].

Za treniranje klasifikatora logističke regresije je korišćen stohastički gradijentni spust. Za treniranje modela logističke regresije sa L_1 regularizacijom je kotišćen SAGA [25] algoritam, dok je za treniranje linearne metode potpornih vektora korišćena kriterijumska funkcija šarke (eng. hinge loss function).

Pasivno-agresivni klasifikator je detaljno opisan u [23]. Ovaj algoritam mašinskog učenja je pogodan za treniranje na velikom broju primera. U analizi su korišćene obe varijante iz rada, PA-I i PA-II. Algoritam koristi parametar agresivnosti C, koji služi kao balans između što manje promene parametara modela i adaptacije na nove primere.

Klasifikator slučajne šume je sačinjen od više klasifikatora stabla odluke (eng. decision tree). Finalna predikcija klasifikacije se kreira usrednjavanjem verovatnoća predviđanja pojedinačnih stabala. Ovaj algoritam je dao sjajne rezultate u oblasti detekcije napada [12, 27, 117], pa je kao takav korišćen i u ovoj analizi kako bi se videla njegova uspešnost u učenju iz atributa ekstraktovanih iz tekstova veb zahteva.

2.4.2 Duboki modeli

U analizi su korišćena tri duboka modela: LSTM, TextCNN i CNNLSTMCNN. Na početku svih dubokih modela se nalazi adaptivni sloj ugradnje. Ovaj sloj transformiše ulazne grupe sekvenci ID-eva, koje imaju oblik $B \times S$ (B - veličina grupe, S - maksimalna dužina sekvence), u trodimenzioni niz veličine $B \times S \times E$ (E - veličina vektora ugradnje). Cilj ovog sloja je reprezentacija tokena u visoko dimenzioni vektorski prostor pogodan za kreiranje dobrih predikcija. Ove reprezentacije (vektori ugradnje) se uče u toku procesa treniranja, zajedno sa svim ostalim parametrima modela.

LSTM je najkorišćenija arhitektura ćelije rekurentne neuronske mreže. Za razliku od standardnih arhitektura koje su joj prethodile, ona je pokazala znatno bolje rezultate u obradi dužih sekvenci. Ima široku primenu u literaturi detekcije napada [108, 111, 54]. U eksperimentima će biti korišćena dvosmerna varijanta ovog modela. Zadnja stanja skrivenih slojeva oba smera biće nadovezana, i to će dodatno biti procesirano jednim linearnim slojem sa jednim izlaznim neuronom. U ovom neuronu se primenjuje sigmoidalna aktivaciona funkcija. Izlaz iz ovog neurona predstavlja verovatnoću da je veb zahtev maliciozan.

TextCNN model je baziran na radu [60]. Na svom početku sadrži tri 1-dimenziona konvoluciona sloja različitih veličina kernela, koji primenjuju konvoluciju nad dimenzijom koja predstavlja dužinu sekvence. Zatim se nad izlazom svakog od njih primenjuje ReLU nelinearnost, kao i selekcija globalnog maksimuma po dimenziji dužine sekvence. Sva tri izlaza se zatim nadovežu (izlazi su 2-dimenzioni nakon selekcije globalnih mak-

simuma, i njihova prva dimenzija predstavlja veličinu grupe B). Zatim se koristi sloj odustajanja (eng. dropout), a nakon toga jedan linearni sloj sa jednim izlaznim sigmoidalnim neuronom. Možemo da vidimo iz arhitekture modela da on traži šablone na lokalnom nivou (raspon zavisi od veličina kernela), a zatim bira najjače aktivacije ovih šablona po svim mestima duž sekvence. To je pogodno u problemu koji rešavamo, jer na ovaj način možemo da otkrijemo lokalna pojavljivanja malicioznih komandi, a da zatim analizirajući globalno izvedemo zaključke o pojavljivanju komandi na nivou veb zahteva.

CNNLSTMCNN je model koji kombinuje prethodna dva modela. Kao i kod modela TextCNN, na početku ovog modela se nalaze tri 1-dimenziona konvoluciona sloja različitih veličina kernela. Ulazi konvolucionih slojeva su dopunjeni tako da izlazi slojeva sa različitim veličinama kernela imaju jednaku dimenziju koja odgovara dužini sekvence. Zatim se izlazi sva tri konvoluciona sloja konkatenišu po dimenziji koja odgovara broju kernela i primenjuje se ReLU nelinearnost. Izlaz ove operacije je 3-dimenzioni, gde prvoj dimenziji odgovara veličina grupe, drugoj ukupan broj kernela u sva 3 konvoluciona sloja, a trećoj dužina sekvence, i on se koristi kao ulaz dvosmerne rekurentne LSTM mreže. Skriveni slojevi u oba smera se konkatenišu, i to predstavlja ulaz za tri dodatna konvoluciona sloja različitih veličina kernela. Kao i kod TextCNN modela, i ovde se nad izlazom sva tri dodata konvoluciona sloja primenjuje ReLU nelinearnost i selekcija globalnog maksimuma po dimenziji koja odgovara dužini sekvence. Tri dobijena 2-dimenziona izlaza se onda konkatenišu po drugoj dimenziji (koja odgovara broju korišćenih kernela). Na kraju se primenjuje jedan linearni sloj sa jednim izlaznim sigmoidalnim neuronom, kao i kod prethodna dva modela.

2.5 Evaluacija

Plitki modeli korišćeni za detekciju napada su implementirani pomoću Scikit Learn [87] biblioteke za mašinsko učenje. Duboki modeli su implementirani koristeći PyTorch [86] biblioteku. Za treniranje i testiranje plitkih modela je korišćena centralna procesorska jedinica (CPU), dok je za duboke modele korišćena grafička procesorska jedinica (GPU).

Plitki modeli su korišćeni u svojim osnovnim oblicima iz Scikit Learn biblioteke. Svi modeli su kompatibilni sa retkim podacima (eng. sparse), što je pogodno za način ekstrakcije atributa koji koristimo. Za prikupljanje tokena u procesu ekstrakcije atributa korišćeni su isključivo zahtevi iz trening skupa. Korišćena je konstanta za minimalni broj pojavljivanja $MBP = 2$ na prikupljenom TBWIDD korpusu i $MBP = 10$ na FWAF korpusu (razlog veće konstante je veći broj primera u FWAF korpusu).

Slojevi ugradnje u dubokim modelima ugrađuju tokene u 128-dimenzioni vektorski prostor. Kod LSTM modela je korišćena veličina skrivenog stanja od 128. TextCNN model ima tri konvoluciona sloja od po 50 kernela. Svaki sloj ima različitu veličinu kernela, i one iznose 3, 5 i 7. Korišćena je verovatnoća odustajanja od 0.5. CNNLSTMCNN model takođe ima po 50 kernela veličina 3, 5 i 7 u konvolucionim slojevima pre i posle dvosmernog rekurentnog LSTM sloja. Veličina skrivenog stanja rekurentnog sloja je 128.

Trening skup kreiranog TBWIDD korpusa ima 836 regularnih i 6465 malicioznih zahteva, dok trening skup FWAF korpusa ima 886195 regularnih i 31299 malicioznih zahteva. Kako je u oba korpusa vidna razlika između broja regularnih i malicioznih

zahteva, modeli trenirani na njima mogu biti pristrasni u korist klase sa većim brojem primera. Da bi se ublažio ovaj problem, pored standardnog treniranja, implementirane su i tehnike poduzorkovanja (eng. undersampling) i preuzorkovanja (eng. oversampling). U tehnici poduzorkovanja se na slučajan način bira onoliki broj primera iz klase sa više primera koliko ima primera u klasi sa manjim brojem primera. Svi ostali primeri iz klase sa većim brojem primera se izbacuju iz skupa za treniranje. U tehnici preuzorkovanja se u toku kreiranja trening skupa koristi više kopija svih primera iz klase sa manjim brojem primera, umesto samo jedne. Broj kopija je jednak količniku broja primera u klasi sa više i klasi sa manje primera, zaokruženom na donji ceo broj.

Za treniranje dubokih modela je korišćena kriterijumska funkcija binarne unakrsne entropije (eng. binary cross-entropy), čija je formula data u (2.1). Sa X i Y su redom označeni svi zahtevi i labele iz trening skupa, dok oznake malim slovima x_n i y_n predstavljaju pojedinačne primere. Verovatnoća malicioznosti zahteva koju predvidi model je označena sa $f(x_n)$. Kriterijumska funkcija opisuje grešku između tačnih labela i izlaza modela.

$$L_{BCE}(X, Y) = -\frac{1}{|X|}\sum_{n=1}^{|X|} y_n \log f(x_n) + (1 - y_n)\log(1 - f(x_n)) \tag{2.1}$$

Modeli su trenirani po 20 epoha, koristeći veličinu grupe od 64. Svi zahtevi su sortirani po broju tokena koje ekstraktor atributa izvuče iz njih, a u slučaju da dva ili više zahteva ima isti broj tokena korišćen je njihov leksikografski poredak. Nakon sortiranja, susedni zahtevi su redom s početka udruživani u grupe veličine 64, osim poslednje grupe koja može da bude i manje veličine. Ovakvim pristupom kreiranju grupa postiže se da sekvence unutar grupe budu sličnih dužina, što može drastično da ubrza proces treniranja i primene modela. U svakoj epohi se redosled kojim model koristi grupe za treniranje kreira na slučajan način. Kada se koristi tehnika preuzorkovanja, svaki zahtev iz klase sa manjim brojem primera može da se pojavi više puta. Sa ciljem da se izbegne veliki broj pojavljivanja istih zahteva unutar jedne grupe, u ovom slučaju se zahtevi sortiraju samo po broju tokena (bez korišćenja leksikografskog poretka u slučaju da više zahteva ima isti broj tokena). Za treniranje svih dubokih modela je korišćen Adam [61] optimizacioni algoritam, sa veličinom koraka 0.001.

Za poređenje rezultata predikcije različitih modela korišćen je broj lažno pozitivnih primera (eng. false positives), broj lažno negativnih primera (eng. false negatives), kao i metrika geometrijske sredine predikcije, koja je pogodna za poređenje rezultata na korpusima u kojima se broj primera po klasama značajno razlikuje. Često korišćene metrike, kao što je tačnost klasifikacije, često nisu dobar pokazatelj kvaliteta treniranih modela u korpusima gde se broj primera po klasi značajno razlikuje. One mogu da zanemare lošu tačnost modela na primerima iz klase sa manjim brojem primera, ukoliko je tačnost velika na primerima iz klase sa većim brojem primera. Metrika geometrijske sredine uzima u obzir tačnost modela na primerima iz obe klase. Ona koristi broj istinito/lažno pozitivnih/negativnih primera da bi ocenila klasifikator, i formula za njeno izračunavanje je data u (2.2).

$$\begin{aligned} \text{Senzitivnost} &= \frac{\text{IP}}{\text{IP} + \text{LN}} \\ \text{Specifičnost} &= \frac{\text{IN}}{\text{LP} + \text{IN}} \\ \text{G-Sredina} &= \sqrt{\text{Senzitivnost} \cdot \text{Specifičnost}} \end{aligned} \tag{2.2}$$

U formulama je broj istinito pozitivnih primera označen sa **IP**, broj istinito negativnih sa **IN**, broj lažno pozitivnih sa **LP**, a broj lažno negativnih sa **LN**. U tabelama sa rezultatima eksperimenata će se koristiti skraćenica **GS** za metriku geometrijske sredine. Oznake I i II su korišćene da označe dve različite verzije pasivno-agresivnog algoritma koji je opisan u radu [23]. Termini 3gram, karakter i 1-3gram označavaju redom prvu, drugu i treću strategiju ekstrakcije tokena opisanu u delu 2.3.

2.5.1 Klasično testiranje

U tabelama 2.6 su dati rezultati klasičnog testiranja na kreiranom TBWIDD korpusu. U ovom testiranju regularni i maliciozni zahtevi su na slučajan način podeljeni između trening i validacionog skupa. Tabela pokazuje da je većina modela uspela da postigne visoke vrednosti metrike geometrijske sredine.

Rezultati na validacionom skupu su dati u tabeli 2.6a. Metrika geometrijske sredine na validacionom skupu opisuje uspešnost klasifikacije modela na primerima koji imaju istu raspodelu kao i primeri korišćeni za standardno treniranje. LSTM model sa trigramima kao atributima je postigao najveću vrednost geometrijske sredine od 99.55%. Slične, ali nešto slabije, rezultate postigli su i ostali duboki modeli, linearna metoda potpornih vektora i pasivno-agresivni klasifikatori. Preuzorkovanje je unapredilo rezultate klasifikatora logističke regresije (i sa i bez L_1 regularizacije). Klasifikator slučajne šume i linearna metoda potpornih vektora su postigli najbolje rezultate primenom tehnike poduzorkovanja.

Tabela 2.6b prikazuje rezultate na skupu za testiranje. Najbolje rezultate na ovom skupu su postigli logistička regresija, linearna metoda potpornih vektora i TextCNN model, koristeći tehniku poduzorkovanja. Kao i na validacionom skupu, logistička regresija sa L_1 regularizacijom je postigla najbolje rezultate koristeći tehniku preuzorkovanja. Samo je LSTM model ostvario svoj najbolji rezultat koristeći standardno treniranje, što ukazuje na važnost tehnika poduzorkovanja i preuzorkovanja kada se u podacima za treniranje i testiranje značajno razlikuje odnos broja regularnih i malicioznih zahteva. Rezultati na test skupu dobro odslikavaju performanse modela u realnoj primeni, jer ovaj skup sadrži sve zahteve prikupljene u jednom fiksiranom vremenskom periodu, i zbog toga ima realan odnos broja regularnih i malicioznih zahteva. Kako validacioni skup i skup za testiranje imaju drugačiji odnos broja regularnih i malicioznih primera, može se očekivati i određena razliku u dobijenim metrikama geometrijske sredina na ova dva skupa. Posmatrajući rezultate i na validacionom i na test skupu, linearna metoda potpornih vektora koja je trenirana na n-gramima dužine između jedan i tri koristeći tehniku poduzorkovanja je ukupno pokazala najbolje performanse.

Tabela 2.6: Rezultati na kreiranom TBWIDD korpusu

(a) Validacioni skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
LSTM	3gram	**3**	**2**	**99.55**	3	4	99.51	4	2	99.41
TextCNN	3gram	4	2	99.41	**4**	**1**	**99.42**	9	1	98.72
CNNRNNCNN	3gram	5	1	99.28	**4**	**3**	**99.39**	7	2	98.98
LSTM	karakter	5	5	99.21	4	31	98.88	**4**	**3**	**99.39**
SVM	1-3gram	7	2	98.98	**4**	**4**	**99.37**	5	2	99.27
SVM	3gram	7	1	99.0	**4**	**4**	**99.37**	5	3	99.25
Pasivno-Agr. I	1-3gram	**5**	**1**	**99.28**	5	8	99.16	7	2	98.98
Pasivno-Agr. II	1-3gram	**5**	**1**	**99.28**	5	8	99.16	6	2	99.13
TextCNN	karakter	**5**	**2**	**99.27**	5	4	99.23	**5**	**2**	**99.27**
Pasivno-Agr. I	3gram	**5**	**3**	**99.25**	5	11	99.1	7	3	98.97
Pasivno-Agr. II	3gram	**5**	**3**	**99.25**	5	6	99.19	6	2	99.13
CNNRNNCNN	karakter	6	1	99.14	**5**	**7**	**99.18**	6	0	99.16
Slučajna Šuma	1-3gram	8	0	98.88	**6**	**0**	**99.16**	8	0	98.88
Logistička Reg.	1-3gram	10	0	98.6	6	2	99.13	**6**	**1**	**99.14**
Logistička Reg.	3gram	9	0	98.74	6	2	99.13	**6**	**1**	**99.14**
Slučajna Šuma	3gram	8	0	98.88	**6**	**2**	**99.13**	7	0	99.02
Logistička Reg. L_1	3gram	10	2	98.56	8	23	98.47	**6**	**6**	**99.05**
Logistička Reg. L_1	1-3gram	16	2	97.71	7	39	98.32	**7**	**4**	**98.95**

(b) Test skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
Logistička Reg.	1-3gram	44	0	99.81	**4**	**0**	**99.98**	11	0	99.95
SVM	1-3gram	15	0	99.94	**5**	**0**	**99.98**	20	0	99.92
TextCNN	karakter	7	0	99.97	**5**	**0**	**99.98**	24	0	99.9
Logistička Reg. L_1	1-3gram	237	0	99.0	124	1	95.24	**6**	**0**	**99.97**
Pasivno-Agr. I	1-3gram	16	0	99.93	**8**	**0**	**99.97**	32	0	99.86
Pasivno-Agr. II	1-3gram	17	0	99.93	**8**	**0**	**99.97**	30	0	99.87
Logistička Reg. L_1	3gram	324	0	98.62	299	0	98.73	**12**	**0**	**99.95**
CNNRNNCNN	karakter	37	0	99.84	31	0	99.87	**15**	**0**	**99.94**
CNNRNNCNN	3gram	37	0	99.84	**16**	**0**	**99.93**	33	0	99.86
LSTM	3gram	**16**	**0**	**99.93**	44	0	99.81	64	0	99.73
Pasivno-Agr. II	3gram	28	0	99.88	**16**	**0**	**99.93**	34	0	99.86
SVM	3gram	35	0	99.85	**16**	**0**	**99.93**	32	0	99.86
Pasivno-Agr. I	3gram	25	0	99.89	**17**	**0**	**99.93**	34	0	99.86
TextCNN	3gram	18	0	99.92	**17**	**0**	**99.93**	45	0	99.81
LSTM	karakter	**21**	**0**	**99.91**	10	1	95.7	37	0	99.84
Logistička Reg.	3gram	45	0	99.81	**23**	**0**	**99.9**	36	0	99.85
Slučajna Šuma	1-3gram	44	0	99.81	**34**	**0**	**99.86**	38	0	99.84
Slučajna Šuma	3gram	43	0	99.82	**36**	**0**	**99.85**	39	0	99.84

Tabela 2.7: Rezultati na FWAF korpusu

Model	Atributi	Standardno			Poduzorkovanje			Preuzorkovanje		
		LP	LN	GS (%)	LP	LN	GS (%)	LP	LN	GS (%)
TextCNN	karakter	13	14	99.95	1578	6	99.77	**14**	**9**	**99.96**
LSTM	karakter	8	16	99.94	**170**	**4**	**99.96**	5	13	99.95
CNNRNNCNN	karakter	3	69	99.74	**133**	**6**	**99.96**	6	12	99.95
Pasivno-Agr. I	1-3gram	**13**	**37**	**99.86**	472	28	99.83	18	69	99.74
Logistička Reg. L_1	1-3gram	23	76	99.71	762	36	99.77	**72**	**44**	**99.83**
Pasivno-Agr. II	1-3gram	**13**	**47**	**99.82**	508	34	99.81	21	59	99.78
Slučajna šuma	1-3gram	**16**	**83**	**99.69**	2281	39	99.55	67	197	99.25
LSTM	3gram	**48**	**96**	**99.64**	1050	114	99.44	10	942	96.42
Pasivno-Agr. II	3gram	31	186	99.3	3286	76	99.28	**84**	**103**	**99.6**
CNNRNNCNN	3gram	**42**	**112**	**99.58**	1207	108	99.44	10	334	98.75
TextCNN	3gram	**20**	**128**	**99.52**	2125	90	99.38	15	1353	94.82
Logistička Reg. L_1	3gram	40	354	98.67	2177	249	98.78	**146**	**145**	**99.44**
Pasivno-Agr. I	3gram	**37**	**187**	**99.3**	1474	141	99.28	27	200	99.25
SVM	1-3gram	322	1911	92.56	**2243**	**111**	**99.29**	2024	141	99.21
Slučajna šuma	3gram	**372**	**178**	**99.29**	5954	94	98.86	442	190	99.23
SVM	3gram	309	3222	87.13	**2609**	**314**	**98.48**	2276	372	98.31
Logistička Reg.	1-3gram	45	5784	75.41	**4050**	**290**	**98.38**	3903	327	98.26
Logistička Reg.	3gram	28	7452	66.67	**4763**	**585**	**97.18**	4649	693	96.78

Posmatrajući rezultate možemo da zaključimo da izbor strategije ekstrakcije atributa (karakteri ili trigrami) ima blagi uticaj na performanse dubokih modela na kreiranom TBWIDD korpusu. Svi modeli su postigli svoj najbolji rezultat na validacionom skupu koristeći trigrame, ali su TextCNN i CNNRNNCNN modeli ostvarili bolji rezultat na test skupu kada su kao atribute koristili karaktere (jednograme). Skoro svi plitki modeli su postigli svoje najbolje rezultate koristeći n-grame dužine između jedan i tri kao atribute, a jedini izuzetak je L_1 regularizovana logistička regresija na validacionom skupu. Još jedna stvar koju treba pomenuti je da su skoro svi modeli uspeli da detektuju sve maliciozne zahteve na test skupu, uz izuzetak dva modela kada su koristila tehniku poduzorkovanja prilikom treniranja. To je jako dobro, jer su greške detekcije na malicioznim zahtevima obično opasnije nego greške na regularnim zahtevima. Greške na regularnim zahtevima obično kao rezultat imaju odbijanje izvršavanja nekog zahteva regularnog korisnika, što nije dobro, ali je manje opasno po sistem.

Rezultati eksperimentalne evaluacije na FWAF korpusu su dati u tabeli 2.7. Najveću geometrijsku sredinu je postigao TextCNN model sa atributima koji su individualni karakteri iz veb zahteva. On je postigao metriku geometrijske sredine od 99.95% koristeći standardno treniranje, i 99.96% koristeći tehniku preuzorkovanja. Ostali algoritmi koji su postigli dobre rezultate su ostali duboki modeli koristeći karaktere kao atribute, pasivno-agresivni klasifikatori i L_1 regularizovana logistička regresija sa atributima koji su uzastopne sekvence od jednog do tri karaktera iz veb zahteva. LSTM i CNNRNNCNN modeli sa karakterima kao atributima generišu veći broj lažno pozitivnih primera kada koriste tehniku poduzorkovanja. Kako je broj regularnih zahteva dosta veći od broja malicioznih zahteva na ovom skupu, ovi modeli su postigli veću vrednost metrike geo-

metrijske sredine koristeći ovaj pristup. Najgori rezultat je postigla obična logistička regresija, koja nije uspela da detektuje veliki broj malicioznih primera. Linearna metoda potpornih vektora takođe ima veliki broj lažno pozitivnih i lažno negativnih primera na ovom skupu. Oba ova modela su postigla bolje rezultate koristeći tehniku poduzorkovanja.

Suprotno situaciji na TBWIDD korpusu, ovde jasno možemo da vidimo da duboki modeli postižu bolje rezultate kada koriste individualne karaktere kao atribute. Razlog toga može biti što ovaj korpus sadrži mnogo veći broj trigrama, što može negativno da utiče na generalizaciju. Najbolji plitki modeli koriste tokene veličine od jedan do tri. Duboki modeli ne koriste n-grame različitih dužina na ulazu, kako bi osigurali striktan redosled, ali pokazuju bolje rezultate nego plitki modeli kada koriste trigrame kao atribute. Jedini izuzetak je pasivno-agresivni II klasifikator kada koristi tehniku preuzorkovanja. On je postigao slične rezultate kao duboki modeli. FWAF korpus je veliki, a kako duboki modeli obično postižu bolje rezultate na većim korpusima, ovakvi rezultati nisu iznenađujući.

2.5.2 Testiranje na napade nultog dana

Prilikom testiranja na napade nultog dana, umesto slučajne podele zahteva na trening i validacioni deo, ovde se zahtevi (kako regularni tako i maliciozni) koji su registrovani do određenog vremenskog trenutka koriste za treniranje, dok se zahtevi nakon tog trenutka koriste za validaciju. Ovaj pristup bolje odslikava situaciju korišćenja ovih modela u praksi. Oni se prvo treniraju koristeći podatke prikupljene u nekom vremenskom periodu, da bi nakon toga bili korišćeni u zvaničnim sistemima za klasifikaciju novih zahteva koji pristignu. Novi zahtevi mogu kasnije da se koriste za retreniranje modela, kako bi se adaptirao na promene koje se vremenom pojave. Te promene mogu da budu u regularnom saobraćaju, ali i u vidu novih tipova napada koji nisu ranije primećeni.

Tabele 2.8 sadrže rezultate testiranja na napade nultog dana na kreiranom TBWIDD korpusu. Ako pogledamo rezultate na validacionom skupu (tabela 2.8a), možemo da primetimo da su svi modeli uspeli da ostvare metriku geometrijske sredine od preko 94.5% bar sa jednim načinom treniranja. Rezultati su nešto slabiji u odnosu na klasično testiranje, što je i očekivano s obzirom na to da su skupovi za treniranje i validaciju prikupljeni u odvojenim vremenskim intervalima. Najbolju geometrijsku sredinu od 95.95% je postigao LSTM model sa karakterima kao atributima, ali su i drugi modeli kao što su TextCNN i linearna metoda potpornih vektora ostvarili dobre rezultate. Najbolji rezultati na test skupu (tabela 2.8b), koji je isti skupu koji je korišćen i kod klasičnog testiranja, su uglavnom postignuti korišćenjem tehnike poduzorkovanja u procesu treniranja. Svi modeli su uspeli da detektuju sve maliciozne zahteve, i njihove performanse se jedino razlikuju po broju regularnih zahteva koji su greškom klasifikovani kao maliciozni. Pasivno-agresivni klasifikator, logistička regresija, linearna metoda potpornih vektora i TextCNN model generišu manje od 10 pogrešno klasifikovanih primera na ovom skupu sa bar jednim načinom treniranja.

Rezultati testiranja na napade nultog dana na FWAF korpusu su dati u tabeli 2.9. Kako su maliciozni zahtevi istog tipa u ovom korpusu uglavnom jedni do drugih, klasifikacija zahteva koji su posle neke pozicije u korpusu na osnovu znanja koje je prikupljeno

Tabela 2.8: Rezultati testiranja na napade nultog dana na kreiranom TBWIDD korpusu

(a) Validacioni skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
LSTM	karakter	**28**	**4**	**95.95**	32	7	95.32	33	7	95.17
TextCNN	karakter	39	3	94.36	38	6	94.46	**32**	**16**	**95.16**
LSTM	3gram	**31**	**28**	**95.1**	32	49	94.59	90	6	86.47
SVM	1-3gram	40	1	94.25	**32**	**31**	**94.9**	37	5	94.62
Logistička Reg.	3gram	40	0	94.26	**35**	**8**	**94.86**	40	2	94.23
TextCNN	3gram	**35**	**9**	**94.85**	35	25	94.57	39	18	94.11
CNNRNNCNN	3gram	**35**	**10**	**94.83**	34	111	93.22	38	5	94.47
Slučajna šuma	1-3gram	40	0	94.26	**36**	**2**	**94.82**	39	0	94.41
Logistička Reg. L_1	3gram	40	2	94.23	**35**	**11**	**94.81**	36	7	94.73
CNNRNNCNN	karakter	38	6	94.46	35	50	94.14	**36**	**3**	**94.8**
Slučajna šuma	3gram	40	0	94.26	**36**	**4**	**94.79**	39	1	94.4
Logistička Reg.	1-3gram	41	0	94.12	**37**	**5**	**94.62**	39	2	94.38
Pasivno-Agr. I	3gram	**37**	**6**	**94.6**	30	135	93.37	37	14	94.47
Pasivno-Agr. II	3gram	**37**	**6**	**94.6**	33	58	94.29	37	11	94.52
SVM	3gram	40	1	94.25	32	59	94.42	**37**	**6**	**94.6**
Pasivno-Agr. I	1-3gram	**37**	**7**	**94.59**	31	144	93.07	37	17	94.42
Logistička Reg. L_1	1-3gram	40	2	94.23	39	20	94.07	**37**	**11**	**94.52**
Pasivno-Agr. II	1-3gram	**37**	**11**	**94.52**	33	86	93.8	37	17	94.42

(b) Test skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
Pasivno-Agr. I	3gram	24	0	99.9	**4**	**0**	**99.98**	31	0	99.87
Logistička Reg.	1-3gram	45	0	99.81	**5**	**0**	**99.98**	16	0	99.93
Pasivno-Agr. I	1-3gram	30	0	99.87	**5**	**0**	**99.98**	35	0	99.85
Pasivno-Agr. II	3gram	29	0	99.88	**6**	**0**	**99.97**	31	0	99.87
SVM	1-3gram	13	0	99.95	**6**	**0**	**99.97**	15	0	99.94
SVM	3gram	20	0	99.92	**8**	**0**	**99.97**	21	0	99.91
TextCNN	karakter	34	0	99.86	17	0	99.93	**8**	**0**	**99.97**
Pasivno-Agr. II	1-3gram	32	0	99.86	**10**	**0**	**99.96**	35	0	99.85
CNNRNNCNN	3gram	**14**	**0**	**99.94**	16	0	99.93	48	0	99.8
Logistička Reg.	3gram	45	0	99.81	**20**	**0**	**99.92**	39	0	99.84
LSTM	karakter	56	0	99.76	32	0	99.86	**30**	**0**	**99.87**
LSTM	3gram	**31**	**0**	**99.87**	35	0	99.85	5030	0	75.87
CNNRNNCNN	karakter	37	0	99.84	**32**	**0**	**99.86**	**32**	**0**	**99.86**
Slučajna šuma	1-3gram	44	0	99.81	**33**	**0**	**99.86**	42	0	99.82
TextCNN	3gram	37	0	99.84	**35**	**0**	**99.85**	42	0	99.82
Slučajna šuma	3gram	43	0	99.82	**37**	**0**	**99.84**	40	0	99.83
Logistička Reg. L_1	1-3gram	318	0	98.65	232	0	99.02	**122**	**0**	**99.48**
Logistička Reg. L_1	3gram	329	0	98.6	314	0	98.67	**291**	**0**	**98.76**

Tabela 2.9: Rezultati testiranja na napade nultog dana na FWAF korpusu

Model	Atributi	Standardno			Poduzorkovanje			Preuzorkovanje		
		LP	LN	GS (%)	LP	LN	GS (%)	LP	LN	GS (%)
TextCNN	karakter	9	250	99.06	526	194	99.21	**37**	**118**	**99.55**
LSTM	karakter	4	1832	92.92	**69**	**231**	**99.13**	3	355	98.67
CNNRNNCNN	karakter	10	1510	94.2	**51**	**259**	**99.02**	5	755	97.14
Pasivno-Agr. II	3gram	36	9038	57.11	1930	4983	79.08	**69**	**682**	**97.42**
SVM	1-3gram	629	9984	50.53	**2657**	**1230**	**94.97**	2457	4401	81.7
Pasivno-Agr. I	1-3gram	9	3928	84.09	**302**	**1921**	**92.53**	11	5437	77.11
LSTM	3gram	33	8068	63.13	**3476**	**2578**	**89.47**	8	11102	41.52
Pasivno-Agr. II	1-3gram	12	5164	78.42	**327**	**3082**	**87.73**	13	4829	80.0
Slučajna šuma	1-3gram	**38**	**4537**	**81.35**	2765	4927	79.25	121	7561	66.05
Logistička Reg. L_1	1-3gram	15	6929	69.53	**533**	**4603**	**80.99**	53	5304	77.75
CNNRNNCNN	3gram	**63**	**4856**	**79.87**	1343	6934	69.38	8	11005	42.38
TextCNN	3gram	36	6877	69.81	**1404**	**6840**	**69.88**	14	11345	39.27
Logistička Reg. L_1	3gram	37	10407	47.34	**2209**	**7103**	**68.39**	123	9552	53.65
Pasivno-Agr. I	3gram	26	10239	48.65	**1397**	**8001**	**63.41**	13	10416	47.27
Logistička Reg.	1-3gram	143	12831	20.84	**6068**	**8403**	**60.63**	5831	10336	47.53
Slučajna šuma	3gram	**418**	**9524**	**53.82**	3143	9907	50.92	523	9583	53.4
SVM	3gram	705	12108	31.17	**3147**	**10491**	**46.49**	2718	10999	42.28
Logistička Reg.	3gram	84	13139	14.32	**6488**	**11348**	**38.91**	5969	11822	34.18

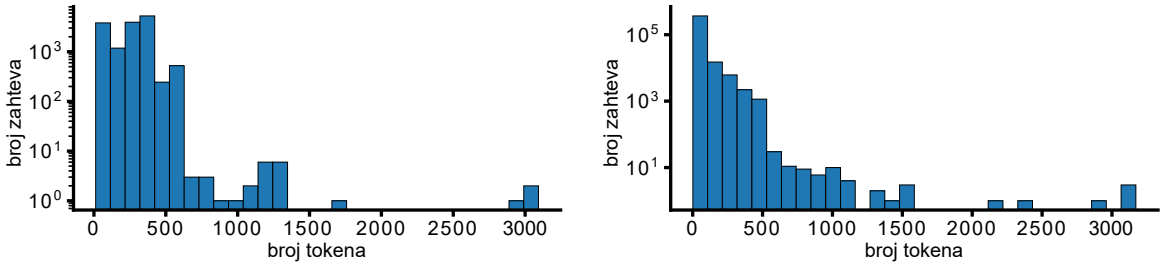
na osnovu zahteva pre te pozicije je težak problem. Za veliki broj modela ovo je bio previše težak zadatak, što se najbolje odslikava po broju malicioznih zahteva koji su pogrešno klasifikovani kao regularni. Najbolje rezultate su postigli duboki modeli, koristeći individualne karaktere kao atribute. Kako se tipovi malicioznih zahteva značajno razlikuju u trening i validacionom skupu, kompaktna reprezentacija sa malim brojem različitih tokena (karaktera) je izuzetno pogodna u ovom slučaju. Najbolji među dubokim modelima sa karakterima kao atributima je TextCNN model, koji je uspeo da postigne metriku geometrijske sredine od preko 99% sa svakim od tri načina treniranja (standardni, poduzorkovanje i preuzorkovanje). Ovaj model je postigao sjajne rezultate i u klasičnom testiranju i u testiranja na napade nultog dana na oba korpusa.

2.5.3 Vreme treniranja i predikcije

Još jedan aspekt na koji treba obratiti pažnju prilikom odabira modela je vreme potrebno za njegovo treniranje, kao i brzina kojom vrši predikcije nakon što je istreniran. U tabeli 2.10 su prikazana ova vremena za sve modele na oba korpusa. Vremena potrebna za treniranje modela su izražena u sekundama, dok su vremena potrebna za vršenje predikcije izražena u mikrosekundama. Vreme potrebno za vršenje predikcije na TBWIDD korpusu predstavlja vreme potrebno da se izvrše predikcije za sve zahteve iz validacionog i test skupa, podeljeno sa ukupnim brojem zahteva u ova dva skupa. Kako FWAF korpus nema test skup, kod njega je to vreme koje je potrebno da se izvrše predikcije za sve zahteve iz validacionog skupa, podeljeno sa brojem zahteva u validacionom skupu. Data vremena su dobijena prilikom klasičnog testiranja, bez korišćenja tehnika poduzorkovanja ili preuzorkovanja.

Tabela 2.10: Vreme potrebno za treniranje i vršenje predikcije na TBWIDD i FWAF korpusima

Model	**Atributi**	**Treniranje TBWIDD (s)**	**Predikcija TBWIDD (μs)**	**Treniranje FWAF (s)**	**Predikcija FWAF (μs)**
CNNRNNCNN	3gram	62.13	267.51	6091.91	109.28
CNNRNNCNN	karakter	62.23	253.75	5788.52	102.03
LSTM	3gram	44.48	241.12	3891.23	69.29
LSTM	karakter	43.47	236.09	3676.56	69.29
Logistička Reg.	1-3gram	0.46	135.98	34.98	35.22
Logistička Reg.	3gram	0.19	54.55	15.22	15.07
Logistička Reg. L_1	1-3gram	1.55	133.97	258.32	34.96
Logistička Reg. L_1	3gram	0.74	51.79	469.94	14.74
Pasivno-Agr. I	1-3gram	0.45	136.27	34.38	35.05
Pasivno-Agr. I	3gram	0.19	54.76	15.01	14.83
Pasivno-Agr. II	1-3gram	0.46	136.39	34.1	35.32
Pasivno-Agr. II	3gram	0.18	54.4	15	14.79
Slučajna šuma	1-3gram	1.35	167.79	608.36	103.06
Slučajna šuma	3gram	0.9	68.84	921.2	199.81
SVM	1-3gram	0.44	137.36	34	35.53
SVM	3gram	0.18	54.77	14.66	14.91
TextCNN	3gram	21.6	94.09	2539.4	45.94
TextCNN	karakter	20.49	86.63	2419.02	43.25



(a) TBWIDD (b) FWAF

Slika 2.1: Broj ekstraktovanih tokena po zahtevu

Kao što je i očekivano, dubokim modelima je potrebno više vremena za treniranje nego plitkim modelima. Na FWAF korpusu, logističkoj regresiji sa L_1 regularizacijom i klasifikatoru slučajne šume je potrebno dosta više vremena za treniranje nego ostalim plitkim modelima. Za praktičnu primenu je vreme vršenja predikcije važnije, jer ono predstavlja koliko brzo već trenirani model može da klasifikuje zahteve kada se stavi u rad kao deo sistema. Kada uporedimo ovo vreme kod dubokih i plitkih modela, razlike su dosta manje. Na primer, klasifikator slučajne šume sa trigramima kao atributima je na FWAF korpusu vršio klasifikaciju zahteva sporije nego bilo koji duboki model. Među tri duboka modela, TextCNN model se pokazao kao najbrži u vršenju predikcija. Kako različiti linearni modeli vrše predikcije na isti način, i njihova vremena vršenja predikcija su slična.

Slika 2.1 opisuje broj tokena po zahtevu. Za generisanje grafika je korišćena strategija ekstrakcije u kojoj se tokenima smatraju sve uzastopne sekvence karaktera dužine između jedan i tri. Ostale dve strategije ekstrakcije imaju približno 3 puta manje tokena po zahtevu. Nisu uključeni zahtevi koji se koriste za treniranje, jer oni ne utiču na izračunate brzine vršenja predikcije. Kako ova strategija ekstraktuje više tokena po zahtevu nego druge dve, modeli sporije klasifikuju zahteve kada je koriste.

Poglavlje 3

Inkrementalno učenje klasifikatora HTTP zahteva

U poslednjim decenijama, polje neuronskih mreža je doživelo značajan napredak i izazvalo revoluciju u mnogim oblastima, kao što su računarski vid, obrada prirodnog jezika i robotika. Iako postižu izvanredne rezultate u statičkim zadacima, neuronske mreže se suočavaju sa značajnim izazovima u dinamičkim okruženjima, u kojima se podaci brzo menjaju. U ovakvim situacijama modeli treba vremenom da se adaptiraju na nove tipove podataka, uz zadržavanje prethodno stečenog znanja.

Inktementalno učenje (eng. incremental learning), često nazivano i kontinualnim učenjem (eng. continual learning) ili učenjem tokom života (lifelong learning), predstavlja pristup koji omogućava neuronskim mrežama da uče nove zadatke u sekvencijalnom toku podataka, dok istovremeno zadržavaju prethodno stečeno znanje. Kod ovog pristupa se model samo dotrenira korišćenjem novih primera, umesto potpunog retreniranja na svim podacima koje je potrebno kod standardnog učenja.

Ovaj pristup postaje posebno relevantan u kontekstu sajber bezbednosti, jer napadi i pretnje često evoluiraju i prilagođavaju se novim metodama. U dinamičnim i stalno promenljivim okruženjima sajber bezbednosti, sistemi moraju efikasno da prepoznaju nove vrste napada i zlonamernih aktivnosti, dok istovremeno čuvaju svoju sposobnost da prepoznaju već poznate pretnje. Zamke nam pomažu da kontinualno sakupljamo podatke o najnovijim tipovima napada, dok nam inkrementalno učenje dodatno omogućava da model to znanje usvaja takođe na kontinualan način.

U ovom poglavlju ćemo razmotriti osnovne principe inkrementalnog učenja u kontekstu sajber bezbednosti. Jedan od ključnih problema u inkrementalnom učenju je katastrofalno zaboravljanje (eng. catastrophic forgetting), a situacija nije drugačija ni kod modela za detekciju veb napada. Nakon treniranja na novom skupu primera, model gubi znanje koje je stekao tokom treniranja na ranijim skupovima primera, i nije u stanju da ih ispravno klasifikuje, iako je pre treniranja na novom skupu to mogao. Istražićemo različite tehnike i strategije razvijene za ublažavanje ovog problema, uključujući metode zasnovane na memoriji, regularizaciji i adaptaciji arhitekture. Najpre ćemo se nadovezati na ideje iz [51], i biće implementiran model koji inkrementalno uči na primerima iz zamki i iz regularnog saobraćaja, uz značajno umanjeni uticaj katastrofalnog zaboravljanja. Osnovna ideja ovog pristupa je zamrzavanje krajnjeg sloja binarnog klasifikatora neu-

ronske mreže nakon inicijalnog treniranja, i adaptacija ostalih slojeva i rečnika n-grama. Takođe će biti predstavljen i pristup inkrementalnom učenju koji koristi mali skup prethodno viđenih primera kako bi redukovao katastrofalno zaboravljanje. Na kraju će biti kreiran i model koji nadovezuje ideje iz ova dva pristupa.

3.1 Pregled literature

Za razliku od standardnih tehnika mašinskog učenja, primena tehnika inrementalnog učenja u svrhu detekcije veb napada je dosta manje istražena. Jedan takav model su predložili Burbeck i Nadjm-Tehrani [20]. Model se sastoji iz klastera, koji se nalaze u listovima stabla. Svaki unutrašnji čvor stabla sumira sve listove iz svog podstabla. Svaki klaster je opisan brojem primera koje sadrži, njihovom sumom i sumom njihovih kvadrata. Primeri moraju biti predstavljeni kao vektori realnih brojeva, i moguće ih je inkrementalno dodavati. Samo primeri iz regularnog saobraćaja se koriste u modelu. U toku testiranja, bira se klaster iz stabla koji je najbliži datom primeru za koji se vrši predikcija. Ukoliko je distanca veća od unapred određenog limita, primer se klasifikuje kao maliciozni, a u suprotnom kao regularni.

Data i Aritsugi su predstavili model detekcije upada [24] koji inkrementalno uči o novim tehnikama koje napadači koriste. Model se sastoji iz više dubokih neuronskih mreža, koje su međusobno povezane u strukturu nalik stablu. Eksperimentalnom evaluacijom su pokazali da ovaj model može da uči inkrementalno i da smanjuje uticaj katastrofalnog zaboravljanja.

3.2 Korpus

Za treniranje i testiranje modela će biti korišćeni isti korpusi koji su korišćeni i u prethodnom poglavlju, naš TBWIDD korpus i FWAF korpus. Međutim, za inkrementalno učenje će nam biti potrebno više manjih hronološki poređanih mini-korpusa.

Modeli će za treniranje mini-korpuse koristiti jedan po jedan. Cilj teniranja na jednom mini-korpusu biće adaptacija modela da dobro klasifikuje njegove primere, uz što manje zaboravljanje stečenog znanja iz prethodnih mini-korpusa. Koristiće se isti oni duboki modeli koji su korišćeni za neinkrementalno treniranje i testiranje (iz prethodnog poglavlja), uz jedinu razliku da će kod TextCNN modela biti izbačen sloj odustajanja pre poslednjeg sloja.

Pre kreiranja mini-korpusa za inkrementalno treniranje, najpre će i regularni i maliciozni zahtevi biti sortirani kao što je to bilo učinjeno kod testiranja na napade nultog dana. Zatim će zahtevi oba tipa (regularni i maliciozni) biti podeljeni u isti broj grupa međusobno susednih zahteva. Među grupama nema preklapanja, i savki zahtev pripada tačno jednoj grupi. Broj zahteva u grupi među grupama istog tipa može da se razlikuje za najviše jedan zahtev.

Iz svake grupe zahteva (i iz regularnih i iz malicioznih grupa) na slučajan način će biti odvojeno 30% zahteva za kreiranje validacionog skupa (validacioni skup će činiti odvojeni zahtevi iz svih regularnih i svih malicioznih grupa). Nakon toga ostaće nam sekvenca grupa regularnih zahteva i sekvenca grupa malicioznih zahteva, gde svaka grupa sadrži

70% svojih originalnih zahteva. Da bi kreirali mini-korpuse za treniranje, uparivaćemo redukovane grupe regularnih i malicioznih zahteva koje se nalaze na istoj poziciji u svojim sekvencama (sekvence grupa regularnih i grupa malicioznih zahteva su iste dužine).

Kako je FWAF korpus veći od našeg TBWIDD korpusa, on će biti podeljen na 50 mini-korpusa za treniranje, dok će naš korpus biti podeljen na 10. Skup za testiranje koji će se koristiti kod inkrementalnog učenja na našem TBWIDD korpusu biće identičan onom iz prethodnog poglavlja.

3.3 Model baziran na učenju sa manjim zaboravljanjem

U ovoj sekciji ćemo iskoristiti i nadograditi ideje predstavljene u radu učenja sa manjim zaboravljanjem (eng. less-forgetful learning) [51]. Za razliku od inicijalnog rada gde se vrši klasifikacija slika, zadatak našeg modela biće da inkrementalno uči i klasifikuje HTTP zahteve, koji su predstavljeni sekvencama n-grama karaktera (jednograma ili trigrama) koje se ugrađuju u vektorski prostor. Takođe, mi nemamo striktno definisanu tranziciju iz jednog domena u drugi, već primeri iz novog mini-korpusa mogu, ali ne moraju, da pripadaju novom domenu ulaznog prostora.

Poslednji sloj, kao probabilistički linearni klasifikator, ima samo jedan neuron sa sigmoidalnom aktivacionom funkcijom u svim predloženim modelima neuronskih mreža. Ovaj sloj radi u prostoru atributa ekstraktovanih od strane pretposlednjeg sloja. Hiperravan binarnog klasifikatora koja deli prostor je jedinstveno određena težinama vektora neuronskih sinapsi poslednjeg sloja. Vektor težina je ortogonalan na klasifikacionu hiperravan. Promena u ovim sinaptičkim težinama bi izazvala promenu pozicije hiperravni, što može dovesti do toga da neki primeri iz prethodnih mini-korpusa budu pogrešno klasifikovani, što predstavlja katastrofalno zaboravljanje prethodno akumuliranog znanja. Da bi minimizirali katastrofalno zaboravljanje, nakon treniranja na prvom mini-korpusu, zamrznućemo parametre poslednjeg sloja. Dodatno, treniraćemo modele na novim primerima tako da promena u aktivacijama pretposlednjeg sloja bude što manja moguća u odnosu na aktivacije tih novih primera pre treniranja.

Jedna od stvari koju će biti potrebno implementirati je inkrementalni rečnik n-grama. Da bi to postigli, naš vokabular će sadržati dve kolekcije, jednu sa n-gramima koji su već bili prisutni u bar MBP zahteva (među svim mini-korpusima za trening procesiranih do tog trenutka), i jednu sa svim drugim n-gramima viđenim do tog trenutka, uz broj zahteva u kojima je svaki od njih bio viđen. U svakom koraku ćemo ažurirati drugu kolekciju, i prebaciti n-grame koji su viđeni bar MBP puta u prvu kolekciju.

Označimo sa u sve parametre poslednjeg sloja neuronske mreže, a sa w sve druge parametre modela. Označimo sa $f^L(x; w, u, V)$ aktivaciju izlaznog neurona poslednjeg sloja mreže čiji su parametri w i u, koja koristi ekstraktor atributa sa rečnikom n-grama V kako bi kreirala ulaz na osnovu ulaznog zahteva x. U našem slučaju binarne klasifikacije, izlazni neuron u poslednjem sloju koristi sigmoidalnu aktivacionu funkciju. Označimo sa velikim X i Y jedan individualni mini-korpus za treniranje i njegove labele. Malim x_n i y_n označavamo indivudualne zahteve i labele iz mini-korpusa, gde broj n u subskriptu označava poziciju zahteva u mini-korpusu. Broj u superskriptu će označavati poziciju

mini-korpusa u listi svih mini-korpusa korišćenih za treniranje (X^m, x_n^m ili y_n^m). $|X| = \text{card}(X)$ predstavlja broj zahteva u mini-korpusu za treniranje X ($|X^m| = \text{card}(X^m)$).

Model se trenira na prvom mini-korpusu za treniranje na isti način kao kod klasičnog (neinkrementalnog) treniranja, koristeći kriterijumsku funkciju binarne unakrsne entropije (3.1), kao što je prikazano jednačinom (3.2).

$$L_{BCE}(X, Y; w, u, V) = -\frac{1}{|X|}\sum_{n=1}^{|X|} y_n \log f^L(x_n; w, u, V) + (1 - y_n)\log(1 - f^L(x_n; w, u, V)) \tag{3.1}$$

$$w^1, u^* = \arg\min_{w,u} L_{BCE}(X^1, Y^1; w, u, V^1) \tag{3.2}$$

Nakon toga, poslednji sloj će biti zamrznut, tj. neće biti dozvoljeno da se njegove težine u^* menjaju tokom treniranja na narednim mini-korpusima. Označimo sa w^m sve težine modela osim težina u poslednjem sloju, nakon što je model završio treniranje na m-tom mini-korpusu. V^m će predstavljati rečnik n-grama kreiran na osnovu prvih m mini-korpusa za treniranje.

Označimo sa $f^{L-1}(x; w, V)$ aktivacije pretposlednjeg sloja za ulazni zahtev x, parametre modela w i vokabular n-grama V. Treniranje na svim narednim mini-korpusima ($m > 1$) vršiće se optimizacijom kriterijumske funckije (3.4).

$$L_E(X^m; w^{m-1}, w, V^{m-1}, V^m) = \\ = \frac{1}{2|X^m|}\sum_{n=1}^{|X^m|} \left\| f^{L-1}(x_n^m; w^{m-1}, V^{m-1}) - f^{L-1}(x_n^m; w, V^m) \right\|_2^2 \tag{3.3}$$

$$w^m = \arg\min_{w} L_{BCE}(X^m, Y^m; w, u^*, V^m) + \lambda(m-1)L_E(X^m; w^{m-1}, w, V^{m-1}, V^m) \tag{3.4}$$

Prvi deo kriterijumske funckije predstavlja funkciju greške unakrsne entropije L_{BCE}, i on je zadužen da nauči model da ispravno klasifikuje zahteve iz trenutnog (m-tog) mini-korpusa.

Drugi deo kriterijumske funkcije predstavlja Euklidovu funkciju greške L_E (3.3), i njegov zadatak je da minimizuje promenu aktivacija neurona u pretposlednjem sloju po svim primerima iz mini-korpusa za treniranje. Za svaki primer, rastojanje se računa između aktivacija verzije modela koja je dobijena nakon treniranja na prvih $m-1$ mini-korpusa (model sa parametrima w^{m-1} i rečnikom V^{m-1}), i trenutne verzije modela (za koju se uče parametri w i koja koristi rečnik V^m). Koristeći ovu dodatnu funkciju greške, mi podstičemo neuronsku mrežu da uči da ekstraktuje atribute, koje predstavljaju aktivacije pretposlednjeg sloja, slične ekstraktovanim atributima neuronske mreže dobijene treniranjem na prvih $m-1$ mini-korpusa. Ovo radimo zato što je klasifikaciona hiperravan, definisana parametrima u poslednjem sloju, zamrznuta i treba da ostane ispravna granica odluke za zahteve iz prethodnih mini-korpusa. Kako u ovom pristupu ne koristimo zahteve iz prethodnih mini-korpusa, L_E ćemo optimizovati sa zahtevima iz trenutnog (m-tog) mini-korpusa, kao aproksimaciju.

Kako je prvi deo kriterijumske funckije zadužen za prikupljanje znanja iz novog m-tog mini-korpusa za treniranje, a drugi za čuvanje već stečenog znanja iz prvih $m-1$ mini-korpusa, koristićemo hiperparametar λ da bi balansirali između ova dva cilja. Dodaćemo

takođe i član $(m-1)$ u kriterijumsku funkciju, sa ciljem da smanjimo pristrasnost prema zahtevima iz skorijih mini-korpusa.

Rečnik V^m sadrži sve n-grame iz rečnika V^{m-1} (i u istom redosledu), ali može da sadrži i neke nove n-grame posle njih. Zbog toga može da se desi da w sadrži više vektora ugradnje nego w^{m-1}. Pre treniranja na m-tom mini-korpusu, ti novi vektori uradnje biće inicijalizovani na slučajan način, dok će svi ostali vektori ugradnje biti inicijalizovani njima odgovarajućim vrednostima iz w^{m-1}.

Treba napomenuti da nije neophodno da postoje dva odvojena modela u memoriji tokom treniranja. Koristeći model koji je završio svoje treniranje na $(m-1)$-om mini-korpusu, najpre izračunavamo vrednosti $f^{L-1}(x_n^m; w^{m-1}, V^{m-1})$ za sve zahteve x_n^m iz X^m. Nakon toga ažuriramo rečnik do stanja V^m koristeći rečnik u stanju V^{m-1} i zahteve iz novog mini-korpusa za treniranje, kao što je ranije objašnjeno. Novodobijeni rečnik može da ima više n-grama nego što sloj ugradnje iz w^{m-1} ima redova, pa ćemo zato dodati na w^{m-1} po jedan novi vektor ugradnje za svaki novi n-gram iz rečnika, i time dobiti novi skup parametara w. Sada imamo sve što nam je potrebno da bi vršili treniranje na m-tom mini-korpusu, i to činimo optimizacijom kriterijumske funkcije 3.4.

3.4 Modeli koji koriste bafer

Bafer (eng. buffer) predstavlja deo računarske memorije koji je izdvojen da bi se u njega čuvali podaci potrebni za međukorake pri izvođenju određenih radnji. U kontekstu inkrementalnog učenja, funkcija bafera je u čuvanju izvesnog broja ranije viđenih primera u cilju stabilnije akumulacije znanja tokom vremena. Koristićemo dve strategije inkrementalnog učenja koje su zasnovane na baferu. Jedna od njih se nadograđuje na učenje sa manjim zaboravljanjem, koje je opisano u prethodnom poglavlju, dodavanjem malog bafera sa prethodno viđenim primerima. Druga strategija je zasnovana isključivo na baferu.

Bafer koji će biti korišćen za obe strategije se sastoji iz dva dela sa mestom za jednaki broj elemenata koje može da uskladišti. Jedan deo služi za skladištenje malicioznih, a drugi za skladištenje regularnih zahteva. Oba dela se pune koristeći uzorkovanje iz rezervoara (eng. reservoir sampling) [103], koje daje svim viđenim primerima jednaku šansu da budu uskladišteni. Kada pokušamo da usnimimo novi primer u bafer, a deo bafera koji je određen za njegovo skladištenje (regularni ili maliciozni) nije popunjen, primer se samo doda u bafer. Ukoliko je njegov deo bafera pun, primer se skladišti na poziciju drugog primera iz istog dela bafera sa verovatnoćom jednakom količniku između veličine tog dela bafera i ukupnog broja primera za koje je pokušano dodavanje u tom delu bafera (uključujući i trenutni primer). U tom slučaju, primer ima jednaku šansu da zameni bilo koji od uskladištenih primera iz svog dela bafera. Obe inkrementalne strategije pokušavaju da uskladište primere iz svakog mini-korpusa nakon što završe treniranje sa njim. Takođe, obe strategije koriste isti inkrementalni rečnik i sloj ugradnje u kome raste broj redova vremenom kao i model učenja sa manjim zaboravljanjem koji ne koristi bafer.

Strategija učenja sa manjim zaboravljanjem koja koristi bafer čuva tri vrednosti za svaki zahtev: tekst zahteva, labelu i vektor sa aktivacijama pretposlednjeg sloja. Ako

je zahtev iz m-tog mini-korpusa za treniranje, aktivacije koje se čuvaju u baferu se računaju koristeći verziju modela koja se dobija nakon završenog treniranja na m-tom mini-korpusu. Trening na prvom mini-korpusu je isti kao i u verziji modela koja ne koristi bafer. Nakon toga se poslednji sloj zamrzava, kao i kod strategije bez bafera. Razlika je prilikom treniranja na narednim mini-korpusima. Ova verzija strategije koristi i primere iz trenutnog mini-korpusa i sve primere iz bafera (uključujući i regularne i maliciozne) za treniranje. Označimo sa $\overline{X}^m$, $\overline{Y}^m$ i $\overline{R}^m$ redom sve primere (i regularne i maliciozne), njihove labele i njihove aktivacije iz pretposlednjeg sloja, koji se nalaze u baferu pre treniranja na m-tom mini-korpusu (bafer u ovom trenutku sadrži neke od primera iz prvih $m-1$ mini-korpusa). Označimo malim slovima $\overline{x}_n^m$, $\overline{y}_n^m$ i $\overline{r}_n^m$ odgovarajuće individualne elemente iz bafera.

Prvi deo kriterijumske funkcije je sličan kao u 3.4, uz jedinu razliku što se sada koriste primeri i iz trenutnog mini-korpusa i iz bafera, umesto samo primera iz trenutnog mini-korpusa, što je bio slučaj kod startegije učenja sa manjim zaboravljanjem bez bafera. Prvi deo sada možemo da zapišemo kao $L_{BCE}(X^m \cup \overline{X}^m, Y^m \cup \overline{Y}^m; w, u^*, V^m)$. Drugi deo kriterijumske funkcije je sada nešto drugačiji, zato što se minimizuju rastojanja vektora aktivacija pretposlednjeg sloja u odnosu na različite vektore za primere iz trenutnog mini-korpusa i za primere iz bafera. Za primere iz trenutnog mini-korpusa, ovo rastojanje se računa u odnosu na aktivacije pretposlednjeg sloja verzije modela i rečnika kakvi su postojali pre treniranja na trenutnom mini-korpusu (isto rastojanje koje je korišćeno i u 3.3). Računanje ovog rastojanja za primere koji se nalaze u baferu se vrši nešto drugačije. Za njih se ovo rastojanje računa u odnosu na aktivacije pretposlednjeg sloja koje su uskladištene u baferu. Umesto da se koristi L_E funkcija greške koja je opisana u 3.3, u strategiji učenja sa manjim zaboravljanjem koja koristi bafer koristiće se $\overline{L}_E$ funkcija greške, koja je opisana u 3.5. Kompletna kriterijumska funkcija je prikazana u 3.6.

$$\overline{L}_E(X^m, \overline{X}^m, \overline{R}^m; w^{m-1}, w, V^{m-1}, V^m) = \frac{1}{2(|X^m| + |\overline{X}^m|)} \cdot \left(\sum_{n=1}^{|X^m|} \left\| f^{L-1}(x_n^m; w^{m-1}, V^{m-1}) - f^{L-1}(x_n^m; w, V^m) \right\|_2^2 + \sum_{n=1}^{|\overline{X}^m|} \left\| \overline{r}_n^m - f^{L-1}(\overline{x}_n^m; w, V^m) \right\|_2^2 \right) \tag{3.5}$$

$$w^m = \arg\min_w L_{BCE}(X^m \cup \overline{X}^m, Y^m \cup \overline{Y}^m; w, u^*, V^m) + \\ + \lambda(m-1)\overline{L}_E(X^m, \overline{X}^m, \overline{R}^m; w^{m-1}, w, V^{m-1}, V^m) \tag{3.6}$$

Inkrementalna strategija koja se zasniva samo na baferu skladišti dve vrednosti za svaki zahtev: tekst zahteva i labelu. U ovoj strategiji se treniranje modela na prvom mini-korpusu vrši na isti način kao i kod neinkrementalnog treniranja, koristeći funkciju greške binarne unakrsne entropije (2.1). Prilikom treniranja na bilo kom drugom mini-korpusu ($m > 1$), u ovoj strategiji se prvo napravi proširena verzija mini-korpusa, koja sadrži sve primere iz tog mini-korpusa i sve primere koji se trenutno nalaze u baferu

(i regularne i maliciozne). Nakon toga, model se trenira na isti način koristeći funkciju greške binarne unakrsne entropije, ali sa proširenom verzijom mini-korpusa. Ova strategija ne zamrzava parametre poslednjeg sloja, i njen jedini mehanizam kojim sprečava zaboravljanje prethodno stečenog znanja je korišćenje primera iz bafera.

3.5 Evaluacija

Svi hiperparametri kod inkrementalnog treniranja imaju iste vrednosti kao i kod klasičnog treniranja (MBP za svaki korpus, broj epoha treniranja, korak treniranja, veličina grupe i optimizacioni algoritam). Kako kod inkrementalnog treniranja postoji više mini-korpusa koji se koriste za treniranje, treniranje sa svakim od njih vršiće se sa onim brojem epoha treniranja koji je korišćen za treniranje na celom korpusu kod klasičnog i treniranja na napade nultog dana. Tehnike poduzorkovanja i preuzorkovanja će biti primenjene na svaki pojedinačni mini-korpus u ovom slučaju. Hiperparametar λ je postavljen na $2 \cdot 10^{-5}$.

Tabele 3.1 i 3.2 prikazuju rezulate klasifikacije koje su modeli postigli nakon što su završili sa treniranjem na svim mini-korpusima, koristeći tehniku učenja sa manjim zaboravljanjem (bez bafera). I na našem TBWIDD i na FWAF korpusu, TextCNN model sa karakterima kao atributima je ostvario najbolji skor geometrijske sredine. Razlike između finalnih skorova geometrijskih sredina među modelima su manje na našem korpusu. Slično kao i kod klasičnog testiranja i testiranja na napade nultog dana, primena poduzorkovanja uglavnom poboljšava efikasnost na skupu za testiranje na našem TBWIDD korpusu. Na FWAF korpusu, svi modeli su ostvarili svoje najbolje rezultate koristeći karaktere kao atribute.

Kod inkrementalnih strategija koje koriste bufer, korišćena je maksimalna veličina bafera od 50 regularnih i 50 malicioznih zahteva na našem TBWIDD korpusu, i 500 regularnih i 500 malicioznih zahteva na FWAF korpusu. Prilikom primene poduzorkovanja na mini-korpusima za treniranje, jedino će se sa zahtevima koji su ostali posle poduzorkovanja pokušati dodavanje u bafer. Kako preuzorkovanje kreira duplikate nekih zahteva iz mini-korpusa za treniranje, a čuvanje više kopija istog zahteva u baferu ne pruža nikakvu dodatnu informaciju o prošlim zahtevima, pokušaće se dodavanje svakog jedinstvenog zahteva iz mini-korpusa tačno jednom.

Tabele 3.3 i 3.4 prikazuju rezultate modela treniranih inkrementalnim učenjem sa manjim zaboravljanjem sa korišćenjem bafera. Na našem korpusu, LSTM model sa trigramima kao atributima je ostvario najbolji rezultat. Sve kombinacije tipa modela i tipa ulaznih atributa su ostvarile svoje najbolje skorove geometrijske sredine veće u varijanti strategije koja koristi bafer na validacionom skupu. Situacija je slična i na test skupu, a jedini izuzetak je TextCNN model sa trigramima kao atributima, koji je ostvario isti rezultat i sa i bez bafera. Poboljšanje u rezultatima može takođe da se vidi i na FWAF korpusu. Sve kombinacije tipa modela i tipa ulaznih atributa su ostvarile svoj najbolji finalni skor geometrijske sredine od 98.76% ili više, dok su samo 2 kombinacije to uspele kada su bile trenirane bez bafera.

Rezultati tesiranja strategije inkrementalnog treniranja zasnovane samo na baferu su dati u tabelama 3.5 i 3.6. U tabelama jasno može da se vidi pozitivan uticaj koji bafer ima

Tabela 3.1: Rezultati inkrementalnog testiranja na našem TBWIDD korpusu koristeći strategiju učenja sa manjim zaboravljanjem bez bafera

(a) Validacioni skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
TextCNN	karakter	**6**	**3**	**99.11**	6	10	98.98	10	1	98.58
TextCNN	3gram	**6**	**5**	**99.07**	**6**	**5**	**99.07**	13	3	98.12
CNNRNNCNN	3gram	**6**	**7**	**99.04**	8	10	98.7	15	0	97.89
CNNRNNCNN	karakter	**4**	**26**	**98.98**	8	12	98.67	11	1	98.44
LSTM	karakter	**5**	**19**	**98.96**	4	146	96.79	9	20	98.39
LSTM	3gram	**4**	**28**	**98.94**	6	33	98.57	10	0	98.6

(b) Test skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
TextCNN	karakter	**31**	**0**	**99.87**	32	0	99.86	32	0	99.86
LSTM	3gram	40	0	99.83	**37**	**0**	**99.84**	2935	0	86.74
TextCNN	3gram	42	0	99.82	**37**	**0**	**99.84**	50	0	99.79
CNNRNNCNN	3gram	42	0	99.82	**40**	**0**	**99.83**	47	0	99.8
LSTM	karakter	**41**	**0**	**99.83**	17	1	95.67	16	2	91.23
CNNRNNCNN	karakter	42	1	95.57	**42**	**0**	**99.82**	**42**	**0**	**99.82**

Tabela 3.2: Rezultati inkrementalnog testiranja na FWAF korpusu koristeći strategiju učenja sa manjim zaboravljanjem bez bafera

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
TextCNN	karakter	12	41	99.85	129	74	99.71	**6**	**39**	**99.85**
LSTM	karakter	9	948	96.41	133	681	97.42	**26**	**322**	**98.79**
TextCNN	3gram	**176**	**417**	**98.41**	1635	641	97.38	738	2132	91.64
LSTM	3gram	**87**	**1323**	**94.94**	2717	2100	91.53	9	4148	83.16
CNNRNNCNN	karakter	4	2406	90.61	**89**	**1931**	**92.53**	0	2199	91.46
CNNRNNCNN	3gram	**140**	**2092**	**91.88**	4863	2951	87.78	911	4675	80.68

Tabela 3.3: Rezultati inkrementalnog testiranja na našem TBWIDD korpusu koristeći strategiju učenja sa manjim zaboravljanjem sa baferom

(a) Validacioni skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
LSTM	3gram	**1**	**11**	**99.66**	6	24	98.73	11	0	98.46
TextCNN	3gram	**3**	**3**	**99.53**	5	7	99.18	11	3	98.41
CNNRNNCNN	3gram	**3**	**4**	**99.51**	4	14	99.19	9	2	98.71
TextCNN	karakter	6	3	99.11	5	10	99.12	**6**	**2**	**99.13**
CNNRNNCNN	karakter	**5**	**13**	**99.07**	6	18	98.84	9	1	98.72
LSTM	karakter	**3**	**31**	**99.03**	6	63	98.03	10	6	98.49

(b) Test skup

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
LSTM	3gram	**24**	**0**	**99.9**	34	0	99.86	3042	0	86.22
LSTM	karakter	**27**	**0**	**99.89**	23	1	95.65	35	1	95.6
TextCNN	karakter	38	0	99.84	**30**	**0**	**99.87**	34	0	99.86
TextCNN	3gram	**37**	**0**	**99.84**	38	0	99.84	61	0	99.74
CNNRNNCNN	karakter	36	1	95.6	40	0	99.83	**38**	**0**	**99.84**
CNNRNNCNN	3gram	**38**	**0**	**99.84**	39	0	99.84	42	0	99.82

Tabela 3.4: Rezultati inkrementalnog testiranja na FWAF korpusu koristeći strategiju učenja sa manjim zaboravljanjem sa baferom

Model	**Atributi**	**Standardno**			**Poduzorkovanje**			**Preuzorkovanje**		
		LP	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**	**LP**	**LN**	**GS (%)**
LSTM	karakter	4	49	99.82	**364**	**26**	**99.86**	11	62	99.77
CNNRNNCNN	karakter	4	92	99.66	**258**	**31**	**99.85**	5	79	99.71
TextCNN	karakter	11	49	99.82	992	48	99.69	**14**	**46**	**99.83**
TextCNN	3gram	**148**	**233**	**99.11**	3837	206	98.73	1551	1868	92.61
LSTM	3gram	**58**	**250**	**99.06**	2948	210	98.83	13	1353	94.84
CNNRNNCNN	3gram	152	423	98.4	**3875**	**197**	**98.76**	1460	1214	95.2

Tabela 3.5: Rezultati inkrementalnog testiranja na našem TBWIDD korpusu koristeći strategiju zasnovanu samo na baferu

(a) Validacioni skup

Model	Atributi	Standardno			Poduzorkovanje			Preuzorkovanje		
		LP	LN	GS (%)	LP	LN	GS (%)	LP	LN	GS (%)
TextCNN	3gram	**3**	**5**	**99.49**	5	26	98.84	9	4	98.67
CNNRNNCNN	karakter	**2**	**13**	**99.49**	5	64	98.15	10	12	98.39
LSTM	3gram	**1**	**25**	**99.41**	6	57	98.14	8	11	98.69
CNNRNNCNN	3gram	**2**	**23**	**99.31**	6	17	98.86	11	7	98.34
TextCNN	karakter	**6**	**8**	**99.02**	5	21	98.93	6	9	99.0
LSTM	karakter	**3**	**41**	**98.84**	6	101	97.34	11	11	98.27

(b) Test skup

Model	Atributi	Standardno			Poduzorkovanje			Preuzorkovanje		
		LP	LN	GS (%)	LP	LN	GS (%)	LP	LN	GS (%)
TextCNN	3gram	**15**	**0**	**99.94**	35	0	99.85	42	0	99.82
TextCNN	karakter	30	0	99.87	**17**	**0**	**99.93**	20	0	99.92
CNNRNNCNN	3gram	**33**	**0**	**99.86**	35	0	99.85	37	0	99.84
LSTM	3gram	37	0	99.84	**33**	**0**	**99.86**	40	0	99.83
CNNRNNCNN	karakter	31	1	95.62	36	1	95.6	**41**	**0**	**99.83**
LSTM	karakter	**25**	**1**	**95.64**	23	2	91.2	34	1	95.61

Tabela 3.6: Rezultati inkrementalnog testiranja na FWAF korpusu koristeći strategiju zasnovanu samo na baferu

Model	Atributi	Standardno			Poduzorkovanje			Preuzorkovanje		
		LP	LN	GS (%)	LP	LN	GS (%)	LP	LN	GS (%)
CNNRNNCNN	karakter	19	85	99.68	**682**	**3**	**99.9**	6	76	99.72
TextCNN	karakter	3	199	99.26	**129**	**71**	**99.72**	13	122	99.54
LSTM	karakter	16	127	99.52	**303**	**77**	**99.67**	3	108	99.6
TextCNN	3gram	**184**	**214**	**99.18**	4362	189	98.72	215	839	96.8
LSTM	3gram	**81**	**303**	**98.86**	3048	218	98.79	37	663	97.5
CNNRNNCNN	3gram	**198**	**330**	**98.74**	3845	334	98.25	178	1086	95.86

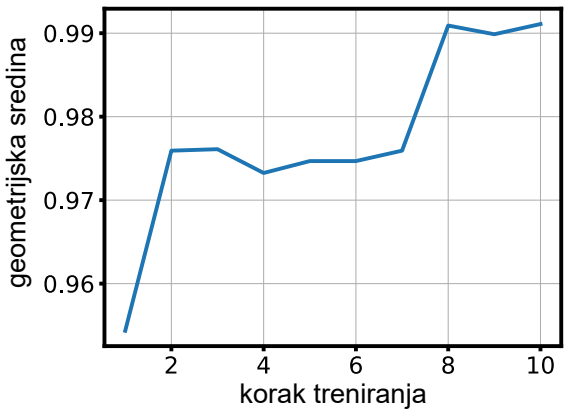
na očuvanje stečenog znanja. Na našem test skupu, polovina kombinacija tipa modela i tipa ulaznih atributa je ostvarila bolje rezultate kada je koristila samo bafer nego kada je koristila strategiju učenja sa manjim zaboravljanjem sa baferom. Situacija je nešto slabija na validacionom skupu, na kome je samo jedna od šest kombinacija ostvarila bolje rezultate kada je koristila samo bafer. Na našem korpusu, TextCNN model sa trigramima kao atributima je ostvario najbolji skor geometrijske sredine kada je koristio samo bafer. Rezultati ove strategije su takođe dobri i na FWAF korpusu, gde je najbolji finalni skor geometrijske sredine ostvaren korišćenjem ove strategije. Dve od šest kombinacija tipa modela i tipa ulaznih atributa su ostvarile bolje najbolje skorove geometrijske sredine koristeći ovu strategiju u odnosu na strategiju učenja sa manje zaboravljanja koja koristi bafer.

Kako je TextCNN model sa karakterima kao atributima pokazao sveukupno dobre rezultate koristeći svaku od tri strategije inkrementalnog učenja, prikazaćemo na graficima 3.1, 3.2 i 3.3 kako se njegove performanse menjaju nakon treniranja na svakom mini-korpusu. Prikazani rezultati su za varijantu modela koja koristi standardno treniranje (bez korišćenja tehnika poduzorkovanja ili preuzorkovanja). Na sva tri grafika može da se primeti da je model bio u stanju da inkrementalno uči, uvećavajući svoje znanje o regularnim i malicioznim zahtevima tokom vremena. Skor geometrijske sredine uglavnom ostaje stabilan ili se poboljšava kroz vreme, uz pojedine retke male padove u skoru.

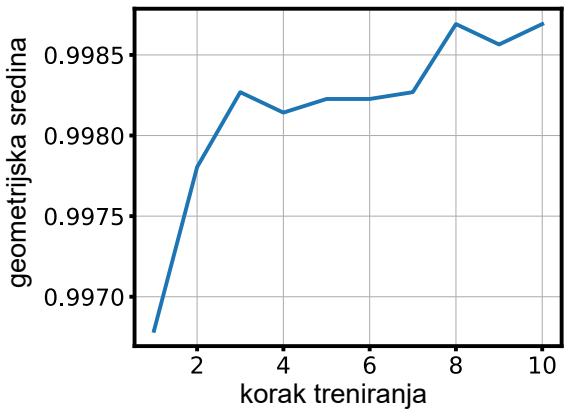
Na FWAF korpusu možemo primetiti da modeli koji koriste preuzorkovanje često postižu veće vrednosti skora geometrijske sredine brže neko kada koriste standardno treniranje. Na slici 3.4 su prikazani grafici TextCNN modela sa karakterima kao atributima koji je inkrementalno treniran na FWAF korpusu korišćenjem preuzorkovanja. Sa grafika se može primetiti da tehnika inkrementalnog učenja sa manjim zaboravljanjem (i sa i bez korišćenja bafera) postiže veoma visoke performanse već nakon treniranja na prvih 10 mini-korpusa, i da nakon toga model nastavlja da inkrementalno uči iz preostalih mini-korpusa za treniranje. Kada je za treniranje korišćena inkrementalna strategija koja koristi samo bafer, treniranje je bilo manje stabilno, ali je kroz vreme model ipak uspeo da akumulira znanje.

Modeli LSTM i TextCNN sa trigramima kao atributima trenirani na standardan način (bez korišćenja tehnika poduzorkovanja ili preuzorkovanja) su postigli odlične rezultate koristeći inkrementalno treniranje sa manje zaboravljanja sa baferom i inkrementalno treniranje zasnovano samo na baferu, redom. Grafici koji prikazuju promenu vrednosti skora geometrijske sredine na validacionim i test skupovima tokom njihovog inkrementalnog treniranja su dati na slikama 3.5 i 3.6. Na našem korpusu, njihovi grafici su stabilni, uspešnost klasifikacije raste tokom vremena, uz povremene retke male padove. Na FWAF korpusu, oba modela su dostigla visok krajnji nivo klasifikacije, ali je treniranje na modelu koji koristi samo bafer bilo manje stabilno. Iznenadni padovi u kvalitetu klasifikacije su nešto što nije poželjno u inkrementalnom učenju, tako da iako su njihovi krajnji nivoi klasifikacije slični, treba odabrati stabilniji model u ovom poređenju.

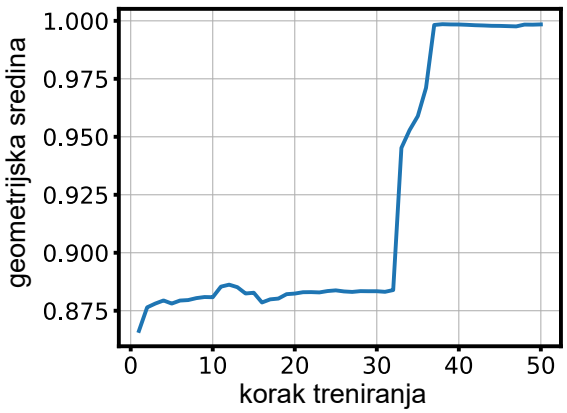
Izbor oko strategije inkrementalnog treniranja može takođe da zavisi i od ograničenja vezanih za privatnost. Ukoliko se od modela očekuje da inkrementalno uči iz skupova podataka iz različitih organizacija, mešanje delova podataka iz različitih izvora može da naruši ta ograničenja. U tom slučaju je inkrementalno učenje sa manjim zaboravljanjem bez korišćenja bafera dobro rešenje (model TextCNN sa karakterima kao atributima je



(a) Validacioni skup - TBWIDD

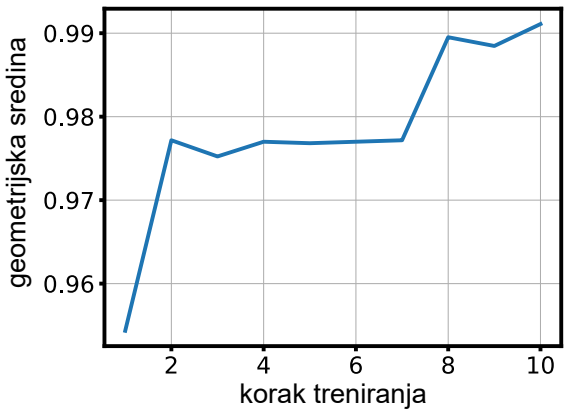


(b) Test skup - TBWIDD

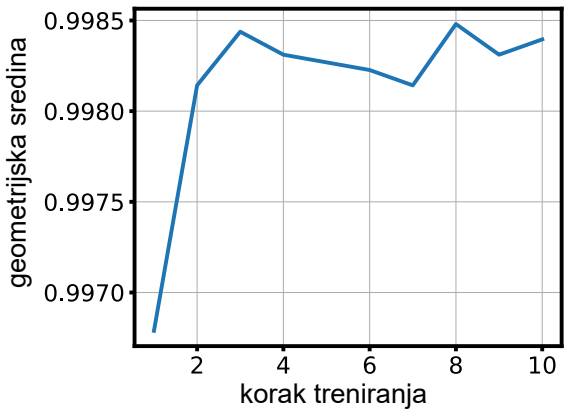


(c) Validacioni skup - FWAF

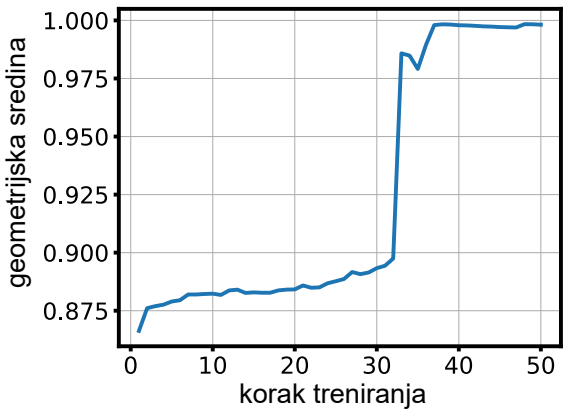
Slika 3.1: Grafici pokazuju kako se skor geometrijske sredine menja kroz korake treniranja za TextCNN model sa karakterima kao atributima koristeći standardno inkrementalno učenje sa manjim zaboravljanjem bez bafera



(a) Validacioni skup - TBWIDD

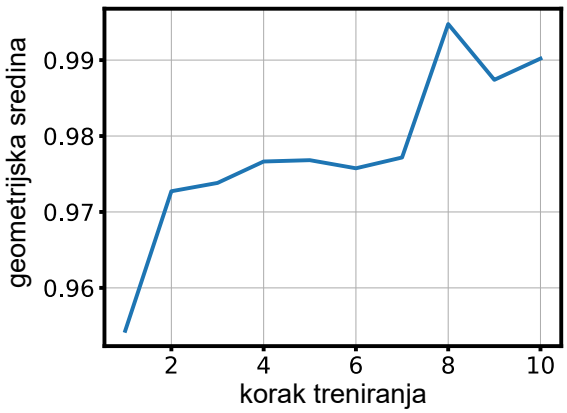


(b) Test skup - TBWIDD

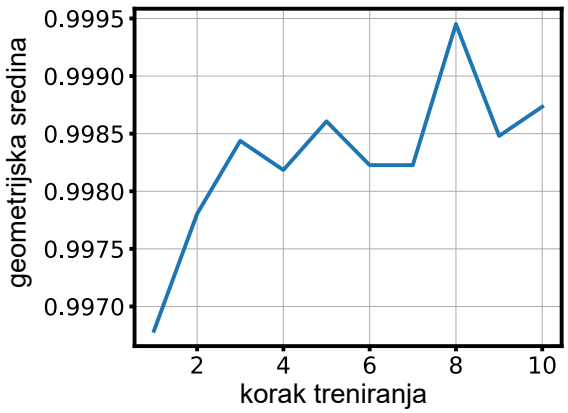


(c) Validacioni skup - FWAF

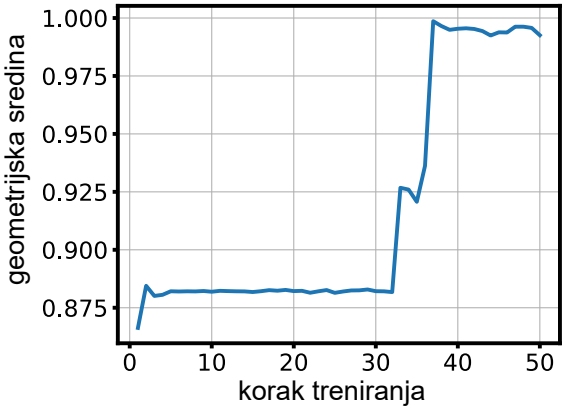
Slika 3.2: Grafici pokazuju kako se skor geometrijske sredine menja kroz korake treniranja za TextCNN model sa karakterima kao atributima koristeći standardno inkrementalno učenje sa manjim zaboravljanjem sa baferom



(a) Validacioni skup - TBWIDD

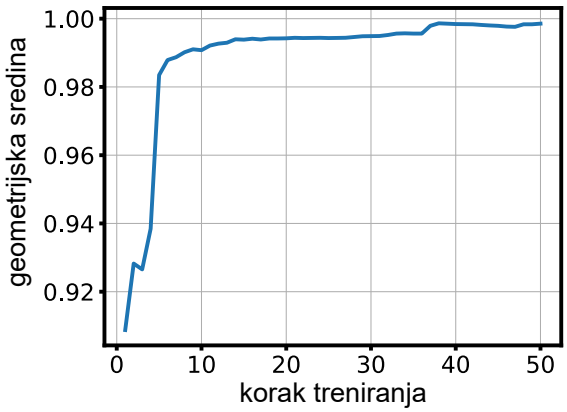


(b) Test skup - TBWIDD

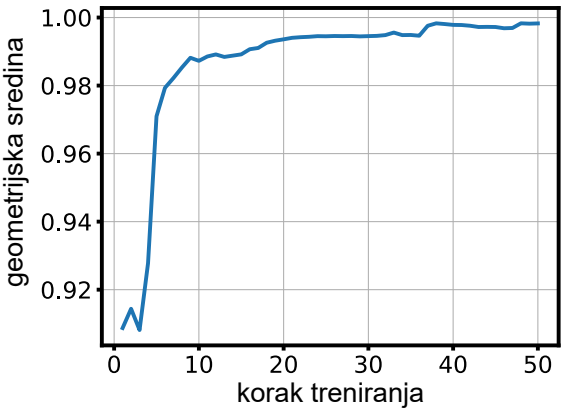


(c) Validacioni skup - FWAF

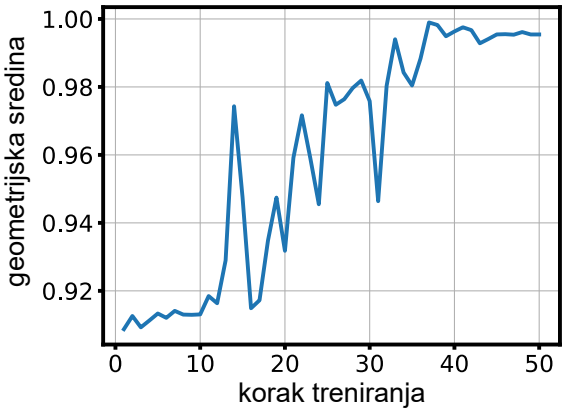
Slika 3.3: Grafici pokazuju kako se skor geometrijske sredine menja kroz korake treniranja za TextCNN model sa karakterima kao atributima koristeći standardno inkrementalno učenje zasnovano samo na baferu



(a) Učenje sa manje zaboravljanja

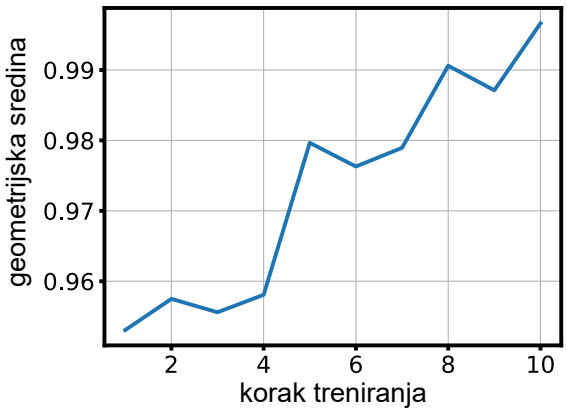


(b) Učenje sa manje zaboravljanja i baferom

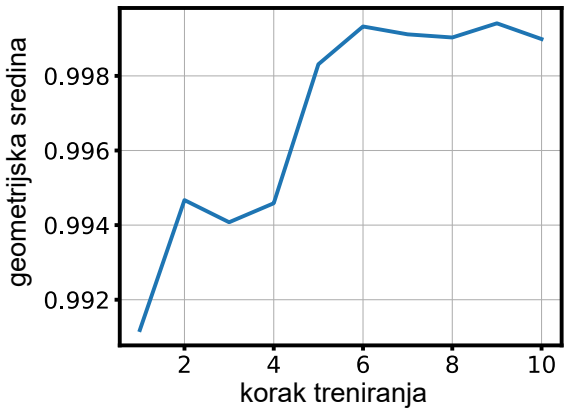


(c) Učenje samo sa baferom

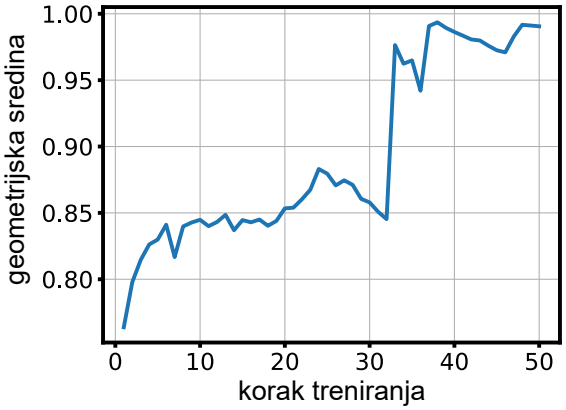
Slika 3.4: Performanse TextCNN modela sa karakterima kao atributima na FWAF korpusu za vreme inkrementalnog treniranja sa preuzorkovanjem



(a) Validacioni skup - TBWIDD

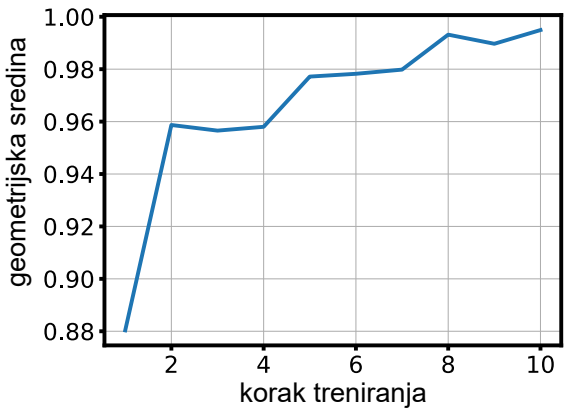


(b) Test skup - TBWIDD

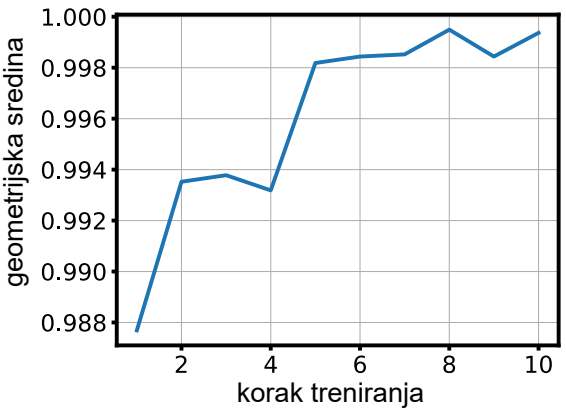


(c) Validacioni skup - FWAF

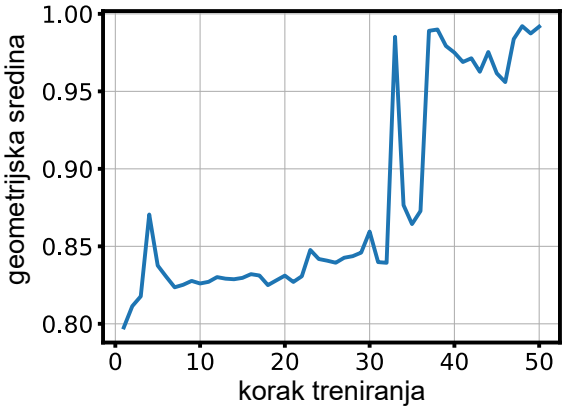
Slika 3.5: Performanse LSTM modela sa trigramima kao atributima treniranog standardnom inkrementalnom strategijom sa manje zaboravljanja i baferom



(a) Validacioni skup - TBWIDD



(b) Test skup - TBWIDD



(c) Validacioni skup - FWAF

Slika 3.6: Performanse TextCNN modela sa trigramima kao atributima treniranog standardnom inkrementalnom strategijom baziranom samo na baferu

pokazao dobre rezultate kod ovakvog treniranja), zato što ne koristi zahteve iz prošlosti da bi ažuriralo model. Ukoliko mešanje zahteva iz različitih izvora (mini-korpusa) ne krši ograničenja privatnosti, dodavanje bafera modelu može da unapredi model, i tim strategijama bi trebalo dati prednost.

Poglavlje 4

Odabir atributa mreže baziran na populaciji

U oblasti bezbednosti računarskih mreža, sposobnost detekcije i odgovora na neovlašćeni pristup i zlonamerne aktivnosti od presudnog je značaja za zaštitu osetljivih informacija i očuvanje integriteta mrežnih sistema. Bilo koji skup akcija koji pokušava da kompromituje funkcionalnost sistema se može smatrati upadom. Sistemi za detekciju upada (eng. intrusion detection system) igraju ključnu ulogu u prepoznavanju potencijalnih štetnih radnji koje mogu ugroziti podatke, sisteme ili usluge organizacija. Efikasnost ovih sistema u velikoj meri zavisi od njihove sposobnosti da analiziraju različite mrežne atribute koji karakterišu ponašanje i protok saobraćaja kroz mrežu. Upad može izazvati promenu u parametrima mrežnog saobraćaja, što sistemi za detekciju upada pokušavaju da iskoriste. Pravovremenom detekcijom malicioznih aktivnosti smanjuje se rizik i poboljšava ukupna sigurnost mreže.

Atributi mreže predstavljaju merljive karakteristike i parametre koji opisuju mrežni saobraćaj. Ovi atributi obuhvataju širok spektar faktora, uključujući detalje na nivou paketa, statistiku protoka, ponašanje veza i obrasce komunikacije. Atributi mreže pružaju detaljan uvid u tokove saobraćaja i komunikacione obrasce. Njihova korisnost proizilazi iz činjenice da mnogi napadi imaju prepoznatljive karakteristike na mrežnom nivou. Na primer, DDoS (eng. distributed denial of service) napadi često dovode do naglog povećanja dolaznog saobraćaja sa više izvora, dok skeniranje portova generiše neuobičajeno veliki broj pokušaja povezivanja na različite portove. Ekstrakcijom, odabirom i analizom ovih atributa, sistemi za detekciju mogu efikasno identifikovati anomalije i prepoznati obrasce koji odgovararu napadima.

Praćenje IP adresa izvora i odredišta pomaže u identifikaciji malicioznih aktera. Često menjanje adrese može ukazati na maliciozno skeniranje ili izviđanje. Posmatranje često korišćenih portova (na primer HTTP na portu 80 ili HTTPS na portu 443) predstavlja standardnu meru prevencije upada. Napadači često ciljaju otvorene portove kako bi iskoristili određene servise. Zbog toga neuobičajeno korišćenje portova ili pokušaj povezivanja na više portova istovremeno može ukazati na pokušaj upada.

Napadači mogu koristiti određene protokole za izviđanje ili infiltraciju. Klasifikacijom saobraćaja prema protokolu lakše je detektovati anomalije. Na primer, nagli skokovi u broju ICMP (eng. internet control message protocol) paketa mogu ukazivati na DDoS

napad baziran na ping porukama. Atributi poput broja paketa, količine prenesenih bajtova po jedinici vremena i slični mogu često da otkriju volumetrijske napade. Iznenadni porast u obimu protoka ili nagle promene u odnosima razmene podataka mogu da ukažu na maliciozne aktivnosti. Obrasci u vremenskim razmacima između paketa mogu ukazati na aktivnosti skeniranja ili na kompleksnije napade koji zaobilaze jednostavnija pravila detekcije.

Iako postoji mnogo atributa mreže, nisu svi podjednako korisni za detekciju upada. Kako računarske mreže postaju složenije, raste i broj dostupnih atributa, što može povećati dimenzionalnost podataka i uneti šum. Ovo poglavlje istražuje odabir ključnih atributa mreže relevantnih za detekciju upada, na osnovu šireg skupa svih zabeleženih atributa mreže. Pravilno odabrani atributi mogu značajno da poboljšaju tačnost i efikasnost detekcije.

U mašinskom učenju, pretreniranje nastaje kada se model previše prilagodi podacima koje koristi za treniranje (memoriše primere za treniranje), a ne uspeva da generalizuje stečeno znanje i da korektne odgovore na novim primerima. Uklanjanjem nebithih ili manje značajnih atributa, sistem detekcije upada može da nauči pravilnije obrasce bez prilagođavanja slučajnom šumu.

Sistemi detekcije upada često moraju da obrađuju velike količine podataka u realnom vremenu. Smanjivanjem broja atributa može da se smanji opterećenje procesora i poboljša efikasnost sistema. Odabir atributa pomaže i kod interpretabilnosti, jer smanjuje obim potencijalnih atributa koji utiču na odluke sistema, i time olakšava posao analitičarima bezbednosti.

4.1 Pregled literature

Više korpusa je kreirano u cilju treniranja i testiranja sistema za detekciju upada. Među ranijim korpusima, KDDCup'99 [56] korpus se i dalje često koristi za evaluaciju modela, iako je napravljen pre više od dve decenije. NSLKDD [100] korpus je kasnije kreiran da umanji neke od problema koje je KDDCup'99 korpus imao, kao što su redundantni zapisi neravnomeran odnos regularnih i malicioznih primera i vrednosti koje nedostaju. Uprkos unapređenjima, NSLKDD ipak ne sadrži sveobuhvatnu reprezentaciju modernih napada sa malim tragom. Vođena tim nedostatkom, istraživačka grupa za sajber bezbednost iz australijanskog centra sa sajber bezbednost je kreirala novi korpus, nazvan UNSW-NB15 [82], koji sadrži bolju reprezentaciju modernih tipova napada. Zbog toga je za treniranje i evaluaciju modela u ovom poglavlju izabran ovaj korpus.

Korpusi za detekciju upada sadrže dosta atributa. Da bi učinili sisteme detekcije bržim i efikasnijim, istraživači su koristili različite strategije za odabir atributa. Janarthanan i Zargari [50] su eksperimentisali sa više metoda odabira atributa koristeći Veka (eng. Weka) alat. Na kraju su odabrali 5 važnih atributa, i sa njima ostvarili tačnost od 81.62%. Khan i dr. [57] su iskoristili tehniku za određivanje značaja atributa kojom su odabrali 11 ulaznih atributa iz UNSW-NB15 korpusa. Da je neki atribut značajan i da ga treba zadržati odlučivali su tako što su na slučajan način promešali vrednosti duž kolone datog atributa, i ukoliko bi skor predikcije opao, to bi značilo da je taj atribut bitan i da ga treba zadržati. Najvišu tačnost od 75.66% su ostvarili koristeći klasifikator slučajne

šume. Više informacija o trenutnim pristupima problemu detekcije upada se mogu naći u studiji koju su sproveli Khraisat i dr. [59].

4.2 Korpus

Svaki primer u UNSW-NB15 korpusu za detekciju upada sadrži 42 ulazna atributa u svojoj originalnoj formi. Spisak svih atributa je dat u tabeli 4.1, a detaljno objašnjenje svakog od njih je datu u originalnom radu [82]. Tri atributa nisu numerička (protokol koji je korišćen, njegovo stanje i servis koji radi), a kako većina klasifikatora mašinskog učenja očekuje numerički ulaz, ovi atributi će biti transformisani pre korišćenja. Transformacija će biti izvršena tako što će se najpre sakupiti sve moguće vrednosti koje nenumerički atribut može da ima, a zatim će se svakoj vrednosti dodeliti po jedan nenegativni ceo broj, počevši od broja nula, i uvećavanjem za 1 za svaku sledeću vrednost.

Tabela 4.1: Atributi UNSW-NB15 korpusa

Redni broj	Ime	Tip	Redni broj	Ime	Tip
1.	dur	realni	22.	dtcpb	celobrojni
2.	proto	nominalni	23.	dwin	celobrojni
3.	service	nominalni	24.	tcprtt	realni
4.	state	nominalni	25.	synack	realni
5.	spkts	celobrojni	26.	ackdat	realni
6.	dpkts	celobrojni	27.	smean	celobrojni
7.	sbytes	celobrojni	28.	dmean	celobrojni
8.	dbytes	celobrojni	29.	trans_depth	celobrojni
9.	rate	realni	30.	response_body_len	celobrojni
10.	sttl	celobrojni	31.	ct_srv_src	celobrojni
11.	dttl	celobrojni	32.	ct_state_ttl	celobrojni
12.	sload	realni	33.	ct_dst_ltm	celobrojni
13.	dload	realni	34.	ct_src_dport_ltm	celobrojni
14.	sloss	celobrojni	35.	ct_dst_sport_ltm	celobrojni
15.	dloss	celobrojni	36.	ct_dst_src_ltm	celobrojni
16.	sinpkt	realni	37.	is_ftp_login	binarni
17.	dinpkt	realni	38.	ct_ftp_cmd	celobrojni
18.	sjit	realni	39.	ct_flw_http_mthd	celobrojni
19.	djit	realni	40.	ct_src_ltm	celobrojni
20.	swin	celobrojni	41.	ct_srv_dst	celobrojni
21.	stcpb	celobrojni	42.	is_sm_ips_ports	binarni

Korpus je u svojoj originalnoj formi podeljen na deo za treniranje i deo za testiranje. Kako će nam biti potrebni i primeri za validaciju, 30% primera je na slučajan način izvučeno iz skupa za testiranje i od njih je kreiran validacioni skup. Preostalih 70% primera iz test skupa će biti korišćeno za testiranje. Finalne veličine skupova za treniranje, validaciju i testiranje koji će se koristiti u evaluaciji su date u tabeli 4.2.

Kako se raspon vrednosti različitih atributa značajno razlikuje, vršena je normalizacija. Normalizacija se vrši tako što se od vrednosti svakog atributa najpre oduzme mini-

Tabela 4.2: Veličine skupova za treniranje, validaciju i testiranje

Skup	Regularni	Maliciozni
Treniranje	56000	119341
Validacija	11087	13613
Testiranje	25913	31719

malna vrednost tog atributa, a zatim se taj broj podeli sa razlikom između maksimalne i minimalne vrednosti tog atributa. Maksimalne i minimalne vrednosti se izračunavaju na osnovu skupa za treniranje.

4.3 Metod odabira atributa

U ovoj glavi će biti predstavljen metod odabira odgovarajućih atributa za detekciju upada baziran na populaciji. Elementi populacije u ovom slučaju biće podskupovi supa svih ulaznih atributa. Svaki element se može predstaviti kao vektor nula i jedinica veličine $N = 42$ (ukupni broj atributa). Jedinica označava da je dati atribut izabran za korišćenje, dok nula označava da nije izabran.

Svaka generacija $g \in \{1, 2, \ldots, G\}$ je definisana svojim vektorom verovatnoća $p^g \in [0, 1]^N$. Vrednost p_i^g označava verovatnoću da i-ti atribut bude odabran u g-toj generaciji. Generacija g se kreira uzorkovanjem S instanci na osnovu njenog vektora verovatnoća p^g. Označimo sa $e_1^g, \ldots, e_S^g$ ove instance (elemente populacije).

U našoj strategiji se neće čuvati instance iz prethodnih generacija. Način na koji instance utiču na sledeću generaciju je preko vektora verovatnoća. U prvoj generaciji, sve verovatnoće p_i^1 će biti inicijalizovane na istu vrednost $p_{inicijalno}$. Za svaku narednu generaciju $g > 1$, vektor verovatnoća se računa na osnovu skorova koje su ostvarile instance iz prethodne $(g-1)$-ve generacije. U našem slučaju, skor instance e_i^g će biti definisan kao tačnost na validacionom skupu koju ostvari model treniran na skupu za treniranje koristeći atribute e_i^g, i ovu vrednost ćemo označiti sa s_i^g.

Da bi izračunali vektor verovatnoća p^g, najpre ćemo izračunati doprinose individualnih elemenata iz prethodne generacije. Označimo sa c_i^g doprinos i-tog elementa generacije g na generaciju $g+1$. Način na koji izračunavamo vrednost c_i^g je dat u (4.1).

$$\begin{aligned} Min^g &= \min_j s_j^g \\ Max^g &= \max_j s_j^g \\ \overline{c}_i^g &= \frac{s_i^g - Min^g}{Max^g - Min^g} \\ c_i^g &= \frac{\overline{c}_i^g}{\sum_{j=1}^{S} \overline{c}_j^g} \end{aligned} \tag{4.1}$$

Ukoliko se desi specijalni slučaj da je $Min^g = Max^g$, sve vrednosti doprinosa treba postaviti na $c_i^g = 1/S$. Kao što se može videti iz formule, instance sa većim tačnostima

će imati veće vrednosti doprinosa. Koristeći vrednosti doprinosa, vektor verovatnoća p^g se dobija pomoću formule (4.2). Treba napomenuti da je u ovoj formuli c_j^{g-1} skalar, a e_j^{g-1} vektor nula i jedinica. U formuli se takođe koristi i hiperparametar α, mali realni broj. Svrha ovog hiperparametra je da svakom ulaznom atributu obezbedi neku pozitivnu verovatnoću sa kojom će ili neće biti odabran u sledećoj generaciji. Na osnovu formule se može zaključiti da će instance sa većim vrednostima doprinosa imati veći uticaj na kreiranje instanci sledeće generacije.

$$\begin{aligned} \overline{p}^g &= \sum_{j=1}^{S} c_j^{g-1} e_j^{g-1} \\ p^g &= (1-\alpha)\overline{p}^g + 0.5\alpha \end{aligned} \tag{4.2}$$

U toku odabira atributa, vršiće se evaluacija ukupno $G \cdot S$ podskupova skupa ulaznih atributa (po S instanci iz svake od G generacija). Ovaj broj je značajno manji od ukupnog broja mogućih podskupova (2^N). Za krajnji podskup atributa biće uzet onaj podskup koji ima najbolju tačnost na validacionom skupu.

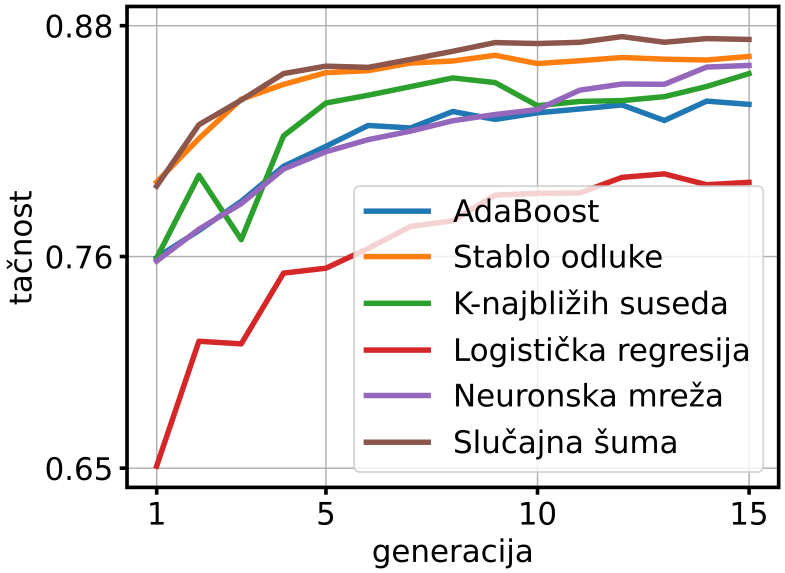
4.4 Evaluacija

U cilju evaluacije predloženog metoda odabira atributa baziranog na populaciji, metod odabira atributa će biti uparen sa više algoritmima mašinskog učenja. Za uparivanje su upotrebljeni klasifikatori mašinskog učenja koji su često korišćeni u sistemima detekcije upada iz literature: stablo odluke, slučajna šuma, AdaBoost, logistička regresija, k-najbližih suseda i potpuno povezana veštačka neuronska mreža. Svi eksperimenti će biti obavljeni u $G = 15$ generacija. U svakoj generaciji biće generisano $S = 15$ instanci (elemenata populacije). Vektori verovatnoća će biti inicijalizovani sa $p_{inicijalno} = 0.1$. Vrednost hiperparametra α koja je korišćena je $\alpha = 0.1$.

Neuromska mreža ima jedan skriveni sloj od 100 neurona sa ReLU aktivacionom funkcijom. Mreža je trenirana algoritmom za optimizaciju Adam, sa korakom treniranja 0.001. Klasifikator k-najbližih suseda je korišćen u varijanti koja koristi 5 najbližih suseda. Slučajna šuma ima 100 stabala. AdaBoost kao svoje osnovne estimatore koristi stabla odluke, i ima maksimalno 50 estimatora.

Slika 4.1 prikazuje prosečnu validacionu tačnost po generaciji za svaki klasifikator. Prosečna vrednost se računa po svim instancama jedne generacije. Na grafiku se može primetiti konstantni rast u prosečnoj tačnosti za sve klasifikatore, uz neke povremene male padove.

Najviše problema da uspešno klasifikuje ulazne signale sa mreže imao je klasifikator logističke regresije. Ovo je linearni klasifikator i po svojoj stukturi nema mogućnosti da nauči složenije veze među ulaznim atributima, kao što to mogu drugi kompleksniji klasifikatori. Najveći pad tačnosti u jednoj generaciji desio se u trećoj generaciji klasifikatora k-najbližih suseda, ali je već u četvrtoj generaciji klasifikator povratio nivo tačnosti i nadmašio onaj koji je imao u drugoj generaciji.



Slika 4.1: Prosečna validaciona tačnost po generaciji

Klasifikacioni rezultati na skupu za testiranje su dati u tabeli 4.3. Iz tabele se može videti da su svi modeli poboljšali svoju tačnost koristeći predloženi metod odabira atributa. Najveći napredak je postignut kod klasifikatora k-najbližih suseda, za koji se tačnost klasifikacije povećala za više od 3%. Najvišu tačnost od 88.91% je ostvario model veštačke neuronske mreže. Klasifikator stabla odluke je takođe ostvario dobru tačnost na skupu za testiranje, i to sa samo 14 odabranih atributa. To je svega jedna trećina od ukupnog broja ulaznih atributa.

Ovo poređenje tačnosti modela kada koriste sve atribute i kada koriste atribute odabrane od strane predložene strategije pokazuje njenu efektivnost. Pored toga što je poboljšao tačnost, predloženi metod je takođe drastično smanjio broj ulaznih atributa. Manji broj atributa može pozitivno da utiče na brzinu klasifikacije prilikom korišćenja modela.

Tabela 4.3: Rezultati klasifikacije na skupu za testiranje

Model	Tačnost (svi atributi)	Tačnost (odabrani atributi)	Odabrano
AdaBoost	0.8522	0.8604	22/42
Stablo odluke	0.8630	0.8824	14/42
K-najbližih suseda	0.8429	0.8762	23/42
Logistička regresija	0.8028	0.8113	17/42
Neuronska mreža	0.8674	**0.8891**	23/42
Slučajna šuma	0.8727	0.8812	15/42

Poglavlje 5

Detekcija fišing mejlova

Korišćenje servisa na mreži je postalo deo svakodnevnice velikog broja ljudi. Oni se koriste za naručivanje proizvoda, zabavu, komunikaciju sa prijateljima i kolegama i slično. Elektronska pošta (eng. electronic mail ili email) je jedan od najčešćih načina za komunikaciju preko interneta, posebno u poslovnom okruženju. Za nju se često koriste i skraćeni termini mejl ili imejl. Istraživačka grupa Radicati je procenila broj poslovnih i privatnih mejlova poslatih u 2021. godini na 319.6[1] milijardi. Za istu godinu su ocenili i broj korisnika elektronske pošte na 4.3 milijarde. Sve ovo ukazuje na važnost postojanja bezbedne i pouzdane komunikacije elektronskom poštom.

Jadna od glavnih pretnji u modernoj sajber bezbednosti su fišing napadi. Napad često počinje fišing mejlom, koji liči na regularan mejl, ali obično sadrži link ka fišing sajtu. Cilj napadača je da pridobije poverenje od strane korisnika elektronske pošte, koji neće posumnjati da se nešto maliciozno deštava i koji će kliknuti na maliciozni link. Kada korisnik dođe na fišing sajt, od korisnika se najčešće traži da unese neke poverljive informacije. To može da bude bilo šta što kasnije napadačima može da donese neku dobit, uključujući lozinke, brojeve bankovnih kartica i slično. Fišing sajtovi često liče na sajtove nekih poznatih organizacija, u koje korisnici već imaju poverenje. To može da ih navede da na sajtu unesu svoje poverljive informacije, verujući da koriste originalne sajtove proverenih organizacija. Nekada napadači mogu i da imitiraju nekog prijatelja od poverenja ili kolegu, i da direktnije traže neke usluge od korisnika elektronske pošte.

Radna grupa koja se bori protiv fišinga[2], je primetila dupliranje broja fišing napada u toku jedne godine. Takođe su primetili porast broja fišing sajtova koji koriste SSL/TLS sertifikate na 84%. Kako se broj fišing napada vremenom povećava, tako se povećava i značaj istraživanja u oblasti njihove detekcije. U zavisnosti od sadržaja koji se analizira, postoji vise podoblasti istraživanja. Jedan pravac istraživanja je detekcija na osnovu URL-ova (eng. Uniform Resource Locator) [21, 96, 71]. Pored URL-a, Feng i dr. [28] su dodatno koristili i sadržaj stranice u cilju bolje detekcije. Mehanović i dr. [75] su primenili selekciju ručno kreiranih atributa veb stranica. Nad tako odabranim atributima

[1]The Radicati Group Inc. Email Statistics Report, 2017-2021. Dostupno na: `https://www.radicati.com/wp/wp-content/uploads/2017/01/Email-Statistics-Report-2017-2021-Executive-Summary.pdf`, 2017.

[2]The Anti-Phishing Working Group. Phishing Activity Trends Report, 4th Quarter 2020. Dostupno na: `https://docs.apwg.org/reports/apwg_trends_report_q4_ 2020.pdf`, 2021.

su vršili klasifikaciju koristeći više klasičnih modela mašinskog učenja. Pošto detekcija fišing napada u inicijalnom trenutku kada korisnik dobije fišing mejl može drastično da ukloni opasnost, fokus u ovom poglavlju biće na prevenciji fišing napada detekcijom fišing mejlova.

Većina trenutne literature se fokusira na selekciji najkorisnijih atributa iz mejlova i primeni nekih klasičnih algoritama mašinskog učenja. Postoji nedostatak u pristupima koji uče atribute direktno iz fišing mejlova. U ovom poglavlju će biti predstavljen pristup koji uči ulazne atribute automatski pomoću učenja vektora ugradnje karaktera i reči iz tekstova mejlova. Ovaj pristup je univerzalniji, i jednostavno je kasnije ažurirati model sa novim tipovima mejlova.

Klasifikator koji će biti prikazan u ovom poglavlju biće zasnovan na neuronskoj mreži i simultano će koristiti vektore ugradnje karaktera i reči. Klasifikator kombinuje više konvolucionih i rekurentnih slojeva kako bi ekstraktovao i lokalne i globalne atribute. Za svrhu testiranja modela će se koristiti podaci iz dva često korišćena korpusa iz literature, SpamAssassin [72] javnog korpusa i Nazario [83] fišing korpusa. Korpusi sadrže mejlove u sirovom formatu, pa će se najpre iz svakog od njih skriptom izvući tekstualni deo. U svrhu potvrde efikasnosti klasifikatora, biće sprovedeno detaljno testiranje i analiza korišćenjem ekstraktovanih podataka iz ovih korpusa. Eksperimentalni rezultati ukazuju da je ovaj pristup isti ili bolji od najboljih dostupnih modela iz literature.

5.1 Pregled literature

Tradicionalno su za detekciju fišing napada bile korišćene crne liste (eng. blacklists). Iako je ovaj pristup bio efikasan pri detekciji već poznatih fišing napada, on nije bio u stanju da detektuje modifikacije tih napada ili neke nove napade. Da bi prevazišli ovaj nedostatak, istraživači su počeli da primenjuju algoritme mašinskog učenja. Shaukat i dr. [98] su odradili pregled primena modela mašinskog učenja u sajber bezbednosti u poslednjoj deceniji.

Islam i Abawajy [47] su predstavili troslojni model za klasifikaciju između fišing i regularnih mejlova. Koristili su 21 atribut ekstraktovan iz zaglavlja (eng. header) i tela (eng. body) mejlova. U svakom sloju su koristili neki različiti dobro poznati algoritam mašinskog učenja. Ako algoritmi iz prva dva sloja daju istu predikciju, tu predikciju su usvajali kao finalnu predikciju celog modela. U suprotnom algoritam iz trećeg sloja je odlučivao o predikciji. Uspeli su da ostvare tačnost od 97%. Fette i dr. [29] su koristili samo 10 ulaznih atributa, od kojih su 7 binarni. Preostala 3 atributa su broj linkova i domena i maksimalan broj tačaka u nekom od linkova mejla. Binarni atributi uključuju prisustvo linkova koji koriste IP adresu, proveru da li je domen skoro registrovan i slično. Model je baziran na klasifikatoru slučajne šume koji sadrži 10 sabala odluke, i detektuje preko 96% fišing mejlova. Gualberto i dr. [36] su najpre kreirali dokument-termin matricu, u kojoj je svaki email predstavljen jednim redom, a svaki termin jednom kolonom. Zatim su vršili redukciju dimenzionalnosti i na kraju primenili XGBoost (eng. extreme gradient boosting) klasifikator. Model je dostigao tačnosti od 99.95% i 99.58%, u zavisnosti od toga da li je ili nije koristio i dodatno znanje iz Wordnet-a [77] (leksičke baze sa semantičkim relacijama među rečima).

Gangavarapu i Jaidhar [32] su predstavili bio-inspirisanu metaheuristiku kako bi izvukli podskup značajnih atributa iz skupa od 40 atributa vezanih za sadržaj i ponašanje i 200 atributa iz Doc2Vec-a. Koristili su višeslojni perceptron za klasifikaciju nepoželjne pošte, i ostvarili su tačnost od 99.4% prilikom klasifikacije između regularnih i fišing mejlova. Koristeći fastText [3] kako bi predstavili tekst u obliku vektora, Ganesh i dr. [31] su kreirali klasifikator fišing mejlova koji je dostigao F_1-skor of 99%. Yasin i Abuhasan [114] su koristili inteligentno pretprocesiranje podataka kako bi izvukli skup od 16 atributa pogodnih za problem detekcije fišinga. Nakon evaluacije njihovog pristupa pomoću 10-struke unakrsne validacije (eng. 10-fold cross-validation), ostvarili su tačnost od 99.1% koristeći klasifikator slučajne šume. Gangavarapu i dr. [33] su sproveli detaljnu analizu u kojoj su poredili više različitih tehnika ekstrakcije atributa i metoda mašinskog učenja u cilju klasifikacije nepoželjne elektronske pošte. Odabirom svega 21-og atributa iz originalnog skupa od 40 atributa, uspeli su da ostvare tačnost od 99.4% koristeći klasifikator slučajne šume. Akinyelu i Adewumi [5] su izvukli skup od 15 bitnih atributa i takođe primenili klasifikator slučajne šume. Oni su uspeli da dostignu tačnost od 99.7% na korpusu koji sadrži fišing i regularne mejlove u odnosu 1:9.

Koristeći skip-gram arhitekturu iz Word2Vec-a [76] za učenje vektora ugradnje reči, i LSTM arhitekturu rekurentne ćelije, Vinayakumar i dr. [89] su ostvarili tačnost od 99.1% u 10-strukoj unakrsnoj validaciji. Feng i dr. [26] su predložili pristup za detekciju fišing mejlova zasnovan na rekurentnoj konvolucionoj neuronskoj mreži [66]. Oni su takođe koristili Word2Vec za učenje vektora ugradnji karaktera i reči. Ostvarili su tačnost od 99.848% i F_1-skor od 99.331%. Moradpoor i dr. [81] su iskoristili Word2Vec vektore ugradnje i 4 dodatna atributa kao ulaz njihovog klasifikatora baziranog na neuronskoj mreži. Ostvarili su tačnost od 91.5% na skupu za testiranje.

Hagleš i dr. [38] su predložili pristup za detekciju fišing mejlova baziran na rekurentnoj neuronskoj mreži. Odabrali su 5000 tokena (uključujući i specijalne tokene) iz tekstova mejlova, dodelili svakom tokenu jedinstveni identifikator (ID), i predstavili su svaki mejl kao niz ID-eva tokena. Svaki ID tokena će kasnije biti zamenjen svojim vektorom ugradnje veličine 200. Tako dobijeni niz vektora ugradnje biće ulaz rekurentne neuronske mreže. Svi vektori ugradnje se uče zajedno sa ostalim parametrima modela. Ovaj model je ostvario tačnost od 98.91% i F_1-skor od 98.63%. Učeći vektore ugradnje reči zajedno sa ostalim parametrima konvolucione neuronske mreže, Hiransha i dr. [43] su dostigli tačnosti od 94.2% i 96.8%, u zavisnosti od toga da li su ili nisu koristili zaglavlje mejla prilikom klasifikacije. Radovi [38] i [43] su od pomenutih radova najsličniji metodu koji će u ovom poglavlju biti predstavljen, jer se u oba simultano uče vektori ugradnje i svi ostali parametri neuronske mreže.

5.2 Korpus

Svi mejlovi koji će biti korišćeni za treniranje i evaluaciju modela biće iz dva korpusa: SpamAssassin [72] javnog korpusa i Nazario [83] fišing korpusa. Prvi sadrži regularne mejlova, a drugi fišing mejlove.

Iz SpamAssassin javnog korpusa su korišćeni mejlovi iz fajlova čije se ime završava

[3]FastText. Dostupno na: `https://fasttext.cc/`

sa "ham". Uzeti su mejlovi iz ukupno tri ovakva fajla: "20030228_easy_ham.tar.bz2", "20030228_hard_ham.tar.bz2"i "20030228_easy_ham_2.tar.bz2". Prvi ("easy_ham") sadrži regularne mejlove koji ne sadrže HTML elemente i relativno su laki za klasifikaciju. Fajl "hard_hamšadrži komplikovanije regularne mejlove, koji uključuju HTML elemente, imaju rečenice koje zvuče sumnjivo i slično. Treći fajl sadrži regularne mejlove koji su naknadno dodati u korpus. Ostali fajlovi sadrže drugačije i zastarele verzije ovih fajlova, ili spam mejlove, pa kao takvi neće biti uključeni u analizu. Finalni korpus sadrži ukupno 4150 regularnih mejlova.

Fišing mejlovi su uzeti iz fajla "phishing3.mbox"iz Nazario fišing korpusa. Ti isti mejlovi su korišćeni i u analizi [36] koju su Gualberto i dr. sproveli. Ukupno 2279 fišing mejlova je uvezeno u finalni korpus koji će biti korišćen za treniranje i evaluaciju.

I regularni i fišing mejlovi su u neobrađenom formatu u originalnim korpusima. Iz svakog mejla će najpre biti ekstraktovan predmet (eng. subject) i tekstualni sadržaj iz tela mejla. Neki mejlovi sadrže više delova, i u tom slučaju će predmet i tekstovi iz svih tih delova biti spojeni dodavanjem karaktera novog reda između. Tekstovi iz fajlova koji su pridodati mejlovima biće ignorisani u ovoj analizi. Neki delovi su u HTML formatu, i iz njih će biti korišćeni samo tekstualni delovi i linkovi (često nazivani i vezama). Iz svakog linka če biti izvučen i tekst koji se prikazuje i URL adresa ka kojoj vodi dati link. Biblioteka html2text[4] je korišćena za ekstrakciju teksta iz linkova.

5.3 Model za detekciju fišing mejlova

5.3.1 Rečnici karaktera i reči

Neuronske mreže obrađuju numeričke podatke, dok mejlovi sadrže tekstualne podatke. Kako bi procesirali mejlove neuronskom mrežom, iz svakog od njih ćemo najpre izvući tokene. Koristićemo dva tipa tokena, karaktere i reči. Definišemo reč kao sekvencu uzastopnih slova koja nije deo neke veće sekvence uzastopnih slova. Na primer, ukoliko mejl sadrži tekst "Pozdrav, Nikola", dve reči će biti ekstraktovane, "Pozdrav"i "Nikola". Isti tekst sadrži petnaest karaktera: "P", "o", ž", đ", "r", "a", "v", ",", , "N", "i", "k", "o", "l", "a".

Biće kreiran rečnik svih tokena za svaki od dva tipa tokena posebno. Samo će tokeni iz skupa podataka za treniranje biti korišćeni za kreiranje rečnika. Kako vektori ugradnje tokena naučeni iz svega par pojavljivanja nekog tokena mogu da budu zavaravajući, samo će tokeni koji se pojavljuju u najmanje deset mejlova biti uključeni u rečnik. Na ovaj način se smanjuje i ukupan broj parametara modela koje treba naučiti, i samim tim model postaje kompaktniji. Deset je izabran kao dobar balans između izbacivanja retkih tokena koji predstavljaju šum i zavaravaju model, i zadržavanja drugih tokena koji sadrže korisne informacije za klasifikaciju.

Oba rečnika sadrže dva dodatna tokena, jedan koji predstavlja nepoznate tokene i jedan za dopunjavanje (eng. padding). Nepoznatim se smatraju svi tokeni koji se pojavljuju u manje od 10 mejlova u trening skupu, kao i svi tokeni koji se ne pojavljuju u trening skupu (oni tokeni na koje će model naići prvi put tokom testiranja). Tretiranjem

[4]Html2text. Dostupno na: `https://pypi.org/project/html2text/`

retkih tokena kao nepoznatih za vreme treniranja poboljšava se generalizacija modela, jer se model priprema da procesira i nove retke tokene koji se prvi put pojavljuju tokom testiranja. Kako je 10-struka unakrsna validacija korišćena za evaluaciju modela, i svaka od 10 faza evaluacije koristi različiti skup za treniranje, rečnici koji će biti kreirani tokom svake od faza se mogu međusobno razlikovati.

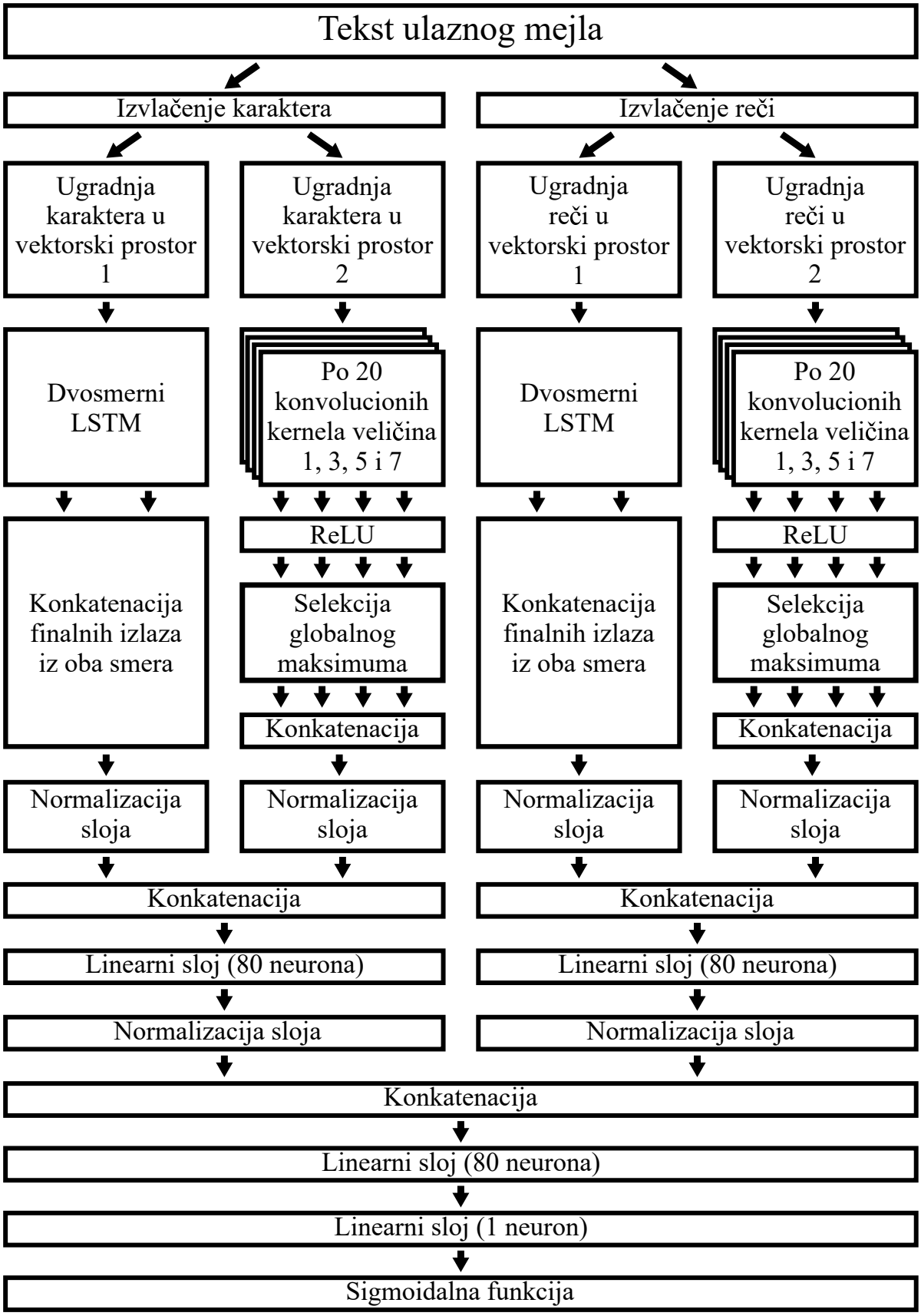
Svaki token iz rečnika ima jedinstveni identifikacioni broj (ID). ID sa vrednošću 0 je rezervisan za token dopunjavanja, a ID sa vrednošću 1 za sve nepoznate tokene. Kada je potrebno procesirati neki mejl, najpre se ekstraktuju dve sekvence tokena, sekvenca karaktera i sekvenca reči. Nakon toga se ove dve sekvence transformišu, tako što se svi tokeni koji imaju svoj ID zamene njime, a svi ostali tokeni se zamene jedinicama. Kako će model procesirati mejlove u mini-grupama, kraće sekvence će biti dopunjene nulama, tako da sve sekvence istog tipa (sa karakterima ili sa rečima) u jednoj mini-grupi imaju istu dužinu. Rezultat ovog procesa su dve 2-dimenzione matrice sa ID-evima. Broj redova svake od njih je jednak veličini mini-grupe, dok je broj kolona jednak dužini najduže sekvence u mini-grupi (broj kolona dve matrice može da se razlikuje, jer je često broj karaktera i reči u tekstu mejla različit).

5.3.2 Arhitektura modela

Model će početi procesiranje svake od dve matrice ID-eva sa blokom iste strukture ali različitih parametara. Svaki blok ima dva sloja ugradnje, jedan za rekurentno i jedan za konvolucionalno procesiranje (postoje ukupno četiri sloja ugradnje u celoj arhitekturi). Sve matrice parametara slojeva ugradnje imaju po 40 kolona (veličina vektora ugradnje). Broj redova je jednak broju elemenata u rečniku (broj zavisi od toga da li se radi o sloju ugradnje karaktera ili reči). U svakom od dva bloka, ID-evi iz matrice ID-eva će biti zamenjeni odgovarajućim vektorima ugradnje iz ovih matrica. Na ovaj način će se u svakom od blokova kreirati po dva 3-dimenziona izlaza slojeva ugradnje, jedan za rekurentno i jedan za konvoluciono procesiranje. Prva dimenzija predstavlja veličinu mini-grupe, druga dužinu najduže sekvence u grupi, a treća veličinu vektora ugradnje (40).

Za rekurentno procesiranje je korišćena dvosmerna LSTM rekurentna ćelija. Ona je odabrana zato što ima dosta manje problema sa iščezavajućim gradijentima u odnosu na ranije arhitekture rekurentnih ćelija. Dvosmerna varijanta dodatno obogaćuje novokreiranu reprezentaciju ulazne sekvence. Poslednja skrivena stanja iz oba smera će biti nadovezana, i to će činiti rekurentnu reprezentaciju bloka. Veličina skrivenog stanja koja će se koristiti biće 40, pa će rekurentna reprezentacija bloka biti 2-dimenziona, gde prva dimenzija predstavlja veličinu mini-grupe, a druga veličinu dva nadovezana skrivena sloja LSTM ćelije (80). Rekurentna ćelija procesira čitavu ulaznu sekvencu, pa na taj način ekstraktuje globalne atribute bazirane na sadržaju celog mejla.

Konvoluciono procesiranje je bazirano na radu Kim-a i dr. [60]. Procesiranje počinje sa četiri 1-dimenziona konvoluciona sloja sa različitim veličinama kernela (1, 3, 5 i 7). Svaki od njih sadrži po 20 kernela, i dopunjavanje će biti primenjeno na obe strane ulaza za veličinu kernela (dopunjavanjem se proširuje druga dimenzija iz 3-dimenzionog ulaza). Veličine kernela i njihov broj su određeni empirijski, sa ciljem maksimizacije rezultata klasifikacije. Konvolucija se primenjuje duž sekvence. Nakon toga će biti prime-



Slika 5.1: Arhitektura modela

njena ReLU nelinearnost i selekcija globalnog maksimuma nad izlazom svakog od četiri konvoluciona sloja. Kako su kraće sekvence tokena u mini-grupi dopunjene nulama, primenjivanjem dopunjavanja u konvolucionim slojevima obezbeđuje se da izlaz selekcije globalnog maksimuma za neki mejl bude isti bez obzira na dužine sekvenci tokena ostalih mejlova iz iste mini-grupe. Selekcija globalnog maksimuma se koristi da izračuna najjače aktivacije duž ulazne sekvence. Nakon njene primene svaki od 4 izlaza biće 2-dimenzioni, gde prva dimenzija reprezentuje mini-grupu, a druga broj kernela (20). Da bi se kreirala konvoluciona reprezentacija bloka, sva ova 4 izlaza biće nadovezana po drugoj dimenziji (koja će sada imati veličinu $4 \cdot 20 = 80$). Konvolucioni slojevi ekstraktuju lokalne atribute. Veličina kernela direktno utiče na broj uzastopnih elemenata ulazne sekvence koji će uticati na ekstrakciju atributa. Više različitih veličina kernela se koristi sa ciljem da bi se ekstaktovali raznovrsniji lokalni atributi.

Nakon toga će biti primenjena normalizacija sloja nad obe reprezentacija (rekurentnu i konvolucionu) za svaki od dva bloka (jedan koji analizira karaktere i jedan koji analizira reči). Normalizacija sloja se koristi kako bi unapredila efikasnost i brzinu treniranja, ali i da unapredi generalizaciju modela. Da bi se kreirala reprezentacija bloka, najpre se nadovežu njegove dve normalizovane reprezentacije (rekurentna i konvoluciona), a zatim se primeni potpuno povezani linearni sloj. Linearni sloj ima 160 ulaznih i 80 izlaznih neurona.

Reprezentacije oba bloka, jednog koji procesira karaktere i jednog koji procesira reči, se najpre normalizuju normalizacijom sloja, a zatim se tako dobijeni izlazi spajaju. Nakon toga se primenjuje jedan potpuno povezani linearni sloj, koji kombinuje reprezentacije oba bloka u jednu reprezentaciju. Ovaj sloj ima 160 ulaznih i 80 izlaznih neurona. Na kraju se koristi još jedan linearni sloj sa jednim izlaznim neuronom. Ovaj neuron ima sigmoidalnu aktivacionu funkciju, i njegov izlaz reprezentuje verovatnoću da se radi o fišing mejlu. Cela arhitektura je prikazana na slici 5.1.

5.4 Evaluacija

Konfiguracioni detalji modela i procedure treniranja su dati u tabeli 5.1. Za treniranje modela je korišćena kriterijumska funkcija binarne unakrsne entropije. Parametri se ažuriraju korišćenjem Adam [61] optimizacionog algoritma, sa veličinom koraka 0.001. Model se trenira u 4 epohe, korišćenjem mini-grupa veličine 32. Da bi se kreirale mini-grupe, najpre se mejlovi sortiraju po dužini teksta, a zatim se susedni mejlovi grupišu u mini-grupe. Kreiranje mini-grupa sa mejlovima sličnih dužina, a samim tim i sličnim brojem tokena, drastično može da ubrza treniranje modela. Pre svake epohe se na slučajan način određuje redosled po kojem će model da koristi mini-grupe za treniranje. Čest je slučaj da se u literaturi odbace suviše mali ili veliki mejlovi iz korpusa, ili da se oni skrate ili prošire na određenu dužinu. Iako ovo može da bude praktično za testiranje na korpusu, ovako kreirani klasifikatori mogu lako da budu prevareni u realnoj primeni. Zbog toga ćemo sve mejlove koristiti u svojoj punoj dužini.

U cilju evaluacije modela prikupljeni su fišing mejlovi iz Nazario fišing korpusa i regularni mejlovi iz SpamAssassin javnog korpusa. Za svaki mejl je ekstraktovan tekst iz njegovog zaglavlja i tela. Kako su veličine ova dva korpusa relativno male, često korišćeni

Tabela 5.1: Konfiguracioni detalji modela

Sloj	Broj	Detalji
Sloj ugradnje	4	veličina vektora ugradnje: 40
Dvosmerni LSTM	2	veličina skrivenog sloja: 40
1d konvolucija	2	20 kernela veličine 1 20 kernela veličine 3 20 kernela veličine 5 20 kernela veličine 7
Normalizacija sloja	6	$\epsilon = 10^{-5}$
Linearni sloj	3	veličina ulaza: 160, veličina izlaza: 80
	1	veličina ulaza: 80, veličina izlaza: 1
Selekcija globalnog maksimuma	2	
Optimizacioni detalji	algoritam	Adam
	broj epoha	4
	veličina koraka	10^{-3}
	veličina grupe	32

Tabela 5.2: Broj regularnih i fišing mejlova u svakoj grupi

Broj grupe	Regularni	Fišing	Broj grupe	Regularni	Fišing
1	415	227	7	415	228
2	415	228	8	415	228
3	415	228	9	415	228
4	415	228	10	415	228
5	415	228			
6	415	228	Total	4150	2279

metod podele na skupove za treniranje, velidaciju i testiranje može da uzrokuje nestabilne rezultate. Zbog toga će se kao metod evaluacije koristiti 10-struka unakrsna validacija. Svi mejlovi će biti na slučajan način podeljeni u 10 grupa, zadržavajući u svakoj grupi približno isti odnos između broja fišing i regularnih mejlova kao što je bio odnos broja primera u dva originalna korpusa. Tabela 5.2 prikazuje tačan broj regularnih i fišing mejlova u svakoj grupi. U svakom od 10 eksperimenata, mejlovi iz jedne grupe će se koristiti za testiranja, dok će mejlovi iz preostalih 9 grupa biti korišćeni za treniranje. Na taj način će svaki mejl biti korišćen tačno jednom za testiranje, i ti rezultati će biti korišćeni za računanje svih evaluacionih metrika.

Da bi se evaluirao predloženi model za detekciju fišing mejlova, korišćene su sledeće metrike: tačnost, preciznost, senzitivnost, F_1-skor i stopa lažno pozitivnih (skraćeno SLP; eng. false positive rate). Metrike ovih formula su date u jednačinama od (5.1) do (5.5), u funkciji od broja istinito pozitivnih (**IP**), lažno negativnih (**LN**), lažno pozitivnih (**LP**) i istinito negativnih (**IN**) primera. U cilju upoređivanja različitih varijanti predloženog modela, biće korišćene matrice grešaka (eng. error matrix ili confusion matrix). Takođe će biti prikazana i ROC kriva (eng. receiver operating characteristic) našeg modela.

ROC kriva prikazuje odnos senzitivnosti i stope lažno pozitivnih u različitim graničnim tačkama. Celokupan proces treniranja i evaluacije je sumiran pseudo kodom datim u algoritmu 1.

$$\text{Tačnost} = \frac{IP + IN}{IP + LN + LP + IN} \tag{5.1}$$

$$\text{Preciznost} = \frac{IP}{IP + LP} \tag{5.2}$$

$$\text{Senzitivnost} = \frac{IP}{IP + LN} \tag{5.3}$$

$$\text{F}_1\text{-skor} = \frac{2 \cdot \text{Preciznost} \cdot \text{Senzitivnost}}{\text{Preciznost} + \text{Senzitivnost}} \tag{5.4}$$

$$\text{SLP} = \frac{LP}{LP + IN} \tag{5.5}$$

Algoritam 1: Procedura teniranja i evaluacije

Podaci: $F_1 = (X_1, Y_1), \ldots, F_{10} = (X_{10}, Y_{10})$; `// grupe iz korpusa`
for $testInd \leftarrow 1$ **to** 10 **do**
  $model$ = inicijalizujModel();
  $optimizator$ = inicijalizujAdamOptimizator($model.parametri$);
  $D_{trening} = \bigcup_{i=1, i \neq testInd}^{10} F_i$;
  $D_{test} = F_{testInd}$;
  $Rečnik_{karaktera}, Rečnik_{reči}$ = kreirajRečnike($D_{trening}$);
  $miniGrupe$ = podeli(sortiraj($D_{trening}$)); `// sortiranje mejlova po dužini teksta i podela u mini-grupe veličine 32`
  for $epoha \leftarrow 1$ **to** 4 **do**
    pomešaj($miniGrupe$);
    foreach $miniGrupa=(X_{mg}, Y_{mg})$ **of** $miniGrupe$ **do**
      $tokeniKaraktera$ = tokenizuj($X_{mg}, Rečnik_{karaktera}$);
      $tokeniReči$ = tokenizuj($X_{mg}, Rečnik_{\text{reči}}$);
      $izlaz$ = prolazakNapred($model, tokeniKaraktera, tokeniReči$);
      $greška$ = greškaBinarneUnakrsneEntropije($izlaz, Y_{mg}$);
      $gradijenti = \frac{\partial greška}{\partial model.parametri}$; `// izračunato korišćenjem bekpropagacije`
      $optimizator$.ažurirajParametreModela($gradijenti$);
    end
  end
  $P_{testInd}$ = napraviPredikcije($model, X_{testInd}$);
end
izračunajEvaluacioneMetrike($Y_1, \ldots, Y_{10}; P_1, \ldots, P_{10}$);

Tabela 5.3 prikazuje rezultate klasifikacije predloženog modela. Prikazan je rezultat punog modela, kao i rezultati verzija modela koje koriste samo karaktere ili samo reči

Tabela 5.3: Klasifikacioni rezultati modela

Model	Tačnost	Preciznost	Senzitivnost	F_1-skor	SLP
Puni model	0.99813	0.99824	0.99649	0.99736	0.00096
Samo karakteri	0.99564	0.99429	0.99342	0.99385	0.00313
Samo reči	0.99362	0.99513	0.98684	0.99097	0.00265

Tabela 5.4: Matrice grešaka različitih varijanti predloženog modela

(a) Puni model

		Predikcija	
		Regularan	Fišing
Labela	Regularan	4146	4
	Fišing	8	2271

(b) Samo karakteri

		Predikcija	
		Regularan	Fišing
Labela	Regularan	4137	13
	Fišing	15	2264

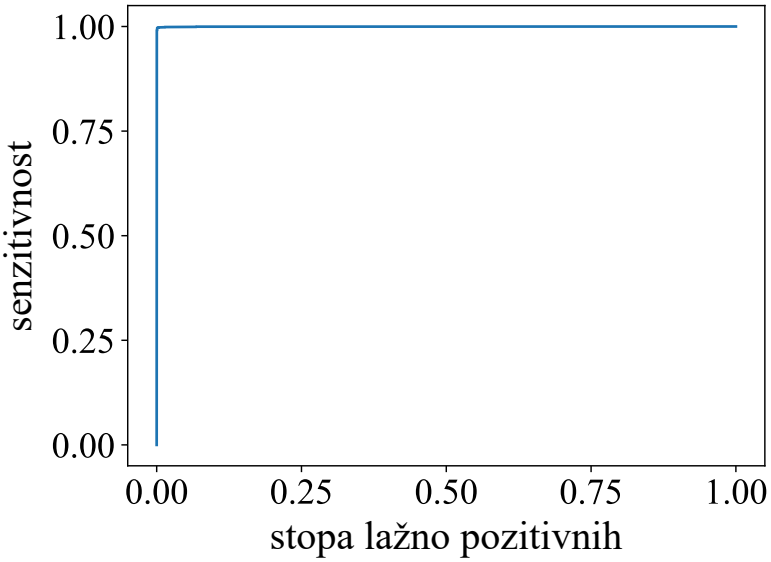
(c) Samo reči

		Predikcija	
		Regularan	Fišing
Labela	Regularan	4139	11
	Fišing	30	2249

(sadrže levu ili desnu stranu iz arhitekture prikazane na slici 5.1 do sloja konkatenacije koji spaja normalizovanu rekurentnu i konvolucionu reprezentaciju, kao i sve nakon poslednjeg sloja konkatenacije). Iz tabele se može videti da je varijanta modela koja koristi i karaktere i reči postigla bolje rezultate nego modeli koji koriste samo jedan tip tokena. Ovaj model je dostigao tačnost od 99.81% i F_1-skor od 99.74%. Model koji koristi samo karaktere se pokazao kao bolji od modela koji koristi samo reči. Ova dva modela su postigla tačnosti od 99.56% i 99.36%, redom. Model koji koristi samo reči je ostvario nešto bolju preciznost i stopu lažno pozitivnih. Glavna razlika između ova dva modela je u senzitivnosti, gde se model koji koristi samo karaktere pokazao kao značajno bolji. Puni model (koji koristi i karaktere i reči) je dao bolje rezultate nego druge dve varijante modela po svih pet evaluacionih metrika.

Tabela 5.4 prikazuje matrice grešaka punog modela i modela koji koriste samo jedan tip tokena. Puni model je pogrešno klasifikovao svega 4 regularnih i 8 fišing mejlova. Ova varijanta modela ujedno ima i najmanji broj lažno pozitivnih i najmanji broj lažno negativnih primera. Varijanta modela koja koristi samo reči ima nešto manji broj lažno pozitivnih primera u odnosu na varijantu modela koja koristi samo karaktere, dok ova druga ima dva puta manje lažno negativnih primera od nje. Grafik 5.2 prikazuje ROC krivu predoženog modela. Kako je teško vizuelno razlikovati između ROC krivih ove tri varijante modela, na grafiku je prikazan samo grafik ROC krive punog modela. Sa krive se jasno vidi da model uspešno razdvaja između fišing i regularnih mejlova.

U tabeli 5.5 je dato poređenje rezultata predloženog modela sa rezultatima trenutno najboljih modela za detekciju fišing mejlova iz literature. Dva modela [29, 36] imaju i varijante koje dodatno ekstraktuju znanje iz eksternih izvora, kao što su klasifikacioni



Slika 5.2: ROC kriva modela

Tabela 5.5: Poređenje sa drugim modelima

Referenca	Tačnost(%)	F_1-skor(%)
Predloženi pristup	99.81	99.74
Fang i dr. [26]	99.85	99.33
Akinyelu i dr. [5]	99.7	98.45
Gualberto i dr. [36]	99.58	99.58
Gangavarapu i dr. [32]	99.4	nedostupno
Gangavarapu i dr. [33]	99.4	99.4
Yasin i dr. [114]	99.1	99.1
Halgaš i dr. [38]	98.91	98.63
Fette i dr. [29]	98.87	94.68
Islam i dr. [47]	97	nedostupno
Moradpoor i dr. [81]	91.5	92.06

rezultati iz eksternih filtera nepoželjne pošte ili znanja iz WordNet biblioteke. Da bi poređenje učinili pravednijim, u tabeli će za ove modeli biti prikazani rezultati verijanti modela koje ne koriste znanje iz eksternih izvora. Tačnost i F_1-skor nisu dati u [29], pa su zato ove vrednosti izračunate na osnovu datih vrednosti za stopu lažno pozitivnih i stopu lažno negativnih. Tačnost i F_1-skor svih ostalih modela koji su korišćeni za poređenje su dati u tabeli, osim F_1-skorova modela [47] i [32], koji su izostavljeni jer nisu dostupni.

Kao što se može videti iz tabele, nijedan model nema veći F_1-skor od predloženog modela. Drugi najbolji F_1-skor od 99.58% je ostvario model koji su kreirali Gualberto i dr. [36]. Ovaj model takođe ima i nižu tačnost u odnosu na naš model. Model koji su predložili Fang i dr. [26] je jedini model koji je ostvario za nijansu veću tačnost, ali je taj model testiran na veoma neuravnoteženom korpusu, što može da se vidi po F_1-skoru koji su dobili, a koji je značajno niži od našeg. Svi ostali modeli imaju nižu i tačnost i F_1-skor u odnosu na naš model.

Dva pristupa koja su najsličnija našem su predložili Halgaš i dr [38] i Hiransha i dr. [43]. Oba pristupa su bazirana na arhitekturama neuronskih mreža, koje uče vektore ugradnje zajedno sa ostalim parametrima mreže. Ova dva modela su ostvarila tačnosti od 98.91% i 96.8%, redom. Sa tačnošću od 99.81%, naš model predstavlja značajan napredak u odnosu na ove pristupe.

Poglavlje 6

Detekcija fišing veb sajtova

Fišing napadi obično kreću tako što napadači podele linkove ka svojim fišing sajtovima. Oni se često dele preko elektronske pošte, ali ovakvi linkovi mogu da se nađu i na društvenim mrežama, forumima i sličnim sajtovima. Nakon što korisnik klikne na maliciozni link, biva preusmeren ka fišing sajtu. U nekim tipovima fišing napada, kao što je tabnabbing napad, korisnik može biti preusmeren ka fišing sajtu čak i kada posećuje zvanični sajt organizacije koju fišing sajt lažno predstavlja.

Na osnovu izveštaja radne grupe koja se bori protiv fišinga[1] u januaru 2021. godine je zabeležen novi rekord od 245,711 novih fišing sajtova kreiranih samo u tom mesecu. U martu iste godine je zabeleženo preko 200,000 fišing napada. Ovako veliki brojevi ukazuju na ozbiljnost problema, kao i na važnost kreiranja pouzdanih metoda za detekciju fišing sajtova, čime bi se ublažili rizici koje ovakvi napada donose.

Postoji više načina odbrane od fišing napada. Jedan od najintuitivnijih su pravna rešenja. Broj napada se može umanjiti kreiranjem zakona kojima se kažnjavaju ovakve aktivnosti i procesuiranjem napadača. Ipak, fišing sajtovi često postoje u veoma kratkom periodu, pa je potrebno brzo reagovanje organa za sprovođenje zakona. Drugi pristup je u obrazovanju korisnika interneta o sajber opasnostima. Iako ovaj metod može biti efikasan, obično ga je teško implementirati, jer zahteva od korisnika da ulože značajno vreme učeći o fišing metodama. Pored toga, napadači vremenom postaju sve sofisticiraniji u svojim tehnikama imitiranja legitimnih veb sajtova.

Tehnička rešenja su najzastupljeniji vid borbe protiv fišing napada. Jedan pristup je u detekciji samih izvora na kojima korisnici nailaze na fišing linkove, i jedan takav metod je prikazan u prethodnoj glavi. Drugi pristup je da se direktno vrši provera da li je neki sajt fišing sajt, i to će biti pristup koji će se koristiti u ovoj glavi. Modeli mašinskog učenja mogu da učine odbrambene mehanizme boljim u detekciji prethodno neviđenih fišing sajtova. Ekstrakcijom korisnih atributa iz fišing sajtova i primenom naprednih algoritama mašinskog učenja, moguće je identifikovati veći broj fišing sajtova. Diskretni opisni atributi veb stajtova doprinose boljoj interpretabilnosti modela, i omogućavaju jasnije razumevanje faktora koji najviše utiču na klasifikacione odluke.

U ovoj glavi će biti predstavljen model detekcije fišing veb sajtova baziran na konvolucionoj neuronskoj mreži. Crpeći inspiraciju iz primene slojeva ugradnje u procesiranju

[1]The Anti-Phishing Working Group. Phishing Activity Trends Report, 1th Quarter 2021. Dostupno na: `https://docs.apwg.org/reports/apwg_trends_report_q4_2020.pdf`

govornog jezika [76], najpre ćemo dizajnirati sloj ugradnje atributa sajtova u vektorski prostor. Ovaj sloj će biti prilagođen diskretnim opisnim atributima veb sajtova koji se često koriste u literaturi. Zatim će biti odrađena adaptacija sloja ponderisanja CancelOut [18] za korišćenje nad vektorima ugradnje atributa sajtova. Na kraju će biti dizajnirana i sama arhitektura konvolucione mreže koja će koristiti ponderisane vektore ugradnje. U cilju potvrde efektivnosti predloženog modela, biće sprovedena detaljna analiza i testiranje na tri korpusa. Rezultati testiranja će biti upoređeni sa rezultatima testiranja drugih pristupa iz literature, gde će se pokazati da predloženi pristup ima slične ili bolje rezultate od ostalih modela iz literature.

6.1 Pregled literature

Kao što je već naglašeno, postoji veći broj pristupa detekciji fišinga. Kako se metod koji će u ovom poglavlju biti predstavljen zasniva na diskretnim opisnim atributima, analiziraćemo slične ovakve pristupe iz literature.

Ali i Ahmed [9] su predložili arhitekturu duboke neuronske mreže za detekciju fišing veb sajtova. Koristeći tehnike odabira i ponderisanja atributa bazirane na genetskom algoritmu, uspeli su da ostvare tačnosti od 90.39% i 91.13%, redom. Ali i Malebary [10] su primenili optimizaciju roja čestica (eng. particle swarm optimization) za ponderisanje atributa veb sajtova. Svoj model su evaluirali koristeći tehniku 10-struke unakrsne validacije. Najvišu tačnost od 96.83% su ostvarili koristeći klasifikator slučajne šume. Nakon eksperimentisanja sa tri različite tehnike odabira atributa, Thabtah i Abdelhamid [101] su kreirali klasifikator koji koristi samo dva ulazna atributa. Klasifikator je dostigao tačnost od 91.26% u 10-strukoj unakrsnoj validaciji.

Mohammad i dr. [79] su predložili model neuronske mreže sa jednim skrivenim slojem i sa automatskom adaptacijom strukture, u cilju detekcije fišing sajtova. Algoritam je u stanju da automatski podesi veličinu koraka treniranja i da inkrementalno dodaje nove neurone u skriveni sloj. Njihov pristup je ostvario tačnost od 92.48% na skupu za testiranje. Hadi i dr. [37] su pristupili problemu algoritmom asocijativne klasifikacije (eng. associative classification algorithm), koji primenjuje pravila asocijacije kako bi napravio predikcije. Ovaj model ima dobru interpretabilnost, a ostvario je tačnost između 92% i 93%. Još jedan metod koji koristi pristup asocijativne klasifikacije je predložio Alqahtai [13], i on je ostvario tačnost od 95.20% i F_1-skor od 95.11%.

Rajab [90] je primenio dve tehnike odabira atributa. Sa 11 i 9 atributa je uspeo da ostvari klasifikacione greške koje su samo 1.32% i 1.02% veće nego kada bi koristio svih 30 atributa. Penmatsa i Kakarlapudi [88] su primenom mravljeg algoritma (eng. ant colony optimization algorithm) odabrali 23 atributa iz originalnog skupa. Primenom klasifikatora slučajne šume nad tim atributima, uspeli su da ostvare tačnost od 97.26% i F_1-skor od 97.3%. Motlagh i Bardsiri [84] su predložili arhitekturu sa dva skrivena sloja, i njome dostigli tačnost od 93.42%. Abad i dr. [1] su eksperimentisali sa tri algoritma mašinskog učenja u cilju detekcije fišing veb sajtova. Najbolji rezultat postigli su linearnom metodom potpornih vektora, kojom su ostvarili tačnost od 96.71%.

Al-Ahmadi i Lasloum [6] su predstavili model baziran na višeslojnom perceptronu, kojim su ostvarili tačnost od 96.65% i F_1-skor od 96.65% prilikom klasifikacije između

fišing i regularnih veb sajtova. Vrbančić i dr. [104] su eksperimentisali sa dva algoritma inteligencije roja (eng. swarm intelligence), algoritmom slepog miša [113] i algoritmom hibridnog slepog miša [30], kako bi podesili hiperparametre neuronske mreže (broj epoha treniranja, veličinu grupe, korak treniranja i broj neurona u skrivenom sloju). Najbolji model su dobili korišćenjem algoritma slepog miša za podešavanje hiperparametara, i on je ostvario tačnost od 96.5% tokom 10-struke unakrsne validacije. Isti autori su kasnije primenili algoritam inteligencije roja svica (eng. firefly swarm intelligence) [105], kojim su ostvarili tačnost od 96.65% i F_1-skor od 96.61%.

Thabtah i dr. [102] su razvili rešenje za detekciju fišinga zasnovano na neuronskoj mreži. Njihov algoritam dinamički podešava strukturalne parametre mreže za vreme treniranja, sa ciljem dobijanja klasifikatora sa visokom tačnošću i dobrom generalizacijom. Model je dostigao tačnost od 93.06%. Al-Sarem i dr. [8] su predložili metod iz više koraka za detekciju fišing veb sajtova. Najpre su trenirali više modela mašinskog učenja bez optimizacije hiperparametara. Zatim su unapredili modele koristeći genetski algoritam u cilju nalaženja optimalnih hiperparametara. Tada su odabrali najbolje modele, i od njih kreirali ansambl modela za klasifikaciju, koji je ostvario tačnost od 97.16%. Jalal i dr. [49] su eksperimentisali sa algoritmom sličajne šume, stablom odluka, linearnim modelom i neuronskom mrežom. Najvišu tačnost od 95.7% je postigao algoritam slučajne šume.

Lakshmi i dr. [67] su rezvili model dubokog učenja za detekciju fišing veb sajtova. Koristili su algoritam Adam za optimizaciju, i model je ostvario tačnost od 96%. Parra i dr. [85] su predložili distribuirani metod dubokog učenja za detekciju fišing napada. Za detekciju fišing veb sajtova je korišćena konvoluciona neuronska mreža, koja je ostvarila tačnost od 94.3%. Al-Milli i dr. [7] su predstavili 1-dimenzionu konvolucionu neuronsku mrežu za detekciju fišing veb sajtova, i njome ostvarili tačnost od 94.31%. Binarni klasifikator zasnovan na LSTM rekurentnoj ćeliji su predložili Wang i dr. [107]. On se pokazao boljim u odnosu na klasifikator slučajne šume, i ostvario je tačnost od 95.47%. Metodi predstavljeni u [6, 67, 9, 102, 85, 7, 107] su bazirani na neuronskim mrežama, i kao takvi su najsličniji metodu koji će biti predstavljen u ovom poglavlju.

6.2 Korpus

Kako bi se kreirao efikasan model za detekciju fišing veb sajtova baziran na mašinskom učenju, potrebno je najpre odabrati korisne atribute iz veb sajtova. Po istraživanju koje su radili Mohammad i dr. [78], postoje četiri kategorije značajnih atributa za datekciju fišing veb sajtova: atributi adresne linije (eng. address bar), atributi abnormalnosti, HTML i JavaScript atributi i atributi domena. Svi ovi atributi automatski mogu da se ekstraktuju iz veb sajtova, bez potrebe za ljudskom ekspertizom u procesu ekstrakcije. Pored toga što su ovi atributi korisni za detekciju fišing veb sajtova, ovi atributi su takođe opisni, što poboljšava interpretabilnost odluka modela. Zbog svih ovih prednosti koje ovakvi atributi poseduju, oni će biti korišćeni u modelu koji će biti predstavljen u ovom poglavlju.

Za evaluaciju modela će biti korišćena tri korpusa. Prvi korišćeni korpus (Korpus-1) je takozvani korus fišing veb sajtova, koji se često koristi u literaturi detekcije fišinga. Korpus sadrži 11,055 veb sajtova, od čega je 4,898 fišing veb sajtova i 6,157 regularnih

veb sajtova. Svaki veb sajt je u korpusu predstavljen sa 30 atributa: 12 atributa adresne linije, 6 atributa abnormalnosti, 5 HTML i JavaScript atributa i 7 atributa domena. Svi atributi su dati u tabeli 6.1.

Tabela 6.1: Atributi korpusa fišing veb sajtova

Atributi adresne linije
Korišćenje IP adrese umesto domena
Dugačak URL kako bi sakrio sumnjive delove
Korišćenje servisa za skraćivanje URL-a (TinyURL)
URL sadrži "@šimbol
Redirekcija korišćenjem "//"
Dodavanje prefiksa ili sufiksa domenu odvojenog sa -"
Broj poddomena
Korišćenje HTTPS-a
Period registracije domena
Korišćenje Favikona van adresne linije
Korišćenje nestandardnih portova
Postojanje "HTTPS"tokena u delu URL-a za domen
Atributi abnormalnosti
URL-ovi ka sadržaju sa spoljnim domenom
Linkovi sa spoljnim domenima
Linkovi u <meta>, <script> i <link> oznakama
Obrađivač poslatih formi
Slanje informacija na mejl
Abnormalan URL
HTML i JavaScript atributi
Preusmeravanje na sajtu
Prilagođavanje statusne linije
Onemogućen desni klik
Korišćenje iskačujućih prozora
IFrame redirekcija
Atributi domena
Starost domena
DNS zapis
Saobraćaj veb sajta
PageRank
Google Indeks
Broj linkova koji pokazuju ka stranici
Atribut baziran na statističkim izveštajima

Drugi korpus (Korpus-2) sadrži 2,456 primera. Svaki veb sajt je u korpusu predstavljen sa istih 30 atributa koji su korišćeni i u prvom korpusu. Korpus sadrži 1,362 fišing veb sajtova i 1,094 regularnih sajtova. I Korpus-1 i Korpus-2 se nalaze na UCI

repozitorijumu za mašinsko učenje[2], na kome se nalazi i detaljan opis atributa.

Treći korpus (Korpus-3) je kreirala Neda Abdelhamid i nalazi se na ripozitorijumu za mašinsko učenje Univerziteta u Ervajnu [3]. Svaki veb sajt je predstavljen sa 9 atributa u ovom korpusu. Korpus sadrži 702 fišing i 548 regularnih primera. Korpus takođe sadrži i 103 primera koji nisu označeni ni kao regularni ni kao fišing, i ovi primeri nisu uzeti za analizu. Detaljan opis atributa korpusa je dat u uvodnom radu [2].

Svaki atribut ima dve ili tri moguće vrednosti. Vrednost atributa -1 ukazuje da je na osnovu tog atributa verovatnije da se radi o fišing veb sajtu. Vrednost atributa 1 ukazuje da je na osnovu tog atributa verovatnije da se radi o regularnom veb sajtu. Atributi sa tri moguće vrednosti mogu da imaju i vrednost 0, što ukazuje da na osnovu tog atributa veb sajt može da bude sumnjiv.

Jedan primer atributa koji može da ima dve moguće vrednost je "korišćenje IP adrese umesto domena". Ukoliko se IP adresa koristi umesto domena (na primer ukoliko je URL stranice "http://125.98.3.123/fake.html"), vrednost atributa će biti -1, što ukazuje da je verovatnije da se radi o fišing veb sajtu. Ukoliko se ime domena nalazi u URL-u umesto IP adrese, atribut će imati vrednost 1. Primer atributa sa tri moguće vrednosti je "broj poddomena". Ukoliko URL ne sadrži poddomene vrednost atributa će biti 1. Ukoliko sadrži jedan poddomen, vrednost će biti 0, a ako sadrži dva ili više poddomena, vrednost će biti -1. Napadači nekada koriste više poddomena kako bi sakrili originalni domen njihovog veb sajta i obmanuli korisnike.

6.3 Model za detekciju fišing sajtova

6.3.1 Ugradnja atributa veb sajtova u vektorski prostor

Kako vrednosti atributa -1 i 1 redom ukazuju na fišing i regularne karakteristike atributa, a vrednost 0 označava nešto između ove dve karakteristike, poželjno je to i inkorporirati u sam sloj ugradnje atributa veb sajtova u vektorski prostor. Za svaki ulazni atribut postojaće dva vektora ugradnje: jedan za fišing karakteristike (e^i_{-1}) i jedan za regularne karakteristike (e^i_1). Oba su d-dimenzioni vektori realnih brojeva, i u konkretnoj implementaciji modela d će biti 100. Sa i je označen indeks atributa. Ukoliko sa n označimo ukupan broj ulaznih atributa, sloj ugradnje će sadržati ukupno $2n$ vektora ugradnje.

Da bi se ulazni atributi veb sajta ugradili u vektorski prostor, biće učinjeno sledeće. Ukoliko je vrednost i-tog atributa -1, njegov vektor ugradnje biće e^i_{-1}. Ukoliko je vrednost 1, vektor ugradnje će biti e^i_1. Ukoliko je vrednost atributa 0, iskombinovaćemo vektore ugradnje za fišing i regularne karakteristike, pa ćemo vektor ugradnje ovog atributa modelirati sa $(e^i_{-1} + e^i_1)/2$.

Ovakav sloj ugradnje će transformisati vektor ulaznih atributa veb sajta u matricu dimenzije $n \times d$. Nad ovom matricom će kasnije biti primenjeno ponderisanje, nakon

[2]UCI Machine Learning Repository - Phishing Websites Dataset. Irvine, University of California, School of Information and Computer Science, 2012. Dostupno na: `https://archive.ics.uci.edu/ml/datasets/Phishing+Websites`

[3]Irvine, CA: University of California, School of Information and Computer Science, Machine Learning Repository, 2016. Dostupno na: `https://archive.ics.uci.edu/dataset/379/website+phishing`

čega će biti korišćena kao ulaz konvolucionog modela, što će biti opisano u nastavku. Parametri svih vektora ugradnje biće ažurirani tokom procesa treniranja, zajedno sa svim ostalim parametrima modela.

6.3.2 Ponderisanje vektora ugradnje atributa

Kako je teško razumeti šta tačno utiče na odluke dubokih neuronskih mreža, one se često opisuju kao crne kutije (eng. black-box). To uglavnom nije slučaj kod tradicionalnih modela mašinskog učenja. U linearnim modelima, svakom atributu je pridružen jedan parametar, koji opisuje njegov uticaj na odluku. U modelima baziranim na stablima odluke, svaki čvor je lako tumačiti. Jedna strategija koja pomaže kod tumačenja dubokih neuronskih mreža je ponderisanje atributa.

Korišćenje ponderisanja i selekcije atributa u cilju boljeg tumačenja je već istraženo u nekim modelima detekcije fišing veb sajtova iz literature [9, 10, 88, 101]. U trenutnim pristupima, proces se sastoji iz dva koraka. u prvom koraku se vrši ponderisanje atributa, dok se u drugom koraku trenira model koristeći atribute sa najvećim težinama. Obavljanje ova dva zadatka nezavisno umanjuje mogućnost da procesi treniranja modela i ponderisanja atributa međusovno poboljšaju jedan drugog.

Pristupi s kraja na kraj (eng. end-to-end) su sve prisutniji u dubokom učenju. Oni pojednostavljuju proces kreiranja, skladištenja i korišćenja dubokih modela mašinskog učenja, a obično i ostvaruju bolje rezultate. Jedan metod koji omogućava ponderisanje atributa i treniranje neuronske mreže po ovakvom pristupu je CancelOut [18] sloj ponderisanja.

U njihovom pristupu, model na ulazu dobija N-dimenzioni vektor. Sloj ponderisanja sadrži vektor parametara $W_{CO} \in \mathbb{R}^N$. Oni koriste sigmoidalnu nelinearnost kako bi ograničili težine na opseg od 0 do 1. Ukoliko sa x označimo ulazni vektor, izlaz sloja je proizvod po elementima između ulaznih vrednosti i ograničenih težina, $x \odot \sigma(W_{CO})$.

U našem modelu neće biti ponderisanja individualnih ulaznih vrednosti. Kako je svaki ulazni atribut veb sajta ugrađen u vektorski prostor, vršiće se ponderisanje nad vektorima ugradnje. Označimo sa $X \in \mathbb{R}^{n \times d}$ izlaz sloja ugradnje za jedan veb sajt. On sadrži n vektora ugradnje, po jedan za svaki ulazni atribut. Najpre ćemo svaki od njih normalizovati koristeći Euklidovu normu. Sloj ponderisanja će imati vektor od n parametara, $W_{CO} \in \mathbb{R}^n$. Umesto vršenja ponderisanja nad individualnim vrednostima, svaki parametar je odgovoran za jedan ulazni atribut, koji reprezentuje jedan red iz X. Slično kao i u originalnom CancelOut sloju, za ograničavanje opsega će biti korišćena sigmoidalna nelinearnost. Izlaz sloja ponderisanja je dat u jednačini (6.1).

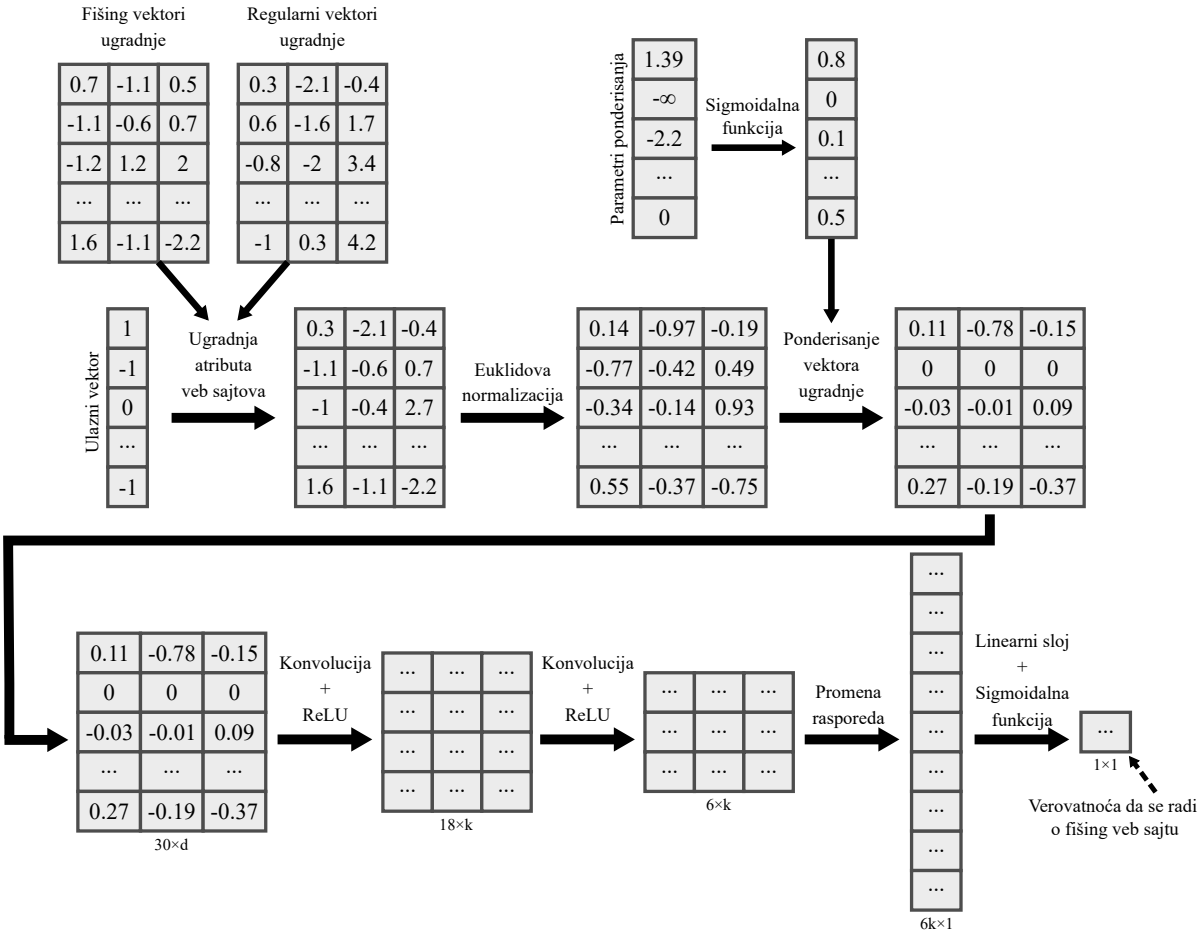
$$\text{diag}(\sigma(W_{CO}))\,\text{row\_norm}(X) \tag{6.1}$$

Sa $\text{diag}(\sigma(W_{CO}))$ je označena dijagonalna matrica konstruisana od elemenata vektora $\sigma(W_{CO})$. Sa $\text{row\_norm}(X)$ je označena matrica dobijena normalizacijom svakog reda matrice X korišćenjem Euklidove norme. Proizvod ove dve matrice, $\text{diag}(\sigma(W_{CO}))$ i $\text{row\_norm}(X)$, je standardni matrični proizvod. Izlaz ovog sloja će biti korišćen kao ulaz ostatka mreže, i svi parametri sloja će biti ažurirani zajedno sa ostalim parametrima mreže u procesu treniranja.

6.3.3 Arhitektura modela

Nakom ugradnje originalnih atributa veb sajta, i ponderisanja vektora ugradnje, biće korišćena konvoluciona neuronska mreža za povezivanje vektora ugradnje i kreiranje kompleksnijih reprezentacija. Arhitektura sadrži dva konvoluciona sloja. Svaki od njih ima po 100 kernela. Veličina kernela biće 13 za modele treniranje na korpusima Korpus-1 i Korpus-2, a 3 za model treniran na korpusu Korpus-3, pošto on ima manju veličinu ulaznog vektora.

U cilju postizanja kompleksnijih nelinearnih transformacija ulaznog vektora atributa, ReLU nelinearnost će biti primenjena nakon svakog konvolucionog sloja. Nelinearnost se primenjuje po elementima. Izlaz nakon primene ReLU nelinearnosti nad izlazom drugog konvolucionog sloja će biti preraspoređen (spljošten, eng. flattened) u vektor veličine 600 za Korpus-1 i Korpus-2, odnosno 500 za Korpus-3. Nad ovim vektorom će na kraju biti primenjen finalni linearni sloj sa jednim izlaznim neuronom. Sigmoidalna aktivacija tog neurona modeluje verovatnoću da se radi o fišing veb sajtu.



Slika 6.1: Predložena arhitektura

Cela arhitektura je prikazana na slici 6.1. Jedina razlika između stvarne arhitekture i prikazane je što je veličina vektora ugradnje d u stvarnoj arhitekturi 100, dok su na

grafiku ovi vektori veličine 3, i što je u stvarnoj arhitekturi broj kernela k u konvolucionim slojevima 100, umesto 3 kako je na grafiku. Kako manja veličina uprošćava ilustraciju, veličina 3 je korišćena na grafiku radi demonstracije.

6.3.4 Alternativne arhitekture

Predloženi model koristi konvolucione slojeve za kreiranje kompleksnijih reprezentacija na osnovu vektora ugradnje atributa veb sajtova. Ipak, to nije jedini način da se to postigne. Jedan način na koji je to još moguće postići je preko ćelija rekurentnih neuronskih mreža. Kako bi istražili alternativne pristupe i uporedili ih sa originalnom arhitekturom koja koristi konvolucione slojeve, odradili smo eksperimente i sa dva alternativna modela.

Prvi alternativni model primenjuje dvosmernu LSTM ćeliju. Ćelija na ulazu koristi vektore ugradnje atributa veb sajtova u vektorski prostor. Izlazi poslednjih skrivenih stanja u oba smera se nadovezuju, i na kraju se primenjuje finalni linearni sloj sa jednim izlaznim neuronom nad tim vektorom. Sigmoidalna aktivacija ovog neurona modelira verovatnoću da se radi o fišing veb sajtu.

Drugi alternativni model je sličan prvom, ali koristi dvosmernu GRU ćeliju umesto LSTM ćelije. GRU je novija arhitektura sa manje mehanizama kapija u odnosu na LSTM arhitekturu. Takođe je karakteriše i uglavnom nešto brže izvršavanje.

Kako i LSTM i GRU koriste mehanizme kapija, teško je opravdati korišćenje ponderisanja nad vektorima ugradnje u ovom slučaju, jer taj sloj funkcioniše na sličan način kao mehanizmi kapija. Empirijska analiza koja sledi će takođe potvrditi ovaj zaključak.

6.4 Evaluacija

Za treniranje modela je korišćena kriterijumska funkcija binarne unakrsne entropije. Vrši se regularizacija svih parametara modela osim parametara pomeraja (eng. bias parameters). Označimo sa W_r vektor koji sadrži sve parametre modela osim parametara pomeraja i parametara korišćenih za ponderisanje. Kriterijumskoj funkciji binarne unakrsne entropije biće pridodata tri regularizaciona člana. Prvi član je L_2 regularizacija parametara modela W_r, što doprinosi boljoj generalizaciji modela. Drugi i treći član se odnose na regularizaciju parametara ponderisanja (W_{CO}). U cilju podsticanja manjih i raznovrsnijih težina, simultano će se maksimizovati njihova varijansa i minimizovati L_1 norma njihovih sigmoidalnih aktivacija, slično kao što je to učinjeno u originalnom CancelOut sloju [18]. Celokupna regularizacija je data u formuli (6.2).

$$\frac{\lambda_1}{2} \|W_r\|_2^2 + \lambda_2 \|\sigma(W_{CO})\|_1 - \lambda_3 \mathrm{Var}(W_{CO}) \tag{6.2}$$

Za izračunavanje varijanse je korišćena Baselova korekcija. U cilju kreiranja balansa između tri regularizaciona člana su korišćeni hiperparametri. Vrednosti hiperparametara su $\lambda_1 = 10^{-4}$, $\lambda_2 = 2 \cdot 10^{-3}$ i $\lambda_3 = 3 \cdot 10^{-3}$ za Korpus-1, $\lambda_1 = 10^{-4}$, $\lambda_2 = 3 \cdot 10^{-2}$ i $\lambda_3 = 4 \cdot 10^{-2}$ za Korpus-2 i $\lambda_1 = 3 \cdot 10^{-3}$, $\lambda_2 = 2 \cdot 10^{-2}$ i $\lambda_3 = 3 \cdot 10^{-2}$ za Korpus-3. U slučajevima kada se model trenira bez ponderisanja vektora ugradnje, koristiće se samo prvi regularizacioni član iz formule (6.2).

Za ažuriranje parametara modela je korišćen Adam [61] optimizacioni algoritam, sa početnim korakom treniranja 0.001. Kad god se desi da se u 5 uzastopnih epoha ne smanji greška na trening skupu, veličina koraka se smanjuje na polovinu trenutne vrednosti. Treniranje modela se vrši u 400 epoha, korišćenjem mini-grupa od po 200 primera. Pre svake epohe se redosled veb sajtova u korpusu promeša na slučajan način, a mini-grupe se formiraju do susednih primera.

Kako su veličine korpusa relativno male, često korišćena podela na skup za treniranje, validaciju i testiranje može da proizvede nestabilne rezultate. Zbog toga je za evaluaciju korišćena 10-struka unakrsna validacija. Najpre su svi veb sajtovi na slučajan način podeljeni u 10 grupa približno istih veličina (grupa može da ima najviše jedan veb sajt više u odnosu na bilo koju drugu grupu). U svakom od 10 eksperimenata, veb sajtovi iz jedne grupe se koriste za testiranje, dok se model trenira veb sajtovima iz preostalih 9 grupa. Ovim pristupom će svaki veb sajt biti korišćen tačno jednom za testiranje. Rezultati testiranja svih veb sajtova će biti korišćeni za računanje evaluacionih metrika. Biće analizirane sledeće evaluacione metrike: tačnost, preciznost, senzitivnost, F_1-skor i stopa lažno pozitivnih.

Tabela 6.2 prikazuje klasifikacione rezultate predloženog modela, sa i bez korišćenja ponderisanja vektora ugradnje. Pored toga što ponderisanje poboljšava interpretabilnost značaja individualnih atributa, tabela pokazuje i da je model koji koristi ponderisanje ostvario nešto bolje rezultate na sva tri korpusa. Na korpusu Korpus-1, varijanta modela koja koristi ponderisanje je ostvarila tačnost od 97.53% i F_1-skor od 97.20%, dok je varijanta koja ne koristi ponderisanje ostvarila tačnost od 97.39% i F_1-skor od 97.03%. Obe varijante su ostvarile dobre rezultate, što može da ukaže na značaj konvolucionih slojeva i ugradnje atributa u vektorski prostor. Rezultati su slični i na korpusu Korpus-2, a nešto slabiji na korpusu Korpus-3, koji je manji, ali izazovniji korpus.

Tabela 6.2: Rezultati predloženog modela

Model	Tačnost	Preciznost	Senzitivnost	F_1-skor	SLP
Korpus-1					
Sa ponderisanjem	0.97531	0.97749	0.96652	0.97197	0.01770
Bez ponderisanja	0.97386	0.97663	0.96407	0.97031	0.01835
Korpus-2					
Sa ponderisanjem	0.97557	0.97588	0.98018	0.97802	0.03016
Bez ponderisanja	0.9715	0.9757	0.97283	0.97426	0.03016
Korpus-3					
Sa ponderisanjem	0.9336	0.94405	0.93732	0.94067	0.07117
Bez ponderisanja	0.9280	0.94348	0.92735	0.93534	0.07117

Klasifikacioni rezultati dva alternativna pristupa su dati u tabeli 6.3. Kao što je ranije pretpostavljeno, oba pristupa su ostvarila bolje rezultate u varijanti bez korišćenja

ponderisanja vektora ugradnje na sva tri korpusa. Performanse oba alternativna pristupa su slične. LSTM je ostvario nešto višu tačnost za Korpus-1, na korpusu Korpus-2 imaju iste tačnosti, dok je GRU postigao nešto višu tačnost za Korpus-3. U poređenju sa originalnim predloženim modelom, koji koristi konvoluciju, oba alternativna pristupa su ostvarila niže rezultate na svakom od tri korpusa. Razlika je najuočljivija na korpusu Korpus-2, na kome je predloženi model sa konvolucijom uspeo da redukuje broj pogrešno klasifikovanih primera za 37.5%.

Tabela 6.3: Rezultati alternativnih pristupa

Model	Tačnost	Preciznost	Senzitivnost	F_1-skor	SLP
Korpus-1					
LSTM sa ponderisanjem	0.95721	0.95329	0.94998	0.95163	0.03703
LSTM bez ponderisanja	0.97060	0.97428	0.95896	0.96656	0.02014
GRU sa ponderisanjem	0.95360	0.95160	0.94324	0.94740	0.03817
GRU bez ponderisanja	0.96997	0.97228	0.95958	0.96589	0.02176
Korpus-2					
LSTM sa ponderisanjem	0.94748	0.96111	0.94347	0.95220	0.04753
LSTM bez ponderisanja	0.96091	0.96820	0.96109	0.96463	0.03931
GRU sa ponderisanjem	0.94870	0.96188	0.94493	0.95333	0.04662
GRU bez ponderisanja	0.96091	0.96613	0.96329	0.96471	0.04205
Korpus-3					
LSTM sa ponderisanjem	0.91760	0.93469	0.91738	0.92595	0.08212
LSTM bez ponderisanja	0.92720	0.93957	0.93020	0.93486	0.07664
GRU sa ponderisanjem	0.91520	0.92330	0.92593	0.92461	0.09854
GRU bez ponderisanja	0.92960	0.93983	0.93447	0.93714	0.07664

U tabeli 6.4 je dato poređenje predloženog modela i ostalih pristupa za detekciju fišing veb sajtova iz literature. Tabela pokazuje da je predloženi model jedan od trenutno najboljih dostupnih metoda. Ugradnja atributa veb sajtova u vektorski prostor i primena konvolucionih slojeva omogućavaju veću kompleksnost reprezentacije našem modelu u poređenju sa tradicionalnim potpuno povezanim arhitekturama neuronskih mreža [79, 84, 104, 105], što je dovelo i do veće tačnosti. Dodavanje sloja ponderisanja vektora ugradnje atributa veb sajtova u arhitekturu je dodatno unapredilo tačnost. Selekcija atributa i ponderisanje je standardna tehnika koja je često primenjivana u literaturi detekcije fišinga [9, 10, 88]. Za razliku od nekih drugih često korišćenih tehnika ponderisanja, naš pristup simultano uči težine ponderisanja i trenira model. Ovaj pristup s kraja na kraj doprinosi jačoj vezi između ponderisanja atributa i klasifikacije, što može da bude razlog veće tačnosti našeg modela.

Tabela 6.4: Poređenje sa drugim metodama

(a) Korpus-1

Model	Tačnost (%)	Senzitivnost (%)
Predloženi pristup sa ponderisanjem	**97.53**	**96.65**
Predloženi pristup bez ponderisanja	**97.39**	**96.41**
Al-Sarem i dr. [8]	97.16	96.83
Ali i Malebary [10]	96.83	95.27
Al-Ahmadi i Lasloum [6]	96.65	96.65
Vrbančič i dr. [105]	96.65	N/A
Vrbančič i dr. [104]	96.5	N/A
Lakshmi i dr. [67]	96.0	N/A
Alqahtani [13]	95.20	N/A
Parra i dr. [85]	94.3	93.67
Thabtah i dr. [102]	93.06	91.12
Motlagh i Bardsiri [84]	93.42	92.27
Mohammad i dr. [79]	92.48	N/A

(b) Korpus-2

Model	Tačnost (%)	Senzitivnost (%)
Predloženi pristup sa ponderisanjem	**97.56**	**98.02**
Predloženi pristup bez ponderisanja	**97.15**	**97.28**
Al-Ahmadi i Lasloum [6]	95.73	95.73
Jalal i Naaz [49]	95.7	96.1
Wang i dr. [107]	95.47	95.37
Al-Milli i Hammo [7]	94.31	N/A

(c) Korpus-3

Model	Tačnost (%)	Senzitivnost (%)
Predloženi pristup sa ponderisanjem	**93.36**	**93.73**
Predloženi pristup bez ponderisanja	**92.80**	**92.74**
Khan i dr. [58]	92.94	89.41
Kulkarni i dr. [65]	91.50	90.97
Ali i Ahmed [9]	91.13	90.79
Almousa i dr. [11]	88.67	N/A
Vrbančič i dr. [105]	86.06	N/A

U poređenju sa ostalim modelima baziranim na neuronskim mrežama, predloženi model je na korpusu Korpus-1 ostvario višu tačnost u odnosu na [6], [67], [85] i [102] redom za 0.88%, 1.53%, 3.23% i 4.47%. Na korpusu Korpus-2, predloženi model je ostvario višu tačnost u odnosu na modele [7] i [107], koji su bazirani na arhitekturama neuronskih mreža sa konvolucionim i LSTM slojevima, za 3.25% i 2.09%. Na korpusu Korpus-3, naš model je ostvario za 2.23% višu tačnost u odnosu na model dubokog učenja predstavljen

u radu [9]. Ovaj model koristi genetski algoritam za selekciju atributa i ponderisanje, što se razlikuje od našeg pristupa s kraja na kraj, u kome simultano vršimo ponderisanje atributa i treniranje klasifikatora.

Tabela 6.5: Prosečna težina dodeljena svakom atributu

Atribut	Ograničena težina
Saobraćaj veb sajta	0.92
Korišćenje HTTPS-a	0.91
Linkovi u <Meta>, <Script> i <Link> oznakama	0.91
Linkovi ka spoljnim domenima	0.9
Broj poddomena	0.9
Dodavanje prefiksa ili sufiksa domenu odvojenog sa -”	0.88
Obrađivač poslatih formi	0.84
Broj linkova koji pokazuju ka stranici	0.83
Google Indeks	0.82
URL-ovi ka sadržaju sa spoljnim domenom	0.76
DNS zapis	0.75
Korišćenje IP adrese umesto domena	0.71
Starost domena	0.7
Korišćenje nestandardnih portova	0.7
Slanje informacija na mejl	0.67
Period registracije domena	0.65
Postojanje ”HTTPS”tokena u delu URL-a za domen	0.65
PageRank	0.65
Preusmeravanje na sajtu	0.63
Redirekcija korišćenjem ”//”	0.53
Korišćenje servisa za skraćivanje URL-a (TinyURL)	0.5
Korišćenje iskačujućih prozora	0.47
Dugačak URL kako bi sakrio sumnjive delove	0.42
Abnormalan URL	0.41
Korišćenje Favikona van adresne linije	0.38
URL sadrži ”@šimbol	0.38
IFrame redirekcija	0.31
Onemogućen desni klik	0.29
Prilagođavanje statusne linije	0.21
Atribut baziran na statističkim izveštajima	0.08

Ukoliko uporedimo senzitivnost našeg modela sa ostalim modelima iz literature, možemo da primetimo da je naš model ostvario bolji rezultat u odnosu na sve druge modele na korpusima Korpus-2 i Korpus-3. Naš model je takođe ostvario istu ili bolju senzitivnost u ondnosu na sve modele na korpusu Korpus-1, osim modela [8], koji je dostigao nešto višu vrednost (za 0.18%). Ipak, naš model je u odnosu na ovaj model

ostvario značajno veću tačnost (za 0.37%).

U cilju poređenja uticaja različitih atributa veb sajta na klasifikaciju, zabeležili smo sigmoidalne aktivacije svih parametara sloja ponderisanja nakon završetka procesa treniranja modela. Kako se evaluacija modela vrši korišćenjem 10-struke unakrsne validacije, za svaki parametar ponderisanja izračunat je prosek sigmoidalnih aktivacija kroz 10 izvršavanja. Te vrednosti su prikazane u tabeli 6.5. Dva atributa sa najvećim težinama su saobraćaj veb sajta i korišćenje HTTPS-a. Najniža težina je dodeljena atributu baziranom na statističkim izveštajima, koji proverava da li IP ili domen pripadaju izveštajima PhishTank-a i StopBadware-a. Upoređujući najviše rangirane atribute iz naše liste sa najviše rangiranim atributima iz tri rangiranja datih u [101], možemo primetiti da se u dva slučaja prvih 20% atributa poklapa, dok se u trećem slučaju razlikuje samo za 1 atribut.

Poglavlje 7

Zaključak

U prvom poglavlju smo se podsetili istorije neuronskih mreža. Zatim smo opisali više različitih gradivnih blokova neuronskih mreža, koje smo kasnije koristili u ostalim poglavljima. Na kratko smo sumirali i trenutne načine primena mašinskog učenja na probleme u sajber bezbednosti.

Drugo poglavlje obrađuje problem detekcije malicioznih zahteva. Predstavili smo efikasan način za prikupljanje malicioznih zahteva korišćenjem zamki. Predložili smo više modela mašinskog učenja za detekciju, uključujući i modele bazirane na neuronskim mrežama. Kreirali smo nekoliko različitih strategija ekstrakcije atributa iz veb zahteva. Neke od njih ekstraktuju vektore atributa fiksne dužine, pogodne za tradicionalne modele, dok druge ekstraktuju atribute u sekvencijalnom obliku, pogodne za određenje modele veštačkih neuronskih mreža. Zatim smo istražili primenu tehnika poduzorkovanja i preuzorkovanja u cilju prevazilaženja problema koje neravnomeran broj regularnih i malicioznih primera može da prouzrokuje. Nakon toga smo odradili analizu detekcije napada nultog dana, tako što smo skupove podataka korišćene za treniranje i evaluaciju kreirali od zahteva koji su pristigli u vremenskim intervalima koji se ne prekpalaju.

U trećem poglavlju smo implementirali tri strategije inkrementalnog učenja na osnovu novih zahteva sa zamki i regularnog saobraćaja, sa ciljem umanjenja problema katastrofalnog zaboravljanja. Jedna od njih se bazira na zamrzavanju poslednjeg sloja binarnog klasifikatora zasnovanog na neuronskoj mreži nakon inicijalnog treniranja, kao i omogućavanju ažuriranja ostalih slojeva mreže i adaptaciji rečnika. Druga strategija koristi mali bafer kako bi prevazišla problem katastrofalnog zaboravljanja. Treća strategija kombinuje ova dva pristupa. Sve tri strategije inkrementalno grade rečnike i vremenom akumuliraju znanje.

U četvrtom poglavlju smo predložili metod odabira atributa mrežnog saobraćaja baziran na populaciji u cilju detekcije upada. Pokazali smo njegovu efektivnost upoređujući tačnost klasifikacije više klasifikatora mašinskog učenja kada koriste sve ulazne atribute i kada koriste podskupove atributa odabranih od strane našeg metoda. Naš metod je poboljšao tačnost svih testiranih klasifikatora, uz značajno umanjenje broja ulaznih atributa.

Peto poglavlje opisuje problem detekcije fišing mejlova. Kao prvi korak detekcije smo radili ekstrakciju tekstualnog sadržaja iz mejlova. Umesto manuelnog inženjerisanja ulaznih atributa, izvlačili smo karaktere i reči iz teksta. Ovaj pristup je univerzalniji, i u

budućnosti neće biti teško primeniti ga na nove tipove mejlova kada se pojave. Zatim smo vršili ugradnju karaktera i reči u vektorski prostor, a vektore ugradnje smo koristili kao ulaz klasifikatora baziranog na neuronskoj mreži koji smo dizajnirali. Vektore ugradnje karaktera i reči u vektorski prostor smo učili zajedno sa svim ostalim parametrima neuronske mreže, koristeći optimizacioni algoritam baziran na gradijentu. Odradili smo detaljnu evaluaciju predloženog modela, u kojoj je naš model pokazao slične ili bolje rezultate od trenutno najboljih modela iz literature.

U šestom poglavlju smo obradili problem detekcije fišing veb sajtova koristeći diskretne opisne atribute. Značajna prednost korišćenja ovakvih atributa je bolja interpretabilnost modela. Najpre smo dizajnirali sloj ugradnje kojim smo modelirali ugradnju ovakvih atributa u vektorski prostor. Inspirisani CancelOut slojem, razvili smo sloj za ponderisanje vektora ugradnje, kojim smo modelirali razlike u uticaju različitih atributa na detekciju fišinga. Predložili smo klasifikator baziran na konvolucionoj neuronskoj mreži, a zatim i odradili detaljno ispitivanje kojim smo potvrdili efikasnost ovog modela za detekciju fišing veb sajtova.

Literatura

[1] Earl Andrea Abad, John Rafael Ferrer, and Prospero Naval. Phishing website classification using features of web addresses and web pages. 02 2020.

[2] Neda Abdelhamid, Aladdin Ayesh, and Fadi Thabtah. Phishing detection based associative classification data mining. *Expert Systems with Applications*, 41(13):5948–5959, 2014.

[3] U Rajendra Acharya, Shu Lih Oh, Yuki Hagiwara, Jen Hong Tan, Muhammad Adam, Arkadiusz Gertych, and Ru San Tan. A deep convolutional neural network model to classify heartbeats. *Computers in biology and medicine*, 89:389–396, 2017.

[4] Abien Fred Agarap. A neural network architecture combining gated recurrent unit (GRU) and support vector machine (SVM) for intrusion detection in network traffic data. In *Proceedings of the 10th International Conference on Machine Learning and Computing, ICMLC 2018, Macau, China, February 26-28, 2018*, pages 26–30. ACM, 2018.

[5] Andronicus A Akinyelu and Aderemi O Adewumi. Classification of phishing email using random forest machine learning technique. *Journal of Applied Mathematics*, 2014(1):425731, 2014.

[6] Saad Al-Ahmadi and Tariq Lasloum. Pdmlp: Phishing detection using multilayer perceptron. *International Journal of Network Security & Its Applications*, 2020.

[7] Nabeel Al-Milli and Bassam H Hammo. A convolutional neural network model to detect illegitimate urls. In *2020 11th International Conference on Information and Communication Systems (ICICS)*, pages 220–225. IEEE, 2020.

[8] Mohammed Al-Sarem, Faisal Saeed, Zeyad Ghaleb Al-Mekhlafi, Badiea Abdulkarem Mohammed, Tawfik Al-Hadhrami, Mohammad T Alshammari, Abdulrahman Alreshidi, and Talal Sarheed Alshammari. An optimized stacking ensemble model for phishing websites detection. *Electronics*, 10(11):1285, 2021.

[9] Waleed Ali and Adel A Ahmed. Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting. *IET Information Security*, 13(6):659–669, 2019.

[10] Waleed Ali and Sharaf Malebary. Particle swarm optimization-based feature weighting for improving intelligent phishing website detection. *IEEE Access*, 8:116766–116780, 2020.

[11] May Almousa, Tianyang Zhang, Abdolhossein Sarrafzadeh, and Mohd Anwar. Phishing website detection: How effective are deep learning-based models and hyperparameter optimization? *Security and Privacy*, 5(6):e256, 2022.

[12] Mohammad Almseidin, Maen Alzubi, Szilveszter Kovacs, and Mouhammd Alkasassbeh. Evaluation of machine learning algorithms for intrusion detection system. In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000277–000282. IEEE, 2017.

[13] Mohammed Alqahtani. Phishing websites classification using association classification (pwcac). In *2019 International conference on computer and information sciences (ICCIS)*, pages 1–6. IEEE, 2019.

[14] Amir Andalib and Vahid Tabataba Vakili. An autonomous intrusion detection system using an ensemble of advanced learners. In *2020 28th iranian conference on electrical engineering (ICEE)*, pages 1–5. IEEE, 2020.

[15] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.

[16] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19, 2006.

[17] Yoshua Bengio and Yann LeCun. Scaling learning algorithms toward ai. 2007.

[18] Vadim Borisov, Johannes Haug, and Gjergji Kasneci. Cancelout: A layer for feature selection in deep neural networks. In *Artificial Neural Networks and Machine Learning–ICANN 2019: Deep Learning: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part II 28*, pages 72–83. Springer, 2019.

[19] Stephen Brown, Rebecca Lam, Shishir Prasad, Sivasubramanian Ramasubramanian, and Josh Slauson. Honeypots in the cloud. *University of Wisconsin-Madison*, 11, 2012.

[20] Kalle Burbeck and Simin Nadjm-Tehrani. Adaptive real-time anomaly detection with incremental clustering. *information security technical report*, 12(1):56–67, 2007.

[21] Orestis Christou, Nikolaos Pitropakis, Pavlos Papadopoulos, Sean McKeown, and William J. Buchanan. Phishing url detection through top-level domain analysis: A descriptive approach. In *International Conference on Information Systems Security and Privacy*, 2020.

[22] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[23] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, 2006.

[24] Mahendra Data and Masayoshi Aritsugi. T-dfnn: an incremental learning algorithm for intrusion detection systems. *IEEE Access*, 9:154156–154171, 2021.

[25] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

[26] Yong Fang, Cheng Zhang, Cheng Huang, Liang Liu, and Yue Yang. Phishing email detection using improved rcnn model with multilevel vectors and attention mechanism. *IEEE Access*, 7:56329–56340, 2019.

[27] Nabila Farnaaz and MA Jabbar. Random forest modeling for network intrusion detection system. *Procedia Computer Science*, 89:213–217, 2016.

[28] Jian Feng, Lianyang Zou, Ou Ye, and Jingzhou Han. Web2vec: Phishing webpage detection method based on multidimensional features driven by deep learning. *IEEE Access*, 8:221214–221224, 2020.

[29] Ian Fette, Norman Sadeh, and Anthony Tomasic. Learning to detect phishing emails. In *Proceedings of the 16th international conference on World Wide Web*, pages 649–656, 2007.

[30] Iztok Fister jr, Dušan Fister, and Xin-She Yang. A hybrid bat algorithm. *Elektrotehniški Vestnik*, 80, 03 2013.

[31] H Barathi Ganesh, R Vinayakumar, M Anand Kumar, and K Soman. Distributed representation using target classes: Bag of tricks for security and privacy analytics. In *Proc. 4th ACM Int. Workshop Secur. Privacy Anal.(IWSPA)*, pages 1–6, 2018.

[32] Tushaar Gangavarapu and CD Jaidhar. A novel bio-inspired hybrid metaheuristic for unsolicited bulk email detection. In *International Conference on Computational Science*, pages 240–254. Springer, 2020.

[33] Tushaar Gangavarapu, CD Jaidhar, and Bhabesh Chanduka. Applicability of machine learning in spam and phishing email filtering: review and approaches. *Artificial Intelligence Review*, 53(7):5019–5081, 2020.

[34] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[35] Abdallah Ghourabi, Tarek Abbes, and Adel Bouhoula. Characterization of attacks collected from the deployment of web service honeypot. *Security and Communication Networks*, 7(2):338–351, 2014.

[36] Eder S Gualberto, Rafael T De Sousa, P De B Thiago, João Paulo CL Da Costa, and Cláudio G Duque. From feature engineering and topics models to enhanced prediction rates in phishing detection. *Ieee Access*, 8:76368–76385, 2020.

[37] Wa'el Hadi, Faisal Aburub, and Samer Alhawari. A new fast associative classification algorithm for detecting phishing websites. *Applied Soft Computing*, 48:729–734, 2016.

[38] Lukáš Halgaš, Ioannis Agrafiotis, and Jason RC Nurse. Catching the phish: Detecting phishing attacks using recurrent neural networks (rnns). In *Information Security Applications: 20th International Conference, WISA 2019, Jeju Island, South Korea, August 21–24, 2019, Revised Selected Papers 20*, pages 219–233. Springer, 2020.

[39] Xiao Han, Nizar Kheir, and Davide Balzarotti. Deception techniques in computer security: A research perspective. *ACM Comput. Surv.*, 51(4):80:1–80:36, 2018.

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[41] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.

[42] GE Hinton. A fast learning algorithm for deep belief nets. *Neural Computation*, 2006.

[43] M Hiransha, Nidhin A Unnithan, R Vinayakumar, K Soman, and ADR Verma. Deep learning based phishing e-mail detection. In *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal.(IWSPA)*, pages 1–5. Tempe, AZ, USA, 2018.

[44] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[45] Samir Ifzarne, Hiba Tabbaa, Imad Hafidi, and Nidal Lamghari. Anomaly detection using machine learning techniques in wireless sensor networks. *Journal of Physics: Conference Series*, 1743, 2021.

[46] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[47] Rafiqul Islam and Jemal Abawajy. A multi-tier phishing detection and filtering approach. *Journal of Network and Computer Applications*, 36(1):324–335, 2013.

[48] Michiaki Ito and Hitoshi Iyatomi. Web application firewall using character-level convolutional neural network. In *2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA)*, pages 103–106. IEEE, 2018.

[49] Kahksha Jalal and Sameena Naaz. Detection of phishing website using machine learning approach. In *International Conference on Sustainable Computing in Science, Technology & Management (SUSCOM)*, 04 2019.

[50] Tharmini Janarthanan and Shahrzad Zargari. Feature selection in unsw-nb15 and kddcup'99 datasets. In *2017 IEEE 26th international symposium on industrial electronics (ISIE)*, pages 1881–1886. IEEE, 2017.

[51] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetful learning for domain expansion in deep neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018.

[52] V. Kanimozhi and T. Prem Jacob. Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. *ICT Express*, 5(3):211–214, 2019.

[53] V. Kanimozhi and T. Prem Jacob. Artificial intelligence outflanks all other machine learning classifiers in network intrusion detection system on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. *ICT Express*, 7(3):366–370, 2021.

[54] Sydney Mambwe Kasongo and Yanxia Sun. A deep long short-term memory based classifier for wireless intrusion detection system. *ICT Express*, 6(2):98–103, 2020.

[55] Sydney Mambwe Kasongo and Yanxia Sun. A deep gated recurrent unit based model for wireless intrusion detection system. *ICT Express*, 7(1):81–87, 2021.

[56] Kdd cup 1999 intrusion detection dataset. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

[57] N Maajid Khan, Nalina Madhav C, Anjali Negi, and I Sumaiya Thaseen. Analysis on improving the performance of machine learning models using feature selection technique. In *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 2*, pages 69–77. Springer, 2020.

[58] Sohail Ahmed Khan, Wasiq Khan, and Abir Hussain. Phishing attacks and websites classification using machine learning and multiple datasets (a comparative analysis). In *Intelligent Computing Methodologies: 16th International Conference, ICIC 2020, Bari, Italy, October 2–5, 2020, Proceedings, Part III 16*, pages 301–313. Springer, 2020.

[59] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1):1–22, 2019.

[60] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.

[61] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[62] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Stefanos Gritzalis. Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset. *IEEE Commun. Surv. Tutorials*, 18(1):184–208, 2016.

[63] Janardhan Reddy Kondra, Santosh Kumar Bharti, Sambit Kumar Mishra, and Korra Sathya Babu. Honeypot-based intrusion detection system: A performance analysis. In *2016 3rd international conference on computing for sustainable global development (INDIACom)*, pages 2347–2351. IEEE, 2016.

[64] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[65] Arun D Kulkarni, Leonard L Brown III, et al. Phishing websites detection using machine learning. 2019.

[66] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[67] L Lakshmi, M Purushotham Reddy, Chukka Santhaiah, and U Janardhan Reddy. Smart phishing detection in web pages using supervised deep learning classification and optimization technique adam. *Wireless Personal Communications*, 118(4):3549–3564, 2021.

[68] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[69] Jingxi Liang, Wen Zhao, and Wei Ye. Anomaly-based web attack detection: a deep learning approach. In *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, pages 80–85, 2017.

[70] Richard Lippmann, Robert K. Cunningham, David J. Fried, Isaac Graf, Kris R. Kendall, Seth E. Webster, and Marc A. Zissman. Results of the DARPA 1998 offline intrusion detection evaluation. In *Recent Advances in Intrusion Detection, Second International Workshop, RAID 1999, West Lafayette, Indiana, USA, September 7-9, 1999*, 1999.

[71] Samuel Marchal, Jérôme François, Radu State, and Thomas Engel. Phishstorm: Detecting phishing with streaming analytics. *IEEE Transactions on Network and Service Management*, 11:458–471, 2014.

[72] J Mason. The apache spamassassin public corpus. *The Apache SpamAssassin Project*, 2005.

[73] Iik Muhamad Malik Matin and Budi Rahardjo. Malware detection using honeypot and machine learning. In *2019 7th international conference on cyber and IT service management (CITSM)*, volume 7, pages 1–4. IEEE, 2019.

[74] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[75] Dželila Mehanović and Jasmin Kevrić. phishing website detection using machine learning classifiers optimized by feature selection. *Traitement du Signal*, 37:563–569, 2020.

[76] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*, 2013.

[77] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[78] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. An assessment of features related to phishing websites using an automated technique. In *2012 international conference for internet technology and secured transactions*, pages 492–497. IEEE, 2012.

[79] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. Predicting phishing websites based on self-structuring neural network. *Neural Computing and Applications*, 25:443–458, 2014.

[80] Bahram Mohammadi and Mohammad Sabokrou. End-to-end adversarial learning for intrusion detection in computer networks. In Karl Andersson, Hwee-Pink Tan, and Sharief Oteafy, editors, *44th IEEE Conference on Local Computer Networks, LCN 2019, Osnabrueck, Germany, October 14-17, 2019*, pages 270–273. IEEE, 2019.

[81] Naghmeh Moradpoor, Benjamin Clavie, and Bill Buchanan. Employing machine learning techniques for detection and classification of phishing emails. In *2017 Computing Conference*, pages 149–156. IEEE, 2017.

[82] Nour Moustafa and Jill Slay. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *2015 Military Communications and Information Systems Conference, MilCIS 2015, Canberra, Australia, November 10-12, 2015*, pages 1–6. IEEE, 2015.

[83] J Nazario. Phishing corpus, 2007.

[84] F Parandeh Motlagh and A Khatibi Bardsiri. Detecting fake websites using swarm intelligence mechanism in human learning. *International Journal of Engineering*, 31(10):1642–1650, 2018.

[85] Gonzalo De La Torre Parra, Paul Rad, Kim-Kwang Raymond Choo, and Nicole Beebe. Detecting internet of things attacks using distributed deep learning. *Journal of Network and Computer Applications*, 163:102662, 2020.

[86] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[87] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[88] Ravi Kiran Varma Penmatsa and Padmaprabha Kakarlapudi. Web phishing detection: feature selection using rough sets and ant colony optimisation. *International Journal of Intelligent Systems Design and Computing*, 2(2):102–113, 2018.

[89] Vinayakumar Ra, Barathi Ganesh HBa, Anand Kumar Ma, Soman KPa, Prabaharan Poornachandran, and A Verma. Deepanti-phishnet: Applying deep neural networks for phishing email detection. In *Proc. 1st AntiPhishing Shared Pilot 4th ACM Int. Workshop Secur. Privacy Anal.(IWSPA)*, pages 1–11. Tempe, AZ, USA, 2018.

[90] Majed Rajab. Visualisation model based on phishing features. *Journal of Information & Knowledge Management*, 18(01):1950010, 2019.

[91] Romil Rawat and Shailendra Kumar Shrivastav. Sql injection attack detection using svm. *International Journal of Computer Applications*, 42(13):1–4, 2012.

[92] Wei Rong, Bowen Zhang, and Xixiang Lv. Malicious web request detection using character-level cnn. In *Machine Learning for Cyber Security: Second International Conference, ML4CS 2019, Xi'an, China, September 19-21, 2019, Proceedings 2*, pages 6–16. Springer, 2019.

[93] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para.* Cornell Aeronautical Laboratory, 1957.

[94] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation, parallel distributed processing, explorations in the microstructure of cognition, ed. de rumelhart and j. mcclelland. vol. 1. 1986. *Biometrika*, 71(599-607):6, 1986.

[95] Chaimae Saadi and Habiba Chaoui. Cloud computing security using ids-am-clust, honeyd, honeywall and honeycomb. *Procedia Computer Science*, 85:433–442, 2016.

[96] Ozgur Koray Sahingoz, Ebubekir Buber, Önder Demir, and Banu Diri. Machine learning based phishing detection from urls. *Expert Syst. Appl.*, 117:345–357, 2019.

[97] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Paolo Mori, Steven Furnell, and Olivier Camp, editors, *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Madeira - Portugal, January 22-24, 2018*, pages 108–116. SciTePress, 2018.

[98] Kamran Shaukat, Suhuai Luo, Vijay Varadharajan, Ibrahim A Hameed, and Min Xu. A survey on machine learning techniques for cyber security in the last decade. *IEEE access*, 8:222310–222354, 2020.

[99] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A. Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Comput. Secur.*, 31(3):357–374, 2012.

[100] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009*, pages 1–6. IEEE, 2009.

[101] Fadi Thabtah and Neda Abdelhamid. Deriving correlated sets of website features for phishing detection: a computational intelligence approach. *Journal of Information & Knowledge Management*, 15(04):1650042, 2016.

[102] Fadi Thabtah, Rami M Mohammad, and Lee McCluskey. A dynamic self-structuring neural network model to combat phishing. In *2016 international joint conference on neural networks (ijcnn)*, pages 4221–4226. IEEE, 2016.

[103] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

[104] Grega Vrbančič, Iztok Fister Jr, and Vili Podgorelec. Swarm intelligence approaches for parameter setting of deep learning neural network: case study on phishing websites classification. In *Proceedings of the 8th international conference on web intelligence, mining and semantics*, pages 1–8, 2018.

[105] Grega Vrbančič, Iztok Fister Jr, and Vili Podgorelec. Parameter setting for deep neural networks using swarm intelligence on phishing websites classification. *International Journal on Artificial Intelligence Tools*, 28(06):1960008, 2019.

[106] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339, 1989.

[107] Shan Wang, Sulaiman Khan, Chuyi Xu, Shah Nazir, and Abdul Hafeez. Deep learning-based efficient model development for phishing detection using random forest and blstm classifiers. *Complexity*, 2020(1):8694796, 2020.

[108] Wei Wang, Yiqiang Sheng, Jinlin Wang, Xuewen Zeng, Xiaozhou Ye, Yongzhong Huang, and Ming Zhu. Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE access*, 6:1792–1806, 2017.

[109] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[110] B Widrow, GF Groner, MJC Hu, FW Smith, DF Specht, and LR Talbert. Practical applications for adaptive data-processing systems. *WESCON Techn. Papers*, 11:4, 1963.

[111] Peilun Wu and Hui Guo. Lunet: a deep neural network for network intrusion detection. In *2019 IEEE symposium series on computational intelligence (SSCI)*, pages 617–624. IEEE, 2019.

[112] Peilun Wu, Hui Guo, and Nour Moustafa. Pelican: A deep residual network for network intrusion detection. In *50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops, DSN Workshops 2020, Valencia, Spain, June 29 - July 2, 2020*, pages 55–62. IEEE, 2020.

[113] X Yang. S.: A new metaheuristic bat-insspired algorithm. nature inspired cooperative strategies for optimization. *Studies in Computational Intelligence (Springer). pp-65-74*, 2010.

[114] Adwan Yasin and Abdelmunem Abuhasan. An intelligent classification model for phishing email detection. *International Journal of Network Security & Its Applications*, 8(4):55–72, 2016.

[115] Ming Zhang, Boyi Xu, Shuai Bai, Shuaibing Lu, and Zhechao Lin. A deep learning method to detect web attacks using a specially designed cnn. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14–18, 2017, Proceedings, Part V 24*, pages 828–836. Springer, 2017.

[116] Yuening Zhang, Yiming Zhang, Nan Zhang, and Mingzhong Xiao. A network intrusion detection method based on deep learning with higher accuracy. *Procedia Computer Science*, 174:50–54, 2020.

[117] Yuyang Zhou, Guang Cheng, Shanqing Jiang, and Mian Dai. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Computer networks*, 174:107247, 2020.

Biografija

Nikola Stevanović je rođen 17.08.1992. godine u Nišu. Završio je osnovnu školu Ivo Andrić kao nosilac Vukove diplome i đak generacije. Nakon toga je završio Specijalizovano odeljenje za učenike sa posebnim sposobnostima za matematiku u Gimnaziji Svetozar Marković u Nišu, takođe kao nosilac Vukove diplome.

U 2011. godini je paralelno upisao osnovne studije iz matematike i iz informatike na Prirodno-matematičkom fakultetu u Nišu. Obe je završio 2014. godine, studije iz matematike sa prosečnom ocenom 9.92, a studije iz informatike sa prosečnom ocenom 10.00. Iste godine je upisao master akademske studije na Departmanu za računarske nauke, Prirodno-matematičkog fakulteta u Nišu, na studijskom programu za razvoj softvera. Master akademske studije je završio 2016. godine sa prosečnom ocenom 10.00. Te godine je upisao i doktorske akademske studije na Departmanu za računarske nauke istog fakulteta, a zatim i položio sve ispite sa prosečnom ocenom 10.00.

Tokom osnovnog i srednjeg obrazovanja se veoma uspešno takmičio iz oblasti matematike, informatike i fizike. Iz tih oblasti je osvojio veliki broj nagrada na takmičenjima opštinskog, okružnog i republičkog nivoa. Među njima se izdvajaju treća nagrada na republičkom takmičenju iz matematike, treća nagrada na republičkom takmičenju iz fizike i dve druge nagrade na republičkim takmičenjima iz informatike. Učestvovao je na dve srpske informatičke oblimpijade, kao i na dva finala Bubble Cup takmičenja. Tokom studija je dve godine bio član ekipe fakulteta na takmičenju ACM Programming Contest.

Tokom doktorskih studija vršio je istraživanje u oblasti primene veštačkih neuronskih mreža na detekciju veb napada. Prvi je autor dva rada u međunarodnim časopisima, od kojih je jedan objavljen u vrhunskom međunarodnom časopisu, a u drugom je jedini autor. Prvi i jedini autor je jednog rada koji je prihvaćen za objavljivanje u nacionalnom časopisu međunarodnog značaja. Prezentovao je rad i na Prvoj srpskoj internacionalnoj konferenciji o primenjenoj veštačkoj inteligenciji održanoj u Kragujevcu.

Nagrađen je od strane Univerziteta u Nišu kao najbolji student koji je završio osnovne akademske studije Univerziteta u Nišu u školskoj 2013/2014. godini.

Spisak objavljenih radova:

- **Nikola Stevanović**, Branimir Todorović, and Vladan Todorović. Web attack detection based on traps. Applied Intelligence, 52(11):12397–12421, 2022. **(M21)**

- **Nikola Stevanović**. Character and word embeddings for phishing email detection. Computing and Informatics, 41(5):1337–1357, 2022. **(M23)**

- **Nikola Stevanović**. Population-based feature selection for intrusion detection. In First Serbian International Conference on Applied Artificial Intelligence (SICAAI). Kragujevac, Serbia, 2022. **(M34)**
- **Nikola Stevanović**. Embedding and weighting of website features for phishing detection. Facta Universitatis, Series: Mathematics and Informatics. (prihvaćen 22.11.2024. godine) **(M24)**

ИЗЈАВА О АУТОРСТВУ

Изјављујем да је докторска дисертација, под насловом

ВЕШТАЧКЕ НЕУРОНСКЕ МРЕЖЕ ЗА ДЕТЕКЦИЈУ ВЕБ НАПАДА

која је одбрањена на Природно-математичком факултету Универзитета у Нишу:

- резултат сопственог истраживачког рада;
- да ову дисертацију, ни у целини, нити у деловима, нисам пријављивао/ла на другим факултетима, нити универзитетима;
- да нисам повредио/ла ауторска права, нити злоупотребио/ла интелектуалну својину других лица.

Дозвољавам да се објаве моји лични подаци, који су у вези са ауторством и добијањем академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада, и то у каталогу Библиотеке, Дигиталном репозиторијуму Универзитета у Нишу, као и у публикацијама Универзитета у Нишу.

У Нишу, 10.03.2025.

Потпис аутора дисертације:

Никола Стевановић

Никола М. Стевановић

ИЗЈАВА О ИСТОВЕТНОСТИ ШТАМПАНОГ И ЕЛЕКТРОНСКОГ ОБЛИКА ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Наслов дисертације:

ВЕШТАЧКЕ НЕУРОНСКЕ МРЕЖЕ ЗА ДЕТЕКЦИЈУ ВЕБ НАПАДА

Изјављујем да је електронски облик моје докторске дисертације, коју сам предао/ла за уношење у **Дигитални репозиторијум Универзитета у Нишу,** истоветан штампаном облику.

У Нишу, 10.03.2025.

Потпис аутора дисертације:

Никола Стевановић

Никола М. Стевановић

ИЗЈАВА О КОРИШЋЕЊУ

Овлашћујем Универзитетску библиотеку „Никола Тесла“ да у Дигитални репозиторијум Универзитета у Нишу унесе моју докторску дисертацију, под насловом:

ВЕШТАЧКЕ НЕУРОНСКЕ МРЕЖЕ ЗА ДЕТЕКЦИЈУ ВЕБ НАПАДА

Дисертацију са свим прилозима предао/ла сам у електронском облику, погодном за трајно архивирање.

Моју докторску дисертацију, унету у Дигитални репозиторијум Универзитета у Нишу, могу користити сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons), за коју сам се одлучио/ла.

1. Ауторство **(CC BY)**
2. Ауторство – некомерцијално **(CC BY-NC)**
3. **Ауторство – некомерцијално – без прераде (CC BY-NC-ND)**
4. Ауторство – некомерцијално – делити под истим условима **(CC BY-NC-SA)**
5. Ауторство – без прераде **(CC BY-ND)**
6. Ауторство – делити под истим условима **(CC BY-SA)**

У Нишу, 10.03.2025.

Потпис аутора дисертације:

Никола Стевановић

Никола М. Стевановић