Klasifikacija delova politike privatnosti

Milan Jokanović
jokanovic.r224.2021@uns.ac.rs

Aleksandar Nikolić
aleksandar.nikolic021@uns.ac.rs

Apstrakt–Politike privatnosti se često ignorišu od strane korisnika zbog njihovog dugačkog formata i napornog sadržaja. U ovom radu predlaže se model koji za cilj ima da klasifikuje paragrafe politike privatnosti na odgorajuću klasu kako bi se olakšao način na koji korisnici mogu da dođu do informacija koje su njima od značaja. Skup podataka je sačinjen od skupa parova paragraf i klasa anotacija koji su preuzeti iz gotovog skupa od 115 ručno klasifikovanih politika privatnosti od strane eksperata. Za kreiranje modela korišćeno je više različitih pristupa: tradicionalni pristupi SVM i KNN, neuronska mreža LSTM i na kraju transformer BERT koji je dao najbolje rezultate.

Ključne reči — politika privatnosti, klasifikacija, SVM, KNN, LSTM, BERT

1. Uvod

Politike privatnosti su pravno obavezujući dokument između korisnika i veb stranice kojoj korisnik pristupa. Ovo u teoriji znači da korisnici biraju da li žele da daju svoje podatke na korišćenja kako bi pristupali nekoj veb stranici ili ne. Međutim situacija u praksi je daleko različita, većina korisnika ne pročita politiku privatnosti pre nego što je prihvate i samo daju svoje podatke bez obzira na to da li su to oni stvarno hteli ili ne.

Jedan od glavnih razloga zašto korisnici ne čitaju politike privatnosti leži u njihovom kompleksnom i obimnom sadržaju. Korisnici uglavnom nisu zainteresovani da čitaju ceo sadržaj ovakvih dokumenata, već bi hteli da pročitaju samo one delove koji bi njima bili od značaja.

Tema ovog rada je da pomogne korisnicima da dođu do relevantnih informacija iz politika privatnosti, a da pritom ne moraju da prođu kroz njihov celukupan sadržaj. Ovo je postignutno kreiranjem modela koji za cilj ima klasifikaciju svih paragrafa politika privatnosti na neku od predefinisanih klasa. Na ovaj način korisniku je bar u nekoj meri olakšan pristup željenim informacijama, tako što će preskočiti paragrafe koje mu nisu od interesa ili će direktno pronaći ono što želi.

Podaci koji služe za kreiranje ovog modela su preuzeti iz gotovog skupa ručno klasifikovanih dokumenata od strane domenskih eksperata. Klasifikovani dokumenti su isključivo napisani na engleskom jeziku i samim tim i model iz ovog rada takođe može da klasifikuje isključivo dokumente napisane na engleskom jeziku. Model kao svoj sadržaj prima parove anotacije klase i predprocesuranih paragrafa iz dokumenta. Kreirani skup se zatim šalje modelu na treniranje. Po završetku treniranja model je u mogućnosti da određeni paragraf

politike privatnosti klasifikuje na neku od ponuđenih klasa.

U nastavku rada opisane su korišćene metodologije, skup podataka i izvršeni eksperimenti. U narednom poglavlju biće reči o radovima koji su se bavili sličnim problemima. U poglavalju tri detaljno su objašnjene metodologije korišćene za rešavanje problema. Zatim poglavlje četiri objašnjava samu implementaciju modela, a poglavlje pet detaljno objašnjava skup podataka koji se koristi. Poglavlje šest se bavi eksperimentom i problemima koji su pronađeni u okviru eksperimenta. Poslednje sedmo poglavlje daje zaključak na ceo rad.

2. Srodni radovi

U radu "*The creation and analysis of a website privacy policy corpus*" [1] opisan je postupak za kreiranje skupa podataka od 115 politika privatnosti sa klasifikovanim paragrafima. Jedan od problema koji je nastao prilikom rada na kreiranju skupa podataka jeste činjenica da postoji veliki broj dokumenata sa sličnim sadržajem. Tako da bi se kreirao adekvatan skup anotacija potrebno je pronaći međusobno različite dokumente. Kako bi prevazišli ovaj problem korišćena su dva pristupa za biranje dokumenta. Prvi pristup je zasnovan na biranju relevantnih dokumenata, dok se drugi pristup zasniva na biranju dokumenata iz različitih sektora. Zatim su odabrani dokumenti poslati domenskim ekspertima koji su vršili klasifikaciju njihovog sadržaja na predefinisan skup klasa. U ovom radu, “Klasifikacija delova politike privatnosti”, se takođe koristi isti skup podataka i skup klasa kako bi se postigla automatska klasifikacija paragrafa politika privatnosti.

Kanamori Sachiko u svom radu "*Construction of a Support Tool for User Reading of Privacy Policies and Assessment of its User Impact*" [2] predstavlja rešenje za kreiranje alata koji bi pomogao korisnicima prilikom čitanja politika privatnosti napisanih na japanskom jeziku. Ovaj alat funkcioniše tako što automatski analizira sadržaj politike privatnosti i na osnovu toga daje korisniku informacije koje podatke stranica prikuplja i koje *third party* organizacije takođe imaju pristup tim podacima.

Rad "*Privacy practices of Internet users: Self-reports versus observed behaviour*" [3] se bavi analizom zabrinutosti korisnika sa politikom privatnosti. Rad se u početnoj fazi bavio analizom odgovora korisnika na to koliko su zabrinuti sa sadržajem politika privatnosti. Ovi odgovori su sugerisali da je zabrinutost korisnika visoka, međutim daljim istraživanjem pokazano je da većina korisnika nije zainteresovana za sadržaj politike privatnosti i čak da većina korisnika ima loše razumevanje o tome šta su politike privatnosti.

3. Metodologije

U ovoj sekciji je objašnjeno kojim metodologijama i na koji način će se vršiti obrada podataka od pribavljanja do transformacije u vektorsku predstavu koja je pogodna za prosleđivanje modelu mašinskog učenja. Podaci će najpre proći kroz proces filtriranja i biće odbačeni svi ulazi koji nisu na engleskom jeziku. Potom će se vršiti transformacija podataka, gde će se neke reči izbaciti i obraditi interpunkcija. Naposletku će se izvršiti vektorizacija podataka u oblik pogodan za algoritme mašinskog učenja.

3.1. Pretprocesiranje

Pre nego što podaci postanu pogodni za prosleđivanje modelu mašinskog učenja, a i kako bi povećali učinkovitost modela, neophodno je najpre obraditi skup podataka na odgovarajući način. U ovom radu ta obrada podrazumeva sledeće:

a) Enkodovanje klasa

Kako računar ne razume reči na način na koji ih čovek razume, a kako su klase u skupu podataka date u vidu reči, neophodno je transformisati imena klasa u numerički oblik. U ovom slučaju je transformacija urađena primenom *Label Encoding* metode. Ova metoda funkcioniše tako što svakoj klasi dodeli jedinstven broj koji će se na nju odnositi. Mapiranje je prikazano u tabeli 1.

Tabela 1. Klase i njihove oznake (kodovi)
Skoči na sekciju "evaluacija"

Naziv klase	Oznaka
Other	0
Policy Change	1
First Party Collection/Use	2
Data Retention	3
International and Specific Audiences	4
Third Party Sharing/Collection	5
User Choice/Control	6
User Access, Edit and Deletion	7
Data Security	8
Do Not Track	9

b) Prebacivanje korpusa u mala slova

Cilj ovoga je konzistentnost samih reči kasnije prilikom vektorizacije. Odnosno da bi se izbeglo da se ista reč napisana različitom veličinom slova posmatra kao više različitih reči.

c) Izbacivanje stop reči iz korpusa

Pod stop rečima se podrazumevaju reči koje se najčešće pojavljuju u jeziku, a ne predstavljaju nikakvu informaciju od značaja. Primer ovakvih reči u engleskom jeziku su *the, a, an, etc.* Izbačene reči su iz *nltk.corpus* [4] paketa, odnosno *stopwords* kolekcije za engleski jezik. Pored ovih reči izbačena je i reč *may* iz razloga što se analizom podataka zaključilo da se pojavljuje u proseku više od jednom po rečenici, a ne doprinosi njenom značenju.

d) Izbacivanje znakova interpunkcije iz korpusa

U ovom koraku su izbačeni znakovi interpunkcije. Ovaj korak je urađen zato što je procenjeno da sama interpunkcija nikako ne utiče na klasu instance podatka.

e) Lematizacija korpusa

Lematizacija podrazumeva transformaciju reči u njen rečnički oblik. Ovo se radi da bi se smanjio ukupan broj reči u tekstu, ali i da bi se reči grupisale. Pod grupisanjem se smatra svođenje više različitih, semantički istih reči, na isti oblik kao bi ih model mašinskog učenja posmatrao jednako. Lematizovan oblik reči se naziva *lemma*.

3.2. Vektorizacija i *word embedding*

Kako modeli mašinskog učenja zahtevaju numeričku reprezentaciju podataka, a korpus se sastoji od teksta, neophodno je izvršiti transformaciju. Cilj je da se nakon transformacije korpus sastoji od niza vektora koji predstavljaju reči u numeričkom obliku i kojim potom mogu da se koriste za treniranje modela mašinskog učenja. U nastavku su date tehnike pomoću kojih je ovo postignuto.

3.2.1. Tf-Idf

Tf-idf (eng. *term frequency - inverse term frequency*) je model koji služi da numerički prikaže koliko je određena reč bitna u nekoj kolekciji tekstova. *Tf-idf* vrednost je direktno srazmerna broju ponavljanja određene reči u tekstu, a

obrnuto srazmerna broju ponavljanje te reči u celokupnoj kolekciji tekstova. Ovo predstavlja neku vrstu kontra tega, odnosno služi da izniveliše vrednost *tf-idf*-a u slučaju da se neka reč generalno pojavljuje često (u celom korpusu).

3.2.2. GloVe

GloVe predstavlja pretrenirani *word embedding* model koji sadrži vektorsku reprezentaciju reči. Dostupan je u više verzija, odnosno više različitih dimenzionalnosti koje koristi da predstavi reč. Potekao je sa Stanford univerziteta [5]. U ovom radu se koristi i poredi sa *embedding*-om treniranim specifično nad skupom podataka iz ovog rada.

3.3. Algoritam K najbližih komšija (eng. *K-nearest neighbours KNN*)

KNN [6] je model čiji princip ima široku primenu. Moguće je koristiti ga za klasifikaciju i regresiju. U ovom radu *KNN* je korišćen za klasifikaciju.

KNN klasifikator je nadgledani (eng. *supervised*) model mašinskog učenja koji se trenira tako da bude u mogućnosti da klasifikuje ulazne podatke u željeni broj klasa. Ovo se postiže tako što se prilikom treninga dovode podaci za trening koji se smeštaju na odgovarajuću poziciju prostora problema. Prilikom klasifikacije se na ulaz dovodi željeni podatak, koji se potom postavi u prostor i posmatra se koje su klase njegovih K najbližih komšija. Sistemom glasanja se zatim utvrđuje klasa dovedenog podatka (dobija onu klasu koja je najzastupljenija u njegovom okruženju). Računanje distance je moguće raditi bilom kojom metrikom za merenje distance, npr. euklidska udaljenost.

3.4. Mašina potpornih vektora(eng. *Support Vector Machine SVM*)

SVM [8] je model koji se trenira da bude u mogućnosti da klasifikuje ulazne podatke u jednu od dve kategorije. Ovo se postiže tako što mu se inicijalno dostave parovi ulaz-izlaz, te na osnovu algoritma za treniranje izgradi model pomoću kojeg klasifikuje buduće ulaze. *SVM* model je n-dimenzionalni prostor u kojem su podaci predstavljeni kao tačke preko njihovih koordinata. Podaci (tačke) su razdvojene hiper ravni i dodeljuje im se klasa na osnovu pozicije u odnosu na hiper ravan.

3.4.1. Višeklasna (eng. *multiclass*) klasifikacija upotrebom *SVM*-a

Kako je napomenuto u ovoj sekciji, *SVM* je originalno zamišljen kao model za binarnu klasifikaciju. Međutim, kako problem u ovom radu zahteva višeklasnu klasifikaciju, neophodno je bilo koristiti dodatne tehnike koje bi to omogućile. Neke od tehnika koje ovo omogućavaju su: *one-vs-all*, *one-vs-one* i direktni aciklični grafovi. U ovom radu je korišćena tehnika *one-vs-one*. Ona radi tako što originalni problem rastavi na podprobleme koji su tipa binarne klasifikacije. Što rezultuje da nakon razdvajanja postoji binarni klasifikator za svaki par klasa. Konačna klasa se potom utvrđuje sistemom glasanja.

3.5. LSTM

LSTM (eng. *Long short term memory*) [9] je poseban tip rekurentne neuronske mreže (u daljem tekstu RNN), koja ima izmenjenu arhitekturu samog neurona. *LSTM* neuron se sastoji od sledećih elemenata: stanje ćelije (eng. *cell state)*, kapije zaborava (eng. *forget gate*), kapije ulaza (eng. *input gate)* i kapije izlaza (eng. *output gate*). Ulaz u *LSTM* neuron je skriveno stanje pređašnjeg neurona, *Cell state* pređašnjeg neurona i vektor na ulazu (koji u tom trenutku treba da se obrađuje).

Cell state služi za tok informacija unutar mreže. Ove informacije se u svakoj iteraciji ažuriraju *gate*-ovima i igraju ulogu u kreiranju narednog skrivenog stanja.

Cell state se može nazvati i dugoročnom memorijom.

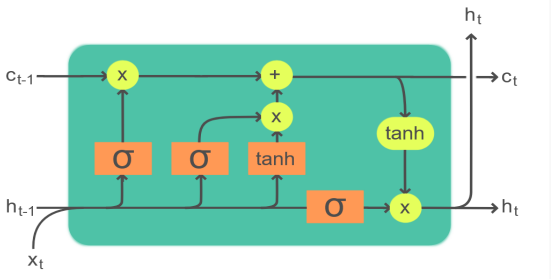
Forget gate je struktura koja uzima prethodno skriveno stanje i trenutni ulaz, spaja ih i propušta kroz sigmoid funkciju (koja vraća vrednosti između 0 i 1). Rezultati se potom množe sa vrednostima iz *cell state*-a i ulaze u njega. Po prirodi množenja što je vrednost iz *forget gate*-a bliža nuli, to će se ona više zaboraviti pri spajanju sa *cell state*-om.

Input gate je struktura koja uzima prethodno skriveno stanje i trenutni ulaz, te ih propušta kroz sigmoid funkciju i tanh funkciju. Rezultat propuštanja kroz sigmoid funkciju ima za cilj da odluči koje su vrednosti bitne da se ažuriraju. Propuštanje kroz tahn funkciju (koja vraća vrednosti između -1 i 1) ima za cilj da svede ulaz na na opseg [-1,1]. Potom se rezultati ove dve operacije množe i rezultat dodaje na *cell state*.

Output gate uzima vrednosti iz *cell state-a* propuštene kroz tanh funkcjiu, kombinovanu vrednost pređašnjeg stanja i trenutnog ulaza propuštenu kroz sigmoid funkciju. Množeći rezultate dobije se skriveno stanje koje se prosleđuje narednoj iteraciji.

Ovakva arhitektura LSTM neurona omogućava dugoročno pamćenje, odnosno filtriranje nebitnih, a pamćenje bitnih podataka. Arhitektura neurona LSTM mreže je prikazana na slici 1.

Slika 1. Neuron LSTM mreže



3.6 BERT

BERT [7] (eng. *Bidirectional Encoder Representations from Transformers*) je *State of the art language model* za *NLP*. Koristi transformer tehnike *attention* mehanizma koje uče iz konteksta relacija između reči u tekstu. U osnovnom obliku transformeri su sačinjeni iz dva dela: enkoder koji čita i dekoder koji daje predikcije. U BERT-ovom slučaju jedino je neophodan enkoder deo kako je njemu zadatak da kreira *language model*.
BERT je predložen kao *language model* nad kojim se može vršiti *fine tuning* za razne primere u domenu NLP-a. Inicijalni modeli su trenirani korišćenjem dve tehnike. Prva je maskiranje reči koja za cilj ima da pogodi koja je reč nalazi iza maske. Druga tehnika radi predikciju da li jedna rečenica sledi iza druge. Ulaznu sekvencu enkodira korišćenjem tri tipa embedding-a koje kombinuje: *Word* koji svakoj reči dodeljuje njen indeks u rečenici, *Sentence* dodeljuje indeks rečenice kojoj reč pripada i *Positional* koji dodeljuje svakoj reči njenu relativnu poziciju u odnosu na početak rečenice.

4. Implementacija

Metodologije o kojima je diskutovano u prethodnom poglavlju implementirane su u okviru *Python* biblioteka *NLTK* (eng. *Natural Language Toolkit*) [14] i *sklearn* [13]. *NLTK* biblioteka pruža funkcije koje omogućavaju manipulacije nad tekstom na jednostavan način. U ovom radu, ova biblioteka je korišćena za deo koraka za pretprocesiranje teksta.
Ostali koraci vezani za pretprocesiranje koji uključuju zamenu ili uklanjanje odgovarajućih tokena mogu se implementirati ugrađenim funkcijama *Python* programskog okruženja. Dodatne pomoćne biblioteke koje su iskorišćene su *pandas* [11] i *numpy* [10].

Napomenimo da biblioteka *sklearn* pruža jedan deo funkcionalnosti za pretprocesiranje polaznog teksta, ali pošto je taj skup funkcija dosta uži od onoga koji pruža *NLTK*, pretprocesiranje će ipak biti oslonjeno na funkcije NLTK biblioteke. Tako da su *sklearn* biblioteci prepušteni samo koraci za vektorizaciju i treniranje tradicionalnih modela (*SVM* i *KNN*).

Za rad sa modelima dubokog učenja i deo vektorizacije je korišćena biblioteka *keras* [12].

5. Skup podataka

Korišćeni skup podataka sačinjen je od 115 politika privatnosti, gde je svaki paragraf dokumenta klasifikovan na odgovarajuću klasu od strane domenskih eksperata.

Konkretan format skupa podataka nije odgovarajući kako su sami paragrafi dalje podeljeni na manje celine koje međusobno dele neke reči. Kako bi se ovaj problem rešio kreiran je algoritam koji prolazi kroz sve celine i uzima odgovarajuće segmente kako bi se kreirao ceo paragraf bez ponavljajućih segmenata.

Po kreiranju ispravnih paragrafa, oni se dalje uparuju sa odgovarajućim klasama. Ovi parovi se zatim predprocesiraju tako što se klase zamenjuju brojevnom reprezentacijom. Dok se za paragrafe izbacuju prepoznate *stop* reči iz engleskog jezika, izbacuju se znakovi interpunkcije, sve reči se pretvaraju u mala slova i nad njima se vrši lematizacija.

6. Eksperiment

Za sprovođenje eksperimenta je potrebno istrenirati klasifikatore, metodama predloženim u poglavlju 3, a zatim sprovesti klasifikaciju nad podacima koji se nalaze u test skupu. O ovome i potencijalnim problemima pisaćemo u nastavku poglavlja. Istrenirani modeli (težine) su sačuvani i nalaze se pod folderom *models* u repozitorijumu.

6.1. Klasifikacija

Za klasifikaciju su korišćena 2 tradicionalna modela (*SVM* i *KNN*), kao i modeli dubokog učenja (*LSTM* i *BERT*). Podela korpusa na trening/test je urađena u odnosu 70%/30%, gde trening deo čini 70%, a test deo 30% ukupnog korpusa.

6.1.1. Tradicionalne metode

Optimizacija hiper-parametara tradicionalnih modela je vršena pomoću mrežne pretrage (eng. *grid search*). Ova metoda omogućava da se unese širok spektar vrednosti hiper-parametara, gde ona potom trenira modele sa svim mogućim kombinacijama unetih hiper - parametara, pamteći parametre najboljeg modela. Validacija tradicionalnih metoda je vršena tehnikom unakrsne validacije (eng. *k-fold cross validation*). Ova tehnika podrazumeva podelu trening skupa na željeni broj delova, gde se potom model trenira nad svim delovima osim jednog. Poslednji deo se koristi da se evaluira trenirani model. U ovom radu je korišćen 4 - fold cross validation, što znači da je trening skup deljen na 4 jednaka dela, gde su 3 korišćena za trening, a 1 za validaciju. Podaci su podeljeni na stratifikovan način (eng. *stratified*), što znači da odnos klasa između trening i validacionog skupa ostaje isti nakon podele.

6.1.2. Metode dubokog učenja

Za potrebe treniranja modela dubokog učenja, korišćen je isti princip podele podataka na trening/test kao i kod tradicionalnih metoda. Što se tiče samih pristupa modelima dubokog učenja eksperimentisano je sa nekoliko pristupa. Za modele dubokog učenja su korišćeni *LSTM* i BERT. Što se tiče *word embedding*-a korišćen je *GloVe* i *word*

embedding koji se trenirao specifično za ovaj problem.

6.2. Evaluacija

Za evaluaciju modela korišćena je F1 mera (eng. *F1-score*), koja kombinuje precision i recall. Od posebnog nam je interesa *weighted* F1 mera, koja još uzima u obzir i prisutnost svake klase. U tabeli 2 su prikazani dobijeni rezultati svih korišćenih modela.

Tabela 2. Rezultati svih korišćenih modela	
Model	Weighted F1 mera
SVM	0.81
KNN	0.77
LSTM	0.79
LSTM + GloVe	0.78
BERT	0.83

Posmatrajući i *classification report KNN modela,* koji je prikazan na slici 2 (*classification report* ostalih modela nije prikazan, ali iako se same vrednosti menjaju, odnos među njima ostaje približno isti), može se uočiti da model pojedine klase prepoznaje daleko teže nego ostale. Pogledati (tabelu 1) za referencu koji broj predstavlja koju klasu. Sa slike 2 se dakle vidi da klase "other" i "data retention" imaju *f1-score* ispod 0.6, što ih čini klasama nad kojima model najlošije radi.

Slika 2. *Classification report* KNN modela

	precision	recall	f1-score	support
0	0.63	0.49	0.55	744
1	0.81	0.80	0.80	139
2	0.80	0.90	0.85	2329
3	0.70	0.50	0.58	105
4	0.67	0.80	0.73	172
5	0.79	0.82	0.81	1381
6	0.71	0.58	0.64	477
7	0.78	0.71	0.74	205
8	0.86	0.70	0.77	236
9	0.94	0.85	0.89	20
accuracy			0.77	5808
macro avg	0.77	0.71	0.74	5808
weighted avg	0.77	0.77	0.77	5808

6.3. Problemi

Prilikom izrade projekta i analize razultata, uočena su dva značajnija problema:

6.3.1. Korpus je nebalansiran

Kao što se vidi sa slike 2 pod kolonom *support* neke klase su zastupljenije deleko više od ostalih. Ovo ima za posledicu da je model u stanju da visoko zastupljene klase nauči jako dobro, dok može da se desi da ne nauči slabo zastupljene klase. Naša pretpostavka je da je to jedan razlog tako slabog rezultata za klasu "data retention". Drugi razlog za tako loš rezultat "data retention" klase je što ima preklapanje (eng. *overlap*) sa ostalim klasama, što rezultuje u pogrešnoj klasifikaciji. Međutim, ako se posmatra zastupljenost klasa, može se primetiti da "do not track" klasa ima svega 20 primeraka u test skupu, a postiže jako dobar rezultat. Objašnjenje za ovo je da su svi primerci ove klase međusobno jako slični, a istovremeno u potpunosti različiti u odnosu na ostale klase, te je model uspeo da ih nauči izuzetno brzo. Ovaj problem bi se mogao rešiti ili izbacivanjem klasa koje imaju mali broj instanci ili dodavanjem instanci u slabo zastupljene klase.

6.3.2. Klasa "other"

Kao što joj i samo ime kaže, klasa other zapravo ne predstavlja jedan tip podatka, već sadrži sve instance koje se ne uklapaju u ostale klase. Samim tim ova klasa je izuzezno nestabilna, što rezultuje u dosta lošem rezultatu kada je ona u pitanju. Kao rešenje bi se mogla predložiti dva pristupa, a to su izbacivanje ove klase u potpunosti ili anotiranje njenih elemenata u postojeće ili nove klase.
U ovom radu izbacivanje ove klase nije primenjeno iz razloga što bi njenim izbacivanjem, izbacili previše podataka (ova klasa je treća po zastupljenosti u

skupu). A kako nismo kompetentni za analizu i anotiranje politika privatnosti, ni pristup re-anotiranja “other” klase takođe nije primenjen.

7. Zaključak

U ovom radu je implementirana klasifikacija delova politike privatnosti. Sama implementacija uključuje tri koraka: pretprocesiranje, vektorizaciju i treniranje modela.

Pretprocesiranje uključuje niz koraka za prilagođavanje ulaznog skupa podatka kako bi se iz njega izvukao maksimum. Potom se vrši tokenizacija i transformisanje skupa podataka u oblik pogodan za prosleđivanje modelu mašinskog učenja. Konačno se treniraju tradicionalni modeli i modeli dubokog učenja.

Za implementaciju se koristilo *python* okruženje. Pomoću biblioteka *keras, pandas, numpy, nltk i sklearn* moguće je na jednostavan način impelementirati sve potrebne funkcije izloženih metodologija.

Na kraju je objašnjen sam eksperiment i prikazani su dobijeni rezultati. Zaključujemo da se najbolje rezultate pokazao *BERT*, dok je *KNN* imao najlošije performanse.

Potom su diskutovana dva problema i dat predlog rešenja koja bi moglo da poboljšaju performanse modela.

8. Reference

[1] Wilson, Shomir, et al. "The creation and analysis of a website privacy policy corpus." *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2016.

[2] Kanamori, Sachiko, et al. "Construction of a Support Tool for User Reading of Privacy Policies and Assessment of its User Impact." *ICISSP*. 2022.

[3] Jensen, Carlos, Colin Potts, and Christian Jensen. "Privacy practices of Internet users: Self-reports versus observed behavior." *International Journal of Human-Computer Studies* 63.1-2 (2005): 203-227.

[4] https://www.nltk.org/api/nltk.corpus.html

[5] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.

[6] Fix, Evelyn; Hodges, Joseph L. (1951). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties (PDF) (Report). USAF School of Aviation Medicine, Randolph Field, Texas. Archived (PDF) from the original on September 26, 2020.

[7] Horev, Rani. "BERT Explained: State of the art language model for NLP. 2018." *URL: https://towardsdatascience.com/bert-explainedstate-of-the-art-language-model-for-nlp-f8b21a9b6270*

[8] Cortes, Corinna; Vladimir Vapnik (1995). "Support-Vector Networks". *Machine Learning*. **20** (3): 273–297. doi:10.1007/BF00994018.

[9] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". *Neural Computation*. **9** (8): 1735–1780.

[10] https://numpy.org/doc/

[11] https://pandas.pydata.org/

[12] https://keras.io/

[13] https://scikit-learn.org/stable/

[14] https://www.nltk.org/