Document editor za pravna dokumenta sa predlogom sledeće reči

Luka Kričković, Đorđije Kundačina

Računarstvo i automatika
Fakultet tehničkih nauka
Univerzitet u Novom Sadu
{krickovic.e220.2021, kundacina.e276.2021}@uns.ac.rs

***Apstrakt*— U ovom radu opisan je document editor koji ima ugradjen podsistem za preporuku sljedeće reči. Ideja za ovakav rad proistekla je iz potrebe da se pravnicima omogući lakse, jednostavnije i brže pisanje pravnih dokumenata. Sistem za preporuku reči (autocomplete) implementiran je uz pomoć pretrenigranog GPT2 modela, koji je dodatno find-tuned (dotreniran) specijalno za nas skup podataka. Krajnji projekat je realizovan u vidu web aplikacije. Dizajniran je korisnički interfejs koji omogućava korisniku kreiranje dokumenta i prilikom kucanja daje mu predloge za sledece reči.**

***Ključne reči*— Pravni domen, mašinsko učenje, document editor, system recommendation, tip dokumenta, procesovanje prirodnog jezika**

I. Uvod

Pravna dokumenta su po svojoj prirodi kompleksna, precizna i koriste opširan skup izraza, koji su strogo vezani za pravni domen, pa predstavljaju relativno velik izazov za kreiranje, odnosno kucanje u digitalnim okruženjima. Trenutno ne postoji veliki izbor specifičnih alata koji mogu da ubrzaju izradu ovakvih dokumenata. Najpoznatiji primer ovakvog sistema je Legito, koji je all in one rešenje sa ugrađenim sistemom za upravljanje dokumentima, a postoje i brojni pomoćni softveri koji se nadograđuju na Word i slične office programe, koji su defakto standard za kucanje pravnih dokumenata.

U ovom radu će biti opisano naše rešenje ovog problema, koje koristi najnovije tehnologije iz domena obrade prirodnog jezika (eng. natural language processing - NLP), kako bi analiziralo uneti tekst i ponudilo predlog sledeće reči i time olakšalo korisniku da sa manje napora dostigne veću efikasnost pri kucanju ugovora.

Tehnologije na kojima baziramo rešenje su konkretno transformer modeli, odnosno GPT2 (eng. Generative Pretrained Transformer 2) model, kojeg su razvili OpenAI, a trenutno je dostupno za preuzimanje i upotrebu preko huggingface repozitorijuma modela. Odabran je konkretno GPT2-medium model, iz razloga što GPT3 nije otvoren publici, odnosno nije open source u trenutku pisanja ovog rada, a veći varijateti GPT2 modela su bili previše kompjuterski zahtevni za izradu ovog rešenja (medium ima 1024 slojeva, dok large ima 1600, i veličine je 6.5gb u baznoj konfiguraciji).

Iako ovaj model ume da generiše kvalitetne i koherentne rečenice, on ne ume da koristi izraze iz pravnog domena, pa ga je bilo potrebno istrenirati nad skupom pravnih tekstova, u našem slučaju ugovora, kako bi davao upotrebljive rezultate.

II. Slična istraživanja

Cilj ovog poglavlja je da opiše metodologije i rezultate predstavljene u radovima na kojima smo mi bazirali naše rešenje. Za izradu našeg sistema su prvenstveno korištena dva rada: “CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review”, Dan Hendrycks, Collin Burns, Anya Chen, Spencer Ball, kao i “Legal Language Modeling with Transformers”, Lazar Perić , Stefan Mijić , Dominik Stammenbach i Elliott Ash.

Prvi rad nam je veoma relevantan, jer predstavlja propratni rad koji dolazi uz skup podataka koji smo primarno koristili - CUAD v1. U ovom radu je opisan način kreiranja njihovog modela za izvlačenje zaključaka iz pravnih tekstova, što iako nama nije bio cilj, funkcioniše na bazi istih tehnologija i jedina je razlika u koraku generisanja teksta, o čemu ćemo pisati u daljim poglavljima. U ovom radu su dali poređenje između različitih transformer modela, od kojih su predložili upotrebu DeBerta (xlarge), jer za skup podataka te veličine (500 različitih ugovora, 13000 anotiranih delova teksta, 41 jedinstvena

labela) daje najbolje rezultate. Način na koji su postavili eksperiment jeste da su koristili četiri različita BERT transformer modela sa svim varijatetima (base, medium, large, xlarge). Odabrane modele su fine tune, odnosno pretrenirali nad istim skupom podataka i izračunali su površinu pokrivenosti (eng. Area Under Curve - AUC) metrike i povezane Recall metrike i napravili poređenje. Interesantan zaključak je da za različite tipove tekstova modeli dobijaju različite preciznosti. Takođe, različiti modeli dobijaju proporcionalne preciznosti za određene tipove tekstova, što važi i za naš model. Konkretno, modeli najbolje funkcionišu kada analiziraju delove tekstova koji se odnose na nadležno pravo (eng. governing law), naziv dokumenta, ugovorne strane (eng. parties), kao i datume važenja (eng. agreement, expiration date). Zaključak ovog rada jeste da DeBerta model funkcioniše dovoljno dobro da se koristi u profesionalnoj primeni i da može advokatima da znatno uštedi vreme izvlačenjem klauzula ugovora, da advokati ne moraju da čitaju ceo tekst.

U drugom radu su predstavljena alternativna rešenja za konkretno naš problem, generisanje pravnog teksta, odnosno predlogom sledeće reči (eng. next word prediction) u domenu pisanja ugovora. Modeli koji su opisani u ovom radu su GPT2 (koji smo mi upotrebili), GPT3, BERT modeli, kao i potpuno novi transformer model kojeg su autori sami kreirali za potrebe ovog eksperimenta, transformer-xl model. Kako bi maksimalno očuvali stil originalnog teksta, autori su predložili minimalnu obradu početnog korpusa. Jedina obrada je bila u vidu zamene znakova interpunkcije sa specifičnim oznakama koje modeli razumeju, npr. <|startoftext|> za početak paragrafa i <|endoftext|> za kraj paragrafa u slučaju GPT modela. Nakon toga, tekstove su iskoristili da prilagode GPT2 model, ali i da potpuno istreniraju njihov transformer-xl model, što znači da GPT2 zna da generiše engleski tekst, ali je specijalizovan za pravni domen, dok njihov transformer-xl samo zna da analizira pravni tekst. Ovo značio da transformer-xl ne može toliko efikasno da se specijalizuje za druge domene, što je prednost u ovom slučaju jer će za ovaj specifičan slučaj davati bolje rezultate. Prvenstveno su poredili rezultate korišćenjem Flesch–Kincaid ocene čitljivosti (veća ocena označava bolju čitljivost). Tekstovi iz korpusa, odnosno pravne presude koje su sakupili su imali gotovo identičnu ocenu čitljivosti kao i pretrenirani GPT2 model (redom 30.37, 30.47), što je interesantno jer pokazuje da GPT2 u odličnoj meri održava karakteristike početnog teksta, odnosno proizvodi tekst koji je podjednako kompleksan kao i tekstovi koje su napisali ljudi. Transformer-xl model autora je dobio rezultat 40.92, što znači da proizvodi neznatno manje kompleksne tekstove, što može biti i prednost i mana zbog prirode problema. Zaključak ovog rada je dobijen anketiranjem velikog broja pravnika, kao i ljudi koji se ne bave pravom. Generalno, tekstovi iz GPT2 i transformer-xl modela su bili toliko precizno i realistično generisani da ljudi nisu bili u stanju da razaznaju koji je tekst generisan, a koji je napisao čovek. Prosečna mera uverenja korisnika da je tekst generisan je 57.6% za transformer-xl i 55.4% za GPT2 (što bliže 50% to je bolje u ovoj situaciji, jer 50% označava da su korisnici morali nasumično da pogađaju između ponuđenih odgovora). Ovi rezultati su identični i za pravnike i za ljude koji nisu stručni u pravnom domenu.

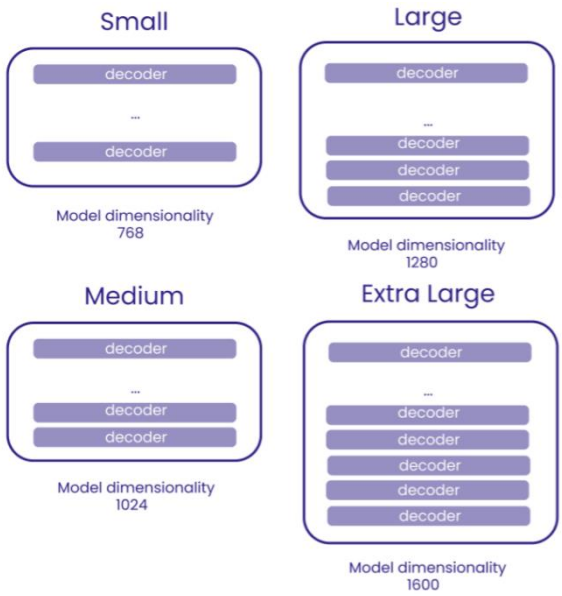
III. Metodologija

U ovom radu su korištena dva eksperimenta, nakon čega je bolje rešenje pretvoreno u aplikaciju pomoću koje korisnici mogu praktično da isprobaju naš sistem. Prvi korak bio je prikpuljanje tekstova, odnosno izmena skupa podataka da odgovara našim potrebama, nakon čega su izvršeni eksperimenti i izrađena je aplikacija za demonstraciju rešenja.

A. Izmena skupa podatala

Skup podataka koji smo koristili je CUAD v1 (eng. Contract Understanding Atticus Dataset), koji je opisan u delu sa sličnim radovima. Problem kod ovog skupa podataka je što je namenjen izvlačenju zaključaka iz ugovora, pa tekstovi nisu dati u punom formatu, nego su dati isparčani po klauzulama, sa delovima teksta koji su izbačeni, što bi nama predstavljalo ogroman problem. Uz skup podataka, kao propratnu dokumentaciju je moguće preuzeti i skup svih upotrebljenih ugovora u pdf formatu, što smo upotrebili na sledeći način. Prvi korak je bio konverzija pdf u txt datoteke, koje smo uvezli u python notebook i spojili u jednu veliku csv datoteku. Ova datoteka je imala samo jednu kolonu

koja je sadržala ceo tekst dokumenta, što ne predstavlja veoma koristan skup podataka jer je prekompleksan za slanje direktno u model. Kako bismo mogli slati tekstove u model, morali smo da im ograničimo dužinu na 500 karaktera, jer je maksimalna veličina ulazne sekvence u GPT2 ograničena, a i CUDA api koji je upotrebljen za treniranje modela je ograničen zbog veličine GPU što nudi Google Colab. Nakon ovoga, svakom redu je dodeljena oznaka kom tipu ugovora pripada. Ovaj podatak su autori ugradili u naziv datoteke tako što su na kraju naziva samog ugovora dodali '_Tip ugovora.csv', što smo uz pomoć regularnih izraza (eng. regex) izvdojili i dodelili odgovarajućim klauzulama. Takođe, kod oba eksperimenta je potrebno kreirati ekstenziju huggingface python klase Dataset, po zvaničnoj dokumentaciji sa njihovog sajta.



Slika. 1 Poređenje varijacija GPT2 modela

B. Predlog sledeće reči uz pomoć pretreniranog GPT2

Prvi eksperiment je koristio GPT2 koji je pretreniran (eng. fine tuned) nad tekstovima iz našeg skupa podataka. Ulaz u ovaj model je tokenizovan tekst, koji je pritom minimalno obrađen. Konkretno, koristili smo GPT2 tokenizator, koji se dobije u sklopu transformers python biblioteke. On standardno zameni znakove interpunkcije i specifične simbole sa oznakama koje prepoznaje (npr. <|startoftext|> i <|endoftext|>). Dalje, ovaj tekst se pretvara u vektorski format, koji je maksimalne dužine 1024 karaktera i prosleđuje se u model.

Model se sastoji iz velikog broja transformerskih dekodera, tačna cifra zavisi od odabrane varijacije. U našem slučaju, model ima 1024 sloja, sa ukupno 345 miliona parametara. Ovaj model se preuzima sa huggingface repozitorijuma istreniran tako da neposredno nakon preuzimanja može da generiše veoma koherentan tekst na engleskom jeziku, ukoliko se ne radi o tekstovima unutar specifičnog domena. Šema modela je prikazana na slici 1.

Pretreniranje (eng. fine tune) se obavlja uz pomoć Trainer klase iz transformers biblioteke, koja za parametre dobija TrainingArgs klasu iz iste biblioteke. Ovime se u Trainer prenosi odgovarajuća kolona iz skupa podataka, u kojoj se nalaze tekstovi na kojima se specijalizuje model, uz dodatne parametre. Konkretni parametri što smo mi koristili su: output_dir='./results' (direktorijum gde će se čuvati logovi i rezultati nakon treniranja modela), num_train_epochs=1 (broj epoha prilikom treniranja), logging_steps=100 (granularnost logova), save_steps=5000 (na koliko se koraka pravi backup modela), per_device_train_batch_size=1 (veličina uzorka pri treniranju po procesu), per_device_eval_batch_size=1 (veličina uzorka pri evaluaciji po procesu), warmup_steps=10 (broj koraka zagrevanja), weight_decay=0.05 (degradacija nagrada za nenadgledano učenje), logging_dir='./logs' (direktorijum za logovanje grešaka).

Nakon treniranja, potrebno je implementirati i funkciju generisanja i dekodiranja enkodovanog teksta kojeg proizvodi GPT2 model. Ovo se postiže pozivanjem model.generate() funkcije, gde se kao parametar prosleđuje tokenizator sa učitanim <|startoftext|> karakterom, kao i postojećim tekstom kojeg treba dopuniti. Generisani tekst de dekoduje uz pomoć GPT2 tokenizatora, GPT2Tokenizer.decode() metodom, što proizvodi

ljudski čitljive tekstove. Da bi se dobila samo sledeća reč, iz nastalog teksta se izbacuje ulazni tekst upotrebom python str.replace() metode.

C. GPT2 sa labelama

Prethodni eksperiment je dao veoma dobre rezultate, ali je problem što se ovaj model ponaša veoma stohastično, što se može videti na osnovu raspona rezultata preciznosti (+/- 5%, o tačnoj vrednosti će biti reči u rezultatima). Razlog za ovakve fluktuacije je što se model ponaša kao crna kutija (eng. black box). Konkretno, kada model dobije tekst, on uzima deo po deo unosa i uči pravila generisanja daljeg teksta. Kako nema nikakve naznake kom tipu ugovora tekst pripada, naš model ne za sa sigurnošću kakav nastavak teksta može biti. Kako bismo sanirali ovaj efekat, uvodimo oznake, odnosno labele sa nazivom tipa ugovora u tekst. Konkretno, ulazni tekst u model se okružuje sa <<Naziv tipa ugovora>></Naziv tipa ugovora>>. Ovime garantujemo da prilikom starta generisanja teksta naš model ima bolju ideju kakav tekst sledi, odnosno time smo obučili naš model da razlikuje tekstove po tipovima ugovora. Ostatak eksperimenta je potpuno ista kao u prethodnom primeru. Ukratko, podaci se šalju u tokenizator, iz kojeg se sa istim parametrima prosleđuje u Trainer klasu, koja specijalizuje model za ugovore, sa dodatnom informacijom o tome koji je tip ugovora u pitanju.

D. Implementacija aplikacije

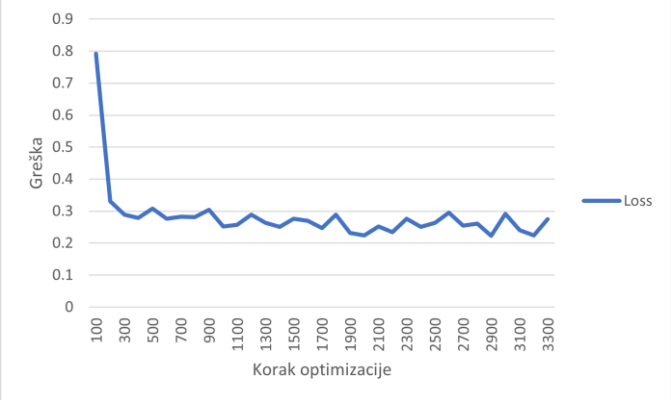
Odabrali smo format web aplikacije, sa klijentskom aplikacijom u React radnom okviru i serverskom aplikacijom u python programskom jeziku, odnosno Flask radnom okviru. Razlog zašto smo odabrali baš ove tehnologije su lične preference za klijentsku aplikaciju, ali za serversku aplikaciju je python bio logičan izbor zbog lake integracije sa našim istreniranim modelom, kao i zbog jednostavnosti kreiranja strukture sa samo jednim endpoint-om zahvaljujući Flask radnom okviru.

Serverska aplikacija prilikom inicijalizacije preuzima blanko GPT2 model sa huggingface repozitorijuma, nakon čega se menja njegov oblik da bi mogao da se prilagodi dimenzijama našeg pretreniranog modela. Nakon toga, da bi se učitao naš model, poziva se load_state_dict metoda, koja kao parametar dobija Dictionary objekat sa stanjima čvorova. Nakon ovoga inicijalizuje se i tokenizator, koji se dalje koristi kroz aplikaciju. Sam server ima otvoren samo jedan endpoint, "/predict", koji prihvata HTTP Post zahteve, koji sadrže korisnički unos i odabrani tip ugovora.

Klijentska aplikacija se sastoji od početne stranice koja sadrži polje za unos teksta, koje ima ugrađeno formatiranje, odnosno sav unet tekst čuva u html zapisu. Taj zapis se na određeni period šalje na server, gde se odgovor prikazuje korisniku u vidu najviše tri dugmeta sa labelama. Klikom na dugme se na uneti tekst konkatenira novi, generisani tekst.

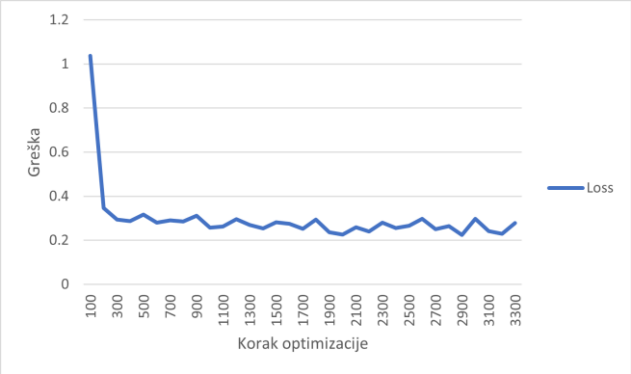
IV. REZULTATI

Nakon procesa prilagođavanja pravnim dokumentima, GPT2 dobija se preciznost od 66.4% do 71.6%, po metrici klizećeg prozora (eng. sliding window). Funkcija opadanja greške je vidljiva na slici 2. Napomena je da se preciznost računa na nivou generisanja teksta određene veličine, a ne na nivou generisanja jedne reči, zbog načina funkcionisanja sliding window metrike.



Slika. 2 Funkcija greške u zavisnosti od koraka optimizacije

Na slici 3 se vidi da verzija GPT2 modela sa labelama ima manje fluktuacije prilikom treniranja modela iz koraka u korak. Takođe, finalna preciznost dobijena je 72.5% (+/- 1%), što znači da ova verzija modela brže i stabilnije konvergira i proizvodi dobre predikcije. Funkcija greške je prikazana na slici 3.



Slika. 3 Funkcija greške u zavisnosti od koraka optimizacije, uz podatak o tipu ugovora

V. Zaključak

Cilj ovog rada je da se kreira sistem koji će pomoći prilikom kreiranja pravnih dokumenata, konkretno ugovora, time što će davati predlog sledeće reči. Naše rešenje, uz dodatne optimizacije, ima potencijala da bude ugrađeno u velik broj već postojećih softverskih rešenja, kako bi većem broju korisnika drastično ubrzalo kucanje i olakšalo upotrebu šireg vokabulara, specifičnog za pravni domen.

Naš sistem je predstavljen putem web aplikacije, ali se naglasak stavlja na model koji vrši predikciju i server koji enkapsulira svu ostalu logiku potrebnu za generisanje teksta. Model koji smo koristili je GPT2 (eng. Generative Pretrained Transformer 2), koji je javno dostupan preko huggingface repozitorijuma. Kroz taj model smo potom probali da generišemo predloge za sledeću reč, ali smo dobili veoma loše rezultate. Nakon toga smo prilagodili neurone izlaznog sloja modela (eng. fine tune), čime smo drastično povećali preciznost. U drugoj iteraciji eksperimenta smo hteli da povećamo preciznost i stabilizujemo rezultate time što smo uveli kontekst u predikcije, odnosno anotirali smo tekst labelama koje su naznačile o kom tipu ugovora se radi.

Prostor za poboljšanje je u načinu na koji anotiramo tekst. Konkretno, anotacija celog teksta tipom ugovora daje konktekst, ali ne naznačava tačno o kom tipu klauzule je reč, nego to naš model sam nagađa. Ukoliko bismo anotirali tekst po klauzulama umesto po tipu ugovora, dobili bismo primetno veću preciznost, jer se određeni tipovi klauzula ponavljaju u različitim tipovima ugovora. Zbog ovoga se pravila pisanja ne mogu toliko dobro grupisati po samom tipu ugovora, nego po tipu klauzule.

VI. Literatura

[1] Dan Hendrycks, Collin Burns, Anya Chen, Spencer Ball, *CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review*, 2nd ed., arXiv, 2021.

[2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, *Language Models are Unsupervised Multitask Learners*, 2020.

[3] Lazar Perić, Stefan Mijić, Dominik Stammenbach, Elliott Ash, *Legal Language Modeling with Transformers*, ASAIL 2020.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, *Attention Is All You Need*, arXiv, 2017.

[5] Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Martin Katz, Nikolaos Aletras, *LexGLUE: A Benchmark Dataset for Legal Language Understanding in English*, arXiv, 2022.