UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Đorđe Petrović

PRIMENA AGENTSKIH AI SISTEMA U OBRAZOVANJU

master rad

Beograd, 2026.

Mentor:

dr Aleksandar Kartelj, vanredni profesor
Univerzitet u Beogradu, Matematički fakultet

Članovi komisije:

dr Vladimir Filipović, redovni profesor
Univerzitet u Beogradu, Matematički fakultet

dr Mladen Nikolić, vanredni profesor
Univerzitet u Beogradu, Matematički fakultet

Datum odbrane: _____

Naslov master rada: Primena agentskih AI sistema u obrazovanju

Rezime: Veliki jezički modeli poslednjih godina omogućili su razvoj agentskih sistema, u kojima model ne generiše samo tekst, nego bira i poziva spoljašnje alate radi izvršavanja složenih zadataka. Obrazovni kontekst je pogodno tlo za takvu paradigmu, ali opšti jezički model ne poznaje sadržaj konkretnog kursa, ne može da navede izvor svoje tvrdnje i sklon je da uverljivo iznese netačan podatak.

U ovom radu projektovan je, implementiran i izmeren agentski asistent za učenje namenjen studentima Matematičkog fakulteta Univerziteta u Beogradu, dostupan preko interfejsa platforme Telegram. Sistem pod istim interfejsom objedinjuje više alata: odgovaranje na pitanja iz nastavnih materijala uz citate na prezentaciju i stranu, generisanje kvizova za samoproveru sa ocenjivanjem odgovora prema gradivu i pregled gradiva izveden iz samog indeksa. Sadržaj kurseva odvojen je od logike programa i tretira se kao ulazni podatak, pa se skup podržanih kurseva proširuje bez izmene izvornog koda.

Težište istraživačkog dela je na odgovaranju iz gradiva, alatu koji nosi najveći deo vrednosti sistema i počiva na generisanju potpomognutom pretragom (engl. *retrieval-augmented generation*, RAG). Merenjem se pokazuje da uzrok lošeg kvaliteta odgovora nije sloj generisanja, nego sloj pripreme podataka i pretrage. Utvrđeni su i otklonjeni konkretni uzroci: jednojezični model vektorizacije koji srpsku ćirilicu smešta u degenerisan deo vektorskog prostora, segmentacija koja gubi deo teksta, duplirani zapisi u vektorskoj bazi i procena pisma u trenutku upita. Predloženo rešenje počiva na normalizaciji pisma pri indeksiranju, ekstrakciji svesnoj strukture slajda, segmentaciji sa kontekstnim zaglavljem i hibridnoj pretrazi koja gustu i leksičku granu spaja recipročnom fuzijom rangova i dodatno ponovo rangira unakrsnim enkoderom.

Merenja su izvedena nad korpusom nastavnih materijala i nad skupom od 90 pitanja preuzetih iz zvaničnih spiskova ispitnih pitanja dva kursa, kod kojih ni formulacija pitanja ni očekivani izvor nisu nastali uz sistem. Odziv $R@5$ raste sa 0,800 na 0,989, pri čemu preko tri četvrtine tog poboljšanja otpada na zamenu modela vektorizacije. Isti upiti postavljeni ćirilicom i latinicom daju istovetne rezultate. Analiza doprinosa svake komponente pokazuje da unakrsni enkoder od 568 miliona parametara na ovom korpusu ne donosi merljivu prednost nad modelom od 118 miliona, uz šesnaest puta duže vreme odziva.

Ključne reči: agentski sistemi, veliki jezički modeli, obrazovne tehnologije, generisanje potpomognuto pretragom, hibridna pretraga, srpski jezik

Sadržaj

Glava 1

Uvod

Veštačka inteligencija, a naročito oblast velikih jezičkih modela, poslednjih godina doživela je ubrzan razvoj koji je otvorio nove mogućnosti primene računarskih sistema u svakodnevnom životu. Jedna od podoblasti koja se intenzivno istražuje jeste razvoj *agentskih sistema*, u kojima jezički model ne generiše samo tekst, nego planira, donosi odluke i poziva spoljašnje alate radi izvršavanja složenih zadataka. Ta paradigma predstavlja pomak u odnosu na tradicionalne konverzacione sisteme: umesto jednog toka pitanje–odgovor, sistem raspolaže skupom radnji nad stvarnim izvorima podataka i sam bira koju će od njih preduzeti.

Obrazovni kontekst je posebno pogodno tlo za takvu primenu. Student se svakodnevno suočava sa potrebom za brzim pristupom nastavnim materijalima i arhivi prethodnih ispita, organizacijom vremena pred ispit i proverom sopstvenog znanja. Iako postoji veliki broj obrazovnih platformi, malo njih objedinjuje te potrebe u obliku asistenta prilagođenog konkretnoj instituciji i njenom sadržaju.

Priprema ispita na fakultetu i dalje se najvećim delom svodi na rad sa materijalima predmetnog nastavnika: prezentacijama sa predavanja, skriptama i spiskovima ispitnih pitanja. Ti materijali su autoritativni, jer se student ocenjuje prema definicijama, oznakama i naglascima koji se u njima javljaju, a ne prema onome što je o istoj temi napisano u nekom udžbeniku ili na internetu. Istovremeno, oni su i nepretraživi na način koji bi studentu koristio. Pitanje poput onoga gde se u gradivu objašnjava razlika između dva srodna pojma nema odgovor koji se dobija pretragom po nazivima datoteka, a pretraga teksta unutar čitača dokumenata pronalazi tačan niz znakova samo ako ga je student pogodio doslovno, u ispravnom pismu i sa ispravnim dijakriticima.

Veliki jezički modeli (engl. *large language models*, LLM) na prvi pogled deluju

kao prirodno rešenje. Oni tečno odgovaraju na pitanja iz gotovo svake oblasti i sposobni su da objasne gradivo na nivou koji studentu odgovara. Međutim, u ovom konkretnom zadatku pate od tri ograničenja koja se ne mogu otkloniti izborom većeg modela. Prvo, model ne poznaje sadržaj konkretnog kursa: njegovo znanje je parametarsko i potiče iz podataka za treniranje, u koje materijali jednog predmeta na jednom fakultetu gotovo sigurno nisu ušli [27]. Drugo, model ne može da navede izvor svoje tvrdnje, pa student nema način da proveri odgovor niti da nađe mesto u gradivu na kome bi nastavio da uči. Treće, model je sklon halucinacijama, odnosno iznošenju netačnih tvrdnji sa istim stepenom uverljivosti kao i tačnih [15]. To je u kontekstu pripreme ispita posebno štetno, jer student po definiciji nije u poziciji da prepozna grešku.

Generisanje potpomognuto pretragom (engl. *retrieval-augmented generation*, RAG) [20] predstavlja odgovor na sva tri ograničenja. Umesto da odgovara iz sopstvenog parametarskog znanja, model prvo pronalazi odlomke iz zadate zbirke dokumenata, a zatim odgovor sastavlja isključivo iz njih. Time sistem dobija pristup materijalu koji nikada nije video tokom treniranja, može da uz svaku tvrdnju navede odlomak iz kog ona potiče, a prostor za halucinaciju se sužava na ono što se u pronađenim odlomcima zaista nalazi.

Ta zamena, međutim, samo premešta težište problema. Ako pretraga ne pronađe odlomak koji sadrži odgovor, nijedan jezički model ne može da nadoknadi tu grešku: on će ili odbiti da odgovori ili, što je nepovoljnije, sastaviti uverljiv odgovor od labavo povezanih odlomaka. Kvalitet ovakvog sistema time postaje, u najvećoj meri, kvalitet njegove pretrage. Upravo to je i tvrdnja koju rad brani i merenjem potkrepljuje: kod sistema ovog tipa nad nastavnim materijalima na srpskom jeziku, uzrok lošeg kvaliteta odgovora gotovo nikada nije sloj generisanja, nego sloj pripreme podataka i pretrage.

Srpski jezik pri tome nije neutralan izbor. U istom korpusu javljaju se oba pisma, jezik je bogato flektivan, dijakritici se u svakodnevnoj upotrebi izostavljaju, a višejezični modeli ga pokrivaju slabije nego jezike sa obimnijim resursima [22, 3]. Svako od tih svojstava opterećuje lanac pretrage na svoj način; njima je posvećen odeljak 2.10, a njihove merljive posledice pokazuje glava 3.

Predmet ovog rada je projektovanje, implementacija i *merenje* agentskog asistenta namenjenog studentima Matematičkog fakulteta Univerziteta u Beogradu, dostupnog preko interfejsa platforme Telegram. Sistem pod istim interfejsom objedinjuje odgovaranje na pitanja iz nastavnih materijala, uz citate koji upućuju na

prezentaciju i stranu, generisanje kvizova za samoproveru znanja iz istih izvora sa ocenjivanjem odgovora prema gradivu, i pregled kurseva, lekcija i tema izveden iz samog indeksa. Sistem radi u oba pisma i projektovan je tako da se proširuje bez izmene programa: kurs je direktorijum sa materijalima, pa ni dodavanje novog kursa ni prenošenje sistema na drugu instituciju ne zahtevaju izmenu izvornog koda. Prvobitno zamišljeni obim obuhvatao je i pripremu planova učenja, praćenje studentskih obaveza, upis ispitnih termina u kalendar i nadzor nad izmenama na stranicama fakulteta; te funkcije nisu deo isporučenog sistema, a razlozi i uslovi pod kojima bi bile dovršene izloženi su u odeljku 6.1.

Ostatak rada organizovan je na sledeći način. Glava 2 daje teorijske osnove: velike jezičke modele i njihova ograničenja, arhitekturu RAG sistema, leksičku i gustu pretragu, fuziju rangiranih listi, ponovno rangiranje, mere kvaliteta pretrage, agentske sisteme i specifičnosti obrade srpskog jezika. Glava 3 opisuje korpus i uzroke slabe pretrage nad njim. Glava 4 predstavlja arhitekturu i implementaciju rešenja. Glava 5 opisuje proveru ispravnosti, skup za evaluaciju, mere i rezultate, uključujući rukovanje pismom, doprinos pojedinačnih komponenti i operativne troškove. Glava 6 sumira rezultate i navodi pravce daljeg rada.

Glava 2

Teorijske osnove

Ova glava daje pojmovni okvir potreban za razumevanje ostatka rada. Redosled izlaganja prati tok podataka kroz sistem: od jezičkih modela i razloga zbog kojih im je potrebna spoljašnja memorija, preko načina na koje se dokumenti pretražuju i rangiraju, do metrika kojima se kvalitet pretrage meri i specifičnosti srpskog jezika koje ceo taj lanac dodatno opterećuju.

2.1 Veliki jezički modeli

Savremeni veliki jezički modeli zasnovani su na arhitekturi *transformer* [31], čija je ključna komponenta mehanizam pažnje (engl. *attention*). Model se predtrenira na velikoj količini teksta na zadatku predviđanja narednog tokena, čime uči statističku strukturu jezika [10, 5], a posredno usvaja i znatnu količinu činjeničnog znanja [27]. Nakon predtreniranja modeli se dodatno podešavaju da prate uputstva, pa se u praksi koriste tako što im se zada tekstualni upit (engl. *prompt*), a odgovor se generiše token po token.

Sposobnost učenja iz konteksta (engl. *in-context learning*), odnosno sposobnost modela da iskoristi informacije date u samom upitu bez ikakvog dodatnog treniranja, osnova je na kojoj počiva RAG. Ako se u upit ugrade odlomci iz dokumenata, model ih tretira kao raspoloživu građu i može iz njih sastaviti odgovor.

Za potrebe ovog rada bitna su tri ograničenja parametarskog znanja:

Zatvorenost. Znanje modela ograničeno je na ono što je bilo prisutno u podacima za treniranje. Interni, privatni ili usko specijalizovani dokumenti, kakvi su nastavni materijali jednog predmeta, po pravilu nisu među njima.

Neproverljivost. Model ne raspolaže pokazivačem na izvor tvrdnje. Traženje izvora naknadno, od samog modela, po pravilu daje izmišljene reference.

Halucinacija. Model generiše najverovatniji nastavak, a ne istinit nastavak. Netačne tvrdnje iznose se sa istom tečnošću kao tačne [15], pa ih korisnik koji uči novu oblast ne može razlikovati.

2.2 Generisanje potpomognuto pretragom

RAG [20] je arhitektura u kojoj se generisanje odgovora uslovljava dokumentima pronađenim u trenutku upita. Formalno, umesto da se odgovor y modeluje kao $p(y \mid x)$ za upit x, uvodi se skup pronađenih dokumenata $D = \{d_1, \ldots, d_k\}$ i modeluje se $p(y \mid x, D)$, pri čemu D zavisi od x preko posebne komponente za pretragu.

Rad [12] razlikuje nekoliko generacija RAG sistema. U *naivnom* obliku sistem se svodi na tri koraka: podeli dokumente na odlomke, pronađi k najsličnijih i prosledi ih modelu. Upravo taj oblik ima polazna konfiguracija analizirana u glavi 3. *Napredni* RAG uvodi obradu pre pretrage (obogaćivanje odlomaka, prepisivanje upita) i posle pretrage (ponovno rangiranje, sažimanje konteksta), dok *modularni* RAG dopušta grananje, ponovljenu pretragu i odlučivanje o tome da li je pretraga uopšte potrebna.

Svaki RAG sistem ima dva razdvojena toka izvršavanja.

Indeksiranje. Dokumenti se učitavaju, iz njih se izdvaja tekst, tekst se deli na odlomke, svaki odlomak se pretvara u vektor pomoću modela vektorizacije, a vektori se zajedno sa metapodacima smeštaju u vektorsku bazu. Ovaj tok se izvršava retko, pri dodavanju ili izmeni gradiva.

Odgovaranje. Upit se normalizuje i pretvara u vektor istim modelom, pronalaze se najbliži odlomci, oni se eventualno ponovo rangiraju, a zatim se najbolji ugrađuju u upit jezičkom modelu zajedno sa uputstvom da odgovori isključivo iz njih.

Način na koji se pronađeni odlomci spajaju sa generisanjem takođe je predmet istraživanja: umesto prostog nadovezivanja u upit, moguće je svaki odlomak kodirati zasebno i spojiti ih tek u dekoderu [13]. U ovom radu korišćen je jednostavniji,

u praksi uobičajen pristup, sa numerisanim blokom dokaza ugrađenim u upit, jer se sve izmene namerno drže ispod sloja generisanja.

Ključna posledica ovakve podele jeste da greška u toku indeksiranja nije nadoknadiva u toku odgovaranja. Odlomak koji nije ušao u indeks, ili je ušao bez konteksta koji ga čini pronalažljivim, za sistem ne postoji.

2.3 Leksička pretraga

Klasična pretraga informacija [24] dokument predstavlja kao vreću reči (engl. *bag of words*) i rangira ga prema poklapanju termina iz upita. Osnovna intuicija, formalizovana inverznom frekvencijom dokumenta [16], jeste da retki termini nose više informacije od čestih.

Standardna funkcija rangiranja u toj porodici je BM25 [29]. Za upit $Q = \{q_1, \ldots, q_n\}$ i dokument d vrednost je

$$\mathrm{BM25}(d, Q) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i, d)\,(k_1 + 1)}{f(q_i, d) + k_1 \left(1 - b + b\,\dfrac{|d|}{\mathrm{avgdl}}\right)}, \tag{2.1}$$

gde je $f(q_i, d)$ frekvencija termina q_i u dokumentu d, $|d|$ dužina dokumenta, avgdl prosečna dužina dokumenta u zbirci, a k_1 i b parametri koji upravljaju zasićenjem frekvencije termina i normalizacijom po dužini. Inverzna frekvencija dokumenta računa se kao

$$\mathrm{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0{,}5}{n(q_i) + 0{,}5} + 1\right), \tag{2.2}$$

pri čemu je N broj dokumenata, a $n(q_i)$ broj dokumenata koji sadrže termin q_i.

Prednost leksičke pretrage je preciznost na retkim, tačnim terminima: oznake poput `CSMA/CD`, `802.11` ili `AIMD` pronalaze se pouzdano jer su i retke i doslovno prisutne u tekstu. Nedostatak je nemoć pred parafrazom, pošto pitanje koje ni jednom rečju ne ponavlja formulaciju iz gradiva dobija vrednost nula, kao i osetljivost na morfologiju, koja je za srpski jezik naročito izražena.

2.4 Gusta pretraga

Gusta pretraga (engl. *dense retrieval*) tekst predstavlja vektorom u prostoru $\mathbb{R}^n$ tako da semantički bliski tekstovi budu geometrijski bliski [17]. Reprezentacije

se dobijaju neuronskim modelom *bi-kodera*: upit i dokument kodiraju se *nezavisno*, pa se vektori dokumenata mogu izračunati unapred i smestiti u indeks.

Sličnost se meri kosinusom ugla između vektora,

$$\mathrm{sim}(u, v) = \frac{u \cdot v}{\|u\| \, \|v\|}, \tag{2.3}$$

što se za normalizovane vektore svodi na skalarni proizvod.

Model se trenira kontrastnim gubitkom: parovi (upit, relevantan odlomak) privlače se, a nerelevantni odlomci odbijaju [17, 28]. Modeli porodice E5 [32, 33] treniraju se asimetrično, pa očekuju eksplicitne prefikse `query:` i `passage:`; izostavljanje prefiksa merljivo smanjuje tačnost. Njihove višejezične varijante grade se nad višejezičnim enkoderima tipa XLM-R [8], čime se dobija i mogućnost međujezičke pretrage, pri čemu upit na jednom jeziku može pronaći dokument na drugom.

Prednost guste pretrage je otpornost na parafrazu i na morfološke varijante; nedostatak je sklonost da tačne, retke oznake „zamuti" u okolni kontekst, kao i činjenica da kvalitet potpuno zavisi od toga koliko je jezik zastupljen u podacima za treniranje modela. Poređenje modela vektorizacije na standardnim skupovima objedinjuje zbirka zadataka MTEB [25], ali ona srpski jezik pokriva slabo, pa se u ovom radu izbor modela zasniva na merenju nad sopstvenim korpusom.

2.5 Hibridna pretraga i fuzija rangiranih listi

Pošto leksička i gusta grana greše na različitim mestima, prirodno je koristiti obe i spojiti njihove rezultate. Spajanje po sirovim vrednostima je nepogodno, jer su skale nesamerljive: BM25 daje neograničenu pozitivnu vrednost, a kosinusna sličnost vrednost iz intervala $[-1, 1]$. Naknadno svođenje na zajedničku skalu uvodi osetljivost na raspodelu i na prisustvo izuzetaka.

Recipročna fuzija rangova (engl. *reciprocal rank fusion*, RRF) [9] zaobilazi problem tako što koristi isključivo *rang* dokumenta u svakoj listi. Za skup rangiranih listi R i dokument d vrednost je

$$\mathrm{RRF}(d) = \sum_{r \in R} \frac{1}{k + \mathrm{rank}_r(d)}, \tag{2.4}$$

gde je $\mathrm{rank}_r(d)$ pozicija dokumenta d u listi r, a k konstanta prigušenja koja umanjuje uticaj visokih pozicija u pojedinačnoj listi. Uobičajena vrednost $k = 60$,

predložena u originalnom radu, sprečava da jedan snažan pogodak u jednoj grani bude nadglasan osrednjim slaganjem u drugoj. Uprkos jednostavnosti, RRF na standardnim skupovima nadmašuje složenije postupke fuzije [9].

2.6 Ponovno rangiranje unakrsnim enkoderom

Bi-koder je brz jer upit i dokument kodira nezavisno, ali upravo zbog toga ne može da modeluje njihovu interakciju. *Unakrsni enkoder* (engl. *cross-encoder*) prima spojen par (upit, odlomak) kao jedan ulaz i propušta ga kroz model do jedinstvene ocene relevantnosti [26]. Time se dobija znatno tačnije rangiranje, po cenu troška srazmernog broju kandidata: ocena se mora izračunati posebno za svaki par, pa se model ne može primeniti na celu zbirku.

Standardno rešenje je kaskadna arhitektura: jeftina komponenta (gusta i/ili leksička pretraga) svede zbirku na nekoliko desetina kandidata, a skupa komponenta (unakrsni enkoder) precizno rangira samo njih. Postoje i međurešenja poput modela ColBERT [18], koji zadržava deo interakcije uz manji trošak.

Za višejezične korpuse koriste se unakrsni enkoderi trenirani na višejezičnim skupovima za rangiranje, kakav je mMARCO [4]. Uobičajena pretpostavka da veći model daje bolje rangiranje proverena je na korpusu korišćenom u ovom radu (odeljak 5.7).

2.7 Segmentacija dokumenata

Dokumenti se pre indeksiranja dele na odlomke (engl. *chunks*). Izbor veličine odlomka je kompromis. Premali odlomci gube kontekst i postaju nepronalažljivi jer ne sadrže dovoljno signala; preveliki razblažuju vektor, mešaju više tema u jednu reprezentaciju i troše prostor u upitu, a poznato je i da jezički modeli slabije koriste informacije koje se nalaze u sredini dugačkog konteksta [21].

Najjednostavniji postupak deli tekst na blokove fiksne dužine sa preklapanjem. Kvalitetniji postupci poštuju strukturu dokumenta, odnosno naslove, pasuse i granice rečenica, i time izbegavaju da odgovor bude presečen na pola.

Poseban problem javlja se kod prezentacija. Pojedinačan slajd je lista nabrajanja čije značenje zavisi od naslova iznad njih; izvučen iz konteksta prezentacije, on za pretragu praktično nema semantiku. Rešenje koje se u novijoj praksi pokazalo delotvornim jeste *kontekstualno obogaćivanje*: svakom odlomku se pre indeksira-

nja dodaje kratak opis mesta koje zauzima u dokumentu [1]. Taj postupak koristi i sistem opisan u ovom radu, u jeftinoj varijanti koja ne zahteva poziv jezičkog modela po odlomku (odeljak 4.4).

2.8 Metrike evaluacije pretrage

Neka je za upit q poznat skup relevantnih dokumenata, a sistem vraća rangiranu listu. Koriste se sledeće mere.

Odziv na k ($R@k$). Udeo upita kod kojih se bar jedan relevantan dokument nalazi među prvih k rezultata. Mera je pogodna za RAG jer je dovoljno da tačan odlomak uđe u upit modela; njegova tačna pozicija manje je bitna.

Srednji recipročni rang (MRR). Za skup upita Q,

$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}, \tag{2.5}$$

gde je rank_i pozicija prvog relevantnog dokumenta za upit i. Mera nagrađuje sisteme koji tačan odgovor stavljaju na sam vrh liste.

Normalizovana kumulativna dobit sa popustom (nDCG@k). Mera [14] koja uzima u obzir pozicije *svih* relevantnih dokumenata:

$$\mathrm{DCG@}k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}, \qquad \mathrm{nDCG@}k = \frac{\mathrm{DCG@}k}{\mathrm{IDCG@}k}, \tag{2.6}$$

pri čemu je rel_i relevantnost dokumenta na poziciji i, a $\mathrm{IDCG@}k$ vrednost $\mathrm{DCG@}k$ za idealno rangiranje.

Evaluacija samog *odgovora*, a ne pretrage, zahteva drugačiji pristup, pre svega mere vernosti odgovora izvorima i relevantnosti odgovora pitanju [11].

2.9 Agentski sistemi i upotreba alata

Jezički model može se koristiti i kao komponenta koja odlučuje o narednom koraku, a ne samo kao generator teksta. Uobičajeno se govori o *agentskom sistemu* kada model, pored generisanja teksta, ima i sledeće:

Skup radnji. Modelu se daju funkcije sa opisanim potpisima, takozvani alati (engl. *tools*), koje deluju na spoljašnji svet: pretražuju bazu, preuzimaju stranicu, upisuju zapis [30]. Model bira koju će pozvati i sa kojim argumentima, a rezultat poziva vraća se u kontekst.

Odlučivanje o toku. Naredni korak nije unapred određen programom, nego zavisi od stanja i od ishoda prethodnih poziva. Pristup ReAct [34] formalizuje to naizmeničnim smenjivanjem rasuđivanja i akcije.

Stanje. Sistem pamti kontekst razgovora i eventualne odluke korisnika između poziva.

Razlika u odnosu na tradicionalni konverzacioni sistem je suštinska: takav sistem preslikava poruku u odgovor, dok agentski sistem preslikava poruku u *niz radnji* čiji je odgovor tek završni ishod. Time postaje moguće pokriti raznorodne zahteve, od pitanja iz gradiva i generisanja kviza do ocene odgovora i pregleda lekcija, jednim interfejsom umesto zasebnim ekranom po funkciji.

Za implementaciju takvih tokova koriste se biblioteke koje sistem predstavljaju kao graf stanja: čvorovi su koraci obrade, grane su prelazi, a stanje se prosleđuje kroz graf i uvećava. Takav model izvršavanja koristi biblioteka LangGraph [19], na kojoj počiva i sistem opisan u ovom radu.

Kada je agentski pristup opravdan

Agentski pristup nosi i cenu. Svaki sloj odlučivanja iznad korisne radnje unosi mogućnost da sistem pogreši u izboru alata i time nikada ne dođe do onoga što je korisniku zaista trebalo; svaki dodatni korak povećava vreme odziva i broj poziva modela, dakle i trošak. Otuda i pravilo da se agentska arhitektura bira tek kada je zadatak zaista otvoren i teško opisiv unapred, dok se u ostalim slučajevima ostaje na jednostavnijem toku sa unapred određenim koracima.

U glavi 4 pokazuje se da je najčešći zahtev, pitanje iz gradiva, namerno izuzet iz odlučivanja i tretiran kao podrazumevani slučaj, dok agentski sloj ostaje zadužen za ono što zaista jeste izbor između raznorodnih radnji.

Agentski sistemi u obrazovanju

Primena jezičkih modela u obrazovanju najčešće se javlja u obliku asistenta nad nastavnim materijalima, sa naglaskom na utemeljenosti odgovora. Prelazak sa

konverzacionog na agentski oblik donosi dve stvari koje su u obrazovnom kontekstu bitne.

Prvo, isti utemeljeni sloj pretrage može da posluži većem broju alata: odgovor na pitanje i provera znanja crpe iz istog indeksa, pa poboljšanje pretrage istovremeno popravlja obe funkcije. Ta pojava, da kvalitet pretrage određuje kvalitet celog agentskog sistema, neposredno je merena u glavi 5.

Drugo, sistem dobija *režime*: ono što student otkuca ne mora uvek značiti isto. Dok je kviz otvoren, slobodan tekst je odgovor koji treba oceniti, a ne novo pitanje; kada je izabrana jedna lekcija, pretraga se drži nje. Bez stanja koje agentski sloj održava, svaka takva razlika morala bi da bude zaseban ekran ili zasebna komanda.

2.10 Specifičnosti obrade srpskog jezika

Dvopismenost

Srpski jezik standardno koristi dva pisma. Svakom ćiriličkom slovu odgovara tačno jedan latinički zapis, pri čemu se tri slova zapisuju kao digrafi (*lj*, *nj*, *dž*), pa je preslovljavanje ćirilice u latinicu jednoznačno. Obrnut smer to nije: latinička niska *nj* u reči „nakovanj" odgovara jednom ćiriličkom slovu, a u reči „injekcija" dvama. Ta asimetrija je za sistem pretrage od praktične važnosti i njome se bavi odeljak 4.3.

Morfologija

Srpski je flektivan jezik sa sedam padeža, tri roda i bogatom glagolskom promenom. Jedna leksema ima veliki broj pojavnih oblika, pa leksička pretraga bez normalizacije promašuje. Klasično rešenje je stemovanje ili lematizacija, koji za srpski postoje, ali zahtevaju dodatne resurse; u ovom radu leksička grana je namerno ostavljena bez lematizacije, jer njen zadatak nije pokrivanje parafraze, što radi gusta grana, nego tačno poklapanje stručnih oznaka.

Dijakritici

Slova *č*, *ć*, *ž*, *š* i *đ* u neformalnoj upotrebi sistematski se pišu bez dijakritika. Za leksičku granu to znači da se poređenje mora vršiti nad oblikom iz kog su dijakritici uklonjeni, pa „Senonov" i „Шеноноv" daju isti token.

Status jezika sa ograničenim resursima

Srpski jezik ima znatno manje anotiranih resursa od engleskog [22, 3]. Postoje namenski predtrenirani modeli za grupu južnoslovenskih jezika [23], ali oni nisu trenirani kao modeli vektorizacije za pretragu. U praksi se stoga pribegava velikim višejezičnim modelima [8, 33, 6], kod kojih kvalitet za srpski jezik nije unapred poznat i mora se izmeriti nad konkretnim korpusom. Otuda i težište koje merenje ima u ovom radu.

Glava 3

Analiza problema

Neposredna primena postupka opisanog u odeljku 2.2 daje funkcionalan sistem: materijali se indeksiraju, na svako pitanje vraćaju se odlomci i od njih se sastavlja odgovor. Upravo je ta prividna ispravnost i najveća opasnost, jer takav sistem uvek vraća neki rezultat, pa činjenica da je pretraga blizu slučajne ostaje nevidljiva sve dok se ne izmeri.

Ova glava opisuje korpus nad kojim je sistem meren, metodološki okvir merenja i četiri uzroka slabog kvaliteta pretrage: jednojezični model vektorizacije, gubitak teksta pri segmentaciji, duplirane zapise u vektorskoj bazi i procenu pisma u trenutku upita. Prva tri potvrđena su zasebnim testom, dok četvrti sledi iz same konstrukcije postupka. Nijedan se ne nalazi u sloju generisanja odgovora. Konfiguracija koja te uzroke sadrži označava se u nastavku kao *polazna*. Od nje potiče i polazni red tabele napretka u glavi 5, s tim što se tamo, radi razdvajanja doprinosa, menja jedna po jedna komponenta.

3.1 Opis korišćenog korpusa

Broj kurseva sa kojima sistem radi nije ograničen niti zapisan u programu, jer je kurs direktorijum sa materijalima, kako je opisano u odeljku 4.7. Za razvoj i merenje bilo je, međutim, potrebno izabrati konkretan korpus, pa su sva merenja u ovom radu izvedena nad materijalima dva kursa Matematičkog fakulteta. Njihova svojstva čine sadržaj merne postavke, a ne granicu sistema, i navedena su u tabeli 3.1. Podaci su preuzeti iz izveštaja koji se automatski generiše pri svakoj izgradnji indeksa.

Za projektovanje sistema presudna su dva svojstva korpusa.

Tabela 3.1: Izmerene karakteristike indeksiranog korpusa

Svojstvo	Vrednost
Broj kurseva	2
Broj datoteka	60
Broj strana	1774
Broj indeksiranih odlomaka	863
Broj izdvojenih znakova	740.086
Prosečno znakova po strani	417
Strane bez teksta (samo slika)	75
Datoteke bez ijednog odlomka	0

Prvo, korpus objedinjuje dva vrlo različita oblika dokumenta. Prezentacije sa predavanja imaju oko 260 znakova po strani, dok skripte imaju oko 1300 znakova po strani. Razlika od oko pet puta glavni je razlog zbog kog jedinstvena strategija segmentacije nije prihvatljiva (odeljak 4.4). Pri tome je prosečan slajd, sa svojih oko 260 znakova, sam po sebi lista nabrajanja bez konteksta: naslov slajda nosi najveći deo semantike strane, a u ravnom tekstu koji vraća uobičajena ekstrakcija ne razlikuje se od ostalih stavki.

Drugo, pismo u korpusu nije ujednačeno. Materijali su pretežno ćirilični, deo je latinični, a nekoliko prezentacija je u celosti na engleskom jeziku. Studenti pitanja postavljaju u oba pisma, često bez dijakritika. Time se problem dvopismenosti, opisan u odeljku 2.10, javlja i na strani dokumenata i na strani upita.

Posebno mesto zauzimaju zvanični spiskovi ispitnih pitanja, koje objavljuju oba kursa i koji uz svako pitanje navode gradivo iz kog se odgovor priprema, bilo prezentaciju sa opsegom slajdova, bilo odgovarajuću sekciju udžbenika. Oni su osnova za merenje kvaliteta pretrage (odeljak 5.3), pri čemu je jedan od njih ujedno i indeksiran kao materijal svog kursa.

3.2 Metodološki okvir

Dijagnoza je vođena jednim pravilom: uzrok se ne pretpostavlja, nego meri, gde god je merenje moguće. Za svaki sumnjivi deo sistema formulisan je najmanji mogući test koji njegovu ispravnost potvrđuje ili obara, nezavisno od ostatka sistema. Razlog je praktičan: u lancu od šest komponenti, u kome svaka može biti uzrok, izmene bez prethodnog merenja nisu proverljive, a pojedinačni doprinos svake izmene ne može se razdvojiti.

Naredni odeljci prikazuju nalaze grupisane prema sloju u kome se nalaze.

3.3 Reprezentacija teksta u vektorskom prostoru

U polaznoj konfiguraciji koristi se model vektorizacije treniran na engleskom jeziku (`all-MiniLM-L6-v2`), što je uobičajen izbor kada jezik korpusa nije uzet u obzir. Da bi se ispitalo koliko takav model predstavlja srpski jezik, izračunata je prosečna kosinusna sličnost između parova *nepovezanih* rečenica, posebno za skup rečenica na engleskom jeziku i za skup rečenica na srpskoj ćirilici. Kod modela koji jezik smisleno predstavlja, nepovezane rečenice treba da budu međusobno udaljene.

Tabela 3.2: Kosinusna sličnost nepovezanih rečenica pod jednojezičnim modelom

Skup rečenica	Prosek	Minimum	Maksimum
Engleski jezik	0,110	−0,011	0,407
Srpska ćirilica	0,755	0,658	0,826

Rezultat u tabeli 3.2 pokazuje da model srpsku ćirilicu smešta u degenerisan deo vektorskog prostora. Za normalizovane vektore, prostor u kome sve nepovezane rečenice stoje na sličnosti od 0,755 znači da se ceo korpus skupio u uzak konus, unutar kog rangiranje nema na čemu da radi: razlike u sličnosti između tačnog i netačnog odlomka postaju manje od šuma.

3.4 Defekti u pripremi podataka

Uz reprezentaciju teksta, ispitan je i put kojim materijal dolazi do indeksa. Tokom razvoja su merenjem uočena dva problema u polaznoj konfiguraciji, od kojih se nijedan ne ispoljava kao otkaz sistema, pa nijedan nije vidljiv bez namenskog merenja.

Gubitak teksta pri segmentaciji. Ako granica prethodnog odlomka zadrži kraj rečenice, a granica sledećeg zbog ciljane dužine odbaci njen početak, ta rečenica ne uđe ni u jedan od odlomaka. Merenje nad kontrolisanim ulazom od numerisanih rečenica potvrđuje da se to zaista dešava: otprilike jedna rečenica

po granici odlomka izostaje iz indeksa. Ovo je važno zato što ostaje neprimećeno u uobičajenoj upotrebi: indeksiranje prijavljuje uspeh, broj odlomaka i strana odgovara očekivanom, a deo sadržaja je ipak nedostupan za pretragu, jer se ta statistika meri po odlomku, a ne po rečenici.

Duplirani zapisi u vektorskoj bazi. Postupak ponovne izgradnje indeksa briše samo zapise vezane za kurs koji se obnavlja, pa zapisi bez ispravnog identifikatora preživljavaju svaku reindeksaciju. Provera stanja baze pokazala je da takvi zapisi čine oko sedminu indeksa. Efekat nije samo zauzimanje prostora: oni troše mesta u listi od k najboljih rezultata, i to upravo za odlomke koji su već prisutni, čime smanjuju raznovrsnost dokaza prosleđenih modelu.

3.5 Rukovanje pismom u trenutku upita

Neusklađenost pisma između upita i dokumenta može se pokušati rešiti i u trenutku upita: za svako pitanje generiše se nekoliko preslovljenih varijanti, izvršava se zaseban upit po varijanti, a rezultati se spajaju redosledom varijanti, a ne po udaljenosti, i zatim skraćuju na prvih k.

Takav postupak uvodi novu grešku. Slabija varijanta upita može da istisne sve tačne pogotke, pošto se rezultati bolje varijante nalaze iza nje i bivaju odsečeni. Reč je o opštoj pojavi u kojoj se nedostatak u jednom sloju nastoji nadomestiti procenom u sloju iznad, umesto da se ukloni uzrok. Rešenje predloženo u odeljku 4.3 uklanja potrebu za tim slojem u celini.

Glava 4

Arhitektura i implementacija rešenja

Ova glava opisuje sistem onakav kakav je isporučen. Izlaganje počinje pregledom arhitekture i agentskim slojem, koji odlučuje koja će se radnja izvršiti, a zatim prati put podataka kroz alat koji nosi najveći deo posla: odgovaranje iz gradiva. Preostali alati i korisnički interfejs opisani su na kraju glave. Uz svaku odluku navodi se i njeno obrazloženje, jer su gotovo sve donete kao odgovor na neki od nalaza iz glave 3.

Opisano je samo ono što sistem zaista radi. Nekoliko prvobitno zamišljenih funkcija nije realizovano i o njima se ovde ne govori kao o postojećim; razlozi i posledice te odluke izloženi su u odeljku 6.1.

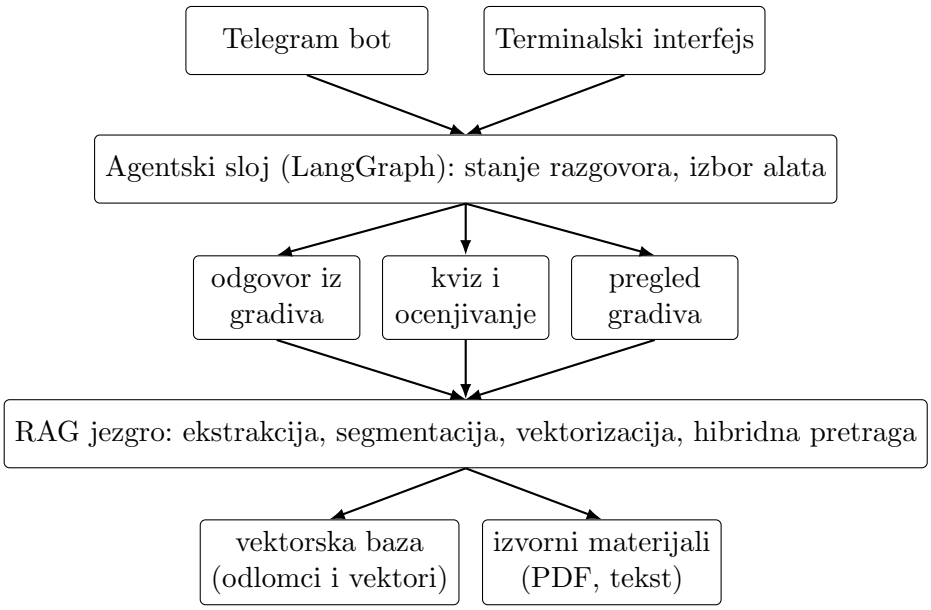
4.1 Pregled arhitekture

Sistem je implementiran u programskom jeziku Python i organizovan u pet slojeva prikazanih na slici 4.1.

Organizacija izvornog koda prati istu podelu, sa jasno odvojenim celinama za jezgro pretrage, za alate koje agent poziva, za graf stanja agenta, za korisničke interfejse i za programe kojima se indeks gradi i kvalitet pretrage meri.

4.2 Agentski sloj i skup alata

Agentski sloj implementiran je pomoću biblioteke LangGraph [19] kao graf stanja sa dva čvora, od kojih jedan poziva model a drugi izvršava alat, i sa uslovnom



Slika 4.1: Slojevi sistema i tok podataka među njima

granom između njih. Stanje razgovora nosi listu poruka, identifikator korisnika, aktivni kurs, opcionu aktivnu lekciju i eventualni otvoreni kviz; grana vodi ka izvršavanju alata kada je alat izabran, a inače završava tok.

Bitno je da to stanje živi u grafu, a ne u pojedinačnom interfejsu. Sve što menja tumačenje naredne poruke, poput izbora kursa, izbora lekcije ili otvorenog kviza, nalazi se na jednom mestu, pa se terminal i Telegram ponašaju identično. Da je taj kontekst smešten u pojedinačni interfejs, ista poruka bi u dva interfejsa bila različito protumačena, sa posledicama opisanim u odeljku 4.8.

Skup alata je ono što ovaj sistem čini agentskim u smislu iz odeljka 2.9: model ne generiše samo tekst, nego bira *koja* će se radnja izvršiti nad kojim izvorom podataka. Raspoloživi alati navedeni su u tabeli 4.1.

Obim odlučivanja prepuštenog modelu

Model interakcije počiva na jednom pravilu: slobodan tekst uvek znači zahtev za odgovorom iz gradiva. Reč je o najčešćem zahtevu i jedinom čiji je kvalitet merljivo potvrđen, pa se ispred njega ne postavlja nikakva klasifikacija namere; sve ostalo je izričita komanda ili dugme. Obrazloženje je isto kao kod uklanjanja ključnih reči za kurs (odeljak 4.7): ručno održavana lista fraza za prepoznavanje namere dugoročno je neodrživa. Razlika je u tome što se ovde problem ne rešava

Tabela 4.1: Alati dostupni agentskom sloju

Alat	Uloga
Odgovaranje iz gradiva	odgovor sastavljen iz pronađenih odlomaka, sa citatima; podrazumevana radnja
Generisanje kviza	pitanja za proveru znanja, izvedena iz pronađenog gradiva
Ocenjivanje odgovora	ocena studentovog odgovora prema gradivu, u tri stepena
Pregled gradiva	spisak kurseva, lekcija i tema, izveden iz indeksa
Šire objašnjenje	objašnjenje iz opšteg znanja modela, odvojeno i bez citata

boljom pretragom, nego prenošenjem izbora na korisnika, kome je namera ionako poznata.

Iz istog razloga opseg pretrage nije skriven. Aktivni kurs i, opciono, aktivna lekcija prikazuju se u zaglavlju svakog odgovora, jer se odgovor ne može ispravno protumačiti bez informacije o tome nad čim je tražen. Unutar izabrane lekcije sistem, kada odgovora nema, to izričito saopštava umesto da tiho pretraži ostatak gradiva, jer bi u suprotnom izbor lekcije bio bez značenja.

Ukupno uzev, modelu je prepušten izbor alata samo kada zahtev nije izričita komanda, izbor kursa izveden je iz rezultata pretrage a ne iz zaključivanja modela (odeljak 4.7), a najčešći zahtev zaobilazi klasifikaciju namere u celini. Ta uzdržanost je namerna: svaki sloj odlučivanja iznad pretrage unosi mogućnost da sistem pogreši u izboru alata i time nikada ne dođe do gradiva, a upravo je ta klasa grešaka bila jedan od nalaza iz glave 3. Agentski pristup je ovde sredstvo za pokrivanje raznorodnih zahteva jednim interfejsom, a ne cilj po sebi.

4.3 Normalizacija pisma

Ovo je najvažnija pojedinačna odluka u sistemu i ona iz koje slede mnoge druge.

Kao što je navedeno u odeljku 2.10, preslovljavanje ćirilice u latinicu je jednoznačno, dok obrnut smer nije. Odatle sledi pravilo:

> *Sav tekst se preslovljava (engl.* transliteration*) u latinicu pre vektorizacije i indeksiranja, a originalni tekst se čuva za prikaz.*

Posledice ovog pravila su:

- Reprezentacija upita i reprezentacija dokumenta *uvek* se poklapaju, bez obzira na to kojim je pismom pitanje postavljeno i kojim je pismom gradivo napisano.
- Ceo sloj nagađanja u trenutku upita, opisan u odeljku 3.5, uklonjen je u celini, a ne ublažen. Ne postoji više nijedna varijanta upita, nijedno spajanje listi i nijedan redosled varijanti koji bi mogao da bude pogrešan.
- Obrnut smer (latinica u ćirilicu) koristi se isključivo za prikaz poruka interfejsa, gde je dvosmislenost bezopasna.

Razlikuju se tri normalizovana oblika teksta, po jedan za svaku upotrebu:

Oblik za vektorizaciju preslovljava tekst u latinicu i sažima praznine, ali zadržava veličinu slova i dijakritike, dakle upravo oblik na kome su višejezični modeli vektorizacije trenirani.

Oblik za leksičku pretragu dodatno prevodi u mala slova i uklanja dijakritike, čime se „Sta je Senonov kapacitet” i „Шта је Шеронов капацитет” svode na identičan niz tokena.

Tokenizacija za leksičku pretragu čuva stručne oznake celovitim, pa su *csma/cd* i *802.11* po jedan token, a ne tri, jer bi deljenje na kosoj crti i tački upravo najprepoznatljivije termine razbilo u beznačajne delove.

Nekoliko detalja lako se previdi, a svaki od njih daje vidljivo pogrešno ponašanje:

- **Digrafi prate veličinu slova okoline.** Ćiriličko Њ u reči koja počinje velikim slovom daje *Nj*, a u reči pisanoj verzalom daje *NJ*. Implementacija zato gleda naredni znak, a kada je on nealfabetski i prethodni.
- **Nazivi datoteka se nikada ne preslovljavaju.** Citat koji upućuje na datoteku pod preslovljenim imenom student ne može da pronađe na disku.
- **Poruke interfejsa pišu se sa punim dijakriticima.** Pošto se za prikaz ćirilicom koriste iste niske, oblik „Pokusaj” preslovio bi se u pogrešno „Покусај” umesto u „Покушај”.

4.4 Priprema i indeksiranje materijala

Sloj koji materijal prevodi u pretraživ oblik opisuje se ovde onoliko koliko je potrebno da bi merenja iz glave 5 bila razumljiva i ponovljiva; njegova mehanika je pretežno inženjerske prirode.

Ekstrakcija svesna strukture. Tekst se izdvaja bibliotekom koja uz sadržaj daje i metrike fonta [2], što omogućava da se naslov slajda prepozna i sačuva odvojeno od tela strane. Naslovom se smatra linija sa najvećim fontom pri vrhu strane, pri čemu se veličina tela teksta određuje kao ona koja pokriva najviše znakova, a ne kao medijana po poziciji linije; kod slajda sa svega dve linije medijana bi pala na sam naslov. Linije koje se ponavljaju na najmanje 40 % strana uklanjaju se kao tekuća zaglavlja, uz ograničenje na kratke linije, jer se duže ponovljene linije po pravilu javljaju kao sadržaj prenet kroz nekoliko uzastopnih slajdova. Strane bez čitljivog teksta prijavljuju se u izveštaju o indeksiranju umesto da budu prećutno preskočene, čime se razdvaja slučaj u kome gradivo nije pronađeno od slučaja u kome ono nije ni ušlo u indeks.

Segmentacija prema obliku dokumenta. Pošto se prezentacije i skripte razlikuju po gustini teksta oko pet puta (odeljak 3.1), tip dokumenta se određuje automatski, prema prosečnom broju znakova po strani, i za svaki se primenjuje druga strategija (tabela 4.2).

Tabela 4.2: Strategije segmentacije po tipu dokumenta

Tip dokumenta	Merena gustina	Strategija
Prezentacije	oko 260 znakova po strani	grupisanje uzastopnih slajdova do 1200 znakova, preklapanje jedan slajd
Skripte i proza	oko 1300 znakova po strani	deljenje po naslovima i pasusima, oko 1000 znakova, preklapanje oko 150 znakova
Spiskovi pitanja	kratke, samostalne stavke	postupak kao za prezentacije, jer svaka stavka nosi zaseban smisao

Preklapanje je pri tome definisano kao klizni prozor preko celih jedinica, rečenica kod proze i slajdova kod prezentacija, a nikada kao sečenje niske; time je

otklonjen gubitak teksta opisan u odeljku 3.4. Invarijanta da se svaka rečenica ulaza javlja bar u jednom odlomku zapisana je kao izvršni test, čime je povratak te klase greške onemogućen. Identifikator odlomka računa se kao kriptografski otisak trojke koju čine kurs, izvorna datoteka i sam tekst odlomka, pa je ponovno indeksiranje idempotentno, a udvajanje zapisa iz iste datoteke postaje nemoguće. Isti sadržaj priložen u dva formata otisak pri tome ne prepoznaje kao jedan, jer se datoteke razlikuju.

Kontekstno zaglavlje odlomka

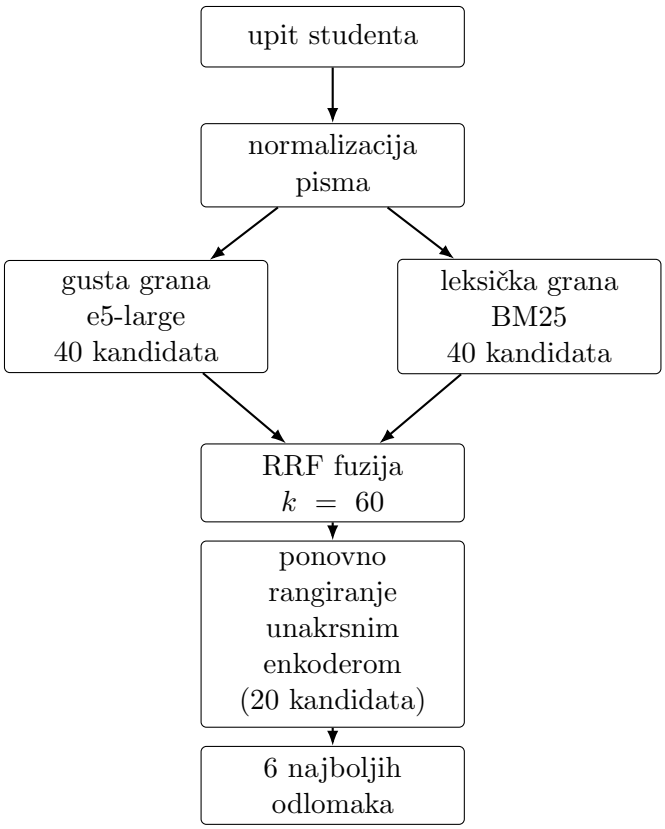
Svakom odlomku dodaje se, pre vektorizacije, kratko zaglavlje koje imenuje kurs, prezentaciju i sekciju, dakle naslov slajda izdvojen pri ekstrakciji, zajedno sa opsegom strana. Zaglavlje postaje deo teksta koji se vektorizuje i koji vidi leksička grana pretrage.

Postupak je u duhu kontekstualnog obogaćivanja [1], ali u varijanti koja ne zahteva poziv jezičkog modela po odlomku, pa je njegova cena zanemarljiva. Efekat je, međutim, znatan: zaglavlje objema granama pretrage daje sadržaj na koji mogu da se poklope i vraća kontekst izgubljen time što je slajd sam po sebi gola lista nabrajanja. Pitanje o nekoj temi često nema nijednu reč zajedničku sa telom slajda, ali ima dve ili tri sa njegovim zaglavljem.

Vektorizacija i skladištenje. Kao model vektorizacije koristi se višejezični model `intfloat/multilingual-e5-large` [33], čime se otklanja uzrok opisan u odeljku 3.3. Model se izvršava lokalno, pa materijal ne napušta računar u fazi indeksiranja, i poziva se sa prefiksima `query:` odnosno `passage:`, koje porodica E5 očekuje, uz normalizaciju vektora kojom se kosinusna sličnost svodi na skalarni proizvod. Vektori i metapodaci smeštaju se u vektorsku bazu [7], a uz svaki odlomak čuvaju se identifikator kursa, naziv datoteke, opseg strana, naslov sekcije i tip dokumenta, dakle svi podaci potrebni za citat i za kasnije izvođenje strukture kursa.

4.5 Hibridna pretraga

Tok obrade jednog upita prikazan je na slici 4.2.



Slika 4.2: Tok obrade upita kroz hibridnu pretragu

Gusta grana. Upit se normalizuje i vektorizuje istim modelom kao i odlomci, pa se iz vektorske baze uzima 40 najbližih kandidata. Pretraga se može ograničiti na kurs ili na skup datoteka jedne lekcije, preko filtera nad metapodacima.

Leksička grana. Nad normalizovanim tekstom svih odlomaka gradi se BM25 indeks (implementacija `rank_bm25`), iz kog se takođe uzima 40 kandidata. Indeks se gradi jednom i ponovo koristi dok se broj odlomaka u bazi ne promeni.

Fuzija. Kandidati se spajaju po formuli (2.4) sa $k = 60$. Time se izbegava poređenje nesamerljivih skala.

Ponovno rangiranje. Prvih 20 spojenih kandidata prosleđuje se unakrsnom enkoderu `cross-encoder/mmarco-mMiniLMv2-L12-H384-v1`, treniranom nad višejezičnim skupom mMARCO [4]. Konačni rezultat čini šest najbolje rangiranih odlomaka. Ukupno vreme obrade upita iznosi oko dve sekunde na procesoru bez grafičkog ubrzanja.

Doprinos svake grane meren je zasebno (odeljak 5.7). Gusta grana hvata parafraze i morfološke varijante, a načelno i veze između jezika, dok leksička hvata tačne stručne oznake koje vektorizacija zamućuje. Ako neka komponenta nije dostupna, na primer ako biblioteka za ponovno rangiranje nije instalirana, sistem se stepenasto degradira na fuziju bez ponovnog rangiranja umesto da otkaže.

4.6 Utemeljeno generisanje odgovora

Šest pronađenih odlomaka pretvara se u numerisan blok dokaza, u kome svaki odlomak nosi svoj citat: naziv datoteke, oznaku slajda ili strane i naslov sekcije. Uz taj blok modelu se prosleđuje uputstvo koje mu nalaže sledeće:

- da odgovara *isključivo* na osnovu priloženih izvoda i da ne dodaje ni ono što zna, ako toga u izvodima nema;
- da iza svake tvrdnje navede redni broj izvoda iz kog ona potiče;
- da, kada izvodi ne sadrže odgovor, ispiše ugovorenu nisku umesto odgovora;
- da odgovori pismom kojim je student pisao i da ne meša pisma unutar istog odgovora, već da izvode po potrebi prepiše u pismo odgovora;
- da stručne termine i skraćenice zadrži onako kako su zapisani u gradivu.

Pismo odgovora određuje se brojanjem slova u pitanju, a ne prisustvom bilo kog ćiriličkog znaka. Latinične skraćenice (DNS, TCP/IP, HTTP) česte su unutar ćiriličke rečenice, pa bi prostija provera pogrešno protumačila pitanje.

Dva sloja odbijanja. Sistem odbija da odgovori na dva nezavisna načina:

1. **Prag na oceni ponovnog rangiranja.** Ako najbolji odlomak ima ocenu ispod praga, odgovor se uopšte ne generiše.
2. **Ugovorena niska iz odgovora modela.** Model je uputstvom obavezan da ispiše unapred dogovorenu nisku kada priloženi izvodi ne pokrivaju pitanje. Ta niska se presreće i zamenjuje porukom studentu.

Tabela 4.3: Podešavanje praga odbijanja prema oceni ponovnog rangiranja

Vrsta pitanja	Najmanja ocena	Najveća ocena
Pitanja iz gradiva	−0,36	+7,36
Pitanja van gradiva	−6,79	−3,30
Izabrani prag: −3,0		

Prag nije pretpostavljen, nego kalibrisan merenjem prikazanim u tabeli 4.3. Izmereni su opsezi ocena za pitanja koja jesu i koja nisu iz gradiva, a prag je postavljen unutar procepa koji se na tom uzorku pokazao.

Skala ocene specifična je za konkretan model ponovnog rangiranja, pa se prag mora ponovo kalibrisati pri svakoj njegovoj zameni; podrazumevana vrednost i ostali podesivi parametri navedeni su u tabeli 5.1. Uzorak pitanja van gradiva na kome je prag kalibrisan pokazao se, međutim, suviše lakim, pa procep iz tabele 4.3 ne važi za teme bliske gradivu; granice te kalibracije razmatraju se u odeljku 5.11. Kako izgleda odgovor sastavljen po ovim pravilima, kao i kako izgleda odbijanje, prikazano je u odeljku 4.10.

Zašto se odgovor ne proverava dodatnim pozivom modela. Naizgled prirodno pojačanje utemeljenosti bilo bi da se posle generisanja odgovora model pozove još jednom, kako bi odgovor proverio. Takva provera, međutim, poredi odgovor sa izlazom istog modela nad istim ulazom, dakle sa samim sobom, pa ne donosi nikakvu novu informaciju, a udvostručava cenu i vreme odziva. Oba opisana sloja odbijanja izbegavaju taj problem time što se oslanjaju na signal spolja: prag se računa iz vrednosti koju daje unakrsni enkoder, a ne jezički model, dok ugovorena niska proizlazi iz poređenja pitanja sa priloženim izvodima, a ne iz ocene sopstvenog odgovora.

Opšte znanje modela: odvojeno, na zahtev, bez citata

Sistem može da odgovori i iz opšteg znanja jezičkog modela, ali pod tri ograničenja:

1. **Stiže kao zasebna poruka**, sa vidljivim upozorenjem, i nikada se ne spaja sa utemeljenim odgovorom;

2. **Nikada ne nosi citate**, jer citat u ovom sistemu znači „ovo piše u tvom gradivu";

3. **Nikada se ne koristi pri proveri znanja**, jer bi student vežbao gradivo koje se od njega ne traži.

U osnovi tog pravila stoji *neslaganje opsega*. Student se ispituje po definicijama svog kursa, pa objašnjenje koje je tačno uopšteno može biti pogrešno za ispit, zbog druge notacije, drugog naglaska ili tema koje kurs namerno izostavlja. Iz istog razloga ocenjivanje odgovora na kviz izričito upućuje na gradivo kao merodavan izvor.

> *Granica sistema nije „šta model zna", nego „šta se od studenta traži na ispitu". Utemeljenost tu štiti i od tačnih odgovora koji se na ispitu ne traže.*

4.7 Kurs kao podatak i usmeravanje pretrage

Uobičajen način da se pretraga usmeri na odgovarajući kurs jeste poređenje pitanja sa ručno održavanom listom ključnih reči po kursu. Taj pristup ima dva nedostatka: lista se mora dopunjavati za svaki novi kurs, a nazivi konkretnih kurseva ulaze u izvorni kod.

Sistem zato kurs prepoznaje samo kada ga student izričito imenuje, a inače ga izvodi iz rezultata pretrage, bez ikakve liste ključnih reči. Osnov za tu odluku je merenje prikazano u odeljku 5.8.

Posledica po arhitekturu je da je **kurs prosto direktorijum sa materijalima**. Nijedan kurs se ne pominje u izvornom kodu. Uz materijale se može priložiti kratak opis kursa, sa identifikatorom, punim imenom, jednim redom opisa i eventualnim alternativnim nazivima po kojima ga student može imenovati, ali ni to nije obavezno: bez njega se identifikator i naziv izvode iz naziva direktorijuma.

Struktura kursa, odnosno koje lekcije postoje i koje teme pokrivaju, čita se iz metapodataka indeksa, odnosno iz naslova slajdova izdvojenih pri ekstrakciji, pa je i pregled gradiva izveden iz podataka umesto da bude ručno zapisan. Dodavanje kursa se time svodi na kopiranje materijala i pokretanje indeksiranja.

Isti princip primenjen je i na imenovanje lekcija. Redni broj iz naziva datoteke upotrebljiv je samo kada kurs numeriše predavanja jedinstveno; kada su materijali

sastavljeni iz više odvojenih serija, svaka sa sopstvenom numeracijom, grupisanje po tom broju spaja nepovezano gradivo. Pošto se iz naziva datoteka struktura predavanja ne može pouzdano rekonstruisati, ona se ni ne pretpostavlja, nego se lekcije prikazuju kao numerisan spisak. Ono što se iz naziva može pouzdano pročitati jeste poreklo materijala, izraženo kratkom oznakom izvora na kraju naziva; ona se prepoznaje automatski i uklanja iz prikaza, a vraća samo kada bi bez nje dva naslova bila identična.

> *Metapodaci izvedeni iz naziva datoteka pouzdani su tačno onoliko koliko je konvencija imenovanja dosledna, pa sistem koristi samo onaj njihov deo za koji je doslednost potvrđena.*

4.8 Kviz i provera znanja

Generisanje kviza koristi isti sloj pretrage kao i odgovaranje. Za zadatu temu pronalazi se $\max(n, k)$ odlomaka, gde je n traženi broj pitanja, a k uobičajeni broj odlomaka za odgovor; od njih se gradi isti numerisan blok dokaza. Model dobija uputstvo da sastavi tačno n pitanja *odgovorivih isključivo iz priloženih izvoda*, da uz svako pitanje navede broj izvoda i da ne navodi odgovore.

Tema se pretražuje onako kako ju je student napisao. Obogaćivanje upita frazama poput „pitanja za vežbu” pre pretrage pomera upit od gradiva ka opštim formulacijama i vraća slabije odlomke, pa se ne primenjuje.

Kviz kao objekat, a ne kao tekst. Pitanja i njihovi odgovori generišu se *u jednom prolazu* i čuvaju zajedno, kao jedinstven objekat. Alternativa, u kojoj bi se sačuvao samo tekst pitanja a odgovor naknadno ponovo izveo iz teme, deluje jednostavnije, ali daje odgovor na *neko* pitanje o toj temi, ne nužno na ono koje je studentu postavljeno. Čuvanjem para pitanje–odgovor to odstupanje je onemogućeno po konstrukciji.

Odgovori se prikazuju tek na izričit zahtev studenta.

Otvoren kviz kao režim rada. Otvoren kviz čuva se u stanju grafa i time menja podrazumevano tumačenje slobodnog teksta: dok je kviz otvoren, ono što student otkuca jeste *odgovor koji treba oceniti*, a ne novo pitanje koje treba potražiti u gradivu.

Ovo je jedini izuzetak od pravila iz odeljka 4.2 da slobodan tekst uvek znači pitanje iz gradiva, i on je nužan: bez njega bi pokušaj odgovora na prvo pitanje kviza bio protumačen kao novo pitanje, pa bi student umesto ocene dobio rešenje potraženo u gradivu. Pošto stanje živi u grafu, oba interfejsa ponašaju se pri tome identično.

Ocenjivanje. Ocenjivanje dobija i konkretno pitanje i sačuvani tačan odgovor, pa ocenu daje u tri stepena, kao tačno, delimično tačno ili netačno, praćenu obrazloženjem i tačnim odgovorom prema gradivu. Uputstvo pri tome kao merodavan izvor izričito navodi *gradivo kursa*, iz razloga opisanog u odeljku 4.6.

Ocena se vezuje za pojedinačno pitanje, i to nije formalnost. Kada bi se poređenje vršilo sa temom, potpuno tačan odgovor na jedno pitanje mogao bi biti ocenjen kao nepotpun zato što ne pokriva i preostala pitanja iz iste teme.

Pismo se pritom *izračunava*, a ne traži od modela. Gradivo je pretežno ćirilično, pa model od koga se zatraži latinica ume da sastavi rečenicu koja meša pisma usred reči. Pošto je preslovljavanje ćirilice u latinicu jednoznačno, izlaz se determinističkim postupkom prevodi u latinicu umesto da se model ubeđuje; ćirilični izlaz se ostavlja kakav jeste, jer je obrnut smer dvosmislen i jer se ionako poklapa sa pismom gradiva.

> *Asistent koji ne pamti šta je upravo pitao ne može da vodi proveru znanja. Razlika između alata koji odgovara na pitanja i asistenta za učenje jeste stanje razgovora.*

Ako model nije dostupan, kviz se degradira na izvode iz gradiva uz zahtev da student odgovori na osnovu njih, što je manje korisno, ali i dalje utemeljeno.

4.9 Korisnički interfejs

Sistem se pokreće u dva režima. Terminalski režim služi razvoju i merenju, a bot na platformi Telegram je interfejs namenjen studentima.

Komande bota i njihovi opisi drže se na jednom mestu, odakle se istovremeno generiše i tekst pomoći i spisak komandi koji se objavljuje platformi, pa dva opisa ne mogu da se razidu. Tastature se grade iz kataloga kursa, pa dodavanje novog kursa ili lekcije dodaje odgovarajuću dugmad bez izmene koda. Odgovori duži od ograničenja platforme dele se na granicama pasusa.

Uvodna poruka se generiše u trenutku pokretanja: navodi kurseve koji su stvarno indeksirani i predlaže primere pitanja sastavljene od stvarnih naslova slajdova, uz odbacivanje naslova koji su administrativne ili navigacione prirode. Time ostaje tačna kako se gradivo menja i pokrivenost demonstrira umesto da je opisuje.

4.10 Primer interakcije

Naredni primer prikazuje tipičnu razmenu, u kojoj je pitanje postavljeno latinicom i bez dijakritika, a odgovor sastavljen isključivo iz pronađenih odlomaka.

Student: *Kako se uspostavlja veza u tri koraka?*

Asistent *[Kurs: RM]***:**

> Veza se uspostavlja razmenom tri segmenta, pri čemu svaka strana potvrđuje početni redni broj one druge [1]. Izbor slučajnog početnog rednog broja štiti od zabune sa segmentima prethodne veze koji su zaostali u mreži [2].
>
> *Izvori:*
> [1] 07.Transportni.sloj.pdf, slajd 21–24, Uspostavljanje veze
> [2] 07.Transportni.sloj.pdf, slajd 25–26, Uspostavljanje veze

Na primeru se vide tri svojstva o kojima je bilo reči: odgovor je sastavljen isključivo iz pronađenih odlomaka, svaka tvrdnja nosi broj izvoda, a citat upućuje na prezentaciju i opseg slajdova, tako da student može otvoriti materijal i proveriti navod.

Ako student zatim zatraži šire objašnjenje, ono stiže kao zasebna poruka, sa vidljivim upozorenjem da nije iz gradiva kursa i bez ijednog citata, u skladu sa pravilom iz odeljka 4.6. Kada gradivo ne pokriva pitanje, sistem ne sastavlja odgovor od labavo povezanih odlomaka, nego saopštava da traženog materijala nema i predlaže preformulisanje pitanja ili proveru izbora kursa. Isto pitanje postavljeno ćirilicom dobija isti odgovor, ispisan ćirilicom.

Glava 5

Evaluacija

Provera sistema odvija se na dva nivoa. Prvi je ispravnost, odnosno pitanje radi li sistem ono što tvrdi da radi; njemu je posvećen odeljak 5.1 i on prethodi svemu ostalom, jer merenje kvaliteta nad neispravnim sistemom nema smisla. Drugi nivo je kvalitet, koji se ocenjuje po tri osnove: tačnosti, operativnim troškovima i upotrebljivosti. Za prvu je u ovom radu dat potpun i ponovljiv postupak merenja, za drugu procena, dok upotrebljivost nije merena.

Tačnost je merena u punom obimu, i to na sloju pretrage. Razlog je što se u tom sloju nalaze sve izmene opisane u glavi 4, a merenje kvaliteta pretrage je objektivno i ponovljivo, za razliku od ocene kvaliteta generisanog teksta. Odeljci 5.3–5.8 bave se time.

Operativni troškovi razloženi su u odeljku 5.9, delom iz merenja, delom iz procene koja je kao takva označena.

Upotrebljivost nije merena. Sistem nije izložen grupi studenata u kontrolisanim uslovima, pa se o zadovoljstvu korisnika u ovom radu ne iznosi nikakav nalaz. Postavka takvog merenja opisana je u odeljku 6.1.

Taj nesrazmer se navodi već ovde; ograničenja merenja razložena su u odeljku 5.11. Cilj merenja tačnosti pri tome nije da pokaže da sistem radi, nego da *razdvoji* doprinos pojedinačnih komponenti.

5.1 Provera ispravnosti

Sistem prati skup automatskih testova organizovanih po modulima, koji se izvršava pri svakoj izmeni. Njihova uloga nije samo zaštita od regresije, nego i *izvršna specifikacija* odluka opisanih u glavi 4: svaka tvrdnja koja se u toj glavi iznosi ima test koji pada ako prestane da važi.

Testovi se mogu razvrstati u četiri grupe.

Jezički sloj. Proverava se da preslovljavanje ćirilice u latinicu ispravno obrađuje digrafe u zavisnosti od veličine slova okoline, da uklanjanje dijakritika svodi oblike sa dijakriticima i bez njih na isti niz jedinica, i da tokenizacija za leksičku pretragu čuva stručne oznake celovitim. Ova grupa je najbrojnija, jer je reč o sloju od koga zavise sve ostale komponente.

Priprema podataka. Ključna je invarijanta pokrivenosti: test deli poznat tekst na odlomke i proverava da se svaka rečenica ulaza javlja bar u jednom od njih. Uz nju se proverava da izbor strategije segmentacije odgovara obliku dokumenta, da kontekstno zaglavlje sadrži očekivane elemente, i da ponovno indeksiranje istog materijala ne stvara duplirane zapise.

Pretraga i generisanje odgovora. Proverava se da se pri vrednosti ispod praga odgovor odbija umesto da bude sastavljen, da se ugovorena niska iz odgovora modela presreće i zamenjuje porukom studentu, da odgovor nosi citate i da se opšte znanje modela nikada ne meša sa utemeljenim odgovorom. Ovi testovi koriste unapred pripremljene odgovore modela umesto stvarnih poziva, pa se izvršavaju bez pristupa mreži i bez troška.

Interfejs i katalog. Proverava se da su komande i njihovi opisi usklađeni, da se struktura kursa ispravno izvodi iz metapodataka indeksa, i da se dugačak odgovor deli na granicama pasusa.

Testovi kojima je potrebna vektorska baza preskaču se kada ona nije izgrađena, umesto da je pokreću, jer je izgradnja indeksa dugotrajna i ne sme biti sporedni efekat pokretanja testova. Time skup testova ostaje upotrebljiv i na računaru na kome indeks nije izgrađen, po cenu toga što se deo provera u tom slučaju ne izvršava.

5.2 Postavka merenja

Merenja su izvršena na istom računaru, na procesoru, bez grafičkog ubrzanja, nad korpusom opisanim u odeljku 3.1. Navedena vremena odziva odnose se na prosečno vreme obrade jednog upita, uključujući vektorizaciju upita, obe grane pretrage, fuziju i ponovno rangiranje.

Izvode ih dva programa priložena uz sistem: jedan računa metrike za jednu konfiguraciju i dopisuje red u zbirnu tabelu rezultata, a drugi izvodi analizu doprinosa pojedinačnih komponenti. Oba upisuju svoje rezultate u datoteke koje se čuvaju uz izvorni kod, tako da je svaki broj naveden u ovoj glavi moguće ponovo dobiti; uputstvo za pokretanje priloženo je uz izvorni kod.

Merenja su izvedena u konfiguraciji iz tabele 5.1.

Tabela 5.1: Konfiguracija u kojoj su izvedena merenja

Parametar	Vrednost
Model vektorizacije	`intfloat/multilingual-e5-large`
Model za ponovno rangiranje	`cross-encoder/mmarco-mMiniLMv2-L12-H384-v1`
Broj kandidata po grani pretrage	40
Broj kandidata prosleđenih ponovnom rangiranju	20
Broj odlomaka u odgovoru	6
Konstanta prigušenja pri fuziji	60
Prag odbijanja odgovora	$-3{,}0$
Pismo odgovora	prema pismu pitanja

5.3 Skup pitanja za evaluaciju

Sva merenja izvedena su nad zvaničnim spiskovima ispitnih pitanja dva kursa, ukupno 90 pitanja (tabela 5.2). Uz svako pitanje spisak navodi i gradivo iz kog se odgovor priprema, pa je referentni izvor izveden iz spiska, a ne sastavljen uz sistem. Time je unapred otklonjena zamerka da je sistem podešavan prema pitanjima na kojima se meri, koja se kod skupa sastavljenog tokom razvoja ne bi mogla ni potvrditi ni opovrgnuti.

Referentni odgovor je za svako pitanje datoteka u kojoj se traženo gradivo nalazi, a za 30 pitanja spisak uz to navodi i opseg strana. Ta razlika je jedini razlog zbog kog se u nastavku razlikuju dva nivoa strogosti.

Tabela 5.2: Sastav skupa pitanja za evaluaciju

Svojstvo	Vrednost
Broj pitanja	90
Pitanja sa označenim opsegom strana	30
Različitih očekivanih datoteka	18
Indeksiranih datoteka u korpusu	60

Latinične varijante. Oba spiska pisana su ćirilicom, pa sama po sebi ne govore ništa o tome šta se dešava kada student isto pitanje otkuca latinicom. Zato je uz njih izgrađen i skup od 90 latiničnih varijanti, dobijen preslovljavanjem teksta pitanja istom funkcijom koju koristi i sam sistem, pri čemu su očekivani izvori ostali nepromenjeni. Ta dva skupa razlikuju se u pismu i ni u čemu drugom, pa razlika u rezultatu meri isključivo rukovanje pismom (odeljak 5.6). Latinična pitanja su, dakle, izvedena, a ne objavljena kao takva.

5.4 Metrike i dva nivoa strogosti

Mere su $R@1$, $R@5$, $R@10$, MRR i nDCG@10, definisane u odeljku 2.8. Relevantnost se procenjuje na dva nivoa strogosti:

Nivo datoteke. Pogodak je bilo koji pronađeni odlomak koji potiče iz očekivane datoteke.

Nivo strane. Dodatno se zahteva da se opseg strana odlomka preklapa sa očekivanim opsegom.

Mera na nivou strane računa se samo nad 30 pitanja koja nose označen opseg, pa se njene vrednosti porede između konfiguracija, a ne sa merom na nivou datoteke.

5.5 Napredak kroz konfiguracije

Tabela 5.3 prati kvalitet pretrage kroz četiri konfiguracije, gde svaka naredna dodaje po jednu komponentu na prethodnu. Polazna tačka je model vektorizacije treniran na engleskom jeziku, čije je ponašanje na srpskoj ćirilici dijagnostikovano

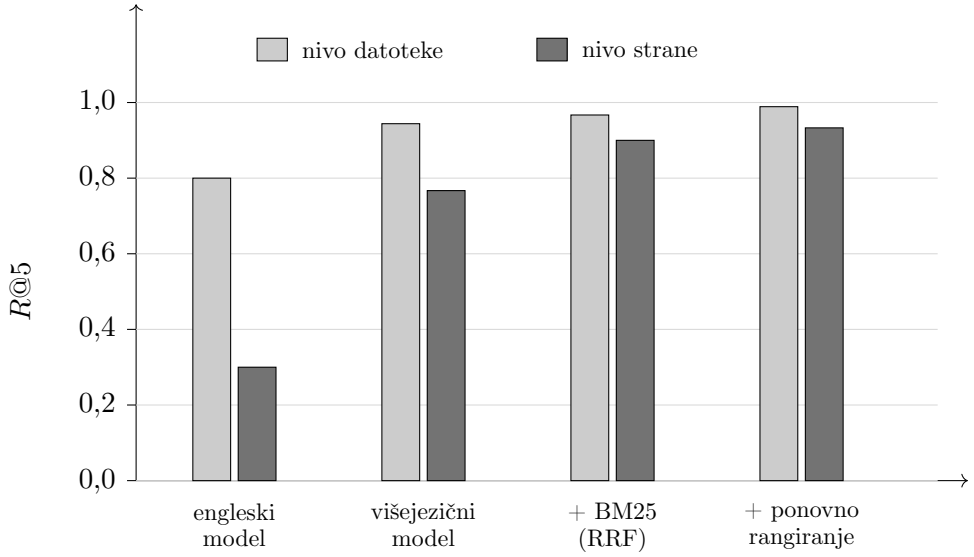
u odeljku 3.3. Sve ostalo, od ekstrakcije i segmentacije do normalizacije pisma, isto je u svim redovima.

Tabela 5.3: Napredak kvaliteta pretrage po konfiguracijama ($N = 90$; nivo strane $N = 30$)

	nivo datoteke					nivo strane
Konfiguracija	*R*@1	*R*@5	*R*@10	MRR	nDCG@10	*R*@5
Engleski model vektorizacije (samo gusta grana)	0,600	0,800	0,889	0,696	0,710	0,300
Višejezični model `e5-large` (samo gusta grana)	0,800	0,944	0,967	0,862	0,852	0,767
+ BM25 hibrid (RRF)	0,856	0,967	1,000	0,911	0,899	0,900
+ ponovno rangiranje (isporučeno)	**0,822**	**0,989**	**0,989**	**0,895**	**0,883**	**0,933**

Broj potpunih promašaja, odnosno pitanja kod kojih među prvih deset rezultata nema nijednog tačnog odlomka, pao je sa deset na jedan, od ukupno 90 pitanja.

Slika 5.1 prikazuje isti napredak za meru *R*@5, na oba nivoa strogosti.



Slika 5.1: Napredak mere *R*@5 kroz konfiguracije, na oba nivoa strogosti ($N = 90$ i $N = 30$)

Glavni nalaz. Najveći pojedinačni skok dolazi od zamene modela vektorizacije: *R*@5 raste sa 0,800 na 0,944, a na nivou strane sa 0,300 na 0,767. Od ukupnog poboljšanja mere *R*@5, koje iznosi 0,189, na taj jedan korak otpada 0,144, dakle

preko tri četvrtine; na nivou strane odnos je približno isti, jer od ukupnih 0,633 na zamenu modela otpada 0,467.

To je direktna potvrda teze rada postavljene u glavi 1. Uzrok slabog kvaliteta bio je u sloju reprezentacije teksta, a ne u mašineriji pretrage nadograđenoj iznad njega, niti u sloju generisanja odgovora, koji tokom celog rada nije menjan.

Poređenje po nivoima strogosti pokazuje i *kako* engleski model omašuje. Razlika na nivou datoteke je osetna (0,800 naspram 0,944), ali na nivou strane dramatična (0,300 naspram 0,767). Drugim rečima, taj model ne samo da često promaši pravu prezentaciju, nego i kada je pogodi, po pravilu ne pogodi mesto u njoj. To je očekivana posledica degeneracije vektorskog prostora izmerene u odeljku 3.3: kada su sve rečenice približno jednako udaljene, rangiranje unutar jednog dokumenta gubi svaki oslonac.

> *Redovi tabele razlikuju se* samo *po navedenoj komponenti. Vrednost 0,144 zato meri doprinos modela vektorizacije, dok zbirni doprinos svih ispravki u sloju pripreme podataka ostaje izvan ovog merenja. Doprinos ispravki opisanih u odeljku 3.4 nije zasebno kvantifikovan, jer polazni oblik tih komponenti nije zadržan u sistemu; zasebno merenje tog doprinosa navedeno je među pravcima daljeg rada (odeljak 6.1).*

Sporedni nalaz. Uvođenje BM25 grane popravlja sve mere odjednom: $R@1$ sa 0,800 na 0,856, $R@10$ sa 0,967 na 1,000 i MRR sa 0,862 na 0,911, uz zanemarljiv trošak, jer se leksički indeks pretražuje za nekoliko milisekundi. Ponašanje poslednjeg koraka, ponovnog rangiranja, složenije je i razloženo je u odeljku 5.7.

5.6 Rukovanje pismom

Normalizacija pisma pri indeksiranju je prva i najdalekosežnija odluka u sistemu (odeljak 4.3), pa uz obrazloženje traži i merenje. Pitanje na koje se odgovara glasi: menja li se išta kada student isto pitanje otkuca latinicom umesto ćirilicom. Merenje je izvedeno nad 90 zvaničnih pitanja i nad njihovih 90 latiničnih varijanti, koje se od njih razlikuju *isključivo* po pismu (tabela 5.4).

Vrednosti su jednake do poslednje decimale, uz isti jedini promašaj u oba pisma. Isto važi i za merenje na nivou strane, gde oba pisma daju $R@5 = 0{,}933$ i $MRR = 0{,}716$.

Tabela 5.4: Isti upiti postavljeni u oba pisma ($N = 90$)

Pismo upita	R@1	R@5	R@10	MRR
Ćirilica, kako spisak glasi	0,822	0,989	0,989	0,895
Latinica, preslovljeno	0,822	0,989	0,989	0,895

Rezultat je neposredna posledica same odluke: pošto se i gradivo i upit preslovljavaju u latinicu pre pretrage, sistem u trenutku pretrage uopšte ne vidi dva pisma, nego jedno. Pitanje postavljeno ćirilicom i njegova latinična varijanta postaju identičan niz znakova, pa je jednakost rezultata očekivana posledica, a merenje potvrđuje da je sprovedena bez izuzetka, uključujući i digrafe i reči sa dijakriticima.

Vredi primetiti šta bi bila alternativa. Da se pismo procenjuje u trenutku upita, kao u dijagnostikovanoj polaznoj konfiguraciji (odeljak 3.5), ovakvo poklapanje ne bi bilo moguće ni načelno, jer bi se dve varijante istog pitanja razlikovale po tome koliko je varijanti generisano i kojim su redom spojene.

5.7 Doprinos pojedinačnih komponenti

Da bi se utvrdilo šta pojedina komponenta zaista donosi, ona se uklanja iz sistema, dok sve ostalo ostaje nepromenjeno, pa se kvalitet ponovo meri; razlika u rezultatu pripisuje se uklonjenoj komponenti. U literaturi o mašinskom učenju takav postupak poznat je pod nazivom *ablation study*. Tabela 5.5 prikazuje rezultate dobijene nad *istim* indeksom, sa modelom vektorizacije `intfloat/multilingual-e5-large`, tako da se poredi isključivo arhitektura pretrage.

Dve komponente ponašaju se bitno različito i vredi ih razdvojiti.

Leksička grana se isplati bez ostatka. Njeno dodavanje gustoj pretrazi podiže R@5 sa 0,944 na 0,967, R@10 sa 0,967 na 1,000 i MRR sa 0,862 na 0,911, a da pri tome ne košta gotovo ništa, jer se BM25 indeks pretražuje za nekoliko milisekundi. To potvrđuje očekivanje iz odeljka 4.5: dve grane greše na različitim mestima, pa se njihovim spajanjem pokriva unija, a ne presek.

Ponovno rangiranje plaća se skupo, a ne donosi sve. Ono podiže R@5 sa 0,967 na 0,989, dakle uvlači u prvih pet dva pitanja koja bez njega ostaju napolju. Istovremeno, međutim, obara R@1 sa 0,856 na 0,822, MRR sa 0,911 na 0,895 i

Tabela 5.5: Doprinos pojedinačnih komponenti pretrage, mereno nad istim indeksom ($N = 90$)

Konfiguracija	R@1	R@5	R@10	MRR	nDCG@10	Vreme odziva
samo gusta grana	0,800	0,944	0,967	0,862	0,852	216 ms
gusta + mmarco-mMiniLMv2 (118 M)	0,833	0,967	0,967	0,896	0,883	2738 ms
gusta + bge-reranker-v2-m3 (568 M)	0,833	0,956	0,978	0,896	0,897	27.300 ms
gusta + BM25	0,856	0,967	1,000	0,911	0,899	132 ms
gusta + BM25 + mmarco (isporučeno)	**0,822**	**0,989**	**0,989**	**0,895**	**0,883**	1944 ms
gusta + BM25 + bge-reranker-v2--m3	0,833	0,978	0,989	0,900	0,882	31.498 ms

R@10 sa 1,000 na 0,989, i to uz vreme odziva petnaest puta duže. Drugim rečima, na ovom skupu ponovno rangiranje popravlja pokrivenost prvih pet, ali blago kvari poredak na samom vrhu liste.

> *Nalaz treba čitati zajedno sa ulogom koju unakrsni enkoder ima izvan rangiranja. Njegov skor nosi i prag ispod kog sistem odbija da odgovori (odeljak 4.6), pa njegovo uklanjanje ne bi bilo samo ubrzanje, nego i gubitak jedine mere kojom se pitanje van gradiva prepoznaje* pre *poziva modela. Drugi sloj odbijanja time ne nestaje, ali deluje tek pošto je model već pozvan i izvode pročitao. Zato je unakrsni enkoder u isporučenom sistemu zadržan, uprkos tome što se po samom rangiranju ne isplati jednoznačno.*

Veličina unakrsnog enkodera nije prednost

Poređenje dva unakrsna enkodera daje nalaz suprotan i intuiciji da je veći model bolji i poretku ova dva modela na opštim skupovima za poređenje. Model od 118 miliona parametara i model od 568 miliona razlikuju se po R@5 za 0,011 i po meri MRR za 0,005, što na 90 pitanja odgovara razlici od jednog pitanja, dakle šumu. Po kvalitetu su izjednačeni.

Ono što ih razdvaja je cena: 1,9 sekunde naspram 31,5 sekundi po upitu, dakle **oko šesnaest puta**. Razlog je u tome što unakrsni enkoder, za razliku od modela vektorizacije, ne može ništa unapred da izračuna, nego mora da propusti kroz mrežu svaki par pitanja i odlomka; pri dvadeset kandidata po upitu, veličina modela ulazi u cenu dvadeset puta po pitanju.

Praktična posledica je da izbor manjeg modela ovde *nije* kompromis između kvaliteta i brzine, jer kvalitet koji se žrtvuje nije merljiv, dok je ušteda u vremenu red veličine. Hipoteza koja bi objasnila zašto veći model ne donosi prednost jeste da je podešavan za duže odlomke, dok su odlomci u ovom sistemu kratke grupe slajdova telegrafskog stila, prosečne dužine oko 1100 znakova; provera te hipoteze ostavljena je za dalji rad.

5.8 Uticaj ograničavanja pretrage na kurs

Sistemi ovog tipa po pravilu sadrže sloj koji upit pre pretrage usmerava na odgovarajući podskup zbirke, u ovom slučaju na jedan kurs. Takav sloj nosi trajan trošak održavanja, jer se lista ključnih reči mora dopunjavati za svaki novi kurs, pa je pre nego što bude zadržan opravdano proveriti šta zapravo donosi. U tu svrhu upoređena su dva režima nad istim skupom pitanja (tabela 5.6), a zasebno je izmereno i koliko se pouzdano kurs može pročitati iz samih rezultata pretrage.

Tabela 5.6: Uticaj ograničavanja pretrage na kurs ($N = 90$)

Režim	R@5
Pretraga ograničena na tačan kurs (gornja granica)	0,989
Pretraga bez ikakvog ograničenja na kurs	0,989

Ograničavanje pretrage na tačan kurs ne donosi *nikakvu* razliku u odnosu na pretragu nad celim indeksom: obe daju istu vrednost. Materijal drugog kursa, dakle, ne uspeva da istisne tačan odlomak iz prvih pet ni kada mu se to dozvoli.

Zaseban podatak, koji nije mera dohvatanja pa stoji izvan tabele, jeste tačnost prepoznavanja kursa: ako se kurs *izvede* iz rezultata, kao kurs najbolje rangiranog odlomka, pogađa se u **svih 90 slučajeva**. Uz takav rezultat prepoznavanje iz rezultata ne može biti lošije od ručno održavane liste ključnih reči, a ne zahteva nikakvo održavanje.

Nalaz je negativan po svom obliku, ali nosi konkretnu projektnu posledicu: na osnovu njega je cela lista ključnih reči uklonjena iz sistema, čime je nestala i klasa grešaka u kojoj pogrešno prepoznat kurs u potpunosti isključuje materijal koji bi na pitanje odgovorio. Reč je o opštem obrascu koji se u radu ponavlja: heuristika iznad slabe pretrage izgleda kao poboljšanje sve dok se pretraga ne popravi, a potom postaje samo izvor grešaka.

5.9 Operativni troškovi

Niski operativni troškovi bili su jedan od ciljeva pri projektovanju sistema, pa ih vredi razložiti. Trošak može da nastane na dva mesta: u sloju pretrage i u sloju generisanja.

Pretraga ne stvara trošak po upitu. Model vektorizacije, leksički indeks i unakrsni enkoder izvršavaju se *lokalno*, na procesoru računara na kome sistem radi. Nijedan od tri koraka pretrage ne šalje upit spoljnom servisu, pa je granični trošak jednog pretraživanja jednak nuli, a cena je izražena isključivo u vremenu, oko dve sekunde po upitu, uz jednokratnih dvanaest minuta za izgradnju indeksa. Posledica je i da gradivo ne napušta računar u fazi indeksiranja.

Generisanje je jedini plaćeni deo. Trošak nastaje samo pri sastavljanju odgovora, kviza ili ocene, i srazmeran je veličini bloka dokaza. Tabela 5.7 razlaže procenu za jedan odgovor.

Tabela 5.7: Procena troška jednog utemeljenog odgovora

Stavka	Količina	Osnov
Prosečna dužina indeksiranog odlomka	1117 znakova	izmereno nad indeksom
Odlomaka u bloku dokaza	6	podrazumevana vrednost
Blok dokaza	oko 6700 znakova	izvedeno
Sistemsko uputstvo i pitanje	oko 1000 znakova	procena
Ulaz, ukupno	oko 7700 znakova	izvedeno
Ulaz u tokenima	2200–3100	pri 2,5–3,5 znakova po tokenu
Izlaz u tokenima	300–600	tipičan odgovor
Trošak po odgovoru	**0,012–0,018 USD**	pri 3 USD/MTok ulaz, 15 USD/MTok izlaz

Broj odlomaka i njihova prosečna dužina su izmereni; pretvaranje znakova u tokene je *procena*, jer tokenizator korišćenog modela nije javno dostupan, a srpski tekst se po pravilu deli na više tokena po znaku nego engleski. Cena po milionu tokena odgovara tarifi modela klase Sonnet u trenutku pisanja rada i podložna je izmenama, pa je merodavan podatak broj tokena, a ne iznos u dolarima.

Uz taj red veličine, hiljadu pitanja mesečno košta petnaestak dolara. Dva svojstva sistema taj iznos dodatno smanjuju: odbijanje odgovora ispod praga uopšte ne poziva model kada gradivo ne pokriva pitanje, a izostavljanje dodatne provere

odgovora drži broj poziva modela po odgovoru na jednom; oba su obrazložena u odeljku 4.6.

Šta nije uračunato. Procena ne obuhvata trošak računara na kome sistem radi, niti eventualno grafičko ubrzanje koje bi vreme odziva pretrage smanjilo za red veličine uz odgovarajući trošak zakupa. Takođe, nije mereno koliko se odgovora u stvarnoj upotrebi odbija pre poziva modela, pa je navedeni iznos gornja granica po pitanju, a ne očekivani prosek.

5.10 Diskusija

Rezultati podupiru tezu rada na tri načina.

Prvo, poboljšanje dolazi ispod sloja pretrage. Zamena modela vektorizacije nosi preko tri četvrtine ukupnog porasta mere $R@5$, dok sistemski upit jezičkom modelu tokom celog rada nije menjan. Odluka o normalizaciji pisma, doneta u istom sloju, pokazuje se merenjem kao potpuna: isto pitanje postavljeno u oba pisma daje istovetne rezultate.

Drugo, komponente koje se u pregledima RAG sistema navode kao glavni izvor kvaliteta [12] daju manji doprinos od reprezentacije teksta, i to tek pošto je osnova ispravljena. Nad degenerisanim vektorskim prostorom polazne konfiguracije nijedna od njih ne bi imala šta da rangira.

Treće, tri nalaza pokazuju da složenije nije automatski bolje. Unakrsni enkoder od 568 miliona parametara ne donosi ništa u odnosu na model od 118 miliona, uz šesnaest puta veći trošak. Sloj usmeravanja po kursu ne donosi ništa u odnosu na pretragu nad celim indeksom. I sámo ponovno rangiranje, iako podiže $R@5$, blago obara $R@1$ i MRR uz petnaestostruko duže vreme odziva, pa se njegovo mesto u sistemu brani ulogom koju ima pri odbijanju pitanja van gradiva, a ne rangiranjem.

Poslednji nalaz vredi izdvojiti, jer se u radu ponavlja u tri različita oblika. Sloj dodat iznad slabe pretrage izgleda kao poboljšanje sve dok se sama pretraga ne popravi; posle toga isti sloj postaje ili suvišan ili izvor grešaka. To važi za listu ključnih reči po kursu, za veći unakrsni enkoder i, delimično, za sámo ponovno rangiranje.

Vreme odziva od oko dve sekunde po upitu prihvatljivo je za interakciju preko Telegrama, gde je i inače prisutno kašnjenje mreže, i najvećim delom potiče od

ponovnog rangiranja na procesoru. Grafičko ubrzanje bi ga smanjilo za red veličine.

5.11 Ograničenja merenja

Skup od 90 pitanja je malog obima, pa jedno pitanje vredi 0,011 i razlike tog reda veličine ne mogu se pouzdano razlikovati od šuma; zaključci se izvode samo iz većih razlika, a upravo zato je poređenje dva unakrsna enkodera u odeljku 5.7 izneto kao izjednačenost, a ne kao prednost jednog od njih. Poreklo pitanja jeste nezavisno od sistema, ali skup time nije postao idealan. Jedan od dva kursa obuhvata svega devet datoteka, pa je za njegova pitanja pogađanje tačne datoteke među prvih pet zadatak koji zasićuje, i zbirna vrednost je utoliko optimističnija nego što bi bila nad ravnomernijim korpusom. Uz to, spiskovi ne pokrivaju sav indeksirani sadržaj, pa deo gradiva ostaje izvan merenja.

Merenja su izvedena na jednoj računarskoj konfiguraciji, procesoru bez grafičkog ubrzanja, nad materijalima na srpskom jeziku i samo dva kursa. Izmerena vremena odziva specifična su za tu konfiguraciju i mogla bi se razlikovati na grafičkoj kartici, a uopštavanje zaključaka o odnosu veličine i kvaliteta modela na druge korpuse i jezike nije opravdano bez ponovljenog merenja.

Merena je pretraga, a ne kvalitet konačnog odgovora; pretpostavka da bolji odlomci daju bolji odgovor je razumna, ali u ovom radu nije proverena, a merenje vernosti odgovora izvorima [11] ostaje za dalji rad. Iz istog razloga sistem nije proveravan sa stvarnim korisnicima, pa nijedan nalaz o njegovoj upotrebljivosti u svakodnevnoj upotrebi nije potkrepljen podacima.

Prag ispod kog se odgovor odbija kalibrisan je merenjem (tabela 4.3), ali nad uzorkom pitanja van gradiva koji se pokazao suviše lakim. Tema koja je gradivu bliska a u njemu je nema, kakvo je Hafmanovo kodiranje u kursu koji obrađuje Hamingov kod, dobija ocenu +1,76, dakle visoko iznad praga; i očigledno nepovezana tema dobija −1,99, i dalje iznad njega. Procep iz tabele 4.3 stoga ne važi uopšteno, pa takve upite zapravo odbija drugi sloj, sud modela nad pročitanim izvodima, a ne mera sličnosti. Sistematska kalibracija nad težim uzorkom ostaje za dalji rad.

Deo troška je procenjen: broj tokena izveden je iz izmerene dužine odlomaka uz pretpostavku o odnosu znakova i tokena, jer tokenizator modela nije javno dostupan (odeljak 5.9).

Konačno, međujezička pretraga ovim merenjem uopšte nije obuhvaćena. Deo

indeksiranog gradiva u celini je na engleskom jeziku, ali ga spiskovi ispitnih pitanja ne pokrivaju, pa nijedno od 90 pitanja ne traži gradivo na drugom jeziku od onog na kome je postavljeno. Sposobnost sistema da srpskim upitom pronađe englesko gradivo ostaje, dakle, neizmerena, iako je reč o slučaju koji se u praksi javlja i za koji višejezični modeli vektorizacije po pravilu daju slabije rezultate.

Glava 6

Zaključak

U ovom radu projektovan je, implementiran i izmeren agentski asistent namenjen studentima Matematičkog fakulteta, dostupan preko interfejsa platforme Telegram. Sistem pod istim interfejsom objedinjuje odgovaranje iz nastavnih materijala, generisanje kvizova sa ocenjivanjem odgovora i pregled gradiva izveden iz samog indeksa. Sadržaj kurseva odvojen je od logike programa, pa se skup podržanih kurseva, a načelno i institucija, proširuje bez izmene izvornog koda.

Središnji alat, i onaj kome je posvećen istraživački deo rada, jeste odgovaranje iz gradiva: sistem odgovara isključivo iz indeksiranog materijala i uz svaku tvrdnju navodi prezentaciju i stranu iz koje ona potiče.

Metodološki, rad polazi od merenja umesto od pretpostavke. Za svaki sumnjivi deo sistema formulisan je zaseban test, čime je omogućeno da se uzroci slabog kvaliteta izoluju pojedinačno. Nijedan od utvrđenih uzroka ne leži u sloju generisanja odgovora, nego u reprezentaciji teksta, pripremi podataka i arhitekturi pretrage.

Predloženo rešenje počiva na četiri odluke. Prva je normalizacija pisma pri indeksiranju, kojom se problem dvopismenosti rešava jednom, na mestu na kome je rešiv, umesto procenom u trenutku upita. Druga je ekstrakcija svesna strukture slajda, koja naslov izdvaja pomoću metrika fonta i uklanja ponavljajuća zaglavlja. Treća je segmentacija prilagođena obliku dokumenta, sa kontekstnim zaglavljem koje svakom odlomku vraća izgubljeni kontekst i sa invarijantom pokrivenosti koja je zapisana kao izvršni test. Četvrta je hibridna pretraga koja gustu i leksičku granu spaja recipročnom fuzijom rangova i ponovo rangira unakrsnim enkoderom.

Merenje je izvedeno nad 90 pitanja preuzetih iz zvaničnih spiskova ispitnih pitanja oba kursa, kod kojih ni formulacija ni očekivani izvor nisu nastali uz sistem. Odziv $R@5$ raste sa 0,800 na 0,989, na nivou strane sa 0,300 na 0,933, a broj

potpunih promašaja pada sa deset na jedan, *bez ijedne izmene sistemskog upita* koji se šalje jezičkom modelu. Preko tri četvrtine tog poboljšanja otpada na zamenu modela vektorizacije, dakle na sloj reprezentacije teksta. Time je teza rada potvrđena.

Odluka o normalizaciji pisma potvrđena je zasebno: isto pitanje postavljeno ćirilicom i latinicom daje istovetne vrednosti svih mera, jer sistem u trenutku pretrage i ne vidi dva pisma, nego jedno.

Merenje doprinosa pojedinačnih komponenti dalo je i nalaze koji se protive uobičajenoj intuiciji. Unakrsni enkoder od 568 miliona parametara ne donosi nikakvu merljivu prednost nad modelom od 118 miliona, uz šesnaest puta duže vreme odziva, pa izbor manjeg modela nije kompromis. Usmeravanje pretrage po kursu na osnovu ručno održavane liste ključnih reči ne donosi nikakvo poboljšanje kada je pretraga dovoljno dobra, pa je uklonjeno u celini; kurs izveden iz najbolje rangiranog odlomka pogađa se u svim izmerenim slučajevima. Oba nalaza primeri su opšteg obrasca u kome sloj iznad slabe pretrage nadoknađuje njen nedostatak, a posle njenog popravljanja postaje suvišan ili postaje izvor grešaka.

Šire posmatrano, iz merenja sledi jedno praktično pravilo: *kvalitet takvog sistema određen je kvalitetom najslabije karike u lancu pripreme podataka, a ne snagom jezičkog modela na njegovom kraju.* Za jezike sa ograničenim resursima, kakav je srpski, ta karika je gotovo uvek reprezentacija teksta.

6.1 Dalji razvoj aplikacije

Isporučeni sistem uži je od prvobitno zamišljenog: priprema planova učenja, praćenje studentskih obaveza, upis ispitnih termina u kalendar i nadzor nad izmenama na stranicama fakulteta nisu njegov deo. Tokom razvoja su te komponente bile započete, ali nisu dovedene do upotrebne vrednosti, pa su izostavljene iz isporučenog sistema umesto da se kao funkcionalnost opisuje nešto što to nije. Dalji razvoj kreće se u sledećim pravcima:

Profil studenta. Trajno stanje sa upisanim kursevima, istorijom postavljenih pitanja i rezultatima rešenih kvizova, kao osnova za personalizaciju: praćenje napretka kroz vreme, prepoznavanje tema na kojima student sistematski greši i usmeravanje provere znanja na njih. Na isti oslonac naslanja se i generisanje plana učenja, sa raspodelom vremena prema obimu gradiva po temi.

Praćenje ostalih kurseva i ispitnih rokova. Upis ispitnih termina u kalendar, podsećanje na obaveze i nadzor nad izmenama na stranicama fakulteta koje student odabere.

Postavljanje sistema u oblak. Prebacivanje sa lokalnog izvršavanja na infrastrukturu u oblaku (engl. *cloud*), radi dostupnosti većem broju studenata.

Literatura

[1] Anthropic. Introducing contextual retrieval. https://www.anthropic.com/news/contextual-retrieval, 2024. Pristupljeno 2026. godine.

[2] Artifex Software. PyMuPDF documentation. https://pymupdf.readthedocs.io/, 2025. Pristupljeno 2026. godine.

[3] Vuk Batanovic, Bosko Nikolic, and Milan Milosavljevic. Reliable baselines for sentiment analysis in resource-limited languages: The Serbian movie review dataset. In *Proceedings of LREC 2016*, page 2688–2696, 2016. https://aclanthology.org/L16-1427/.

[4] Luiz Bonifacio, Vitor Jeronymo, Hugo Queiroz Abonizio, Israel Campiotti, Marzieh Fadaee, Roberto Lotufo, and Rodrigo Nogueira. mMARCO: A multilingual version of the MS MARCO passage ranking dataset. *arXiv preprint arXiv:2108.13897*, 2021. https://arxiv.org/abs/2108.13897.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. https://arxiv.org/abs/2005.14165.

[6] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024. https://arxiv.org/abs/2402.03216.

[7] Chroma. Chroma: the AI-native open-source embedding database. https://www.trychroma.com/, 2025. Pristupljeno 2026. godine.

[8] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzman, Edouard Grave, Myle Ott, Luke Zet-

tlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL 2020*, page 8440–8451, 2020. https://arxiv.org/abs/1911.02116.

[9] Gordon V. Cormack, Charles L. A. Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms Condorcet and individual rank learning methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 758–759, 2009. https://doi.org/10.1145/1571941.1572114.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, page 4171–4186, 2019. https://arxiv.org/abs/1810.04805.

[11] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. RAGAS: Automated evaluation of retrieval augmented generation. In *Proceedings of EACL 2024: System Demonstrations*, page 150–158, 2024. https://arxiv.org/abs/2309.15217.

[12] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023. https://arxiv.org/abs/2312.10997.

[13] Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of EACL 2021*, page 874–880, 2021. https://arxiv.org/abs/2007.01282.

[14] Kalervo Jarvelin and Jaana Kekalainen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002. https://doi.org/10.1145/582415.582418.

[15] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. https://arxiv.org/abs/2202.03629.

[16] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972. https://doi.org/10.1108/eb026526.

[17] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP 2020*, page 6769–6781, 2020. https://arxiv.org/abs/2004.04906.

[18] Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 39–48, 2020. https://arxiv.org/abs/2004.12832.

[19] LangChain. LangGraph documentation. https://langchain-ai.github.io/langgraph/, 2025. Pristupljeno 2026. godine.

[20] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuettler, Mike Lewis, Wen tau Yih, Tim Rocktaeschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020. https://arxiv.org/abs/2005.11401.

[21] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. https://arxiv.org/abs/2307.03172.

[22] Nikola Ljubesic and Filip Klubicka. bsWaC, hrWaC and srWaC – web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, page 29–35, 2014. https://aclanthology.org/W14-0405/.

[23] Nikola Ljubesic and Davor Lauc. BERTic – the transformer language model for Bosnian, Croatian, Montenegrin and Serbian. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, page 37–42, 2021. https://aclanthology.org/2021.bsnlp-1.5/.

[24] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schuetze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. https://nlp.stanford.edu/IR-book/.

[25] Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of EACL 2023*, page 2014–2037, 2023. https://arxiv.org/abs/2210.07316.

[26] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019. https://arxiv.org/abs/1901.04085.

[27] Fabio Petroni, Tim Rocktaeschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. Language models as knowledge bases? In *Proceedings of EMNLP-IJCNLP 2019*, page 2463–2473, 2019. https://arxiv.org/abs/1909.01066.

[28] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of EMNLP-IJCNLP 2019*, page 3982–3992, 2019. https://arxiv.org/abs/1908.10084.

[29] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. https://doi.org/10.1561/1500000019.

[30] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, 2023. https://arxiv.org/abs/2302.04761.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, page 5998–6008, 2017. https://arxiv.org/abs/1706.03762.

[32] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022. https://arxiv.org/abs/2212.03533.

[33] Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multilingual E5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*, 2024. `https://arxiv.org/abs/2402.05672`.

[34] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR 2023)*, 2023. `https://arxiv.org/abs/2210.03629`.

Biografija autora

Đorđe Petrović rođen je 15. juna 2001. godine u Moskvi, u Ruskoj Federaciji. Završio je Gimnaziju „Bora Stanković" u Boru, a osnovne akademske studije na Matematičkom fakultetu Univerziteta u Beogradu upisao je 2019. i završio 2023. godine. Master akademske studije na istom fakultetu, studijski program Informatika, upisao je 2023. godine.

Tokom studija interesovao se za mašinsko učenje i veštačku inteligenciju. Zaposlen je u kompaniji Bosch Srbija kao AI softverski inženjer.