# Obrada prirodnih jezika

## Projekat za školsku 2020/2021. godinu

## Tema projekta

Tema predmetnog projekta za školsku 2020/2021 godinu se tiče problema semantičke pretrage blokova programskog koda – funkcija, klasa i modula. Cilj je da se napravi sistem koji na osnovu upita pisanih na srpskom jeziku vraća one blokove programskog koda koji po funkcionalnosti najviše odgovaraju zadatom upitu (npr. za upit „obim kruga“ vraća funkciju koja za zadat poluprečnik kruga računa njegov obim). Svrha ovakve semantičke pretrage jeste omogućavanje korisniku da pronađe, iskopira u svoj projekat i prilagodi postojeći kod koji obavlja željenu funkcionalnost, da pronađe biblioteku koja nudi željenu vrstu obrade, ili da samo bolje razume kako se određena funkcionalnost može implementirati. Radi omogućavanja pretrage, neophodno je da funkcionalnost svakog razmatranog bloka koda bude dokumentovana pomoću odgovarajućih komentara pisanih na srpskom.

Izrada projekta podrazumeva prikupljanje i izdvajanje odgovarajućeg skupa blokova koda, pisanih na određenom programskom jeziku, i njima pratećih dokumentacionih komentara koji opisuju funkcionalnost koda, pisanih na srpskom jeziku. Projekat obuhvata i ručnu anotaciju stepena sličnosti između blokova koda i zadatih upita koji će se koristiti u semantičkoj pretrazi. Tako kreirani skup podataka je zatim potrebno iskoristiti za obučavanje i evaluaciju nekoliko različitih statističkih modela.

Projekti će se izrađivati grupno. Proces prijavljivanja grupe je opisan u odeljku o propozicijama izrade projekta. Projekat se može implementirati u programskom jeziku i paketu po izboru. Pri izradi projekta biće od interesa blokovi koda napisani na jednom od sledećih programskih jezika: Java, JavaScript/TypeScript, Python, PHP, C, C++, C#, Objective C, MATLAB/Octave. Pošto anotacija podataka podrazumeva uvid u kod pisan na određenom programskom jeziku, studenti bi trebalo da se opredele za rad sa onim programskim jezikom koji im je bar u nekoj meri poznat. Iako je osnovna tema projekta ista za sve studente, svaka projektna grupa će razmatrati samo jedan programski jezik i jednu kombinaciju statističkih pristupa za semantičku pretragu. Neophodno je učešće svih članova grupe u svim fazama izrade projekta, tj. nije dozvoljena podela posla između članova grupe po fazama. U nastavku će biti detaljnije opisana svaka od faza u izradi projekta.

### Faza 1 - Prikupljanje podataka

Proces prikupljanja podataka podrazumeva formiranje dovoljno velikog seta parova (komentar, blok koda) za zadati programski jezik. Kao izvor podataka za formiranje ovakvog seta mogu poslužiti lični repozitorijumi studenata (prethodni projektni zadaci za druge predmete, diplomski radovi, itd.), javno dostupne baze repozitorijuma poput GitHub-a, itd. Zbog korišćenja ovako prikupljenih podataka u kasnijim naučnim istraživanjima, važno je da prikupljeni podaci budu ili objavljeni pod licencama koje ne zabranjuju redistribuciju delova koda, ili, ako se radi o projektima samih studenata, da svi članovi grupe budu saglasni da se ovako prikupljeni podaci mogu dalje redistribuirati. Pri prikupljanju podataka obavezno treba ograničiti izbor samo na one parove (komentar, blok koda) kod kojih komentar opisuje funkcionalnost posmatranog bloka koda tj. opisuje šta blok koda radi. Drugim

rečima, komentare koji su bilo kog drugog tipa (npr. opšte informacije o autoru koda, objašnjenja o tome kako se neka funkcija poziva, automatski generisani komentari od strane okruženja, zakomentarisani delovi koda, itd.) i njima odgovarajuće blokove koda treba ignorisati pri izgradnji skupa podataka. Od studenata se ne zahteva da analiziraju da li su dokumentacioni komentari ažurni, tj. da li tačno opisuju funkcionalnost njima pripadajućih blokova koda. Kao blokovi koda mogu se izdvajati funkcije/metode, klase ili moduli, ali nije dozvoljeno kao zasebne blokove koda razmatrati jednostavne getter/setter metode. Pri tome, dozvoljeno je da jedan isti deo koda figurira u više parova, ako se javlja kao deo hijerarhijski različitih blokova koda (npr. jednom kao kod metode u okviru klase, a drugi put kao deo celokupnog koda klase). S druge strane, jedan komentar se može odnositi samo na jedan blok koda.

Očekuje se da među repozitorijumima/projektima koji se koriste za izdvajanje podataka bude onih za koje je izvesno da sadrže funkcije/metode, klase ili module koji su u nekoj meri relevantni za upite koji će biti korišćeni u semantičkoj pretrazi. Drugim rečima, treba izbeći situaciju da između prikupljenog skupa parova (komentar, blok koda) i korišćenih upita ne postoji nikakva povezanost. U slučaju realne nemogućnosti pronalaženja dovoljne količine relevantnih parova sa komentarima na srpskom, dozvoljeno je da se manji broj parova dobije prevodom odgovarajućih komentara sa engleskog na srpski jezik.

Formirani skup treba da sadrži minimalan broj parova (komentar, blok koda) koji je određen veličinom grupe. Kao pozitivan dodatan faktor u ocenjivanju će se uzimati u obzir prikupljanje većeg broja parova od navedenog minimuma. Dozvoljeno je da se u komentarima pisanim na srpskom jeziku javi i neki termin napisan na engleskom.

Prikupljene podatke treba sačuvati u sledećem formatu:

1. Prikupljeni parovi (komentar, blok koda) treba da budu sačuvani u vidu skupa UTF-8 enkodovanih TXT fajlova, gde svaki fajl sadrži tekst jednog para (komentar i kod). Tekst komentara treba da bude odvojen od koda jednim praznim redom. Naziv fajla treba da predstavlja jedinstveni identifikator za svaki par.
2. Pregled svih prikupljenih parova treba dati u vidu jednog tab-separated UTF-8 TXT fajla sa sledećim kolonama za svaki par:
    1. *ProgrammingLanguageName* – ime programskog jezika na kome je kod napisan (Java, PHP, C,…). Ovo polje će u okviru jedne grupe uvek imati istu vrednost, određenu programskim jezikom koji grupa razmatra.
    2. *RepoDescription* – jedinstveni identifikator/opis repozitorijuma/projekta iz koga su podaci dobijeni. Ukoliko je repozitorijum javno dostupan, ovaj identifikator treba da predstavlja URL tog repozitorijuma, a u suprotnom treba da sadrži kratak opis namene korišćenog repozitorijuma/projekta.
    3. *SourceDescription* – jedinstveni identifikator/opis izvora tj. fajla sa programskim kodom iz koga su podaci dobijeni. Ukoliko je fajl sa programskim kodom iz koga su podaci dobijeni javno dostupan, ovaj identifikator treba da predstavlja URL tog fajla, a u suprotnom treba da sadrži kratak opis namene korišćenog fajla.
    4. *PairID* – jedinstveni globalni identifikator za posmatrani par (komentar, blok koda). Ovaj identifikator treba da bude identičan sa nazivom (bez .txt ekstenzije) tekstualnog fajla u kome je sačuvan tekst posmatranog para.

5. *CommentText* – tekst komentara iz posmatranog para. U ovom polju treba čuvati samo koristan tekst komentara, bez znakova koji označavaju početak (ili kraj) komentara u posmatranom programskom jeziku. Kod komentara koji se protežu na više linija koda (ali je jasno da se radi o jednom komentaru) znak za novi red treba kodovati sa *n*, dok znakove za tabulaciju treba kodovati sa *t*.

## Faza 2 – Anotacija podataka

U ovoj fazi potrebno je ručno obeležiti stepen sličnosti između prikupljenog skupa blokova koda i skupa upita na prirodnom jeziku. Kao upite studenti treba da koriste sopstvene prevode na srpski sledećih 99 upita pisanih na engleskom jeziku:

https://github.com/github/CodeSearchNet/blob/master/resources/queries.csv

Tokom anotacije ne očekuje se da studenti detaljno razmatraju kvalitet ili optimizovanost koda čiju sličnost sa upitom ocenjuju. Stepen sličnosti između bloka koda i upita je potrebno obeležiti ocenom na skali 0 – 3:

- 3 – potpuno podudaranje – ovaj blok koda deluje kao tačno ono što bi se želelo dobiti kao rezultat zadatog upita. Stoga bi se navedeni blok koda mogao direktno iskopirati ili bi se navedena bibliotečka funkcionalnost mogla direktno pozivati za rešavanje zadatog upita, uz eventualne minorne korekcije.
- 2 – jako podudaranje – ovaj blok koda radi manje-više ono što bi se želelo dobiti kao rezultat zadatog upita. Stoga bi se navedeni blok koda mogao koristiti kao kostur za implementaciju željene funkcionalnosti, ali se ne bi mogao direktno iskopirati ili direktno pozivati kao biblioteka za rešavanje zadatog upita.
- 1 – slabo podudaranje – ovaj blok koda ne radi ono što bi se želelo dobiti kao rezultat zadatog upita, ali sadrži neke korisne elemente ili informacije vezane za upit (npr. definicije struktura u kodu, API-je, itd.).
- 0 – potpuna irelevantnost – ovaj blok koda uopšte nije relevantan za zadati upit.

Anotirane podatke treba sačuvati u formatu tab-separated UTF-8 TXT fajla sa sledećim kolonama za svaki par (upit, blok koda):

1. *ProgrammingLanguageName* – ime programskog jezika na kome je kod napisan (Java, PHP, C,…). Ovo polje će u okviru jedne grupe uvek imati istu vrednost, određenu programskim jezikom koji grupa razmatra.
2. *QueryID* – redni broj upita iz navedenog spiska upita (od 1 do 99)
3. *PairID* – jedinstveni globalni identifikator za posmatrani par (komentar, blok koda). Ovaj identifikator treba da bude identičan sa nazivom (bez .txt ekstenzije) tekstualnog fajla u kome je sačuvan tekst posmatranog para.
4. *QueryText* – tekst upita na srpskom jeziku
5. *CommentText* – tekst dokumentacionog komentara za posmatrani blok koda. U ovom polju treba čuvati samo koristan tekst komentara, bez znakova koji označavaju početak (ili kraj) komentara u posmatranom programskom jeziku. Kod komentara koji se protežu na više linija koda (ali je jasno da se radi o jednom komentaru) znak za novi red treba kodovati sa *n*, dok znakove za tabulaciju treba kodovati sa *t*.
6. *SimilarityScore* – ocena stepena sličnosti upita i bloka koda (0, 1, 2 ili 3).

Podatke bi prilikom anotacije trebalo ravnomerno rasporediti između svih članova grupe, tako da svako anotira približno istu količinu podataka. Pritom, očekuje se da članovi grupe razmotre karakteristične problematične situacije u anotaciji i da ih reše na sistemski ujednačen način u okviru grupe.

Za očekivati je da će velika većina parova (upit, blok koda) biti potpuno nepovezana, tj. zavređivati ocenu 0. Stoga grupe treba da se fokusiraju na pravilnu identifikaciju i obeležavanje onih blokova koda koji imaju bar neki stepen sličnosti sa nekim od upita. Preporučuje se da se za te potrebe obrati pažnja i na tekstove pratećih dokumentacionih komentara i nazive samih funkcija/klasa/modula.

Iako je za potrebe izgradnje skupa podataka dovoljno sprovesti jednostruku anotaciju (tj. anotaciju u kojoj samo jedna osoba anotira određeni par upit – blok koda), radi merenja stepena saglasnosti anotatora potrebno je na kraju procesa anotacije nasumično izdvojiti 10% od ukupnog broja parova. Taj podskup podataka treba da anotiraju svi članovi tima zasebno i bez međusobnih konsultacija. Izdvojeni podskup podataka, zajedno sa dobijenim individualnim anotacijama, treba sačuvati u formi dodatnog tab-separated TXT fajla. Format ovog fajla treba da bude isti kao i format glavnog fajla sa anotiranim podacima, samo što će umesto jedne *SimilarityScore* kolone imati onoliko takvih kolona koliko grupa ima članova, pri čemu svaka takva kolona treba da sadrži ocene po jednog člana grupe. Pomoću ovog fajla tj. ovog podskupa podataka treba izračunati procentualan stepen podudarnosti anotacija između svaka dva člana grupe, kao i grupni prosek binarnih stepena podudarnosti.

## Faza 3 - Obučavanje i evaluacija statističkih modela

Razmatraće se tri različita konceptualna pristupa za rešavanje problema semantičke pretrage, korišćenjem dokumentacionih komentara kao opisa funkcionalnosti koda:

1. Pristup zasnovan na klasifikaciji – u ovom pristupu statistički model klasifikacije kao ulaz dobija dva teksta. Prvi tekst predstavlja upit pretrage, a drugi tekst je dokumentacioni komentar vezan za određen blok koda. Drugi tekst takođe može biti proširen i nazivom funkcije/klase/modula na koji se komentar odnosi. Na osnovu zadatog skupa odlika (npr. reči u prvom i drugom tekstu, dužina prvog teksta, dužina drugog teksta, broj istih reči u oba teksta, broj istih stemova reči u oba teksta, itd.), klasifikator predviđa ocenu stepena sličnosti (0, 1, 2 ili 3) između upita i komentara. Klasifikacioni algoritmi koje treba razmotriti u okviru ovog pristupa su multinomijalni naivni Bajesov klasifikator, logistička regresija i metoda potpornih vektora. Obučavanje i evaluaciju modela je potrebno sprovesti putem 10-slojne stratifikovane unakrsne validacije, korišćenjem odgovarajuće metrike za merenje performansi. Kod logističke regresije i metode potpornih vektora potrebno je sprovesti optimizaciju hiperparametra C / lambda koji određuje jačinu regularizacije, korišćenjem ugnežđene unakrsne validacije. Inicijalnim ispitivanjem, korišćenjem default vrednosti za ostala podešavanja, treba utvrditi koja od varijanti funkcije regularizacije (L1 / L2) i funkcije gubitka kod metode potpornih vektora (L1 / L2) daje bolje rezultate – ako nema primetnih razlika, preporučljivo je koristiti L2 oblike funkcija.
2. Pristup zasnovan na regresiji – postavka ovog pristupa je identična kao i za pristup zasnovan na klasifikaciji, samo što sada izlaz modela nije jedna od četiri klase, već kontinualna numerička vrednost. Obučavanje i evaluaciju modela je potrebno sprovesti putem 10-slojne stratifikovane unakrsne validacije, korišćenjem odgovarajuće metrike za merenje performansi. Kao algoritam regresije razmotriti linearnu regresiju sa L2 regularizacijom, pri čemu je potrebno optimizovati hiperparametar koji određuje jačinu regularizacije, korišćenjem ugnežđene unakrsne validacije.

3. Pristup zasnovan na rangiranju – ovaj pristup je zasnovan na principima dohvatanja informacija, gde se od statističkog modela očekuje da sprovede rangiranje sličnosti između zadatog upita i svih elemenata skupa dokumentacionih komentara tj. pronalaženje komentara koji su najsličniji upitu. Kao i ranije, komentar takođe može biti proširen i nazivom funkcije/klase/modula na koji se komentar odnosi. Sličnost između upita i komentara se može računati kao kosinusna sličnost njihovih *bag-of-words* vektora. Evaluaciju modela treba sprovesti tako što će se za svaki par (upit, komentar) za koji ocena sličnosti nije jednaka nuli napraviti poseban „test set" u koji će pored navedenog para biti uključeno još 99 nasumično odabranih komentara čije ocene sličnosti sa posmatranim upitom jesu jednake nuli. Kao metriku za merenje performansi treba koristiti srednji recipročni rang (*mean reciprocal rank – MRR*).

Za svaku grupu je obavezno da razmotri dva pristupa, od čega prvi mora biti pristup zasnovan na klasifikaciji, dok će se grupe međusobno razlikovati po tome da li kao drugi pristup razmatraju regresiju ili rangiranje. Odluku o tome koji skup pristupa žele da razmatraju članovi grupe moraju prijaviti prilikom formiranja grupe i ta odluka se ne može naknadno menjati.

Za sve navedene algoritme treba ispitati efekte različitih tehnika pretprocesiranja teksta. Počevši od osnovnih *bag-of-words* podešavanja gde se ne koriste sledeće tehnike, sistematski razmotriti sledeće:

- Normalizaciju svih tekstova na mala slova (lowercasing)
- Binarizaciju vrednosti *bag-of-words* odlika
- Frekvencijsko filtriranje reči
- TF, IDF i TFIDF ponderisanje
- Filtriranje stop-reči i/ili stemovanje reči (po izboru)
- Korišćenje bigrama i trigrama

Kao listu stop-reči moguće je koristiti neku od javno dostupnih lista ili formirati sopstvenu. Za stemovanje reči na srpskom koristiti stemer Ljubešića i Pandžića iz paketa *SCStemmers* (dozvoljeno je i korišćenje alternativnih implementacija ovog stemera).

Pored toga, studenti mogu opciono razmotriti i korišćenje odlika vezanih za karakteristike bloka koda na koji se određeni komentar odnosi, pored ili umesto odlika vezanih za karakteristike komentara. Ovu dodatnu evaluaciju moguće je sprovesti za sva tri statistička pristupa.

## Propozicije izrade projekta

Optimalna veličina grupe je četvoro studenata. Minimalan broj parova (komentar, blok koda) koje treba prikupiti za takve grupe je 1000. Dozvoljene su grupe i od troje ili petoro članova, u kojem slučaju minimalan broj parova koji treba prikupiti iznosi 750, odnosno 1250. Očekuje se da za svaki od korišćenih upita prikupljeni skup podataka sadrži minimalno 5 parova (komentar, blok koda) koji su bar u nekoj meri relevantni za upit (tj. zavređuju nenulte ocene sličnosti sa upitom). Ukoliko priroda razmatranog programskog jezika sprečava relevantnost nekih upita za taj jezik, dozvoljeno je kompenzovati manji broj relevantnih parova za takve upite većim brojem relevantnih parova za ostale upite. Kako se zahtevi u pogledu obučavanja i evaluacije statističkih modela ne razlikuju u zavisnosti od veličine grupe, preporučuje se da se studenti organizuju u veće grupe.

Studenti se mogu sami organizovati u grupe, za šta je otvoren i poseban kanal u okviru predmetnog tima na Teams platformi. Pre otpočinjanja rada na projektu, neophodno je formirati i zvanično prijaviti

grupu putem mejla. Prilikom prijave grupe, neophodno je navesti spisak članova grupe, kao i spisak od bar 3 željene kombinacije programski jezik/izbor statističkih pristupa (regresija ili rangiranje), po redosledu interesovanja. Grupi će zatim u najkraćem roku biti zvanično dodeljena ona kombinacija koja već nije zauzeta od strane neke ranije prijavljene grupe.

Za slučaj nemogućnosti samoorganizovanja u grupe, studenti mogu i da se individualno prijave za izradu projekta. U tom slučaju, biće od strane predavača organizovani u grupe sa ostalim studentima koji su se individualno prijavili ili će, u slučaju nedovoljnog broja tako prijavljenih studenata, biti pridruženi nekoj od već formiranih grupa. U oba slučaja, individualno prijavljeni studenti neće imati mogućnost izbora programskog jezika i statističkih pristupa koje razmatraju.

Ova postavka predmetnog projekta će važiti do prolećnog semestra naredne školske godine. Grupe je potrebno formirati i prijaviti najkasnije do 01.08.2021, bez obzira na konkretan ispitni rok u kome se planira odbrana. Individualne prijave treba dostaviti najkasnije do 01.07.2020. Ni grupne ni individualne prijave nakon tih datuma neće biti uzimane u obzir.

Grupe koje žele da brane projekat u određenom ispitnom roku treba da pošalju urađeno rešenje i projektnu dokumentaciju do početka istog ispitnog roka, na adresu: vuk.batanovic@ic.etf.bg.ac.rs.

U projektnoj dokumentaciji treba detaljno opisati svaku od faza izrade projekta. Ovo podrazumeva temeljno opisivanje procesa prikupljanja podataka, izvora podataka, navođenje kriterijuma koji su u tom procesu korišćeni i opisivanje kako je proces obavljen sa tehničkog aspekta. Pored toga, dokumentacija mora sadržati detaljan opis anotacije podataka, uključujući ustanovljene kriterijume anotacije, karakteristične problematične situacije sa kojima se grupa srela tokom anotacije podataka i dogovorene načine za njihovo rešavanje, kao i opis tehničke strane označavanja podataka. Takođe se očekuje da izveštaj sadrži deskriptivni statistički prikaz prikupljenih i anotiranih podataka. Za fazu obučavanja i evaluacije statističkih modela podrazumeva se da izveštaj sadrži pregledni tabelarni prikaz rezultata različitih modela i efekata različitih razmotrenih podešavanja. Dokumentacija ne treba da sadrži iskopirana detaljna objašnjenja iz nastavnih materijala za korišćene tehnike i algoritme.

Ukoliko su projektna rešenja i dokumentacija adekvatni, u dogovoru sa studentima biće određen termin odbrane projekta u toku ispitnog roka. Odbrane će biti moguće u svim ispitnim rokovima predviđenim za predmete iz letnjeg semestra, a održavaće se bilo preko Teams platforme bilo uživo, uzimajući u obzir epidemiološku situaciju u konkretnom roku i dogovor grupe sa predavačem.

Ocene će se dobijati na osnovu broja prikupljenih bodova na skali 0-100, prema sledećoj raspodeli:

- 25 poena – faza prikupljanja podataka
- 25 poena – faza anotacije podataka
- 25 poena – faza obučavanje i evaluacije statističkih modela
- 25 poena – kvalitet i potpunost priložene projektne dokumentacije; poštovanje projektne specifikacije očekivanog formata podataka
- 15 poena – dodatni rad grupe iznad traženih minimalnih zahteva (veća količina prikupljenih i/ili anotiranih podataka, razmatranje odlika vezanih za karakteristike bloka koda, itd.)

Za svaku od prve četiri stavke neophodno je da grupa ostvari barem polovinu od minimalnog broja poena. Drugim rečima, nije moguće odbraniti projekat bez sprovođenja i opisivanja sve tri faze izrade.