Korištenje naprednih tehnika dohvatom-pojačanog generiranja za prepoznavanje futuroloških signala

Šoštarko, Igor

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1598

KORIŠTENJE NAPREDNIH TEHNIKA DOHVATOM-POJAČANOG GENERIRANJA ZA PREPOZNAVANJE FUTUROLOŠKIH SIGNALA

Igor Šoštarko

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1598

KORIŠTENJE NAPREDNIH TEHNIKA DOHVATOM-POJAČANOG GENERIRANJA ZA PREPOZNAVANJE FUTUROLOŠKIH SIGNALA

Igor Šoštarko

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1598

Pristupnik:	**Igor Šoštarko (0036540801)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	doc. dr. sc. Mario Brčić

Zadatak: **Korištenje naprednih tehnika dohvatom-pojačanog generiranja za prepoznavanje futuroloških signala**

Opis zadatka:

U današnje doba prezasićenosti informacijama, izuzetno je zahtjevno pronaći sadržaj koji predstavlja signal promjene u smislu znanosti o budućnosti (engl. futures studies). Prirodni slijed je koristiti sustave umjetne inteligencije za pretraživanje nadolazećih podataka. Za takve svrhe su se veliki jezični modeli (engl. Large Language Model, LLM) pokazali izuzetno dobrim početnim rješenjem. No, model je jednom nakon treniranja zamrznut na trenutnoj razini znanja i s prolaskom vremena će sve više imati problema s prepoznavanjem signala novih promjena u odnosu na već dobro poznate signale koji više nisu relevantni. Fino podešavanje je jedno potencijalno rješenje za problem ustajalog znanja, no zahtijeva veća ulaganja i proces podešavanja može potrajati - što smanjuje responzivnost sustava na promjene. Ovaj završni rad usredotočit će se na implementaciju naprednih tehnika dohvatom-pojačanog generiranja (engl. Retrieval-Augmented Generation, RAG) u svrhu poboljšanja osnovnog sustava temeljenog samo na LLM-u. Tehnike RAG-a pohranjuju novo znanje u vanjsku vektorsku bazu podataka koju potom LLM modeli mogu koristiti kao vanjsku memoriju. Troškovi takve implementacije su niži od podešavanja, te je vrijeme do inkorporiranja novih znanja puno kraće. Glavni cilj rada je primijeniti napredne tehnike RAG-a za eliminiranje "ustajalih" signala - podržavajući pritom dvije perspektive. Prva perspektiva je općeg populacijskog znanja, druga je korisnička perspektiva koja eliminira već konzumirane signale za pojedinog korisnika.

Rok za predaju rada: 14. lipnja 2024.

Zahvaljujem se mentoru doc. dr. sc. Mariju Brčiću na pomoći i usmjeravanju tijekom izrade završnog rada.

Sadržaj

1. Uvod

U današnje vrijeme zasićenosti informacijama postaje sve teže prepoznati koje vijesti nose informacije o potencijalno značajnim promjenama u budućnosti, tj. futurološke signale. Ljudi su ograničeni u svojim mogućnostima te u samoj brzini analiziranja velike količine informacija, pa se nameće pitanje o tome koji im pristup može pomoći. Uzimajući u obzir veliki razvoj tehnologija umjetne inteligencije u zadnjih nekoliko godina, kao početna se ideja nameće njihovo korištenje. Konkretno, veliki jezični modeli (eng. Large Language Model, LLM) se čine kao dobar početak pošto mogu analizirati tekst te generirati novi tekst na temelju drugog teksta. No takvi modeli su trenirani do određene točke u vremenu, te iz tog razloga nisu dobar alat za analiziranje informacija koje su nastale značajno nakon njihovog perioda treniranja. Takav problem se može riješiti na više načina. Jedan moguć način je dotreniravanje modela, no to može biti dugotrajan proces koji zahtjeva veća ulaganja te koji bi se trebao provoditi vrlo često. U ovom radu se koristi drugi pristup, a to je implementacija tehnika dohvatom-pojačanog generiranja (eng. Retrieval-Augmented Generation, RAG) za davanje konteksta jezičnom modelu o događajima nakon završnog datuma treniranja modela. Cilj RAG-a je naći najrelevantnije informacije koje bi jezičnom modelu pomogle u odluci o tome ako je dani članak futurološki signal, te mu ih predati kao kontekst s člankom koji se provjerava. Tako model može točnije procijeniti članak bez da se dotrenirava, te je taj pristup lakši i brži za implementaciju.

Na početku ćemo dati teoretsku podlogu o najbitnijim tehnologijama korištenim u ovom radu. Ukratko ćemo predstaviti jezične modele i vektorsku reprezentaciju teksta i računanje udaljenosti između tih vektora. Zatim ćemo objasniti implementaciju RAG-a. Na kraju uspoređujemo performanse sustava s implementiranim RAG-om naspram samog jezičnog modela.

2. Jezični modeli

Umjetna inteligencija ima mnogo različitih područja, od kojih je jedno područje obrade prirodnog jezika. To područje ima cilj razumjeti i manipulirati ljudskim jezikom na smislen način. Ključna komponenta tog područja su jezični modeli, koje možemo definirati kao matematički model koji se oslanja na statističke metode i metode strojnog učenja za predviđanje vjerojatnosti niza riječi. Njegov je zadatak razumjeti tekst da može odgovoriti na upite o njemu, izvršiti zadatke koji zahtijevaju razumijevanje jezika, generirati tekst, prevoditi tekst i slično. Ti su modeli trenirani na velikim skupovima tekstualnih podataka, poput teksta iz knjiga, članaka, web stranica i drugih. Postoji više tipova jezičnih modela, od kojih su neki:

- N-gram modeli
- Rekurentne neuronske mreže
- Veliki jezični modeli

N-gram modeli su isključivo statistički modeli koji predviđaju sljedeću riječ u rečenici na temelju prethodnih n-1 riječi. Ako se u obzir uzima samo prethodna riječ, naziva se bigram, ako se dvije onda je trigram, te općenito ako se n-1 riječi je to n-gram model. N-gram je sekvenca uzastopnih n elemenata. Vjerojatnosti svakog n-grama se dobivaju na temelju korpusa teksta. Formulama možemo vjerojatnost rečenice "Danas pada kiša cijeli dan" za bigram prikazati kao:

$$P(\text{Svaki, dan, pada, kiša}) \approx$$

$$P(\text{Svaki} \mid \langle s \rangle) P(\text{dan} \mid \text{Svaki}) P(\text{pada} \mid \text{dan}) P(\text{kiša} \mid \text{pada}) P(\langle /s \rangle \mid \text{kiša})$$

Oznaka $\langle s \rangle$ označuje početak, a $\langle /s \rangle$ kraj rečenice.

Rekurentne neuronske mreže (eng. Recurrent Neural Network, RNN) su nadomjestile n-gram modele. Te mreže su bidirekcionalne, te se tako omogućuje cirkuliranje informacija unutar njih. Na taj se način informacije iz prethodnih koraka mogu upotrijebiti za donošenje odluka u trenutnom koraku. To se pokazalo vrlo dobrim pristupom kod raznih zadataka obrade prirodnog jezika, poput generiranja teksta, prevođenja teksta i sličnih. Najopćenitija arhitektura takvih mreža je potpuno rekurentna mreža (eng. Fully Recurrent Neural Network, FRNN). U takvoj su mreži izlazi svih neurona spojeni na ulaze svih neurona.

2.1. Veliki jezični modeli

Veliki jezični modeli predstavljaju najnaprednije jezične modele te se ističu po svojoj mogućnosti generiranja jezika opće namjene. To su umjetne neuronske mreže koje koriste arhitekturu transformera. U toj arhitekturi se tekst pretvara u tokene, od kojih je svaki pretvoren u korespondentni vektor korištenjem tablice ugrađivanja riječi (eng. Word Embedding Table). Prednost te arhitekture je nepostojanje rekurentnih dijelova, što zahtjeva manje treniranja od rekurentnih neuronskih mreža. Neuronske mreže takvih modela imaju vrlo veli broj parametara, koji broje u milijardama. Neki značajni modeli su:

- `ChatGPT 3.5 (Generative Pre-trained Transformer 3)` Model razvijen od strane OpenAI, koji ima 175 milijardi parametara te se može koristit za razne namjene, poput generiranja teksta, prevođenja, odgovaranja na pitanja i mnoge druge
- `Gemini` Niz modela koje je razvio Google
- `LLaMa (Large Language Model Meta AI)` Niz jezičnih modela koje je razvila tvrtka Meta

Do 2020. godine je fino podešavanje bila glavna metoda prilagođavanja modela specifičnim zadatcima. No veliki jezični modeli su pokazali mogućnost postizanja sličnih rezultata prilagođavanjem promptova koji im se daju na ulaz [1]. To je olakšalo rješavanje mnogih problema, pošto fino podešavanje modela može biti dugotrajan i potencijalno skup proces.

S takvim modelima možemo koristiti **Dohvatom-pojačano generiranje (eng. Retrieval-**

Augmented Generation, RAG) [2]. To je tehnika kojom poboljšavamo rezultate jezičnog modela dajući mu dodatne informacije, pošto modeli generalno nemaju dovoljno detaljno znanje iz mnogih područja. Veliki jezični modeli imaju mnoge nedostatke, poput predstavljanja lažnih informacija kada ne znaju odgovor, korištenje nepouzdanih izvora u generiranju odgovora, kreiranje netočnih odgovora radi krive interpretacije terminologije i slične, te nam korištenje RAG-a može pomoći u smanjivanju utjecaja tih problema na rezultate. Tako također kontroliramo izvor informacija koje koristi jezični model za naše potrebe, te na laki način modelu dajemo novije informacije koje su potencijalno nastale nakon perioda treniranja modela izbjegavajući tako skuplji i dugotrajniji proces finog podešavanja. To je vrlo popularan pristup, te se koristi za razne probleme, pa tako i probleme predviđanja budućih događaja, gdje je pokazao vrlo dobre rezultate [3].

3. Vektorska reprezentacija teksta

Pošto računala, te algoritmi strojnog učenja, rade s numeričkim vrijednostima, postavlja se pitanje kako tekst pretvoriti u numeričke vrijednosti. Taj proces pretvorbe teksta u numeričke reprezentacije se naziva vektorizacija, te je njen rezultat numerički vektor.

3.1. Bag of words

Jedna od najjednostavnijih metoda vektorizacije je bag-of-words (BoW) metoda. Ona se bazira na neporedanoj strukturi podataka u kojoj se bilježi broj ponavljanja svake riječi. Kao takva ne sadrži izvorni redoslijed riječi te iz tog razloga zanemaruje gramatiku. Na slici 3.1. je prikazan primjer teksta vektoriziranog ovom metodom. Od teksta

Slika 3.1. Primjer vektorizacije BoW metodom

"Iva voli jesti sladoled, Ivo također voli jesti sladoled ali i voće" dobivamo strukturu podataka {"Iva":1, "voli":2, "jesti":2, "sladoled":2, "Ivo":1, "također":1, "ali":1, "i":1, "voće":1} proizvoljnog poretka. Svaka riječ ima zabilježen broj ponavljanja. Vektorska reprezentacija unije više tekstova od kojih već imamo vektorske reprezentacije je unija tih reprezentacija kod kojih sumiramo brojeve ponavljanja riječi koje se pojavljuju u više vektora.

Ovdje se također primjenjuje tokenizacija. To je proces podjele teksta u manje jedinice koji se nazivaju tokeni. To su najčešće riječi, kao u danom primjeru na slici 3.1.

3.2. Naprednije tehnike vektorizacije

Iako je BoW dobra početna tehnika, ona ne uzima u obzir redoslijed riječi ni njihov kontekst.

Jedna od naprednijih tehnika je **Term Frequency-Inverse Document Frequency (TF-IDF)** tehnika. Ta tehnika evaluira koliko je relevantna riječ u dokumentu u kontekstu skupa dokumenata. Za riječ t, korpus D i dokument korpusa d, ta se vrijednost može izračunati korištenjem sljedećih izraza:

Frekvencija pojave riječi (TF):

$$\text{Tf}(t, d) = \frac{\text{broj\_pojavljivanja}(t, d)}{\text{ukupan\_broj\_riječi}(d)}$$

Inverzna frekvencija dokumenta (IDF):

$$\text{Idf}(t, D) = \log\left(\frac{\text{ukupan\_broj\_dokumenata}(D)}{\text{broj\_dokumenata\_koji\_sadrže\_riječ}(t, D)}\right)$$

TF-IDF:

$$\text{Tf-idf}(t, d, D) = \text{Tf}(t, d) \times \text{Idf}(t, D)$$

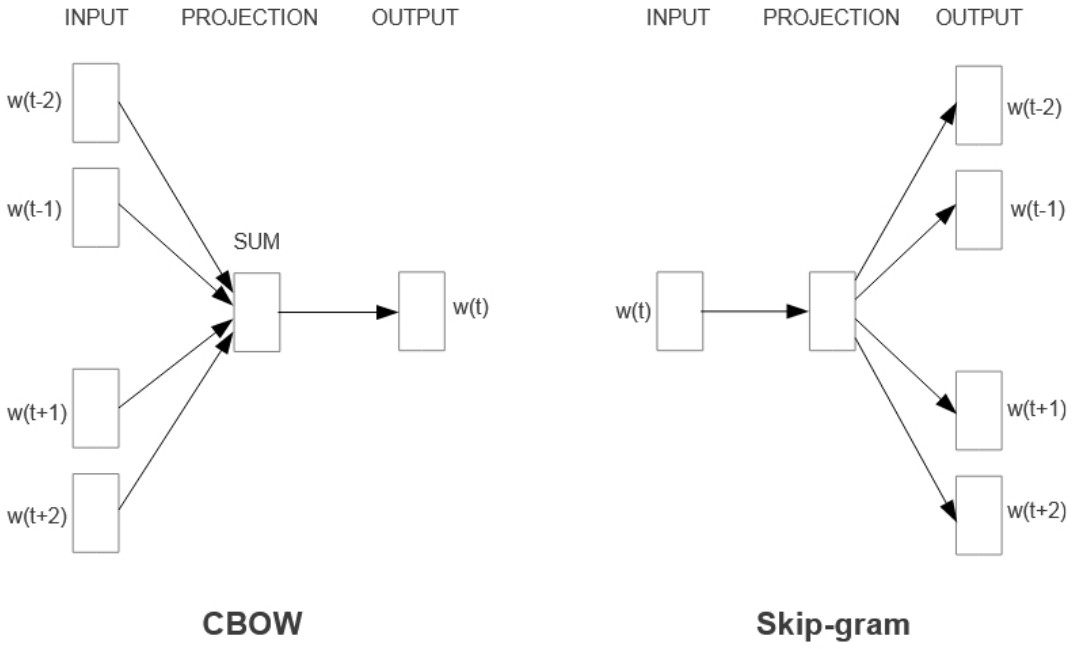
Gdje su:

- broj_pojavljivanja(t, d): broj pojavljivanja riječi t u dokumentu d.
- ukupan_broj_riječi(d): ukupan broj riječi u dokumentu d.
- ukupan_broj_dokumenata(D): ukupan broj dokumenata u korpusu D.
- broj_dokumenata_koji_sadrže_riječ(t, D): broj dokumenata u korpusu D koji sadrže riječ t.

Kao što je vidljivo iz izraza, TF-IDF vrijednost za riječ se izračunava množenjem dvije vrijednosti: frekvencije pojave riječi (eng. Term Frequency) i inverzne frekvencije doku-

menta (eng. Inverse Document Frequency). Frekvencija pojave riječi je vrijednost koju je moguće izračunati na nekoliko načina, od kojih je najjednostavniji brojanje pojava riječi u dokumentu. Inverzna frekvencija dokumenta je mjera koliko je riječ česta u cijelom skupu dokumenata. Jedan od mogućih načina izračunavanja je uzimanje ukupnog broja dokumenata što se dijeli s brojem dokumenata koji sadrže tu riječ, te uzimajući logaritam od tog omjera. Što je vrijednost bliža broju 0, riječ je češća. Vektor za određen tekst se sastavlja od dobivenih vrijednosti.

Metoda **Word2Vec** je naprednija metoda za dobivanje vektorskih reprezentacija riječi. To je skup srodnih modela baziranih na neuronskim mrežama s dva sloja koje su trenirane rekonstruirati lingvistički kontekst riječi. Ulaz mreže je veliki skup tekstualnih podataka, te vrača vektorski prostor u kojem svaka riječ u skupu podataka ima svoj vektor. Jednom kada je model treniran može detektirati sinonime ili nadopunjavati rečenice. Riječi sličnog značenja, poput "trčanje" i "hodanje" su male vektorske udaljenosti. Može koristiti dvije arhitekture: kontinuirani Bag-Of-Words (eng. Continuous Bag-OF-Words, CBOW) ili klizeći skip-gram (eng. Sliding skip-gram). CBOW nastoji predvidjeti trenutnu riječ na temelju konteksta, dok skip-gram model koristi trenutnu riječ za predviđanje okolnog prozora kontekstualnih riječi. Na slici 3.2. su prikazane te dvije arhi-



Slika 3.2. Arhitekture CBOW i skip-gram [4]

tekture. Vektori za svaku riječ imaju istu dužinu, tipično 300, te se generalno sastoje od

decimalnih brojeva.

Bidirectional Encoder Representations from Transformers (BERT) je jezični model koji se bazira na transformerskoj arhitekturi [5]. BERT kod generiranja vektora uzima u obzir dvosmjerni kontekst riječi. S time se razlikuje od metoda poput Word2Vec, koje uvijek istu riječ isto reprezentira. Na primjer, ako imamo rečenice "Sjeo je na banku rijeke" i "Otišao je u banku", BERT će svaku od ove dvije instance riječi "banku" prikazati drugačije. Taj model je donio značajan napredak u obradi prirodnog jezika.

3.3. Udaljenosti vektora

Kada imamo vektorske reprezentacije tekstova, trebamo neku metodu usporedbe tih vektora kako bismo mogli odrediti najrelevantnije tekstove. To određujemo korištenjem vektorske udaljenosti, te postoji više načina njenog izračuna.

Jedna od metoda je **kosinusova udaljenost**. Za njen nam je izračun potrebna kosinusova sličnost. Kosinusna sličnost mjeri kosinus kuta između dva vektora $\mathbf{A} = (a_1, a_2, \ldots, a_n)$ i $\mathbf{B} = (b_1, b_2, \ldots, b_n)$, kao na slici 3.3. gdje je taj kut označen s θ. Izračunava se formulom:

$$\mathrm{S}_C(\mathbf{A}, \mathbf{B}) = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||||\mathbf{B}||}$$

gdje je:

- $\mathbf{A} \cdot \mathbf{B}$ skalarni produkt (dot product) vektora $\mathbf{A}$ i $\mathbf{B}$, te se računa kao

$$\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^{n} A_i B_i$$

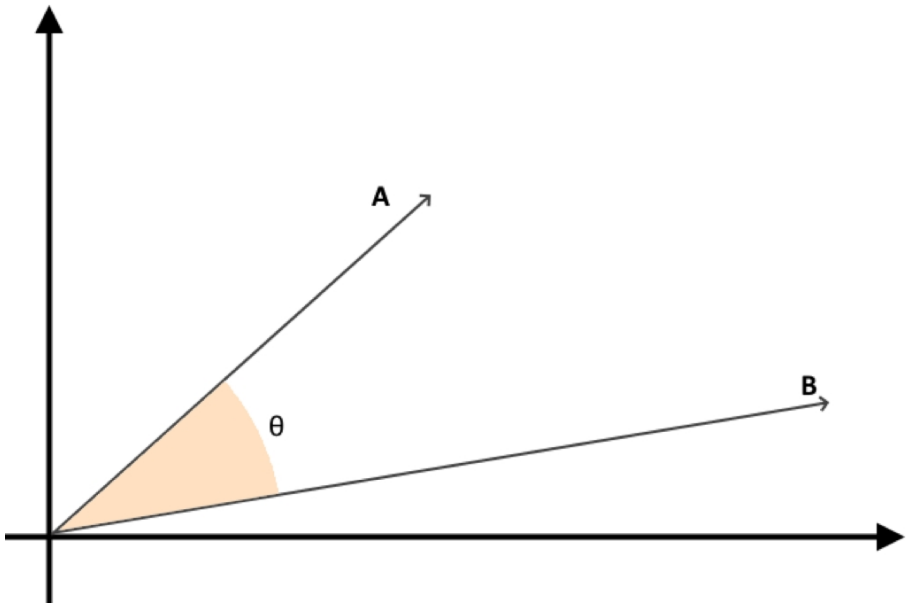
- $||\mathbf{A}||$, $||\mathbf{B}||$ Euklidske norme (duljine) vektora $\mathbf{A}$, $\mathbf{B}$.

Kosinusova udaljenost je:

$$\mathrm{D}_C(\mathbf{A}, \mathbf{B}) = 1 - \mathrm{S}_C(\mathbf{A}, \mathbf{B})$$

Raspon vrijednosti te mjere je [0,2], no taj se interval u praksi često skalira na [0,1].

Kvadrirana Euklidova udaljenost je još jedan mogući način računanja udaljenosti. Neka imamo dva vektora $\mathbf{A} = (a_1, a_2, \ldots, a_n)$ i $\mathbf{B} = (b_1, b_2, \ldots, b_n)$, kvadrirana Euklidova



Slika 3.3. Primjer dva vektora **A** i **B** i njihove kosinusove sličnosti θ u dvodimenzionalnom prostoru

udaljenost između njih se računa formulom:

$$D^2(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{n} (a_i - b_i)^2$$

Ta udaljenost je bilo koja vrijednost veća od nula ili jednaka nuli. Ako je vrijednost jednaka nuli zaključujemo da su vektori jednaki.

Ta je udaljenost vrlo slična **Euklidovoj udaljenosti**, uz razliku da se kod Euklidove udaljenosti uzima korijen zbroja, konkretno:

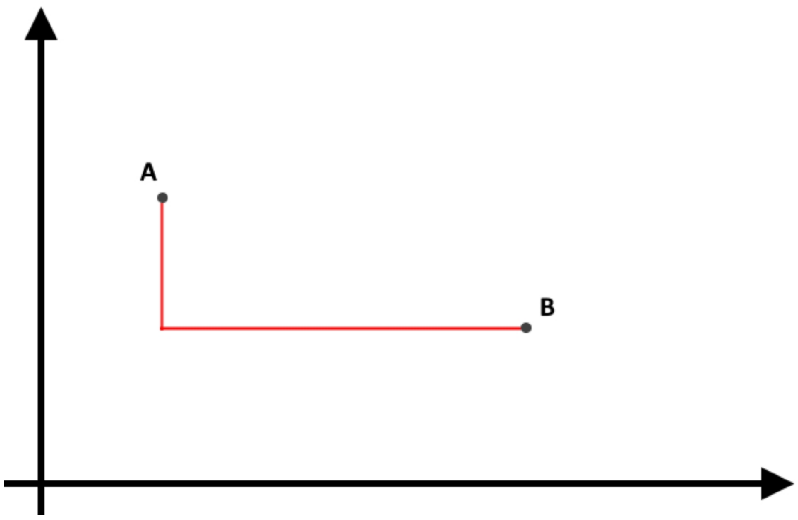
$$D(\mathbf{A}, \mathbf{B}) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Iz tog razloga kvadrirana Euklidova udaljenost može biti pogodnija u nekim slučajevima pošto je jednostavnija i brža za izračun.

Manhattanska udaljenost mjeri udaljenost 2 točke u mreži kvadrata. Ako imamo dva vektora $\mathbf{A} = (a_1, a_2, \ldots, a_n)$ i $\mathbf{B} = (b_1, b_2, \ldots, b_n)$, Manhattanska udaljenost između njih se računa formulom:

$$D_{\text{Manhattan}}(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{n} |a_i - b_i|$$

Primjer takve udaljenosti u dvodimenzionalnom prostoru je prikazan na slici 3.4.



Slika 3.4. Primjer dva vektora **A** i **B** i njihove Manhattan udaljenosti prikazane crvenom linijom u dvodimenzionalnom prostoru

4. Implementacija

U ovom poglavlju razmatramo pristup korišten kod implementacije RAG-a. Sustav na ulaz prima tekst te na temelju tog teksta traži najsličnije tekstove u vektorskoj bazi podataka. Ti tekstovi služe kao kontekst koji jezičnom modelu pomaže u procjeni ako je dani tekst na ulazu futurološki signal. Ako sustav procijeni da je taj tekst signal, dijeli se na više manjih tekstova koji se zatim pohranjuju u bazu podataka. Na takav način sustav dobiva i čuva novo znanje. Ono što nije signal se ne pohranjuje pošto je sustav iz dosadašnjeg znanja mogao zaključiti da to nije signal, te s time pohrana toga ne bi utjecala na rad sustava.

4.1. Vektorska baza podataka

Vektorska baza podataka je baza koja čuva vektorske reprezentacije tekstova kao i same tekstove koje u nju pohranimo, te po potrebi proizvoljne metapodatke. Ona igra ključnu ulogu kod implementacije RAG-a. Vektorska reprezentacija teksta omogućuje brzu i efikasnu usporedbu različitih tekstova, pošto se lako može izračunati njihova međusobna udaljenost.

U implementaciji se koristi FAISS (Facebook AI Similarity Search) vektorska baza podataka. To je brza baza optimizirana za CPU i GPU koja može raditi s vrlo velikim skupovima podataka koji ne stanu u memoriju zahvaljujući mogućnosti rada s particioniranim indeksima [6]. Baza se lako koristi, te se u nju lako mogu dodavati novi tekstovi. Također omogućuje definiranje algoritama za pretragu i ugrađivanje teksta (eng. embedding), te je slobodna svima za korištenje. Iz tih se razloga pokazala kao dobar izbor za bazu podataka.

U ovom slučaju se koristi embedding model "thenlper/gte-small". To je mali model koji ima 33.4 milijuna parametara, što ga čini brzim, ali i nešto lošijim u usporedbi s nekim

većim modelima.

Kao strategija udaljenosti se koristi kosinusova udaljenost, opisana u poglavlju 3.3.

Baza podataka se može inicirati iz proizvoljnog broja dokumenata, te pohraniti na vanjsku memoriju. Nama je pohrana bitna pošto želimo sačuvati to znanje te u nju dodavati novo. Ta baza služi kao način na koji jezičnom modelu dajemo kontekst o novijim događajima koji su se dogodili nakon što je model treniran.

4.2. Podjela teksta

Tekstove koje pohranjujemo u bazu podataka dijelimo da više dijelova. To je korisno iz više razloga, od kojih je jedan preciznije pretraživanje. Možemo pronaći relevantnije dijelove teksta bez potrebe za uzimanjem teksta cijelog originalnog članka. Također su vektorske reprezentacije bolje, pošto je tekst manje heterogen, tj. više specifičan. Pošto jezični model kojeg koristimo ima ograničenje od 16385 tokena, također je korisno imati manji tekst kao kontekst.

Za podjelu teksta koristimo rekurzivno dijeljenje (eng. recursive chunking). Ta metoda koristi listu separatora sortiranu od najbitnijih separatora prema najmanje bitnima, konkretno u ovoj implementaciji one prikazane na slici 4.1. [7]. Rekurzivno dijeljenje dijeli tekst, te nakon toga rekurzivno sve nastale dijelove sve dok ne dođe do željene veličine dijelova teksta.

```
MARKDOWN_SEPARATORS = [
    "\n#{1,6} ",
    "```\n",
    "\n\\*\\*\\*+\n",
    "\n---+\n",
    "\n___+\n",
    "\n\n",
    "\n",
    " ",
    "",
]
```

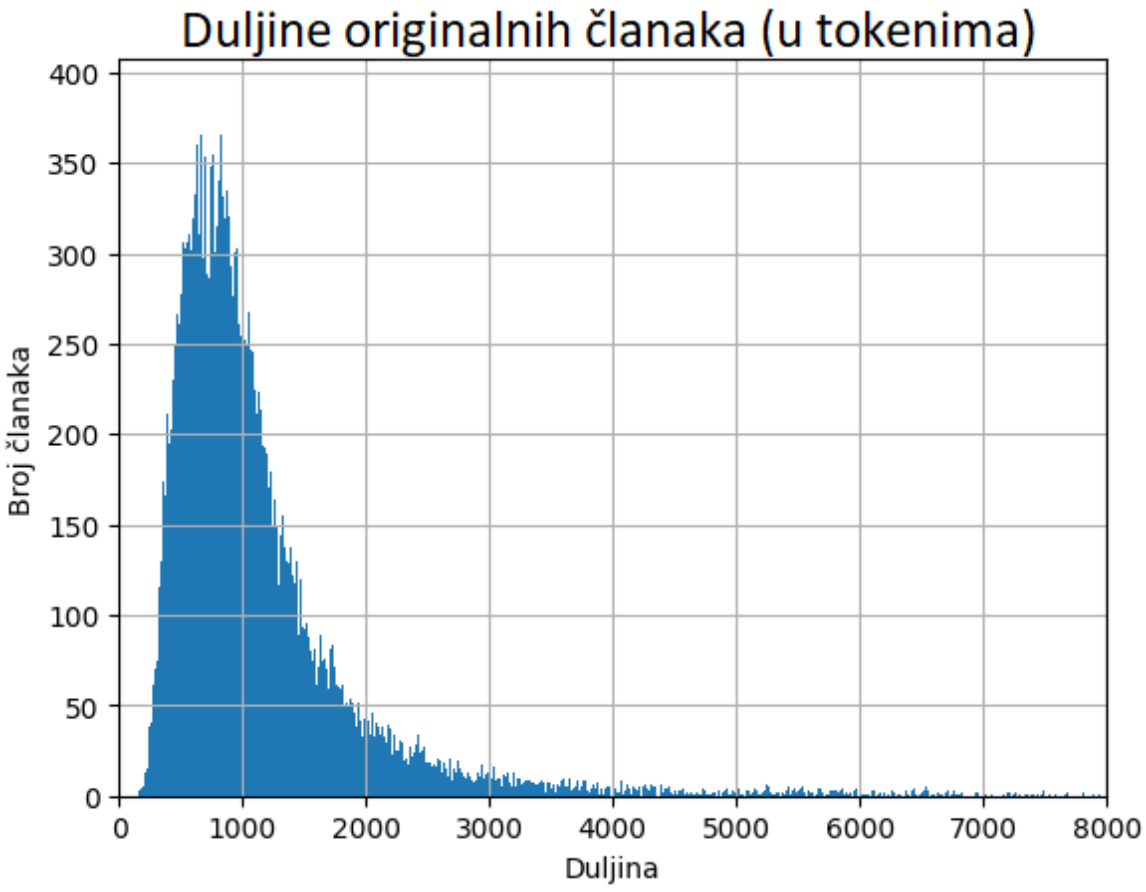
Slika 4.1. Separatori za rekurzivno dijeljenje

Dva bitna parametra kod rekurzivnog dijeljenja su veličina dijela (eng. chunk size) i veličina preklapanja (eng. chunk overlap). Veličina dijela teksta određuje koliko najviše

veliki trebaju biti rezultirajući dijelovi, a veličina preklapanja kolika treba biti veličina preklapanja dijela s njegovim susjednim. Na taj se način nastoji umanjiti prekidanje informacije, pošto će se tekst vrlo često podijeliti unutar rečenica. Konkretno u našoj implementaciji je veličina dijela 512 tokena, a veličina preklapanja 51.

Također je potrebno definirati tokenizer. To je alat koji dijeli rečenice na manje dijelove. Ti dijelovi mogu biti riječi, fraze, podriječi i slično. Tokenizer koji je ovdje korišten je baziran na modelu "Nexusflow/Starling-LM-7B-beta" [8]. To je model koji ima 7 milijarda parametara, te je najbolji model svoje veličine po performansama [9] u vrijeme pisanja ovog rada. Iako postoje veći i bolji modeli, ovdje se ograničavamo na ovakav model pošto zahtjeva manje resursa.

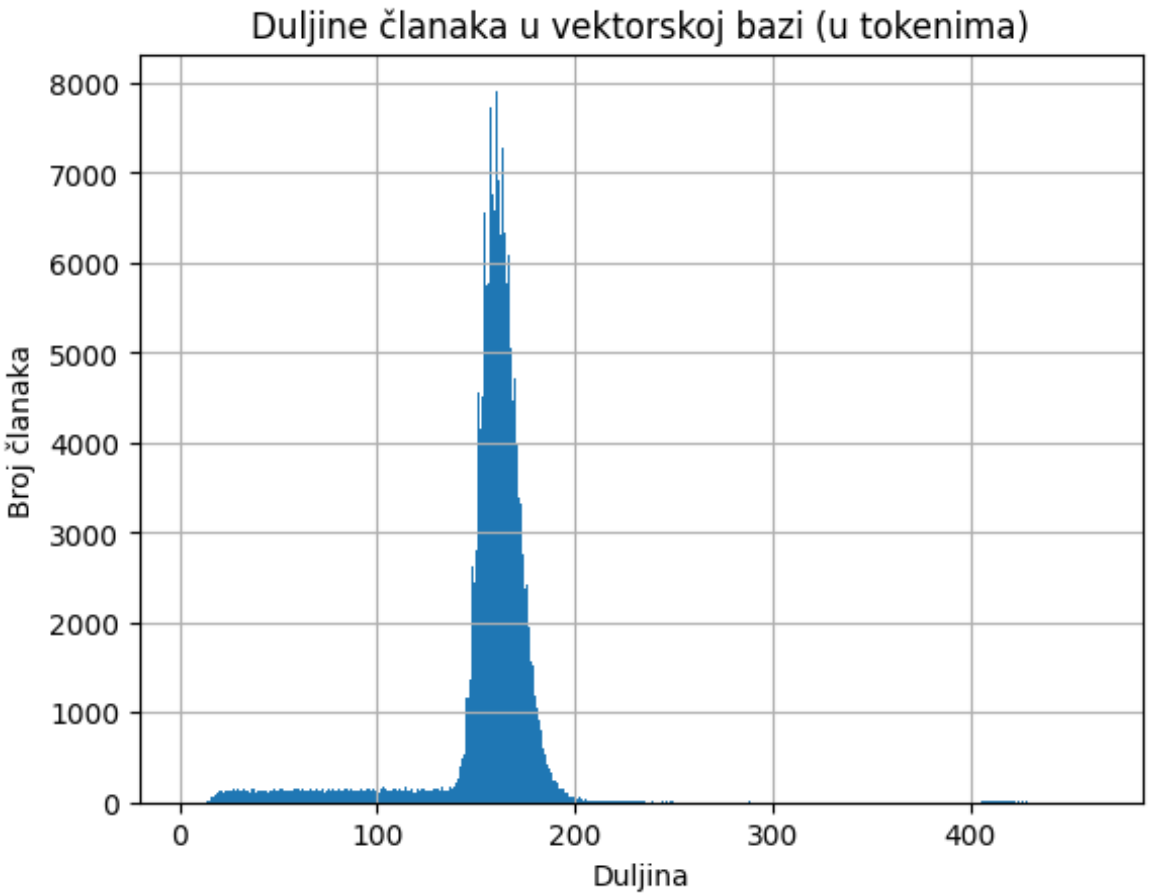
Primjer učinkovitosti dijeljenja teksta možemo prikazati na skupu podataka opisanom u poglavlju 5.1. Na slici 4.2. je prikazan graf koji prikazuje distribuciju duljina originalnih



Slika 4.2. Duljine članaka u tokenima

članaka u tokenima.

Nakon što su ti članci prošli kroz funkciju za dijeljenje teksta, distribucija izgleda kao na slici 4.3. Kao što je vidljivo iz grafova, pohranjeni dijelovi teksta su mnogo manji, te ih je



Slika 4.3. Duljine članaka u tokenima

mnogo više.

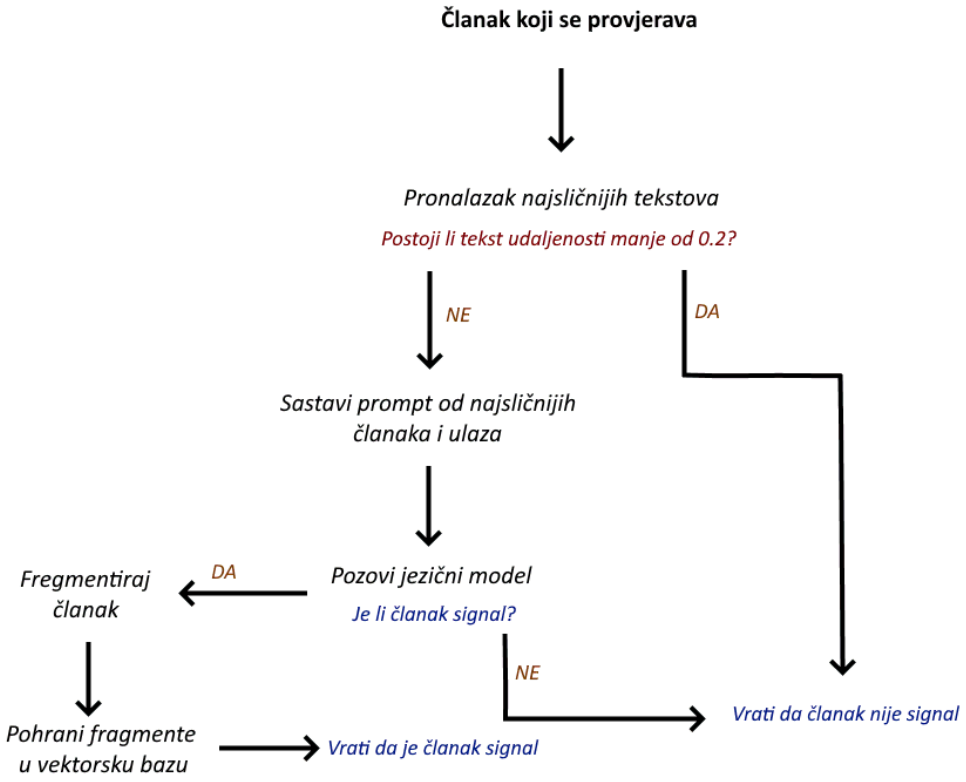
4.3. Proces generiranja odgovora

Sustav na početku kao ulaz prima članak za kojeg se želi provjeriti ako je futurološki signal. Konkretno, sustav je zamišljen da kao ulaz prima tekstualni sadržaj članka. Nakon toga se pretražuje vektorska baza podataka u kojoj se traže najsličniji tekstovi koji će služiti kao kontekst u odlučivanju. Cilj tog pretraživanja je pronaći tekstove koji govore o istim temama kao i članak kojeg sustav provjerava te tako dati što bolji kontekst jezičnom modelu. Broj tekstova koji će se vratiti je proizvoljan parametar koji se može mijenjati. S jedne strane, mali broj tekstova može dovesti do toga da model ne dobije relevantan kontekst ili neku bitnu informaciju koja bi bila presudna kod odlučivanja. No ni prevelik broj vraćenih tekstova nije idealan pošto to može loše utjecati na sposobnost modela da identificira bitne informacije [10]. U ovom radu se ne istražuje optimalna vrijednost tog parametra, nego je to vrijednost koja se može slobodno mijenjati.

U svrhu smanjenja slanja zahtjeva na jezični model, među nađenim tekstovima se provjerava ako je vektorska udaljenost ikojeg teksta i teksta koji je na ulazu manja od 0.2, te ako u bazi takav tekst postoji se zaključuje da ulazni tekst nije signal. To se zaključuje pošto tako mala vektorska udaljenost korespondentnih vektora implicira da tekstovi imaju vrlo sličan sadržaj. To je u ovom slučaju također proizvoljna vrijednost u čije se optimiziranje ne ulazi u sklopu ovog rada.

Dohvačeni tekstovi iz baze podataka postaju sastavni dio prompta za jezični model.

Cijelokupni dijagram procesa je prikazan na slici 4.4. Prompt koji se daje jezičnom mo-



Slika 4.4. Dijagram radnja sustava

delu je ključno dobro dizajnirati kako bi model dao što bolje rezultate. No to često nije jednostavno. Jedna moguća tehnika za to je korištenje jezičnog modela da napiše prompt na temelju naših zahtjeva. Model možemo ispitivati da mjenja prompt sve dok ne dobijemo rezultat kakav želimo. Također je bitno na koji način se dodaje kontekst u prompt, pošto položaj konteksta može značajno utjecati na sposobnost modela da identificira bitne informacije [10].

Prompt na kojem je baziran ovaj sustav je prikazan na slici 4.5. U taj prompt se ubacuje

```
prompt_template = """Anomalies are considered unusual or exceptional instances that deviate
 from the norm or expected patterns. Please analyze the provided text and respond
with 'yes' if it exhibits positive anomalies, or 'no' if it does not. In this context,
positive anomalies refer to information that is groundbreaking, transformative,
or indicative of significant technological advancement and is expected to have longterm effects.
Now, analyze the following text:
"{article}"
"""
```

Slika 4.5. Prompt za jezični model

kontekst te se to predaje jezičnom modelu. U ovoj implementaciji se koristi jezični model ChatGPT 3.5 turbo, te mu se pristupa preko API-ja.

5. Eksperiment

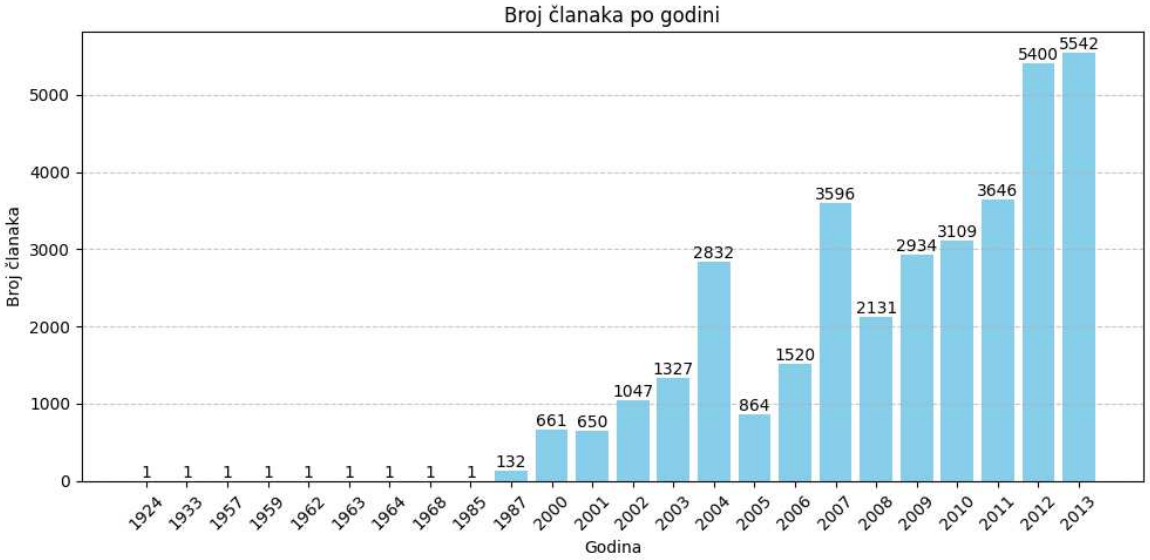
5.1. Skup podataka

U eksperimentu se koristi skup podataka koji se sastoji od 35400 zapisa u json formatu. Taj se skup podataka koristi za testiranje koliko točno sustav prepoznaje futurološke signale, te kao takav treba za svaki članak bilježiti ako je signal. Svaki zapis je skup atributa o jednom članku, a ti atributi su:

- `id`: Jedinstveni identifikator članka
- `title`: Naslov članka
- `date_published`: Datum objavljivanja članka
- `text`: Sadržaj članka
- `link`: URL link do originalnog članka
- `gpt_signal`: Procjena modela ChatGPT 3.5 o tome ako je članak futurološki signal
- `topics`: Popis tema članka
- `growth`: Podatci o rastu interesa tema članka kroz vrijeme
- `signal`: Procjena ako je članak futurološki signal na temelju rasta

Da se dobije što preciznija procjena ako je članak futurološki signal, koristila su se dva načina procjene, prvo s modelom ChatGPT 3.5, te onda koristeći interes kroz vrijeme za glavne teme kojima se bavi članak.
ChatGPT se koristio tako da mu se kao prompt stavi članak te uputa da procijeni ako se ono što se raspravlja u članku pokazalo futurološkim signalom. Da model to može pro-

cijeniti, korišteni su stariji članci za koje se lako može procijeniti koliko je ono što se u njima raspravlja bilo utjecajno, pa je najmlađi članak star oko 10 godina (Slika 5.1.).

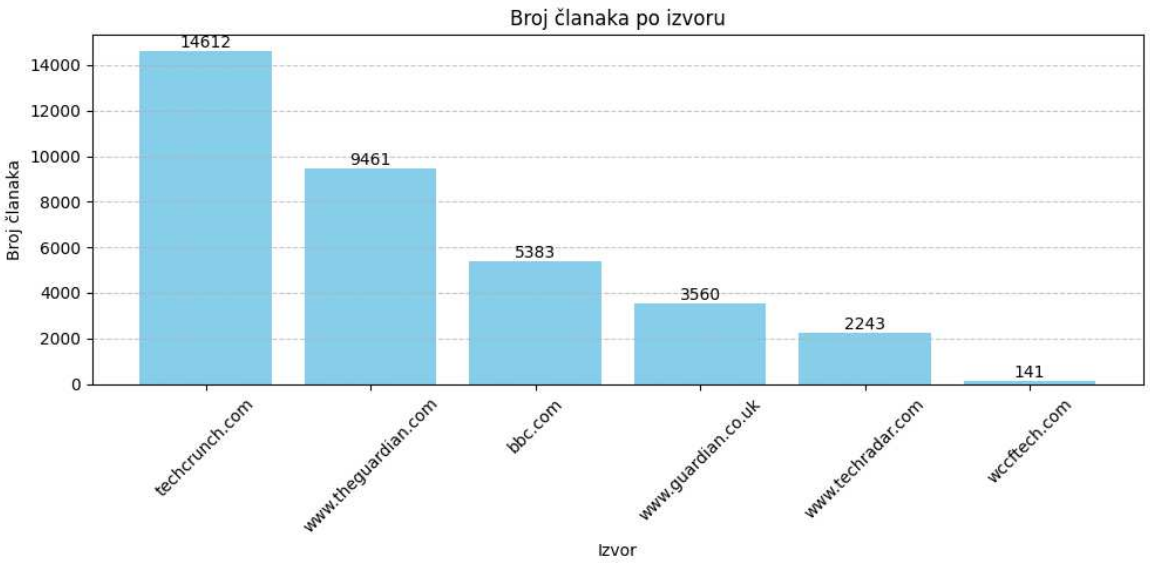


Slika 5.1. Godine objavljivanja članaka u skupu podataka

Da se poveća preciznost procjene, korišteni su podatci o rastu interesa kroz vrijeme za teme članka. Same teme su određene koristeći ChatGPT 3.5, te su podatci o interesima dobiveni koristeći Google Trends, gledajući pritom interese 3, 5 i 10 godina nakon objave članka.

U rezultirajućem skupu podataka, 21.28 % članaka je signal prema rastu a 10.88 % je signal prema ChatGPT-u, te je svaki članak koji je signal prema ChatGPT-u također signal prema rastu. Ako razmatramo samo članke koji imaju oba indikatora signala jednaka, dobivamo da je 12.14 % njih signal.

Članci su dohvaćeni sa 6 web stranica, prikazano na slici 5.2. Poveznice do članaka su dobivene iz dva skupa podataka [11, 12], i rezultata pretraživanja na tražilici Google. Početno je dohvaćeno približno 270000 članaka, no radi raznih ograničenja, prvenstveno financijskih i vremenskih, taj se skup članaka smanjio na 35400 prije početka analize. Smanjivao se prema kriteriju vektorske udaljenosti, gdje su se odbacili članci koji su najudaljeniji, pošto je sadržaj takvih članaka često bio besmislen. Za prvih 20000 članaka iz rezultirajućeg skupa podataka 2D prikaz vektora bi izgledao kao na slici 5.3. nakon njihove podjele na način opisan u poglavlju 4.2.



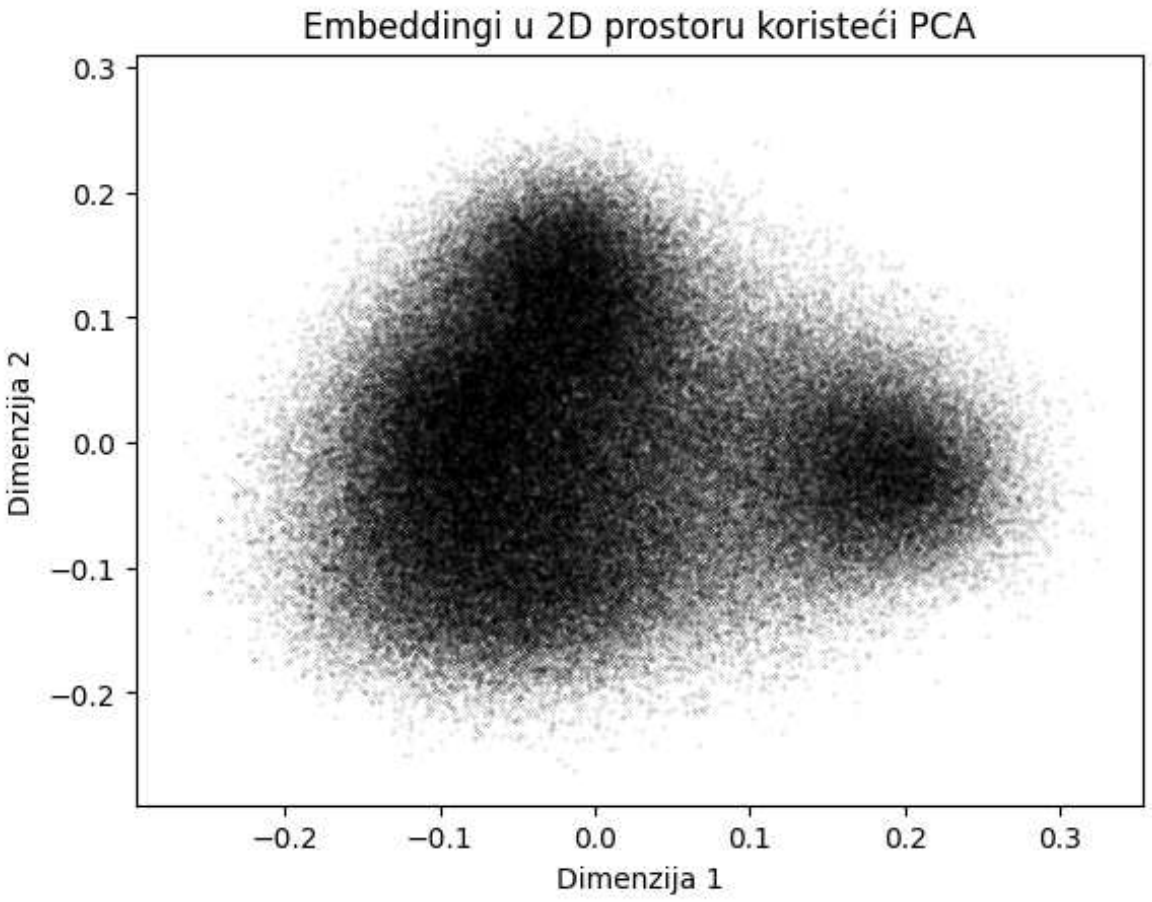
Slika 5.2. Izvor članaka u skupu podataka

5.2. Struktura eksperimenta

Eksperimentom testiramo koliko bolje procjene futuroloških signala dobivamo korištenjem tehnika dohvatom-pojačanog generiranja, te eksperiment kao takav ima dva dijela: testiranje sustava bez RAG-a i testiranje sustava s RAG-om. Cilj eksperimenta je usporediti dobivene rezultate te utvrditi ako RAG poboljšava točnost procjene.

U prvom dijelu eksperimenta koristimo samo jezični model, kojem dajemo prompt s člankom te uputama da procijeni ako je dani članak signal. Pošto radimo s člancima koji su objavljeni prije najmanje 10 godina, također trebamo jezičnom modelu dati uputu da ne uzima u obzir informacije koje ima nakon datuma objave članka, pošto je model treniran do 2021. godine, te da također zanemari informacije koje ima o događajima do mjesec dana prije datuma objave članka. Bez tog zanemarivanja bi simulirali model koji zna sve do tog datuma, tj. model kod kojeg nema potrebe implementirati RAG. Prompt koji se koristi za prvi dio eksperimenta je prikazan na slici 5.4.

Za drugi dio eksperimenta u prompt dodajemo nađeni kontekst. Kontekst se dobiva iz vektorske baze podataka na temelju sličnosti s člankom kojeg provjeravamo kako je to opisano u prethodnim poglavljima, te ako nađemo tekst s vektorskom udaljenosti manjom od 0.2 vraćamo da članak nije signal bez ispitivanja jezičnim modelom. I dalje moramo dati jezičnom modelu uputu da ignorira znanja nakon datuma objave članka i do mjesec dana prije objave, pa je s time i prompt vrlo sličan prethodnom, te je prikazan na slici 5.5. Skup podataka dijelimo na dva djela. Prvih 20000 članaka pohranjujemo u



Slika 5.3. Dvodimenzionalni prikaz vektora

vektorsku bazu podataka koju sustav dobiva te u nju dodaje članke koje smatra signalima. Iako se u bazu dodaju samo signali, ovdje je opravdano dodati svih 20000 članaka pošto, u teoriji, članci koji nisu signali ne bi trebali imati utjecaj na rezultate. Na takav način simuliramo sustav koji je aktivan duže vrijeme. Na ostalih 15400 članaka testiramo sustav iterirajući kroz njih te testirajući svaki. Pošto imamo dva atributa za signal, "signal" i "gpt_signal", bilježimo podudaranje rezultata sustava posebno sa svakim od ta dva atributa. Sustav kao kontekst uzima dva najbliža teksta iz vektorske baze podataka. Iako bi više tekstova dalo potencijalno bolji rezultat ovdje smo se ograničili na dva iz više

```
prompt_template_no_rag="""Anomalies are considered unusual or exceptional instances that deviate from
the norm or expected patterns. Please analyze the provided text and respond with 'yes' if it exhibits
positive anomalies, or 'no' if it does not. In this context, positive anomalies refer to information
that is groundbreaking, transformative, or indicative of significant technological advancement and
is expected to have longterm effects. The answer should only be 'yes' or 'no'. Article was published
on {date}, disregard any information you have after that date or month prior in your prediction.
Now, analyze the following text:
"{article}"""
```

Slika 5.4. Prompt za prvi dio eksperimenta

razloga, poput niže cijene eksperimenta te činjenice da bi s više tekstova mogli doći u situaciju gdje se premaši 16385 tokena u promptu, što je ograničenje korištenog jezičnog modela.

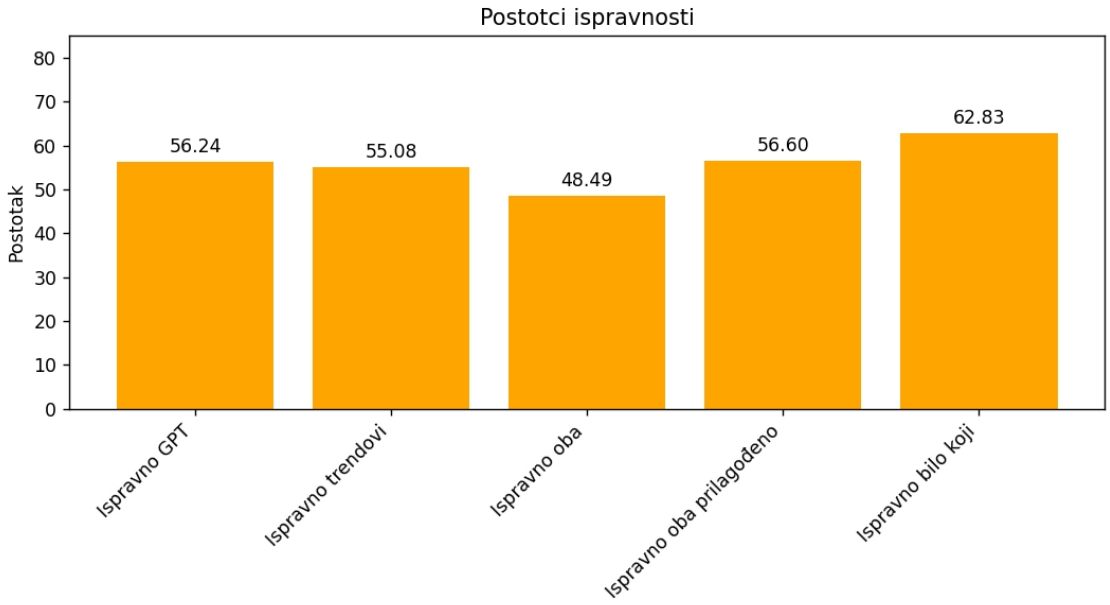
```
prompt_template="""Anomalies are considered unusual or exceptional instances that deviate from
the norm or expected patterns. Please analyze the provided text and respond with 'yes' if it exhibits
positive anomalies, or 'no' if it does not. In this context, positive anomalies refer to information
that is groundbreaking, transformative, or indicative of significant technological advancement and
is expected to have longterm effects. The answer should only be 'yes' or 'no'. Article was published
on {date}, disregard any information you have after that date or month prior in your prediction.
Now, analyze the following text:
"{article}
Here is the context that may help you, this has happened already: {context}"""
```

Slika 5.5. Prompt za drugi dio eksperimenta

Eksperiment je izvršen u Google Collab okruženju, koristeći T4 GPU.

5.3. Rezultati eksperimenta

Prvo ćemo razmotriti dobivene rezultate u prvom dijelu eksperimenta, tj. kada se ne koristi RAG. Dobiveni rezultati su prikazani grafički na slici 5.6. Stupac "Ispravno GPT"

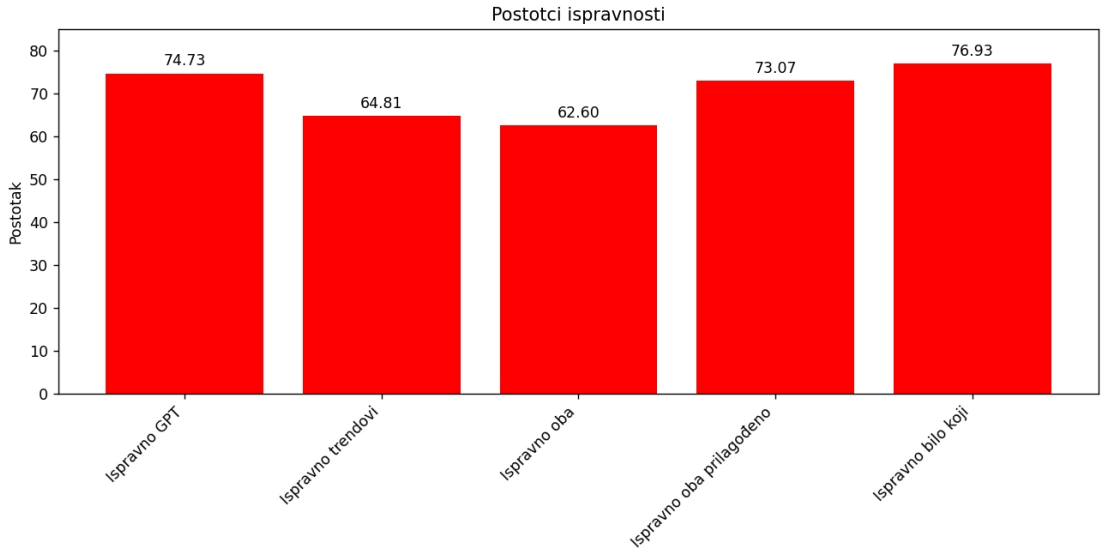


Slika 5.6. Ispravnosti procjena za prvi dio eksperimenta

prikazuje točnost ako se uspoređuju dobivene procjene o signalu s poljem "gpt_signal" u skupu podataka. Ta vrijednost je 56.24 %. Na sličan način, stupac "Ispravno trendovi" prikazuje tu točnost kad se uspoređuje s atributom "signal", tj. procjenom signala na temelju rasta interesa pojmova u člancima. Ta vrijednost je vrlo slična, 55.08 %. Pošto neki

članci nemaju iste vrijednosti atributa "signal" i "gpt_signal", korisno je razmotriti kolika je točnost ako signalima smatramo samo one članke kod kojih oba atributa pokazuju da je članak signal. Tu vrijednost pokazuje stupac "Ispravno oba", te je ta vrijednost 48.49 %. Ali pošto se u mnogim člancima razlikuju vrijednosti atributa "signal" i "gpt_signal", razumno bi bilo kod izračuna postotka točnosti uzimati u obzir samo članke s podudarnim vrijednostima. Od 15400 članaka njih 13192 ima ta dva atributa podudarna. Korigirana vrijednost se prikazuje u stupcu "Ispravno oba prilagođeno" čija je vrijednost 56.60 %. Zadnji stupac "Ispravno bilo koji" predikciju smatra ispravnom ako se predikcija poklapa s bilo kojim od dva atributa, te je ta vrijednost 62.83 %.

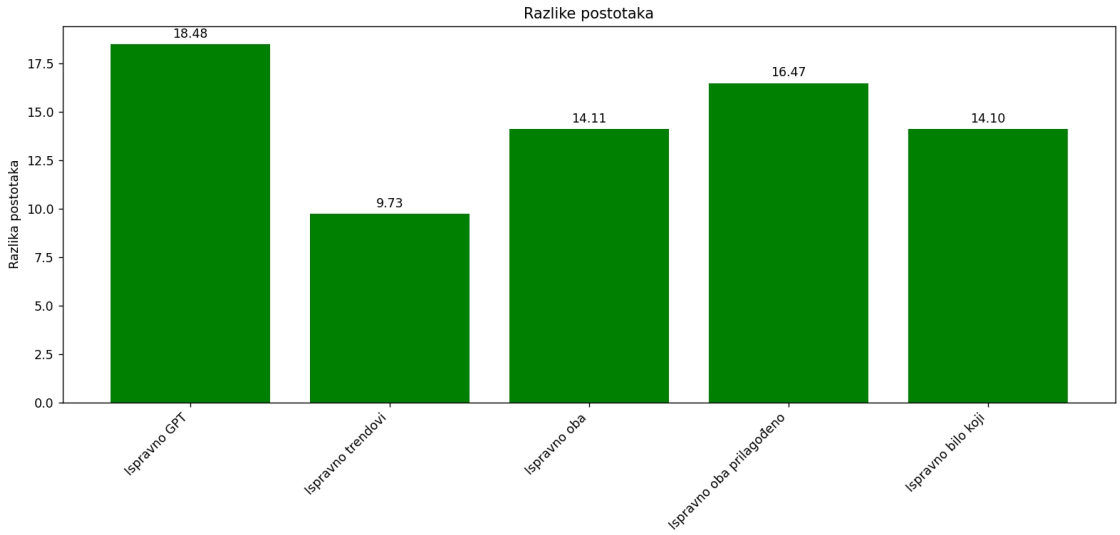
Iz danih postotaka je vidljivo da sustav baziran samo na jezičnom modelu, konkretno modelu ChatGPT 3.5 turbo, može procijeniti ako je članak futurološki signal samo malo bolje od slučajne šanse.
Iduće razmatramo rezultate drugog dijela eksperimenta, u kojem koristimo RAG. Dobivene točnosti su prikazane na grafu na slici 5.7.



Slika 5.7. Ispravnosti procjena za drugi dio eksperimenta

Graf ima stupce korespondentne grafu iz prvog dijela eksperimenta. Pa prema tome, vrijednost stupca "Ispravno GPT" je 74.73 %, stupca "Ispravno trendovi" je 64.81 %, "Ispravno oba" je 62.6 %, "Ispravno oba prilagođeno" je 73.07 % te "Ispravno bilo koji" je 76.93 %.
Iz tih postotaka vidimo značajno povećanu točnost modela kada se koristi RAG. Konkretno, razlike u postotcima su prikazane na slici 5.8., gdje svaki stupac prikazuje razliku

korespondentnog stupca u grafovima na slikama 5.7. i 5.6.

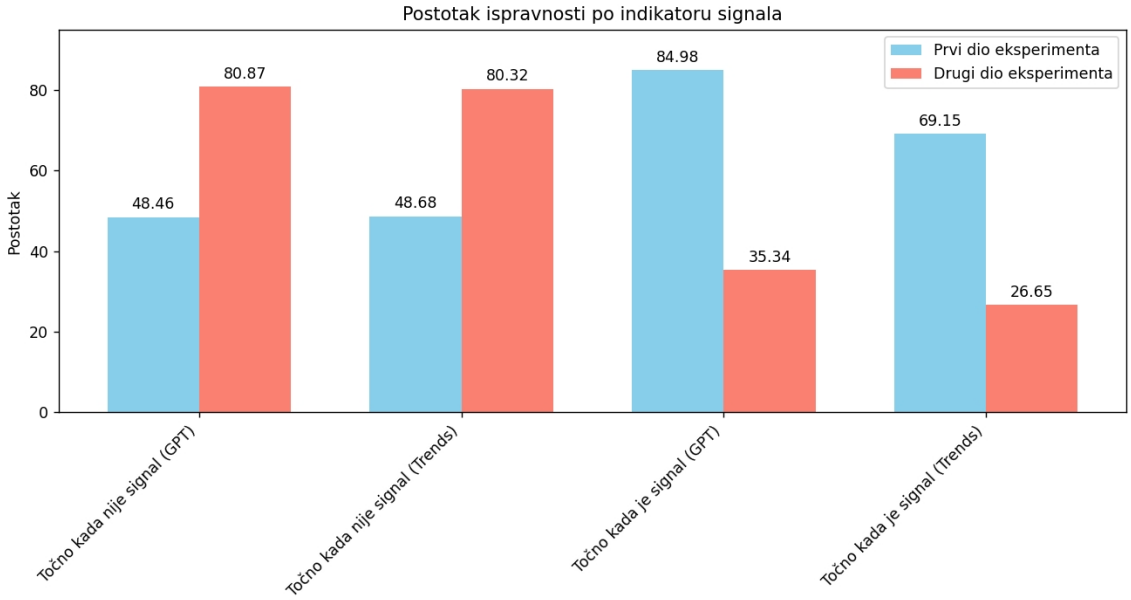


Slika 5.8. Razlike u točnosti prvog i drugog dijela eksperimenta

Korištenjem RAG-a su se postigli značajno bolji rezultati, najviše kada se uspoređivalo atribut skupa podataka "gpt_signal" s dobivenom procjenom modela. Najmanje poboljšanje je bilo kod usporedbe procjena modela s "signal" atributom. To bi mogla biti naznaka da je taj atribut manje točan u procjeni ako je članak zapravo futurološki signal. Također su vrijednosti točnosti "Ispravno oba prilagođeno" i "Ispravno bilo koji" puno bliže.

Korisno je razmotriti u kojim slučajevima je korištenje RAG-a poboljšalo rezultate. To je vidljivo grafički na slici 5.9. Prvi stupac prikazuje postotak članaka za koje je sustav odredio da nisu futurološki signali onda kada prema atributu "gpt_signal" nisu ni bili, gdje prvi stupac pokazuje rezultate za prvi dio eksperimenta, a drugi za drugi dio. Vidimo da se korištenjem RAG-a značajno poboljša točnost sustava kod klasificiranja članaka koji nisu signali, s 48.46 % na 80.87 % gledajući indikator signala "gpt_signal". S druge strane, klasificiranje članaka koji jesu signali se značajno pogorša, padajući s 84.98 % na 35.34 % kod istog indikatora. Sustav i dalje daje značajno veću vjerojatnost klasificiranja kao signal članka označenog u skupu podataka kao signal naspram članka koji nije klasificiran kao signal (35.34 % naspram 19.13 %). Sustav za sve kategorije članaka ima manju vjerojatnost procijeniti da je članak signal.

Postoji nekoliko mogućih objašnjenja smanjenja točnosti klasificiranja signala. Pošto je sam skup podataka rađen na automatiziran način, vrlo je vjerojatno da sam skup sadrži



Slika 5.9. Usporedbe točnosti

pogrešne klasifikacije. Također, pošto je ovo simulacija, jezičnom modelu se daje uputa da ignorira neka znanja koja ima. Jezični modeli često loše slijede takve upute, te bi to ovdje mogao biti vrlo bitan nedostatak. Kod ovakvih problema također sam prompt igra ključnu ulogu, te bi neki drugi prompt potencijalno mogao dati značajno bolje rezultate.

6. Zaključak

U ovom smo radu problemu prepoznavanja futuroloških signala pristupili koristeći veliki jezični model te tehnike dohvatom pojačanog generiranja kako bismo mu dali potreban kontekst.

Opisali smo osnovne koncepte jezičnih modela i vektorske reprezentacije teksta.

Zatim smo ukratko opisali samu implementaciju sustava. Konkretno, na koji način sustav koristi jezične modele, kako generira odgovor te na koji način dobiva relevantan kontekst.

Usporedili smo performanse ovakvog sustava naspram samog jezičnog modela koji ne dobiva nikakav kontekst. Na korištenom skupu podataka smo dobili povećanje postotka točnih klasifikacija od 16.47 % (ako uzimamo u obzir rezultate kod kojih se indikatori signala u skupu podataka poklapaju).

Iako je implementirana strategija pokazala značajno poboljšanje na korištenom skupu podataka, mogućnost sustava da signale točno prepozna kao signale se značajno pogoršala, te poboljšanje točnih klasifikacija proizlazi iz toga što sustav točnije prepoznaje članke koji nisu signali. To je značajan nedostatak ovakvog sustava, no i kao takav pokazuje značajan potencijal u rješavanju ovakvog problema.

Literatura

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, Dario Amode, "Language models are few-shot learners", https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf, 2020.

[2] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks", https://arxiv.org/pdf/2005.11401.pdf, 2020.

[3] Danny Halawi, Fred Zhang, Chen Yueh-Han, Jacob Steinhardt, "Approaching human-level forecasting with language models", https://arxiv.org/abs/2402.18563v1, 2024.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Efficient estimation of word representations in vector space", https://arxiv.org/abs/2307.03172, 2013.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", https://arxiv.org/abs/1810.04805v2, 2018.

[6] "Faiss dokumentacija", https://python.langchain.com/v0.1/docs/integrations/vectorstores/faiss/, [Online;].

[7] A. Roucher, “Advanced rag on hugging face documentation using langchain”, https://huggingface.co/learn/cookbook/advanced_rag, [Online;].

[8] B. Zhu, E. Frick, T. Wu, H. Zhu, K. Ganesan, W.-L. Chiang, J. Zhang, i J. Jiao, “Starling-7b: Improving llm helpfulness & harmlessness with rlaif”, November 2023.

[9] “Chatbot arena”, https://chat.lmsys.org/, [Online;].

[10] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, Percy Liang, “Lost in the middle: How language models use long contexts”, https://arxiv.org/abs/1301.3781, 2023.

[11] Antonio Gulli, http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html, [Online;].

[12] itsayush_k, https://www.kaggle.com/datasets/itsayushk/links-to-all-articles-from-big-tech-news-sites, [Online;].

Sažetak

Korištenje naprednih tehnika dohvatom-pojačanog generiranja za prepoznavanje futuroloških signala

Igor Šoštarko

Kod korištenja velikih jezičnih modela je značajan problem ustajalo znanje modela, što može dovesti do neoptimalnih rezultata, posebice kod problema koji zahtijevaju praćenje najnovijih informacija, poput prepoznavanja futuroloških signala. U ovom radu je predloženo riješenje tog problema korištenjem dohvatom-pojačanog generiranja za čiju se implementaciju koristi vektorska baza podataka. Kao način evaluacije uspješnosti takvog pristupa se provodi usporedba performansa takve implementacije naspram performanse samog jezičnog modela na zadatku prepoznavanja futuroloških signala. Takav je pristup pokazao obečavajuće rezultate, dajući bolje rezultate od samog jezičnog modela, no i određene nedostatke kao što je smanjenje točnosti klasificiranja članaka koji su signali.

Ključne riječi: dohvatom-pojačano generiranje; jezični model; umjetna inteligencija; futurološki signal

Abstract

Utilization of advanced retrieval-augmented generation techniques for futures signal recognition

Igor Šoštarko

One significant issue with using large language models is the problem of stale knowledge, which can lead to sub-optimal results, especially for tasks that require tracking the latest information, such as recognizing futures signals. This paper proposes a solution to this problem by using Retrieval-Augmented Generation implemented with a vector database. To evaluate the success of this approach, the performance of this implementation is compared to the performance of the language model alone in the task of recognizing futures signals. Such an approach has shown promising results, yielding better outcomes than the language model alone, but also certain shortcomings, such as a decrease in the accuracy of classifying articles that are signals.

Keywords: Retrieval-Augmented Generation; Language Model; Artificial Intelligence; Futures Signal