Primer (loše napisanog) opisa rešenja

I. KREIRANJE *CHATBOT-A* KORIŠĆENJEM WORD2VEC BIBLIOTEKE I REKURENTNIH NEURONSKIH MREŽA

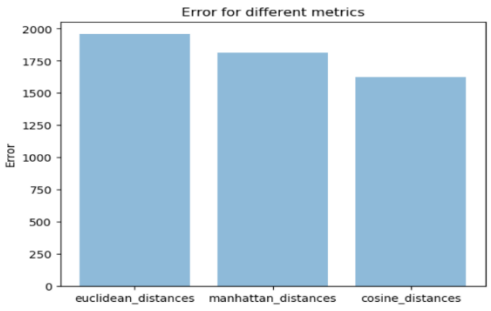
Naš pristup razvoju *chatbot* aplikacije se može podeliti u dve faze. Prvu fazu koja zahteva brz odziv (*high recall*) i drugu fazu visoke preciznosti (*high precision*). Pre prve faze smo odradili predprocesiranje našeg skupa podataka u kom smo došli do zaključka da postoji dosta duplih pitanja i istih pitanja sa različitim odgovorima. Dupla pitanja smo obrisali, a pitanja sa različitim odgovorima smo konkatenirali u jedan jedinstven odgovor.

A. Faza 1: High recall

Prvi korak u razvoju našeg modela bio je vektorizacija reči, odnosno transformacija reči u numeričke vektore. Za ovu svrhu smo isprobali različite metode. Prvo smo probali *CountVectorizer* koji nam transformiše reči u matricu ponavljanja. Model je dosta jednostavan za naš problem nije davao dobre rezultate jer ne uzima u obzir semantiku reči. Drugo stvar koju smo probali je *TFIDF Vectorizer* on je naprednije od prvog pristupa jer pored broja ponavljanja reči uzima u obzir i važnost reči u našem dokumentu. Kao što smo naglasili ni jedan od ova dva pristupa se nije dobro pokazao kod nas, verovatno jer oba pristupa koriste *bag-of-words*, koji ignoriše redosled reči u tekstu.

Zbog gore navedenih nedostataka prva dva pristupa, odlučili smo se za Word2Vec, koji uzima u obzir kontekst između dve reči. Word2Vec smo isprobali smo na 3 različita skupa podataka. Prvi je bio naš osnovni koji smo sredili pre ulaska u fazu 1, dok drugi i treći smo dobili lematizacijom i shematizacijom. Lematizacija je pretvaranje reči u njihov osnovni oblik, dok je stematizacija proces koji vraća reč na njen korenski oblik. Ni jedan od ova dva napredna pristupa nije doneo poboljšanje u rezultatima, tako da smo odlučili da radimo sa prvobitnim skupom podataka.

Kada smo završili vektorizaciju, isprobali smo 3 različite metrike sličnosti da bi pronašli slična pitanja. Isprobali smo *Manhattan* razdaljinu, kosinusnu razdaljinu i Euklidsku razdaljinu. Nakon što smo istestirali sva 3 pristupa, došli smo do zaključka da najmanju grešku pravi kosinusna razdaljina i odlučili smo nju da koristimo (Slika 2). Kao krajnji rezultat faze 1 dobijamo 100 pitanja koja su najsličnija unosu korisnika. Ovim smo postigli da kad krenemo u drugu fazu, naša rekurentna neuronska mreža neće prolaziti kroz ceo skup podataka, nego samo kroz ovih 100 najsličnijih pitanja.



Slika 2 Dijagram izbora metrike 1

B. Faza 2: High precision

Sledeća faza u našem radu bila je odabir najpreciznijeg pitanja iz skupa pitanja koje smo dobili u prvoj fazi. Ovaj deo smo izveli pomoću LSTM modela koji je zasnovan na rekurentnim neuronskim mrežama. Ovaj model je bio pogodan za rešavanje našeg problema iz razloga što uzima u obzir dugoročne zavisnosti između sekvenci. U kontekstu razumevanja prirodnog jezika, model pamti duže vreme, što je ključno za razumevanje konteksta u konverzacijama.

Tokom procesa treniranja, mrežu smo konfigurisali tako da minimizira grešku između svojih predikcija i stvarnih odgovora. Mreža je trenirana da predvidi pitanje tako što je iterativno prilagođavala svoje težine da bi smanjila grešku između izlaznih vrednosti mreže i stvarnog pitanja. Kao što je u uvodu već spomenuto dobili smo preciznost od 69.82%. Ovaj rezultat je dobijen za konfiguraciju mreže od 700 epoha. Ovaj broj je izabran jer je to maksimalan broj za koji smo mi imali vremena da čekamo da se istrenira naš model.

Poslednji korak razvijanje naše *chatbot* aplikacije. Odlučili smo da napravimo *web* aplikaciju kako bi mogli lakše da testiramo naš *chatbot*. Serversku stranu naše aplikacije smo napravili koristeći *Flask framework*, dok smo klijentsku stranu naše *web* aplikacije napravili koristeći *React JS* biblioteku.

C. Alternativni pristupi

Takođe smo razmatrali i druge načine za rešavanje ovog problema. Jedan od njih je bio korišćenje Transformer modela, kao sto jer *BERT* (*Bidirectional Encoder Representations from Transformers*). Ovakvi modeli su se pokazali dosta efikasnim u obradi prirodnog jezika, međutim ovo su veoma kompleksni modeli koji zahtevaju velike količine podataka i računarske snage koja je potrebna za treniranje.

Drugi pristup koji je trenutno najpolarniji je korišćenje generativnog *chatbot*-a, poput *ChatGPT* kog razvija *OpenAI*. Ovi modeli koriste velike količine podataka da bi naučili da generišu prirodni jezik. Problem je što genativne *chatbot*-ove je teško za kontrolisati i u slučaju *ChatGPT* ogromna su čekanja za dobijanje *API* ključa.