Izrada alata za automatsko sažimanje teksta

Mihalić, Danko

SVEUČILIŠTE U ZAGREBU
FILOZOFSKI FAKULTET
ODSJEK ZA INFORMACIJSKE I KOMUNIKACIJSKE ZNANOSTI
SMJER Informatika – istraživački
Ak. god. 2021./2022.

Danko Mihalić, univ.bacc.inf.

Izrada alata za automatsko sažimanje teksta

Diplomski rad

Mentor: izv. prof. dr. sc. Kristina Kocijan

Zagreb, rujan 2022.

Izjava o akademskoj čestitosti

Izjavljujem i svojim potpisom potvrđujem da je ovaj rad rezultat mog vlastitog rada koji se temelji na istraživanjima te objavljenoj i citiranoj literaturi. Izjavljujem da nijedan dio rada nije napisan na nedozvoljen način, odnosno da je prepisan iz necitiranog rada, te da nijedan dio rada ne krši bilo čija autorska prava. Također izjavljujem da nijedan dio rada nije korišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

_____

(potpis)

Želio bih izraziti zahvalnost izv. prof. dr. sc. Kristini Kocijan koja je bez oklijevanja vjerovala u mene i strpljivo vodila kroz pisanje ovog rada.

Sadržaj

1. Uvod

Možda najpoznatiji sažetak «teksta» je formula njemačkog fizičara Alberta Einsteina $E=mc^2$ s početka 20. stoljeća, kojom se povezuju energija i masa. Naravno, Einstein nije tek tako došao do te formule, već je ona produkt proučavanja i sažimanja kompleksnih formula i postulata u fizici. Ta formula odličnim je primjerom kako neki tekst sažeti i pokazati u najjednostavnijem mogućem, a ujedno i razumljivom, obliku.

Sažimanje – zašto i kako? Nije teško odgovoriti na ova pitanja. Najbanalniji odgovor na prvo pitanje je – komfor. Živimo u 21. stoljeću, u svijetu nevjerojatno brzog protoka informacija, gdje je svaka sekunda vrijedna, kako je to sažeto u poslovici: «Vrijeme je novac». Torres-Moreno (2014) daje 6 razloga zašto je potrebno istraživati automatsko sažimanje teksta:

1. sažetak skraćuje vrijeme provedeno čitanjem;
2. sažetci olakšavaju proces odabira podataka prilikom istraživanja dokumenata;
3. automatsko sažimanje poboljšava efektivnost indeksiranja;
4. algoritmi za automatsko sažimanje su manje subjektivni od čovjeka;
5. personalizirani sažeci su korisni u bazama znanja;
6. sustavi za automatsko sažimanje teksta omogućavaju komercijalnim sustavima za obradu teksta da povećaju količine teksta koju mogu obraditi.

Ukoliko nekog zanima određena tema, vrlo je vjerojatno da će najprije pročitati neki sažetak, prije nego uđe u detalje. Ista stvar je s novinskim člancima, najprije se pročita naslov i par rečenica sažetka te na temelju toga donosi zaključak o vlastitoj volji i želji za nastavkom čitanja. Komforni smo, ne želimo utrošiti vrijeme na nešto, čime na kraju tog vremena nećemo biti zadovoljni. U beletristici je već odavno popularno na koricama ili posljednjim stranama knjige imati sažetak, napisan da zainteresira potencijalnog kupca/čitatelja.

Pojavom društvenih mreža, fenomen čitanja sažetaka se samo povećava. Svatko na vlastitom primjeru može vidjeti da radije čita kraće objave od duljih. Kao zanimljivost, može se povezati s tzv. KISS principom[1] američke mornarice iz 60-ih godina.

[1] KISS ili *Keep it simple, stupid* (varijacije su još: *Keep it short and simple, Keep it simple and straightforward, Keep it simple, sailor…*), je pristup u dizajnu sustava koji kaže da je jednostavnost ključna prilikom dizajniranja te da se treba izbjegavati bilo kakvo kompliciranje. Za više informacija vidi: https://en.wikipedia.org/wiki/KISS_principle

Odgovor na pitanje ***kako?*** je također vrlo jednostavan: programski jezik Python je kao stvoren za obradu prirodnog jezika. Sintaksa je relativno jednostavna, mnoge biblioteke poput Natural Language ToolKit (NLTK) i spaCy uvelike olakšavaju rad, podrška korisnika je raširena po cijelom svijetu, a baze znanja su prilično velike. Stoga će se u ovom radu demonstrirati primjena upravo Pythona za rješavanje problema sažimanja teksta.

Na početku ovog rada, u poglavlju *Teorija automatskog sažimanja teksta,* prolazi se kroz pregled teoretskih okvira i pristupa za sažimanje teksta, izlažu se razlozi sažimanja te prednosti i mane tog procesa.

U poglavljima *Implementacija i izrada* te *Eksperiment* prolazi se kroz praktične izvedbu alata za automatsko sažimanje teksta. Predstavljaju se postojeći komercijalni alati te razlozi upotrebe konkretnog programskog jezika. Nakon toga se prelazi na modele prema kojima rješenja rade te se prikazuju rezultati tri različita modela u sažimanju teksta. Rezultati se također uspoređuju s rezultatima dvije web aplikacije za sažimanje teksta čiji princip rada nije poznat.

Pretposljednje poglavlje *Poboljšanja* prolazi kroz programsko rješenje tražeći moguća poboljšanja za budućnost - korisničko sučelje, pristupačnost i sl.

Na samom kraju u poglavlju *Zaključak* ukratko se prolazi kroz rad, rezultate i očekivanja te se donose zaključci u vezi automatskog sažimanja.

2. Teorija automatskog sažimanja teksta

U ovom poglavlju daje se pregled teoretskih okvira automatskog sažimanja teksta, pregled glavnih metoda, njihov opis i principi rada. Također, opisuju se principi strojnog učenja i neuronskih mreža te prednosti i mane tih principa u sažimanju teksta.

Definicija sažimanja teksta prema Lloret i Palomar (2010) kaže da je sažimanje reduktivna transformacija izvornog teksta kroz fokusiranje sadržaja biranjem i/ili poopćenjivanjem onoga što je važno u tom tekstu. Mani (2001) pak kaže da je sažimanje proces koji ima za cilj pretvoriti izvorni tekst u kompaktnu prezentaciju sadržaja za ljudsku upotrebu. Definirajući sažimanje na taj način, naglasio je razlike u odnosu na slična područja koja ne uključuju sažimanje teksta, kao što su: kompresija (engl. *text compression),* pronalaženje dokumenata *(*engl. *document retreival),* indeksiranje *(*engl. *indexing),* vađenje informacija *(*engl. *information extraction),* rudarenje *(*engl. *text mining)* i odgovaranje na pitajna *(*engl. *question answering)*. Mikelić Preradović *et al.* (2014) navode da je sažimanje odabir važnih dijelova ulaznog dokumenta i predstavljanje glavnih ideja izvornog teksta, uz dodatak da je to proces kondenziranja izvornog dokumenta u kraću verziju sebe, na način da se sačuvaju relevantne informacije.

Cilj automatskog sažimanja teksta prema Kumaru i Salim (2012) je fokusiranje izvornog teksta u kraću verziju, pri čemu se čuvaju informacije i općenito značenje. Prema Mani (2001a:1) cilj sažimanja je *«uzimanje izvora informacija, vađenje sadržaja te prikazivanje najvažnijeg sadržaja u kompaktnoj formi na način najprikladniji za korisnika»*.

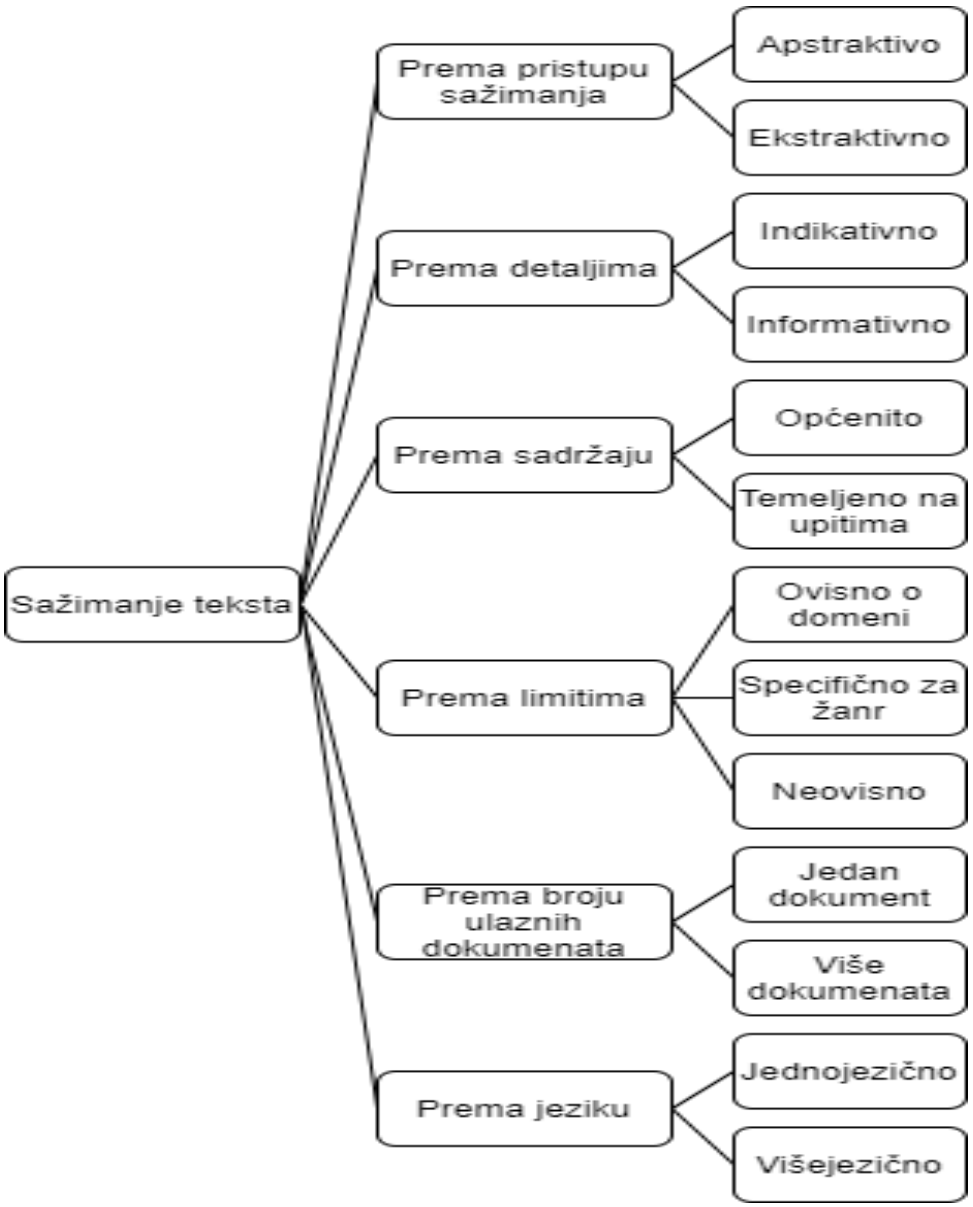
Hovy (2005:584) u svoju definiciju uvodi i **kriterije** prema kojima se tekst može smatrati sažetkom: *«Sažetak je tekst dobiven iz jednog ili više tekstova koji sadrži značajan dio informacija izvornog teksta (tekstova) i nije dulji od polovice originalnog teksta (tekstova)».*

Vidljivo je iz navedenih primjera da različiti autori imaju isti pogled na sažimanje teksta.

2.1. Sažimanje u širem smislu

Različiti autori različito dijele sažimanje teksta. Gholamrezazadeh *et al.* (2009) su preuzeli prevladavajuće podjele i predložili svoju verziju koja se može shvatiti kao sažimanje u širem smislu (slika 1):

1. **prema pristupu sažimanja** – ovdje spadaju *apstraktivne* i *ekstraktivne* metode, koje se detaljnije obrađuju u sljedećim poglavljima. Glavna razlika je u načinu na koji se sažetak stvara; kod ekstraktivnih se koriste i preslaguju dijelovi ulaznog teksta dok je kod apstraktivnih sažetak interpretacija i riječi, rečenice ili fraze iz ulaznog teksta ne moraju postojati u sažetku;
2. **prema detaljima –** ova kategorija se dijeli na *indikativno* i *informativno* sažimanje: indikativno sažimanje prikazuje korisniku samo glavnu ideju teksta i sadrži svega 5 - 10 % izvornog teksta a služi uglavnom kako bi se čitatelja nagnalo da pročita izvorni tekst - kao glavni primjer navode se sažeci na pozadini tadašnjih kutija s DVD filmovima; informativno sažimanje je primjerenije za tehničke dokumente, sadrži 20 -30 % izvornog teksta te može služiti kao njegova zamjena;
3. **prema sadržaju –** sažimanje prema sadržaju autori dijele na *općenito sažimanje* i *sažimanje temeljeno na upitima:* kod općenitog sažimanja tema sadržaja nije važna te se pretpostavlja jednaka važnost svih informacija u tekstu; kod sažimanja temeljenog na upitima, korisnik mora odrediti temu izvornog teksta postavljanjem upita, prije nego se uopće kreće u sažimanje; upiti ujedno služe sustavu za određivanje informacija koje će biti prisutne u konačnom sažetku - primjer sustava za općenito sažimanje prema sadržaju je SUMMARIST (Hovy i Chin, 1999) dok se kao primjer sažimanja temeljenog na upitima navodi WebSumm (Maybury, 1998);
4. **prema limitima –** jedina kategorija koja ima tri podjele, *ovisno o domeni, specifično za žanr* i *neovisno* sažimanje: sažimanje specifično za žanr je pokušaj da se sažetak uklopi u neku standardnu šablonu kao što je znanstveni ili novinski članak, priručnik i slično; sažimanje ovisno o domeni (temi) se vrši na način da se isključuju tekstovi o kojima sustav za sažimanje nema znanja, a znanje koje ima o određenoj domeni je ogromno i koristi ga u sažimanju (iako se tekst sažetka nužno ne nalazi u izornom tekstu); neovisno sažimanje je zapravo proširena verzija sažimanja o domeni gdje sustav bez obzira na manjak znanja ipak pokušava izvaditi glavne ideje teksta te ih uklopiti u nove rečenice;
5. **prema broju ulaznih dokumenata** – u ovoj kategoriji je podjela na sažimanje iz *jednog* ili *više* dokumenata; ukoliko ih je više, moraju biti nekako tematski povezani;
6. **podjela prema jeziku** – slično kao i u prethodnoj kategoriji, ovdje je podjela na sažimanje *jednojezičnih* i *višejezičnih* tekstova: kod jednojezičnih sažetak je na istom jeziku kao i izvorni tekst, dok kod višejezičnih sažetak može biti na drugom jeziku.



Slika 1 Kategorije sažimanja (prilagođeno prema Gholamrezazadeh et al., 2009)

Osim navedenih podjela neki autori (Aggarwal *et al.,* 2009; Zhou i Hovy, 2003) spominju i sljedeće vrste sažimanja:

- verzionirano sažimanje (engl. *update summarization*)
  - [*sažetak = sažimanje(dokument, prethodna_verzija_ili_sažetak_dokumenta)*]
- sažimanje na ključne riječi (engl. *keyword summary*)
  - *ne sažima se u kompaktni tekst, već u ključne riječi ili fraze iz dokumenta*
- naslovno sažimanje (engl. *headline summary*)
  - *sažetak od jednog reda.*

2.2. Sažimanje prema pristupu

Metode, tj. pristup sažimanju teksta mogu biti **ekstraktivne** ili **apstraktivne** (Radev *et al.*, 2002). Pomoću ekstraktivnih metoda sažimanja teksta, identificiraju se važne rečenice i direktno izvade iz originalnog dokumenta, što znači da se sažetak sastoji od originalnih rečenica. Na drugu stranu, apstraktivnim metodama sažimanja teksta, rečenice koje se izvade iz originalnog teksta se obrađuju i restrukturiraju prije nego se ponovo spoje u konačan sažetak (Ganesan *et al.*, 2010). Iz toga proizlazi da apstraktivne metode moraju sadržavati dublju jezičnu analizu i obradu prirodnog jezika.

Razlike između ekstraktivnih i apstraktivnih metoda se mogu vidjeti na primjeru kada tekst sadrži više rečenica o istom objektu s različitim atributima. Npr.:

1. *Baterija novog tableta XY traje vrlo dugo, potrebno ju je napuniti samo jednom u nekoliko dana.*
2. *Baterija novog tableta XY je dosta velika, ali ujedno i jeftina.*
3. *Baterija novog tableta XY je dosta velika i traje vrlo dugo!*

Ekstrativnim metodama, koja god od ove tri rečenice se uzima za sažetak, neće dati potpunu informaciju, tj. sažetak će biti pristran. Neka od apstraktivnih metoda bi u ovom primjeru generirala u sažetak nešto poput:

- *Baterija novog tableta XY je jeftina, dugo traje, ali je ujedno i dosta velika.*

Takav sažetak je evidentno bolji u vidu prijenosa informacija. Sažetak je kompaktniji i prenosi sve informacije, a može se reći i prikladniji za čitatelja na manjim ekranima. O ovim metodama će još kasnije biti riječ.

Prema Hovy (2005), sažimanje teksta se sastoji od 3 faze:

1. **identifikacija teme -** kako bi se identificirala tema teksta, uobičajeno je da se svakoj proizvoljno određenoj jedinici unosa (riječ, rečenica, odlomak…) pridoda neka brojčana vrijednost metodama statističkog ili strojnog učenja;
2. **interpretacija -** faza interpretacije je ona koja razlikuje ekstraktivne od apstraktivnih metoda; tijekom ove faze, teme koje su identificirane kao važne, spajaju se i prezentiraju na novi način i izražavaju novim rečeničnim formulacijama, koristeći riječi ili terminologiju koja nije nužno zastupljena u izvornom tekstu, a ukoliko se sažimanje vrši ekstraktivnim metodama, ovo je ujedno i posljednja faza sažimanja;

3. **generiranje sažetka -** ukoliko se sažimanje vrši apstraktivnim metodama, potrebna je faza generiranja sažetka pomoću tehnika obrade prirodnog jezika[2], kako bi taj sažetak imao gramatičkog i semantičkog smisla.

2.3. Evaluacija sažetka

Jedan od važnih koraka sažimanja teksta je način na koji se vrši evaluacija konačnog sažetka. Metode evaluacije se mogu podijeliti u dvije kategorije (Mani, 2001a): intrinsična i ekstrinsična evaluacija.

Intrinsična evaluacija je testiranje sažetka na samom sebi. Kao kriterije intrinsičnih metoda, Mani navodi koherentnost sažetka, te njegovu informativnost. Kod koherentnosti radi se o tome da tekst mora biti čitljiv čovjeku te da riječi i rečenice ne smiju biti izvađene iz konteksta. Kod informativnosti pak se radi o očuvanju informacija iz originalnog teksta. Kako je sažetak u samoj definiciji kraći od originala, logično je da će u njemu biti manje informacija. Mjera informativnosti govori koliko informacija iz originalnog teksta je ostalo u sažetku.

Ekstrinsična evaluacija podrazumijeva testiranje koliko je sažetak dobar za obavljanje nekog drugog zadatka. Kao primjer Mani navodi upute za uporabu (ako je originalni tekst nekakva uputa, moguće je provjeriti kvalitetu sažetka na način da se pomoću njega izvrše te upute), ali napominje da postoji velika količina ovakvih tipova evaluacija.

Kod evaluacije se doduše javlja problem subjektivnosti. Neovisno o tome je li tekst saželo računalo ili čovjek, postoje određeni faktori na koje se ne može utjecati, kao što su:

- kome je sažetak namijenjen
- čemu sažetak služi
- različite procjene onog što je bitno u tekstu itd.

Dvije različite osobe mogu dati dva različita sažetka istog teksta, no time se ulazi već u psihologiju čovjeka. Neovisno o tome koja metoda evaluacije se koristi, potrebno je držati se istog kriterija.

[2] Obrada prirodnog jezika (engl. *Natural Language Processing – NLP*) je područje računalne znanosti unutar polja umjetne inteligencije koje spaja lingvistiku i računala kako bi se analizirao i procesuirao, prepoznao, razumio i generirao ljudski, prirodni jezik. Za više informacija vidi: https://en.wikipedia.org/wiki/Natural_language_processing

Evaluacija sažetka također može biti i automatska, bez ljudskog faktora. Chin-Yew Lin je 2004. predložio tzv. ROUGE (engl. *Recall-Oriented Understudy for Gisting Evaluation*) metodu za evaluaciju kvalitete sažetka na način da se sažetak uspoređuje sa sažecima koje su pripremili ljudi. Logika za kojom se vodio je da ukoliko je računalo napravilo dobar sažetak, tada se on određenim dijelom preklapa sa sažecima koje su napravili ljudi. Ono što iskače jest da su svakako potrebni referentni sažeci napravljeni od strane ljudi. Varijacije koje je predložio su:

- ROUGE-N – usporedba se radi na n-gramima. N-gram je slijed od N riječi. Kvaliteta sažetka se određuje tako da se radi suma omjera svih sljedova riječi koje se poklapaju u računalnom sažetku te sažetku napravljenom od strane čovjeka.
- ROUGE-L – u ovoj varijaciji se traži najdulji zajednički dijelovi između sažetaka rađenog od strane čovjeka i računala.
- ROUGE-W – ova varijacija traži ROUGE-L varijaciju na kojoj se rade težinski (engl. *weighted)* koeficijenti dobivenih sažetaka. Razlika između ove i ROUGE-L varijacije je kao razlika između aritmetičke i medijanske sredine.
- ROUGE-S – S označava *Skip-bigram*, odnosno, varijaciju ROUGE-N gdje se proizvoljno mogu preskočiti riječi i uzeti različiti n-grami.
- ROUGE-SU – ova varijacija je «produžetak» prethodne na način da se dodaje unigram kojim se poništava mogućnost ROUGE-S varijacije u kojoj je rezultat 0 u slučaju da je prvi n-gram identičan posljednjem.

2.4. Određivanje relevantnosti

Određivanje relevantnosti riječi ili rečenice spada u domenu obrade prirodnog jezika. Kao što je rečeno, prvi korak je identifikacija teme teksta. Kako bi se to postiglo, najčešće se tekst dijeli na manje jedinice (riječi, ali većinom rečenice), te im se pridodaje određena brojčana vrijednost pomoću koje se onda određuje relevantnost. Kriteriji za određivanje su različiti, ovisno o različitim autorima – *pozicija rečenice u tekstu* (Edmundson, 1969), *indikatori ključnih fraza* (Luhn, 1958), *frekvencija riječi i fraza* (Lloret, Ferrández *et al.*, 2008), *preklapanje upita i naslova* (Radev, Blair-Goldensohn *et al.*, 2001) itd.

Teško je reći koji kriteriji su bolji ili lošiji, obzirom da je nemoguće napraviti zlatni standard za usporedbu. Moguće je napraviti različite sažetke potpuno jednake kvalitete u smislu informativnosti i koherentnosti. Kao i kod evaluacije, potrebno je držati se istog kriterija

prilikom određivanja relevantnosti. Upravo u određivanju relevantnosti se dolazi do problema na koji su ukazali Gong i Liu (2001). Naime, predstavili su tablicu istraživanja prosječno odabranih rečenica prilikom izrade sažetaka (tablica 1) iz koje se zorno može vidjeti problem subjektivnog pogleda osobe koja radi sažetak u određivanju kriterija relevantnosti neke rečenice u tekstu.

Tablica 1 Prosječan broj odabranih rečenica prilikom izrade sažetaka

	Broj	**Prosječno odabranih rečenica po dokumentu**
Ukupan broj rečenica u 549 dokumenata	7 053	29,0
Rečenice odabrane jednom osobom	1 283	5,3
Rečenice odabrane dvjema osobama	604	2,5
Rečenice odabrane trima osobama	290	1,2
Ukupno odabrane rečenice	2 177	9,0

2.5. Ekstraktivne metode sažimanja teksta

Za razliku od apstraktivnih metoda, ekstraktivnim se metodama tekst pokušava sažeti na način da se odabere podskup riječi koje nose značenje u tom tekstu. Rangiraju se važni dijelovi rečenice te se prema tome stvara sažetak. Za rangiranje postoje različiti algoritmi i tehnike, što uključuje podrangiranje temeljeno na važnosti i sličnosti. Modernije ekstraktivne metode često koriste strojno učenje i obradu prirodnog jezika za identifikaciju ključnih pojmova i relacija među njima.

Prema Gupti i Lehalu (2010) ekstraktivne metode dijele se na:

- frekvencija termina – inverzna frekvencija dokumenata (engl. *Term Frequency-Inverse Document Frequency – TF-IDF*),
- metoda bazirana na klasterima (engl. *Cluster based method*),
- metoda sažimanja temeljena na grafovima (engl. *Graph theoretic approach*),

- metoda sažimanja temeljena na strojnom učenju,
- LSA metoda,
- metoda konceptualno dobivenih sažetaka (engl. *An approach to concept-obtained text summarization*),
- metoda sažimanja pomoću neuronskih mreža,
- metoda sažimanja temeljena na nejasnoj logici (engl. *Automatic text summarization based on fuzzy logic*),
- metoda sažimanja koristeći regresiju za procjenu težinskih faktora značajki (engl. *Text summarization using regression for estimating feature weights*),
- ekstraktivna metoda sažimanja više dokumenata,
- metoda sažimanja temeljena na upitima,
- višejezično ekstraktivno sažimanje tekstova.

Značajke koje bi se trebale uzimati u obzir prilikom sažimanja (Gupta i Lehal, 2010):

a. ključne riječi
b. naslovne riječi
c. mjesto rečenice u tekstu
d. duljina rečenice
e. prave imenice (imena, nazivi…)
f. riječi započete ili napisane tiskanim slovima
g. ključne fraze
h. lista pristranih riječi
i. font i oblikovanje riječi
j. zamjenice
k. kohezija među rečenicama
l. kohezija rečenica-centroid
m. pojavljivanje manje značajnih informacija
n. analiza diskurza.

Sve ekstraktivne metode koriste jednu ili više od ovih značajki u stvaranju sažetaka. U nastavku rada će biti više riječi o svakoj od navedenih 12 metoda.

2.5.1. Frekvencija termina – Inverzna frekvencija dokumenta (TF-IDF)

Frekvencija termina (TF) predložena je u radu *A Statistical Approach to Mechanized Encoding and Searching of Literary Information* H. P. Luhna 1957. kao dio njegove paradigme za dohvaćanje informacija (engl. *Information Retrieval paradigm*). Autori García-Hernández i Ledeneva (2009) su odlučili primijeniti formulu na ekstraktivno sažimanje teksta koja glasi:

$w_i(t_j) = f_{ij}$, gdje je f_{ij} frekvencija termina *j* u dokumentu *i.*

Druga formula koju su autori odlučili iskoristiti je tzv. inverzna frekvencija dokumenata (IDF), koju su predložili Salton i Buckley (1988) u radu *Term-weighting approaches in automatic text retrieval:*

$w_i(t_j) = log\left(\frac{N}{n_j}\right)$, gdje je *N* broj dokumenata u kolekciji, a n_j je broj dokumenata u kojima se termin *j* pojavljuje

Konačnu TF-IDF formulu autori predlažu u obliku:

$$w_i(t_j) = f_{ij} \times log\left(\frac{N}{n_j}\right)$$

Na kraju napominju da se ovakva metoda primjenjuje na sažimanje jednog dokumenta te se kolekcijama dokumenata zapravo smatraju kolekcije rečenica koje čine jedan tekst.

2.5.2. Metoda bazirana na klasterima

Ova metoda bazira se na pretpostavci da se teme u dokumentima obrađuju na neki organizirani način i da su podijeljene na sekcije. Takva podjela se primjenjuje i na sažetke.

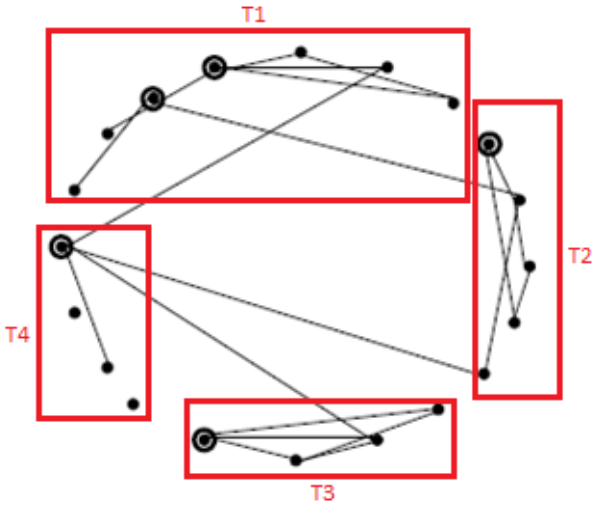
Svaki klaster se smatra jednom temom, svaka tema se prikazuje riječima s najvećim rangom preko TF-IDF metode. Rečenice se odabiru temeljem sličnosti rečenice u odnosu na temu tog klastera C_i, temeljem lokacije te rečenice unutar dokumenta L_i i temeljem sličnosti u odnosu na prvu rečenicu dokumenta kojem pripada F_i. Ukupni rezultat (S_i) pojedine rečenice je težinska suma ta tri faktora.

2.5.3. Metoda sažimanja temeljena na grafovima

Kao što se može vidjeti u prethodnim metodama, prvi korak prema sažimanju teksta je identificiranje tema koje se obrađuju u dokumentu. Ova metoda upravo to omogućuje. Rečenice u dokumentu se prikazuju kao čvorovi u grafu. Postoji čvor za svaku rečenicu. Dvije rečenice su spojene ako postoje neke zajedničke riječi, tj. njihova sličnost je veća od neke granice (npr. kosinusna sličnost).

Čim više spojeva neki čvor ima, tim je važnost te rečenice veća. Samim time je veća vjerojatnost da će biti uključene u sažetak. Slika 2 prikazuje primjer ove metode. Može se vidjeti da graf ima 4 isječka, od kojih tri sadrže po jednu važniju rečenicu (veći čvorovi), dok

jedan isječak sadrži dvije važnije rečenice. Ta četiri isječka ukazuju na 4 teme (na slici označene s T1...T4) te ukupno pet rečenica koje će biti uključene u sažetak.



Slika 2 Prikaz čvorova i njihovih veza

2.5.4. Metoda sažimanja temeljena na strojnom učenju

Primjenu strojnog učenja za sažimanje teksta prvi su razvili Kupiec *et al*. (1995) razvojem alata za sažimanje koristeći Bayesov klasifikator[3] kako bi iskombinirali glavne značajke iz korpusa znanstvenih članaka i njihovih sažetaka. Povezanost se može mjeriti brojem zajedničkih riječi, sinonima, anafora itd. Ističe se i vjerojatnost uključivanja rečenice u sažetak ovisno o tome je li prethodna rečenica bila uključena ili nije.

Bayesovo pravilo koje se primjenjuje u sažimanju pomoću strojnog učenja glasi:

$$P\ (s \in < S \mid F_1, F_2, \dots, F_N) = \frac{P\ (F_1, F_2, \dots, F_N \mid s \in S)\ \times\ P\ (s \in S)}{P\ (F_1, F_2, \dots, F_N)}$$

gdje je

- *s* rečenica iz kolekcije dokumenata,
- F_1, F_2, …, F_N su značajke korištene u klasifikaciji,
- *S* je sažetak koji se generira,

[3] Naivni Bayesov klasifikator je jedan od statističkih modela u strojnom učenju za određivanje razreda (grupa) kojima pojedina riječ pripada. Popularna primjena naivnog Bayesovog klasifikatora je u filtriranju spam e-pošte. Za više informacija vidi: https://en.wikipedia.org/wiki/Naive_Bayes_classifier

- $P(s \in <S \mid F_1, F_2, \ldots, F_N)$ je vjerojatnost da će rečenica *s* biti izabrana u sažetak, ako posjeduje značajke $F_1, F_2, \ldots, F_N$.

2.5.5. LSA metoda

Latentna semantička analiza dobila je takav naziv jer grupira dokumente koji su semantički međusobno povezani, čak i kada nemaju zajedničkih riječi. Temelji se na dekompoziciji singularne vrijednosti[4] (engl. *Singular Value Decomposition*). Ova metoda se može primijeniti u svrhu prikaza tematske riječi i rečenice u dokumentima a predstavili su je Gong i Liu 2001. u svom radu *Generic text summarization using relevance measure and latent semantic analysis.* Gupta i Lehal (2010) tvrde da je prednost korištenja LSA vektora za sažimanje ta što ljudski mozak lako prepoznaje relacije među njima.

2.5.6. Metoda konceptualno dobivenih sažetaka

Ideja ove metode je dohvaćanje koncepata riječi baziranih na HowNet-u[5] i koristi koncept kao značajku umjesto riječi. Ovakav pristup koristi model konceptualnog vektor prostora da formira grubi sažetak i zatim računa stupanj semantičke sličnosti rečenica kako bi se smanjila redundancija. Tri glavna koraka ove metode su:

1. korištenje HowNet kao alata za dobivanje koncepta teksta i uspostavljanje modela konceptualnog vektor prostora;
2. izračunavanje važnosti koncepta temeljenog na tom modelu konceptualnog vektor prostora;
3. generiranje konačnog sažetka računanjem važnosti rečenice i smanjenje redundantnosti sažetka.

2.5.7. Metoda sažimanja pomoću neuronskih mreža

Korištenje neuronske mreže za sažimanje teksta predložio je Kaikhah (2004). Neuronska mreža se najprije trenira na korpusu članaka, zatim modificira tako da spaja određene značajke teksta i «proizvodi» sažetak od visoko rangiranih rečenica izvornog teksta. Spajanjem značajki

[4] Dekompozicija singularne vrijednosti matrice je faktorizacija prikazana produktom triju matrica, gdje su dvije unitarne a jedna dijagonalna na se nalaze singularne vrijednosti. Za više informacija vidi: https://en.wikipedia.org/wiki/Singular_value_decomposition

[5] HowNet baza znanja: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1276017

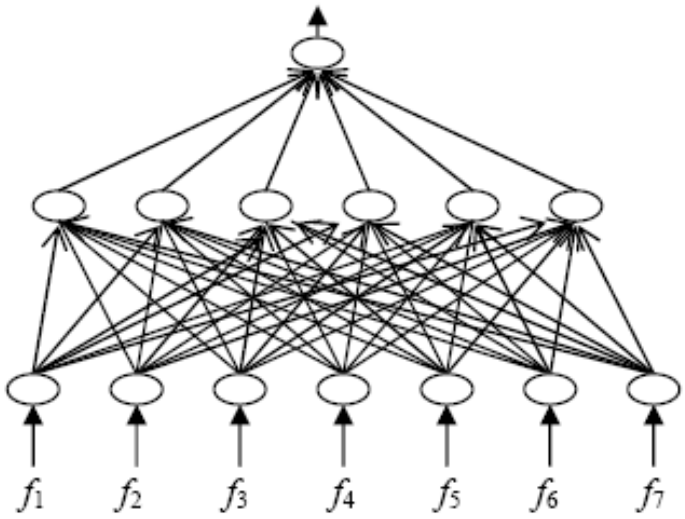
teksta mreža otkriva važnost (i nevažnost) različitih dijelova i tako određuje je li neki dio dovoljno važan da ga uključi u sažetak.

Na početku je potrebno naučiti neuronsku mrežu prepoznavanju tipova rečenica koje bi trebale biti uključene u sažetak (ima li ili nema neku od željenih značajki). Ovo je dugotrajan proces i uključuje čovjeka koji «pokazuje» mreži je li neka rečenica vrijedna ili nije vrijedna za uključivanje u sažetak. Nakon treniranja, rečenice se spajaju ovisno o značajkama koje imaju te se generira sažetak.

Kaikhah (2004) dijeli svaki tekst na listu rečenica. Svaka rečenica se prikazuje kao vektor 7 značajki $[f_1, f_2, \ldots f_7]$. Značajke koje neuronska mreža gleda su:

f_1 – paragraf slijedi naslov
f_2 – pozicija paragrafa u tekstu
f_3 – pozicija rečenice u paragrafu
f_4 – prva rečenica u paragrafu
f_5 – duljina rečenice
f_6 – broj tematskih riječi u rečenici
f_7 – broj naslovnih riječi u rečenici.

Slika 3 je ilustrativnog karaktera i prikazuje neuronsku mrežu nakon vježbanja. Čvorovi s najmanje veza se smatraju slabima i brišu se iz mreže jer se ne uključuju u sažetak. Na ovoj slici svi čvorovi imaju jednak broj veza, stoga se svi uključuju u sažetak. Ovakav slučaj se teško može dogodit u realnoj situaciji jer pretpostavlja da se tekst sastoji od samo jedne ne previše razvijene rečenice (ili jedne rečenice koja se više puta ponavlja).

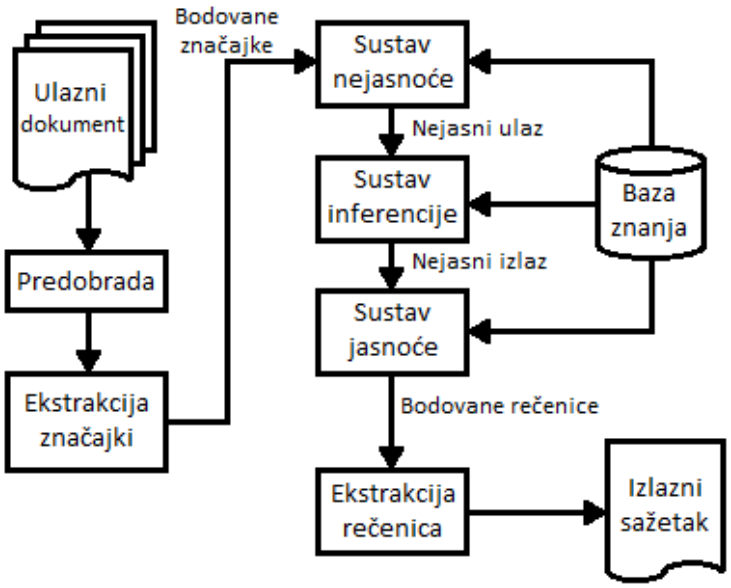


Slika 3 Neuronska mreža nakon vježbanja

Na predloženi model veliki utjecaj ima čovjek koji trenira neuronsku mrežu, te ona odražava njegov stil. Kaikhah (2004) tvrdi da je to prednost njegovog modela u tome što se može prilagoditi ovisno o publici kojoj je namijenjen – drugačije se sažima npr. beletristika od stručnog teksta.

2.5.8. Metoda sažimanja temeljena na nejasnoj logici

Kyoomarsi *et al.* (2008) kažu da metoda sažimanja temeljena na nejasnoj logici gleda karakteristike rečenice kao što su duljina, sličnost s naslovom, sličnost s ključnim riječima i sl., a zatim te karakteristike smatra ulazom sustava nejasne logike. U bazu znanja takvog sustava unose se sva pravila potrebna za sažimanje. Nakon toga se svakoj rečenici pridaje vrijednost između 0 i 1, ovisno o značajkama koje zadovoljava (prema pravilima iz baze znanja). Konačna vrijednost određuje važnost rečenice za uključivanje u konačan sažetak. Važne rečenice se na kraju uključuju IF-THEN pravilom.



Slika 4 Sustav za sažimanje temeljen na nejasnoj logici (prilagođeno prema Kyoomarsi et al., 2008)

Slika 4 pokazuje primjer sustava za sažimanje temeljenog na nejasnoj logici. Sustav nejasne logike se sastoji od 4 komponente:

- sustav nejasnoće (engl. *fuzzifier*)
- sustav inferencije
- sustav jasnoće (engl. *defuzzifier*)
- baza znanja.

U sustavu nejasnoće rečenice se pretvaraju u lingvističke vrijednosti, u sustavu inferencije se pomoću IF-THEN pravila iz baze znanja određuju rečenice koje se uključuju u sažetak i zatim u sustavu jasnoće lingvističke se vrijednosti natrag vraćaju u rečenice. Na kraju se generira sažetak.

2.5.9. Metoda sažimanja koristeći regresiju za procjenu težinskih faktora značajki

Gupti i Singh Lehalu (2010) tvrde da je matematička regresija dobar model za procjenu težinskih vrijednosti značajki teksta. Model su izvorno predložili Fattah i Ren (2008) u svom radu *Automatic Text Summarization.* Tekst ima 10 značajki:

- f_1 – pozicija rečenice
- f_2 – pozitivna ključna riječ u rečenici

- f_3 – negativna ključna riječ u rečenici
- f_4 – centralnost rečenice (sličnost s ostalim rečenicama)
- f_5 – sličnost rečenice s naslovom
- f_6 – uključenost imenica u rečenici
- f_7 – uključenost brojčanih vrijednosti u rečenici
- f_8 – relativna duljina rečenice
- f_9 – broj spojeva među čvorovima (rečenicama)
- f_{10} – zbroj sličnosti među čvorovima.

Ulazni tekst se najprije vježba na određenoj količini teksta, a zatim testira na duplo većoj količini teksta koji mora biti različit od teksta na kojem se vršilo vježbanje sustava.

2.5.10. Ekstraktivna metoda sažimanja više dokumenata

Ova metoda se bavi sažimanjem više tekstova iste tematike. Ovakav sažetak omogućuje korisnicima da se brzo upoznaju s informacijama u većoj količini dokumenata ili klasteru dokumenata. Kako na svaku temu može biti više mišljenja, sažimanjem više dokumenata pokazuju se različiti pogledi na tu temu unutar jednog sažetka.

Jedan od takvih sustava je NeATS (Lin, Hovy, 2002) koji na ulaznu kolekciju novinskih članaka generira sažetak u tri faze:

1. *odabir sadržaja* – cilj ove faze je identifikacija važnih dijelova, termina i koncepata u skupu dokumenata;
2. *filtriranje* – sadržaj se filtrira preko 3 značajke: pozicija rečenice, stigma riječi i maksimalna marginalna relevantnost;
3. *prezentacija.*

2.5.11. Metoda sažimanja temeljena na upitima

U sažimanju teksta temeljenog na upitima, rečenice ulaznog dokumenta rangiraju se prema frekvenciji termina (čestom pojavljivanju riječi ili rečenica). Rečenice koje sadrže fraze iz upita su više rangirane od onih koje sadrže samo pojedine riječi. Nakon toga se rečenice s najvišim rangom slažu u izlazni sažetak. Dijelovi teksta mogu se uzeti iz različitih dijelova dokumenta. Broj rečenica koje se prikazuju u sažetku ovisi o veličini okvira u kojem se sažetak prikazuje bez da ga je potrebno pomicati (engl. *scroll*).

Algoritam za sažimanje temeljeno na upitima:

1. *rangiraj sve rečenice prema frekvenciji termina*
2. *dodaj glavni naslov dokumenta u sažetak*
3. *dodaj prvi naslov u sažetak*
4. *dok (engl. while) se ne prelazi preko veličine sažetka:*
    5. *dodaj sljedeću najviše rangiranu rečenicu*
    6. *dodaj strukturalni kontekst rečenice (ako postoji i ako već nije uključen u sažetak)*
        7. *dodaj sljedeći najviše rangirani naslov iznad teksta (npr. naslov N)*
        8. *dodaj naslov prije N u istoj razini*
        9. *dodaj naslov nakon N u istoj razini*
        10. *ponavljaj korake 7, 8 ,9*
11. *završi klauzulu (engl. end while).*

Druga metoda sažimanja temeljena na upitima je tzv. Bayesovo sažimanje (engl. *Bayesian summarization*) koje glasi:

> *Za kolekciju dokumenata D i upita Q, pretpostavlja se D x Q binarna matrica r_{dq} = 1 ako i samo ako je dokument d relevantan za upit q.*

U sažimanju više dokumenata, r_{dq} će biti 1 kad dokument *d* odgovara upitu *q*.

2.5.12. Višejezično ekstraktivno sažimanje tekstova

Višejezično sažimanje tekstova služi sažimanju izvornog teksta izvornog jezika na željeni jezik (Gupta, Lehal, 2010).

Neki od alata za ovakvu vrstu sažimanja tekstova su:

- SimFinderML, opisan u radu *Identifying Similarity in Text: Multi Lingual Analysis for Summarization* (Kirk Evans, 2005)
- MINDS, opisan u radu *MINDS-Multilingual Interactive document summarization* (Cowie *et al.*, 1998)
- MEAD, opisan u radu *MEAD – a platform for multi document multilingual text summarization* (Radev *et al.*, 2004).

U tablici 2 kronološki je ispisana svaka od navedenih metoda, kratak opis načina na koji se pristupa sažimanju, kao i godina i autor/i koji su o toj metodi pisali.

Tablica 2 Pregled ekstraktivnih metoda sažimanja teksta

Metoda	Godina i autor(i)	Osnovni model sažimanja
Metoda sažimanja temeljena na strojnom učenju	1995. Kupiec, J.; Pedersen, J.; Chen, F..	Koristi se Naivni Bayesov klasifikator kako bi se, temeljem autorima odabranih kriterija, odredila vjerojatnost određene rečenice da se nađe u sažetku.
Metoda sažimanja temeljena na grafovima	1997. Mani, I.; Bloedorn, E.	Rečenice se prikazuju kao čvorovi u grafu koji se zatim povezuju temeljem sličnosti riječi unutar tih rečenica. U sažetak ulaze čvorovi (rečenice) s najviše veza.
Višejezično ekstraktivno sažimanje tekstova	1998. Cowie, J., Mahesh, K., Nirenburg, S., Zajaz, R.	Nekoliko je sustava za višejezično sažimanje tekstova koje služi kako bi se ulazni tekst na jednom jeziku sažeo na nekom drugom.
Metoda bazirana na klasterima	2000. Goldstein, J.; Mittal, V.; Kantrowitz, M.; Carbonell, J.	Rečenice se grupiraju u klastere temeljem lokacije te rečenice unutar dokumenta i temeljem sličnosti u odnosu na prvu rečenicu dokumenta, a u sažetak se stavlja težinska suma tih faktora.
LSA metoda	2001. Gong, Y.; Liu, X.	Semantički se prikazuju tematske riječi i rečenice u dokumentima.
Ekstraktivna metoda sažimanja više dokumenata	2002. Lin, C-Y, Hovy. E.	Bavi se sažimanjem više tekstova iste tematike kako bi se korisnici upoznali s informacijama u većoj količini dokumenata.
Metoda sažimanja pomoću neuronskih mreža	2004. Kaikhah, K.	Neuronska mreža se trenira na korpusu članaka, modificira tako da spaja određene značajke teksta i „proizvodi“ sažetak od visoko rangiranih rečenica izvornog teksta. Treniranje neuronske mreže ima li ili nema neka rečenica željene značajke je dugotrajan proces.
Metoda konceptualno dobivenih sažetaka	2005. Wang, M.; Wang, X.; Xu, C.	Dohvaćanje koncepata riječi baziranih na HowNet bazi podataka korištenjem modela konceptualnog vektor prostora kako bi se formirao grubi sažetak a zatim izračunao stupanj semantičke sličnosti rečenica u kreiranju sažetka.
Metoda sažimanja temeljena na nejasnoj logici	2008. Kyoomarsi, F.; Khosravi, J.; Eslami, E.; Khosravyan Dehkordy, P.; Tajoddin, A.	Gleda karakteristike rečenice koje koristi kao ulaz u sustav nejasne logike. Koristi i bazu znanja u kojoj su sva pravila potrebna za sažimanje. Dodjeljuju se vrijednosti između 0 i 1, stvara se rang lista i prema tome određuje sažetak.

Metoda	Godina i autor(i)	Osnovni model sažimanja
Metoda sažimanja koristeći regresiju za procjenu težinskih faktora značajki	2008. Fattah, M.A.; Ren, F.	Ulazni tekst se najprije vježba na određenoj količini teksta prema 10 određenih značajku, a zatim testira na duplo većoj količini teksta koji mora biti različit od teksta na kojem se vršilo vježbanje.
Frekvencija termina – inverzna frekvencija dokumenata	2009. García-Hernández, R.A.; Ledeneva, Y.	Model koristi formule za frekvenciju termina (TF) i inverznu frekvenciju dokumenata (IDF) prema ranijim radovima za dohvaćanje informacija i tekstova.
Metoda sažimanja temeljena na upitima	2011. Siva Kumar, A.P.; Premchand, P.; Govardhan, A.	Rečenice se rangiraju prema frekvenciji termina. Rečenice koje sadrže fraze iz upita su više rangirane od onih koje sadrže samo pojedine riječi. Prema rangu se stvara lista rečenica koja je zatim dio sažetka.

2.6. Apstraktivne metode sažimanja teksta

Kao što je ranije spomenuto, apstraktivne metode sažimanja teksta obrađuju i restrukturiraju tekst prije nego se spoje u sažetak. Potrebna je obrada prirodnog jezika kako bi se zapravo generirao novi sadržaj. Radev, Hovy i McKeown (2002) kategorizirali su apstraktivne metode kao sve one koje nisu ekstraktivne, što može uvesti konfuziju, no zapravo je vrlo smisleno – svaka metoda koja ne uzima i preraspoređuje tekst, već do sažetka dolazi na neki drugi način, je apstraktivna metoda.

Prema istim autorima, apstrakcija uključuje prepoznavanje da odabrani dijelovi u cjelini čine nešto, što nije nužno eksplicitno navedeno u izvornom tekstu te zamjena tih odabranih dijelova s novim konceptima. Uvjet da sustav prepozna materijal koji se ne nalazi u tekstu je taj da ima pristup nekim vanjskim informacijama (npr. korpusu) na temelju kojeg se vrše daljnji procesi. U nastavku će biti više riječi o tri apstraktivne metode sažimanja: (1) sažimanje temeljeno na grafovima, (2) temeljem pozornosti i (3) redukcijom semantičkih grafova.

2.6.1. Metoda sažimanja temeljena na grafovima

Apstraktivnim metodama se posvetilo manje autora te su takve metode predmetom istraživanja u mnogo manjoj količini od ekstraktivnih. Apsktrakcija je mnogo kompleksija za izvedbu od ekstrakcije. Sustav treba «izmisliti» način kako tekst interpretirati vlastitim riječima. To naravno ne znači da istraživanja i sustavi ne postoje. Jedan takav sustav predložili su Ganesan, Zhai i Han (2010).

Autori predlažu korištenje podataka grafa zvanog Opinosis-Graph kao predstavnika teksta prirodnog jezika i problem apstraktivnog sažimanja riješiti tražeći prikladne puteve u tom grafu. Za razliku od ekstraktivnog sažimanja temeljenog na grafovima, predlažu otvoreni graf, često neusmjeren, kojem su čvorovi rečenice, a rubovi sličnost. Za to predlažu 3 algoritma:

- algoritam za dobivanje *OpinosisGraph(Z)*
- algoritam za sažimanje *OpinosisSummarization(Z)*
- algoritam potprocesa traženja puteva među čvorovima *Traverse( ).*

2.6.2. Metoda sažimanja temeljem pozornosti

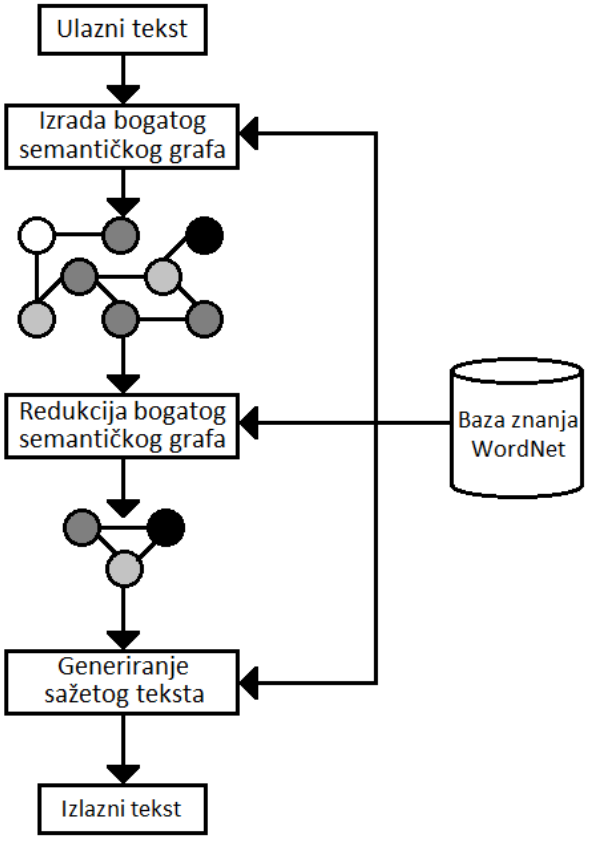
Atori Rush, Chopra i Weston (2015) predstavili su svoj model sažimanja temeljen na neuronskim mrežama. Kombinirali su model neuronskog jezika s enkoderom kontekstualnog ulaza. Enkoder uči preko ulaznog teksta i stvara sažetak. Oni taj svoj pristup nazivaju *Attention-Based Summarization* jer se manje oslanja na lingvističke strukture od ostalih usporedivih sustava. Također navode da se ova metoda lako može istrenirati za veliku količinu podataka, što su i pokazali na korpusu od oko 4 milijuna članaka. Generirali su naslove temeljem prve rečenice članka:

- **ulaz ($x_1, x_2, \ldots, x_{18}$):** *russian defense minister ivanov called sunday for the creation of a joint front for combating global terorism;*
- **izlaz ($y_1, y_2, \ldots, y_8$):** *russia calls for joint front against terorism.*

2.6.3. Metoda redukcije semantičkih grafova

Autori Moawad i Aref (2012) su predstavili svoju ideju o apstraktivnom sažimanju koristeći tehniku redukcije bogatih semantičkih grafova. Njihov je pristup vidljiv na slici 5.

Najprije se sažima ulazni dokument i izrađuje bogati semantički graf originalnog dokumenta. Zatim se generirani graf reducira i iz tog reduciranog grafa generira se apstraktivni sažetak. Prikazali su i *case-study* koja sažima originalan tekst do 50%.



Slika 5 Metoda redukcije semantičkih grafova (prilagođeno prema Moawad i Aref, 2012)

U tablici 3 su kratko opisane apstraktivne metode sažimanja zajedno s autorima koji su metodu predložili i godinom objavljivanja.

Tablica 3 Pregled apstraktivnih metoda sažimanja teksta

Metoda	**Godina i autor(i)**	**Osnovni model sažimanja**
Metoda sažimanja temeljena na grafovima	2010. Ganesan, K.; Zhai, C. X.; Han, J.	Koristi otvoreni, često neusmjeren *Opinosis-Graph* kojem su čvorovi rečenice, a rubovi sličnost.
Metoda redukcije semantičkih grafova	2012. Moawad, I.F.; Aref, M.	Najprije se sažima ulazni dokument i izrađuje bogati semantički graf originalnog dokumenta. Zatim se generirani graf reducira i iz tog reduciranog grafa generira se apstraktivni sažetak.
Metoda sažimanja temeljem pozornosti	2015. Rush, A.M.; Chopra, S.; Weston, J.	Temeljen na neuronskim mrežama. Kombinira se model neuronskog jezika s enkoderom kontekstualnog ulaza. Enkoder uči preko ulaznog teksta i stvara sažetak.

2.7. Poznati sustavi za sažimanje teksta

U tablici 4 je dan kratak pregled nekih od poznatih sustava za sažimanje teksta, poredanih po tipu sažetka koji sustav generira, počevši od ekstraktivnih prema apstraktivnim. U prvom stupcu (Sustav) je naziv sustava, drugi stupac (Ulaz) ukazuje na tip ulaznih dokumenata, tj. podržava li sustav sažimanje samo jednog dokumenta, više dokumenata ili su moguće obje verzije. Treći stupac (Namjena) pokazuje je li sustav dizajniran za neko specifično tematsko područje ili je namijenjen općoj upotrebi. U četvrtom stupcu (Značajke) popisane su značajke temeljem kojih sustav radi. Peti stupac (Izlaz) pokazuje način na koji sustav generira sažetak, tj. je li sažetak generiran nekom od ekstraktivnih (E) ili apstraktivnih (A) metoda.

Tablica 4 Pregled poznatih sustava za sažimanje teksta

Sustav	Ulaz	Namjena	Značajke	Izlaz
ADAM	Jedan dokument	Kemija	- otkrivanje ključnih fraza - frekvencija termina - odabir rečenica	E
ANES	Jedan dokument	Novosti	- težinski odabir fraza i rečenica - prva rečenica uključena u sažetak	E
DimSum	Jedan dokument	Nije poznato	- obrada jezika temeljem korpusa - automatsko uključivanje višerječnih fraza - konceptualna prezentacija teksta	E
SUMMARIST	Jedan dokument	Novosti	- višejezični sustav - otkrivanje teme, interpretacija, generiranje - obrada jezika temeljem baze znanja	E
FociSum	Jedan dokument	Opća namjena	- vađenje informacija i biranje rečenica - određivanje teme prema nazivu dokumenta i višerječnim terminima	E
CENTRIFUSER	Više dokumenata	Zdravstveni članci	- sažimanje temeljeno na upitima - upiti prema sličnosti riječi i rečenica	E
MEAD	Više dokumenata	Novosti	- slične rečenice se isključuju - pozicija i sličnost s prvom rečenicom pri rangiranju	E
NeATS	Više dokumenata	Novosti	- pozicija rečenice - frekvencija termina - ustaljene fraze unutar teme - grupiranje fraza	E

Sustav	Ulaz	Namjena	Značajke	Izlaz
NTT	Jedan dokument	Nije poznato	- strojno učenje za klasificiranje rečenica na važne i nevažne - značajke za određivanje relevantnosti: pozicija, duljina, težinski faktor, sličnost s naslovom	E
GISTSumm	Jedan dokument	Opća namjena	- ključne riječi za identifikaciju najvažnijeg poglavlja - statistička analiza	E
LAKE	Jedan dokument	Novosti	- ključne fraze - strojno učenje - otkrivanje entiteta (NER)	E
MSR-NLP Summarizer	Više dokumenata	Novosti	- sažimanje temeljeno na grafovima	E
CATS	Više dokumenata	Novosti	- statistička analiza rečenica	E
CLASSY	Više dokumenata	Novosti	- sažimanje temeljeno na upitima	E
QASUM-TALP	Više dokumenata	Novosti	- sažimanje temeljeno na upitima	E
ERRS	Jedan i više dokumenata	Novosti	- sažimanje temeljem nejasne (fuzzy) logike	E
FemSum	Jedan i više dokumenata	Novosti	- sažimanje temeljeno na grafovima	E
GOFAISUM	Više dokumenata	Novosti	- TF-IDF sažimanje	E
NetSum	Jedan dokument	Novosti	- strojno učenje - neuronske mreže	E
MultiGen	Više dokumenata	Novosti	- otkrivanje sličnih elemenata kroz jedan ili više dokumenata - spajanje i reformulacija sličnih rečenica	A
Cut&Paste	Jedan dokument	Opća namjena	- leksikalna koherencija rečenica - TF-IDF rezultat - ključne fraze - pozicija rečenice	A
SumUM	Jedan dokument	Tehnički članci	- sintaktička i semantička analiza - identifikacija koncepata i bitnih informacija	A
COLUMBIA MDS	Više dokumenata	Novosti	- statistička analiza - varijanta MultiGen sustava	E i A

Iz tablice 4 je vidljivo da različiti sustavi koriste različite metode za generiranje sažetaka. Također se razlikuju u namjenama, tj. prilagođeni su za određeno područje ili su namijenjeni općoj upotrebi. Neki od sustava su prilagođene verzije drugog sustava radi poboljšanja očekivanog rezultata. Može se primijetiti da mnogo više sustava koristi ekstraktivno sažimanje od apstraktivnog.

Iako nisu u tablici navedeni, postoje i sustavi za sažimanje teksta na hrvatskom jeziku. Jedan takav sustav je CroWebSum (Mikelić Preradović *et al.,* 2010) koji sažima tekst ekstraktivnom TF-IDF metodom na određeni postotak (2 %, 5 %, 10 % ili 20 %) izvornog teksta. Autori ističu probleme fleksije hrvatskog jezika koja uzrokuje poteškoće u sažimanju. Sličan sustav je i CROSUM (Lauc *et al.,* 2005), međutim ograničen je samo na znanstvene radove i članke dok je CroWebSum baziran na novinskim člancima.

3. Implementacija i izrada

Kako ne bi sve ostalo samo na teoriji, u ovom poglavlju se daje kratak uvid u danas dva najveća[6] jezična modela prema broju parametara za obradu i sažimanje prirodnog jezika – Microsoft Turing NLG[7] i OpenAI GPT-3[8]. Oba sustava su komercijalnog tipa te nije bilo moguće iskoristiti ih za izradu vlastite aplikacije i realnu usporedbu, no zorno dokazuju kakav trud i napor, pa i financije su potrebne za razvoj alata i aplikacija koje razumiju prirodni jezik i mogu ga sažeti na način da je teško, ako ne i nemoguće, razaznati razliku između čovjeka i računala.

Također se prikazuje konkretna izrada jednostavnog alata za automatsko sažimanje teksta u programskom jeziku Python[9]. Primijenjene su tri od ranije spomenutih metoda za izradu te je objašnjeno kako se konkretno principi interpretiraju u programskom jeziku. Radi se i usporedba s besplatnim web aplikacijama za sažimanje teksta[10].

3.1. Turing NLG

Turing NLG (engl. *Natural Language Generation*) je jezični model sa 17 milijardi parametara kojeg je razvio Microsofta i objavljen u veljači 2020. te je do pojave GPT-3 u lipnju iste godine prema Microsoftu bio daleko najveći [11]jezični model na svijetu.

Prema Microsoftu, Turing NLG model je sposoban završavati nedovršene rečenice na engleskom jeziku (čija primjena se može vidjeti npr. u MS Teams aplikaciji), može generirati odgovore na pitanja i sažetke ulaznih tekstova te općenito generirati tekst na zadanu temu. Također napominju da su primijetili povezanost između veličine i raznolikosti modela i kvalitete sažetaka – čim je veći model, tim sažetak djeluje prirodnije i teže ga je prepoznati od sažetka koji bi napisao čovjek.

---

[6] https://analyticsindiamag.com/open-ai-gpt-3-language-model/

[7] https://msturing.org/

[8] https://en.wikipedia.org/wiki/GPT-3

[9] https://en.wikipedia.org/wiki/Python_(programming_language)

[10] Pozadina funkcioniranja web aplikacija nije poznata, stoga na usporedbu treba staviti zvijezdicu – drugačiji algoritmi daju drugačije rezultate.

[11] https://www.wired.com/beyond-the-beyond/2020/02/web-semantics-microsoft-project-turing-introduces-turing-natural-language-generation-t-nlg/

U tablici 5 je prikazan primjer sažimanja teksta s Turing NLG iz Microsoftove objave za javnost[12]. Objašnjenje su dali uspoređujući rezultate internetske tražilice koje su zatim proveli kroz Turing NLG algoritam, ograđujući se da je ovakav pristup možda važniji u nekim drugim sustavima, poput osobnih asistenata pokretanih umjetnom inteligencijom (engl. *Personal Assistant AI*) kada su potrebni sažeti, precizni odgovori.

Tablica 5 Primjer Turing NLG sažimanja prema Microsoftu (prilagođeno s engleskog)

Pitanje	**Za koga je Jason Mraz bio zaručen?**
Tekst	Mraz se zaručio za pjevača/autora i dugogodišnjeg bliskog prijatelja Tristana Prettymana na badnjak 2010. godine. Zaruke su raskinuli šest mjeseci kasnije.
Odgovor	*Jason Mraz je bio zaručen za Tristana Prettymana*

Osim sažimanja, Turing NLG je sposoban odgovoriti na pitanja bez ulaznog teksta, ovisno o znanju koje se nalazi u samom modelu. U primjeru je dano pitanje o kraju Drugog svjetskog rata na koje je Turing NLG točno odgovorio. Na žalost, ni jednu od ovih opcija trenutno nije moguće koristiti prema open-source licencama.

3.2. GPT-3

GPT-3 (engl. *Generative Pre-trained Transformer 3*)[13] je trenutno najveći jezični model na svijetu sa 175 milijardi parametara – čak 10 puta više od ranije spomenutog Turing NLG-a. Razvila ga je tvrtka OpenAI kao nasljednik njihovog GPT-2 modela. Prema autorima, kvaliteta teksta koji se može generirati ovim modelom je toliko visoka da je gotovo nemoguće razlikovati ga od teksta napisanog od strane čovjeka. Kao i Turing NLG, sposoban je za sažimanje teksta, odgovaranje na pitanja te generiranje teksta na zadanu temu. Za razliku od Turing NLG, GPT-3 osim engleskog podržava više jezika poput njemačkog, ruskog, japanskog[14] i sl. Tvrtka je odlučila da ga neće davati pod open-source licencom, već samo u komercijalne svrhe, no može se zatražiti pristup preko API poziva za razvoj aplikacija (također

[12] https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/

[13] Članak o GPT-3 modelu, razvoju, funkcioniranju i rezultatima dostupan je na adresi: https://arxiv.org/pdf/2005.14165.pdf

[14] https://www.sigmoid.com/blogs/gpt-3-all-you-need-to-know-about-the-ai-language-model/

komercijalno), a kao glavni razlog tome spominje se bojazan da bi se sustav mogao zloupotrijebiti za generiranje lažnih vijesti i dezinformacija.

Zanimljiva upotreba GPT-3 algoritma je nastala od strane Andrewa Mayne i njegove AI Writer[15] aplikacije koja omogućuje ljudima pisanu korespodenciju s povijesnim ličnostima. Aplikacija koristi GPT-3 sustav kako bi temeljem pisanih djela određene povijesne ličnosti generirala tekst pisan u stilu te osobe. Iako se ne može tvrditi da bi izabrana osoba uistinu tako odgovarala, zanimljivo je kako tehnologija približava povijest i uvid u način razmišljanja određenih ljudi. Neka od povijesnih ličnosti čija djela su unesena u AI Writer aplikaciju su: Alan Turing, Stephen Hawking, Isaac Newton, Isaac Asimov, Benjamin Franklin, Edgar Allan Poe...

3.3. Python

Zašto baš Python za izradu alata za sažimanje i općenito obradu prirodnog jezika? Zbog svoje specifične sintakse, Python se relativno jednostavno može koristiti za obradu teksta. Postoji ogroman broj biblioteka (engl. *library*) koje dodatno olakšavaju rad u određenom području. Dvije najpoznatije biblioteke za obradu prirodnog jezika su spaCy[16] i NLTK[17] te se obje koriste za izradu zasebnih programa za sažimanje teksta.

Također, postoji i velika on-line zajednica koja radi na poboljšanjima te je spremna pomoći korisnicima. Potrebno je još reći kako je Python prevoditeljski tip programskog jezika (engl. *interpretter,* različiti princip od *compiler* tipa programskih jezika kao što je npr. C++) i radi na svim većim platformama i operacijskim sustavima.

3.3.1. NLTK

Prema izvornoj web stranici, NLTK (engl. *Natural Language ToolKit* – skup alata za obradu prirodnog jezika) je platforma za izradu Python programa koji koriste prirodni jezik kao skup podataka. Mnoge stvari, poput tokenizacije, klasifikacije, stemizacije, tagiranja,

[15] Aplikacija za pisanu korespodenciju s povijesnim ličnostima. Za više vidi: https://www.aiwriter.app/

[16] https://spacy.io/

[17] https://en.wikipedia.org/wiki/Natural_Language_Toolkit

parsiranja itd. su ugrađene u NLTK i korisnik ih samo treba pozvati, što uvelike olakšava cijeli proces programiranja.

3.3.2. Model sažimanja temeljen na grafovima

Kako bi se usporedili razni pristupi sažimanju, osim sažimanja pomoću NLTK biblioteke, izvodi se i sažimanje temeljeno na grafovima. Algoritam uči na korpusu te temeljem naučenog rangira rečenice koje se prije toga tokeniziraju. Za tokenizaciju se koristi već poznata NLTK biblioteka, no nema drugog utjecaja na sažimanje.

3.3.3. spaCy

Kao i NLTK, spaCy je biblioteka za naprednu obradu prirodnog jezika koja koristi neuronske mreže za stvaranje sažetka. Podržava kategorizaciju teksta, POS (engl. *part of speech*) i NER (engl. *named entity recognition*) označavanje, tokenizaciju, parsiranje teksta itd. Sadrži ugrađene statističke modele za 17 jezika, uključujući i hrvatski.

4. Eksperiment

U ovom poglavlju predstavljaju se 3 modela sažimanja teksta na hrvatskom i engleskom jeziku. Tekst koji se koristi je preuzet s Wikipedije, a radi se o životopisu hrvatske operne pjevačice Ruže Pospiš-Baldani. Hrvatska i engleska verzija teksta se donekle razlikuju u sadržaju te broju rečenica, riječi i odlomaka, kako je pokazano u tablici 6.

Tablica 6 Usporedba teksta na hrvatskom i engleskom jeziku

	Hrvatski	Engleski
Rečenice	11	7
Riječi	382	222
Odlomci	7	2

Slike 6 i 7 pokazuju razliku u količini teksta hrvatske i engleske verzije.

Ruža Pospiš Baldani (Varaždinske Toplice, 25. srpnja 1942.)[1] je hrvatska operna pjevačica (mezzosopran). Od 2010. dopisna je članica Razreda za glazbenu umjetnost i muzikologiju Hrvatske akademije znanosti i umjetnosti.[2]

Na prvoj izvedbi u Hrvatskoj opere "*Rat i mir*" Sergeja Prokofjeva u HNK u Zagrebu (20. studenog 1961.) posebnu je pozornost privukao glas orguljske ljepote, baršunaste mekoće i bogatih preljeva u maloj ulozi Muratova ađutanta. Pripadao je studentici Muzičke akademije Ruži Pospiš, poslije udanoj Baldani.

Ruža Pospiš za čitave je svoje duge četrdesetogodišnje karijere od 1961. do 2001. ostala vjerna Zagrebačkoj operi i usporedno stjecala svjetski ugled. Počelo je to već dvije godine nakon debuta, u napuljskome San Carlu u ulozi Venere u Offenbachovu "*Orfeju*" u podzemlju, nastavilo se 1964. godine na Holland festivalu s Marinom u "*Borisu Godunovu*" i Dubrovačkim ljetnim igrama s Oktavijom u "*Krunidbi Popeje*" pod ravnanjem Lovre pl. Matačića, 1965. nastupom na festivalu u Edinburghu u Haydnovoj operi "*Ribarice*" i 16. veljače 1966. godine s prvim nastupom u Metropolitanu kao Maddalena iznad prosjeka u "*Rigolettu*".

Angažirana u Metu, bila je 1970. njegova ***Carmen*** u režiji Jean-Louisa Barraulta, koju je upoznala i široka publika u tri radijska prijenosa from coast to coast (od obale do obale). U 1975. godine, kada se proslavljala stota obljetnica praizvedbe "*Carmen*", bila je zacijelo njezina najtraženija interpretkinja, od premijere u Covent Gardenu do Bečke državne opere i Madrida.

Suradnja s velikim dirigentima Karajanom, Karlom Richterom, Claudiom Abbadom vodila ju je na Salzburške svečane igre i Uskrsne svečane igre u istome gradu, u *Scalu*, *Bavarsku državnu operu* u Münchenu, *Teatro del Liceo* u Barceloni, u najveće koncertne dvorane *Berlinske* i *Bečke filharmonije*, Carnegie Hall, u Vatikan pred Papu Pavla VI.

Njezin jedinstveno lijep glas, velika muzikalnost i prekrasan legato uz atraktivnu scensku pojavu činili su je raskošnom *Carmen*, *Dalilom* i *Amneris*, golem glasovni potencijal izražajnom Azucenom, uznositost interpretacije primjerenim "Orfejom", zrelost psihološkoga poniranja u lik izvrsnom Marfom u Hovanščini, proglašenom u njemačkom časopisu Opernwelt najboljom kreacijom u 2000., a uzvišena mirnoća fraze idealnom interpretkinjom velikih vokalno-instrumentalnih djela Bacha, Beethovena, Verdija, Händela.

Slika 6 Hrvatska verzija teksta o Ruži Pospiš-Baldani

Ruža Pospiš-Baldani (Croatian pronunciation: [rǔ͡ʒa pǒspiʃ baldâni]; born 25 July 1942) is a Croatian operatic mezzo-soprano.

Baldani was born in Varaždinske Toplice[1] and made her professional opera debut in 1961 at the Croatian National Theatre in Zagreb as Konchakovna in Alexander Borodin's *Prince Igor*. She remained active at that theatre and at the National Theatre in Belgrade throughout the 1960s. In 1965 she made her debut at the Metropolitan Opera in New York City as Maddalena in Giuseppe Verdi's *Rigoletto*. From 1970-1978 she was committed to the Bavarian State Opera. Between 1973 and 1987 she was a frequent guest artist at the Vienna State Opera; drawing particular acclaim there as Brangäne in Richard Wagner's *Tristan und Isolde*. In 1976 she made her debut at the Paris Opera as Amneris in Verdi's *Aida*, and made her first appearance at the Opéra de Monte-Carlo in the title role of Georges Bizet's *Carmen*. She has since appeared as a guest artist at the Cologne Opera, the Edinburgh Festival, the Greek National Opera, the Hamburg State Opera, the Houston Grand Opera, the Hungarian State Opera House, La Scala, the Liceu, the Lyric Opera of Chicago, the National Opera of Sofia, the Salzburg Festival, the San Francisco Opera, the Savonlinna Opera Festival, the Teatro dell'Opera di Roma, the Teatro di San Carlo, and the Teatro Municipal in Rio de Janeiro among others.[2]

Slika 7 Engleska verzija teksta o Ruži Pospiš-Baldani

Cilj eksperimenta je vidjeti koji od tri modela daje bolje rezultate na oba jezika i to prema sljedećim kriterijima:

- vidljivost subjekta;
- duljina sažetka;
- koherentnost teksta.

4.1. Sažimanje članaka s Wikipedije pomoću NLTK biblioteke

Ovo je prvi primjer programa za sažimanje teksta te se ujedno može smatrati jednostavnijim modelom. NLTK biblioteka rangira riječi prema učestalosti te generira sažetak. Uz NLTK biblioteku, potrebne su još biblioteka *beautiful soup*[18] te *lxml*[19], koje služe za dohvaćanje tekstova s web stranica.

4.1.1. Model 1

Najprije s Wikipedije proizvoljno odaberemo članak. Za potrebe ovog primjera, uzet je članak o hrvatskoj opernoj pjevačici Ruži Pospiš Baldani[20]. NLTK biblioteka sažima tekst ekstraktivnim metodama na sljedeći način:

Unos članka → podjela na rečenice → izbacivanje zaustavnih riječi → izračun učestalost pojavljivanja → rangiranje temeljem učestalosti pojavljivanja → odabir top N rečenica za sažetak

```
[1] import bs4 as bs
[2] import urllib.request
[3] import re
[4] import nltk
[5] from stop_words import get_stop_words
[6] scraped_data =
urllib.request.urlopen('https://hr.wikipedia.org/wiki/Ru%C5%BE
a_Pospi%C5%A1_Baldani')
[7] article = scraped_data.read()
[8] parsed_article = bs.BeautifulSoup(article,'lxml')
[9] paragraphs = parsed_article.find_all('p')
[10] article_text = ""
[11] for p in paragraphs:
[12]     article_text += p.text
```

Kodni blok 1. Dohvaćanje web stranice koristeći NLTK biblioteku

Prvi dio koda (*Kodni blok 1*) dohvaća željenu web stranicu, odvaja paragrafe i stavlja ih u listu. Funkcija `urlopen` (Kodni blok 1, red [6]) dohvaća web stranicu, `read` (Kodni blok 1, red [7]) čita tekst na njoj i zatim biblioteke `BeautifulSoup` i `lxml` (Kodni blok 1, red [8]) tekst parsiraju.

[18] https://pypi.org/project/beautifulsoup4/

[19] https://pypi.org/project/lxml/

[20] https://hr.wikipedia.org/wiki/Ru%C5%BEa_Pospi%C5%A1_Baldani

Wikipedija tekstovi su standardizirano unutar HTML <p> oznaka. Kako bi se izvadio samo tekst, bez tih oznaka, zove se funkcija `find_all` (Kodni blok 1, red [9]). Tom funkcijom traži se sav tekst podijeljen u paragrafe i stavlja ih u listu. Nakon toga, potrebno je pročistiti tekst od metaoznaka. Iz teksta izbacujemo sve nepotrebne znakove poput brojeva, uglatih zagrada, višak razmaka i sl. (*Kodni blok 2*).

```
[13] article_text = re.sub(r'\[[0-9]*\]', ' ', article_text)
[14] article_text = re.sub(r'\s+', ' ', article_text)

[15] formatted_article_text = re.sub('[^a-zA-Z]', ' ',
article_text )
[16] formatted_article_text = re.sub(r'\s+', ' ',
formatted_article_text)
```

Kodni blok 2.

Sada postoje dvije varijable koje sadrže tekst. Varijabla `article_text` (Kodni blok 1, red [12]) sadrži originalni članak, dok će se varijabla `formatted_article_text` (Kodni blok 2, red [16]) koristiti za računanje frekvencije pojavljivanja riječi, koje će prema takvom rangiranju zatim biti zamijenjene riječima iz varijable `article_text` (Kodni blok 2, red [15]).

Sljedeći korak je tokenizacija članka u rečenice. Koristi se varijabla `article_text` jer sadrži interpunkcijske znakove. Tokenizacija se vrši kodom kao što je prikazano u Kodnom bloku 3.

```
[17] sentence_list = nltk.sent_tokenize(article_text)
```

Kodni blok 3.

Nakon tokenizacije, traži se učestalost pojavljivanja svake riječi unutar `formatted_article_text` varijable. To se čini tako da se iz NLTK biblioteke preuzmu tzv. zaustavne riječi (engl. *stopwords*) (Kodni blok 4, red [18]). Iako postoje zaustavne riječi za neke druge slavenske jezike poput češkog, poljskog, slovačkog te ruskog, bugarskog i ukrajinskog na ćirilici, na žalost, hrvatski jezik nije podržan. Obzirom na to, može se ostaviti engleski ili pokušati s nekim od slavenskih jezika. Rezultati su nešto drugačiji, no ne značajno.

```
[18] stopwords = get_stop_words('english') #czech, polish,
slovak
[19] word_frequencies = {}
[20] for word in nltk.word_tokenize(formatted_article_text):
[21]    if word not in stopwords:
[22]        if word not in word_frequencies.keys():
```

```
[23]             word_frequencies[word] = 1
[24]         else:
[25]             word_frequencies[word] += 1
```

Kodni blok 4.

Petljom se prolazi kroz rečenice i traže zaustavne riječi, a zatim uspoređuje postoje li već u rječniku `word_frequency` (Kodni blok 4, red [22]). Ako se riječ pojavila prvi puta, stavlja se u rječnik kao ključ, dok se vrijednost postavlja na 1. Ukoliko riječ već postoji u rječniku, vrijednost se povećava za 1. Nakon prolaska kroz cijeli tekst, dijeli se broj pojavljivanja riječi s brojem pojavljivanja najčešće riječi (*Kodni blok 5*):

```
[26] maximum_frequncy = max(word_frequencies.values())
[27] for word in word_frequencies.keys():
[28]     word_frequencies[word] =
(word_frequencies[word]/maximum_frequncy)
```

Kodni blok 5.

Po računanju učestalosti pojavljivanja, računa se rezultat svake rečenice zbrajanjem rezultata riječi svake rečenice.

```
[29] sentence_scores = {}
[30] for sent in sentence_list:
[31]    for word in nltk.word_tokenize(sent.lower()):
[32]        if word in word_frequencies.keys():
[33]            if len(sent.split(' ')) < 45:
[34]                if sent not in sentence_scores.keys():
[35]                    sentence_scores[sent] =
word_frequencies[word]
[36]                else:
[37]                    sentence_scores[sent] +=
word_frequencies[word]
```

Kodni blok 6.

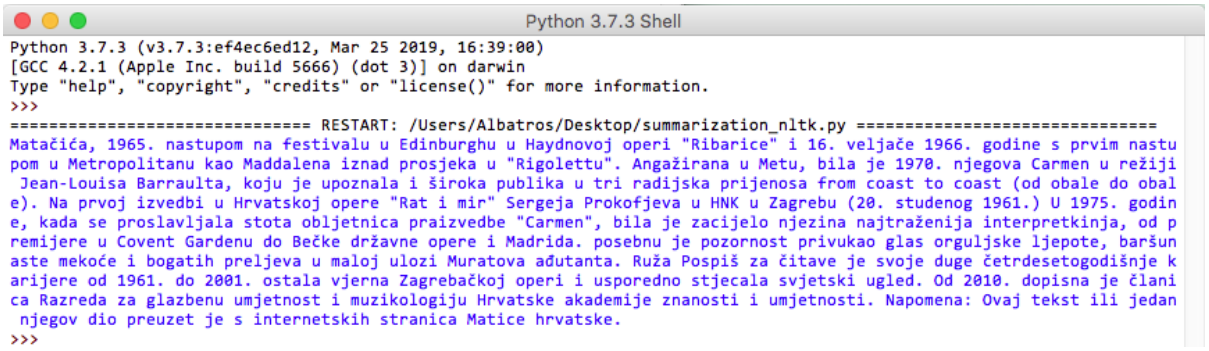
Ključevi novog rječnika `sentence_scores` su rečenice, dok su vrijednosti rezultat te rečenice. Svaka rečenica se ponovo tokenizira na riječi. Obzirom da sažetak treba biti kratak, određeno je da se za njega odabiru samo rečenice s manje od 45 riječi. Broj riječi je proizvoljan. Kada se sve rečenice stave u rječnik, zajedno s vrijednostima, uzima se 8 rečenica s najvećim rezultatom. Broj rečenica koje se stavljaju u sažetak je također proizvoljan.

```
[38] import heapq
[39] summary_sentences = heapq.nlargest(8, sentence_scores,
key=sentence_scores.get)
[40] summary = ' '.join(summary_sentences)
[41] print(summary)
```

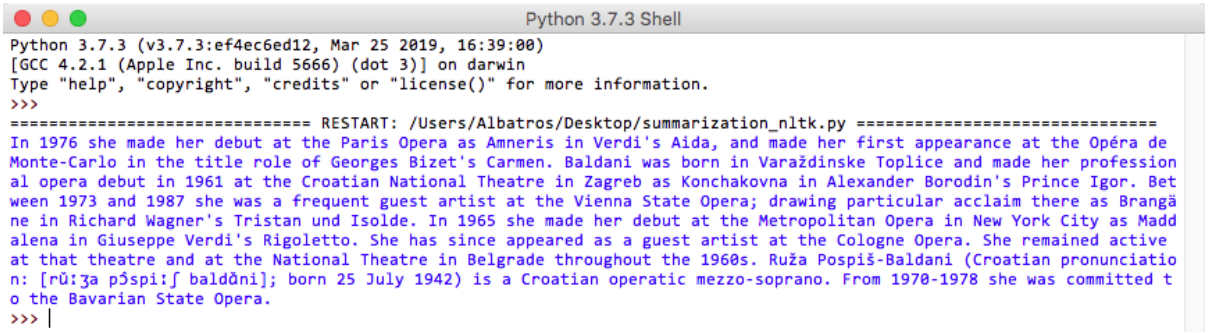
Kodni blok 7.

4.1.2. Rezultat 1

Na slikama 8 i 9 prikazani su rezultati sažimanja hrvatske i engleske verzije istog članka. Za potrebe sažimanja preuzet je samo dio koji se odnosi na sekciju «Životopis». Kako su stranice različitih jezika neovisne jedna o drugoj, tako se tekst na njima može više ili manje razlikovati. Upravo je to i ovdje slučaj – engleska verzija teksta samo je djelomičan prijevod hrvatske verzije, te osim životopisa ni nema daljnjeg teksta. Zbog nepostojanja hrvatskih zaustavnih riječi unutar NLTK biblioteke, engleska verzija djeluje koherentnija. Kada se upotrijebe zaustavne riječi drugih slavenskih jezika, rezultat je sličan.



Slika 8 Prikaz hrvatskog sažetka s Wikipedije pomoću NLTK-a



Slika 9 Prikaz engleskog sažetka s Wikipedije pomoću NLTK

Temeljem ovakvog rezultata, može se zaključiti da nedostatak hrvatskih zaustavnih riječi u NLTK biblioteci utječe na kvalitetu sažetka. Na početku sažetka nedostaje subjekt i uvod o kome je uopće riječ, međutim ostatak teksta je sažet i daje informacije o subjektu koje se mogu smatrati korisnima. Uz neke prilagodbe NLTK biblioteke poput automatskog uključivanja prve rečenice u sažetak, te dodavanja hrvatskih zaustavnih riječi, ovaj model je svakako upotrebljiv. Također treba uzeti u obzir da se članci na Wikipediji s vremenom mijenjaju te je moguće dobiti nešto drugačiji sažetak prilikom pokretanja programa u budućnosti, stoga se u radu

prilaže tekst (prilog 4 i 5) koji se koristi u analizi sva 3 modela na hrvatskom i engleskom jeziku.

4.2. Implementacija sažimanja temeljenog na grafovima u Pythonu

U implementaciji sažimanja temeljenog na grafovima računaju se težinske vrijednosti važnih dijelova rečenice i na taj način se stvara sažetak. Koristi se pristup nenadziranog učenja za nalaženje sličnosti i rangiranja rečenica. Program uči na temelju GloVe algoritma (Pennington *et al.*, 2014) koji sadrži vektorske prikaze riječi temeljene na nekom korpusu. Članak koji se koristi kao primjer u ovom radu je članak s Wikipedije, stoga je primjereno koristiti GloVe Wikipedija korpus[21]. Slijedi primjer implementacije ekstraktivnog modela.

4.2.1. Model 2

Prije samog početka programiranja, potrebno je odabrati model na kojem će se program bazirati. Odabran je model prema ekstraktivnoj metodi sažimanja na sljedeći način:

Unos članka → podjela na rečenice → izbacivanje zaustavnih riječi → izrada matrice sličnosti → generirajne rangiranja temeljem matrice → odabir top N rečenica za sažetak

Matrica sličnosti se radi temeljem tzv. kosinusne sličnosti[22], za što već postoji Python biblioteka `cosine_similarity`. To je moguće iz razloga što se svaka rečenica prikazuje kao vektor, a time je moguće izračunati i kuteve između njih. Kut je 0, ako su rečenice slične. U kodnom bloku 8 se dohvaćaju potrebne biblioteke.

```
[1] import numpy as np
[2] import pandas as pd
[3] import nltk
[4] from nltk.tokenize import sent_tokenize
[5] from nltk.corpus import stopwords
[6] import re
[7] from sklearn.metrics.pairwise import cosine_similarity
[8] import networkx as nx
```

Kodni blok 8.

[21] https://nlp.stanford.edu/data/glove.6B.zip

[22] Kosinusna sličnost je način mjerenja koji se koristi za utvrđivanje sličnosti teksta, bez obzira na njegovu veličinu. Za više vidi: https://www.machinelearningplus.com/nlp/cosine-similarity/

Nakon dohvaćanja potrebnih biblioteka, otvaramo dokument s tekstom koji se sažima kodom prikazanim u kodnom bloku 9.

```
[9] df = pd.read("ruza_pospis_hr.txt")
```

Kodni blok 9.

U kodnom bloku 10 tekst se tokenizira na pojedinačne rečenice i stavlja u listu `sentences` (Kodni blok 10, red [12]). Koristi se funkcija `sent_tokenize` iz NLTK biblioteke.

```
[10] sentences = []
[11] for s in df:
[12]   sentences.append(sent_tokenize(s))
[13] sentences = [y for x in sentences for y in x]
```

Kodni blok 10.

Prije početka građenja matrice, kao i u primjeru s NLTK bibliotekom, tekst se «čisti» od nepotrebnih znakova te se sva slova pretvaraju u mala (engl. *lowercase*), kao što je vidljivo u kodnom bloku 11. Za brisanje zaustavnih riječi, koristi se NLTK biblioteka.

```
[14] clean_sentences = pd.Series(sentences).str.replace("[^a-
zA-Z]", " ")
[15] clean_sentences = [s.lower() for s in clean_sentences]
[16] stop_words = stopwords.words('english')
[17] def remove_stopwords(sen):
[18]     sen_new = " ".join([i for i in sen if i not in
stop_words])
[19]     return sen_new
[20] clean_sentences = [remove_stopwords(r.split()) for r in
clean_sentences]
```

Kodni blok 11.

Varijabla `clean_sentences` (Kodni blok 11, red [14]) se koristi za stvaranje rečeničnih vektora pomoću naučenih podataka iz GloVe skupa. U kodnom bloku 12 ti vektori se stavljaju u rječnik `word_embeddings`.

```
[21] word_embeddings = {}
[22] f = open('glove.6B.100d.txt', encoding='utf-8')
[23] for line in f:
[24]    values = line.split()
[25]    word = values[0]
[26]    coefs = np.asarray(values[1:], dtype='float32')
[27]    word_embeddings[word] = coefs
[28] f.close()
```

Kodni blok 12.

U sljedećem kodnom bloku (13) se se stvaraju vektori za ulazni tekst. Najprije se dohvaćaju vektori u skupu od 100 elemenata. Broj elemenata određuje veličina vektora u GloVe skupu, moguće je dohvatit vektore od 50, 100, 200 i 300 elemenata. Što se veći vektori dohvaćaju, to je vrijeme izvršavanja programa dulje. Nakon toga se radi prosjek tih vektora kako bi se dobio jedan vektor po rečenici.

```
[29] sentence_vectors = []
[30] for i in clean_sentences:
[31]   if len(i) != 0:
[32]     v = sum([word_embeddings.get(w, np.zeros((100,))) for
w in i.split()])/(len(i.split())+0.001)
[33]   else:
[34]     v = np.zeros((100,))
[35]   sentence_vectors.append(v)
```

Kodni blok 13.

Nakon toga izrađuje se matrica kosinusne sličnosti (Kodni blok 14, red [40]) kako bi se pronašle sličnosti među rečenicama.

```
[36] sim_mat = np.zeros([len(sentences), len(sentences)])
[37] for i in range(len(sentences)):
[38]   for j in range(len(sentences)):
[39]     if i != j:
[40]       sim_mat[i][j] = []
cosine_similarity(sentence_vectors[i].reshape(1,100),
sentence_vectors[j].reshape(1,100))[0,0]
```

Kodni blok 14.

Kada su sve rečenice reprezentirane u matrici `sim_mat` (Kodni blok 14, red [36]), pretvaramo tu matricu u grafove. Čvorovi grafa su rečenice, a rubovi među njima su vrijednosti sličnosti iz matrice. Za rangiranje prema veličini, koristi se PageRank[23] (Kodni blok 15, red [42]). algoritam. Na kraju se uzima proizvoljan broj rečenica temeljem rangiranja (u ovom slučaju 8, kako bi se držali isti kriteriji kao i u prethodnom primjeru) i ispiše sažetak.

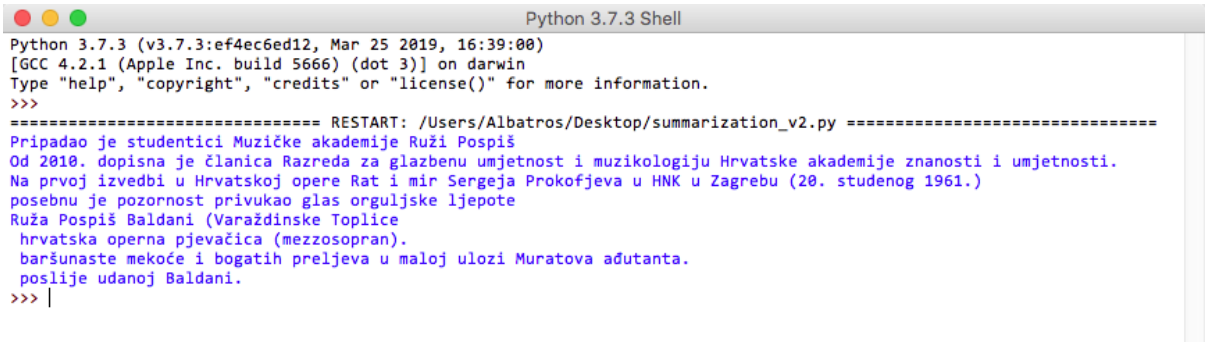
```
[41] nx_graph = nx.from_numpy_array(sim_mat)
[42] scores = nx.pagerank(nx_graph)
[43] ranked_sentences = sorted(((scores[i],s) for i,s in
enumerate(sentences)), reverse=True)
[44] for i in range(10):
[45]   print(ranked_sentences[i][1])
```

Kodni blok 15.

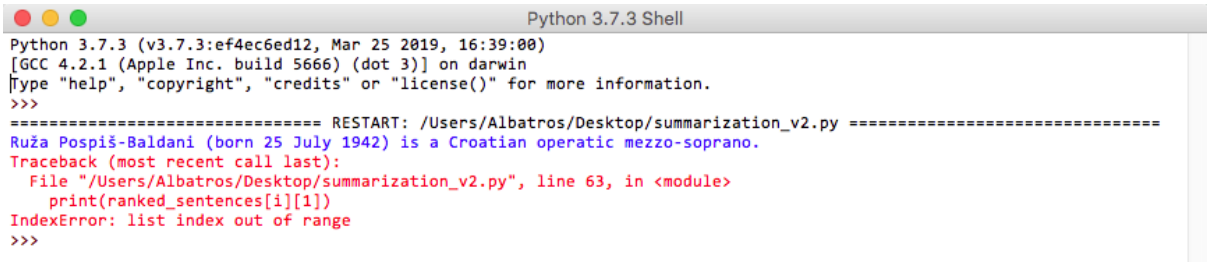
[23] PageRank je algoritam koji koristi Google za rangiranje web stranica prilikom prikazivanja rezultata pretraživanja. Za više vidi: https://en.wikipedia.org/wiki/PageRank

4.2.2. Rezultat 2

Na slikama 10 i 11 prikazani su rezultati sažimanja hrvatske i engleske verzije istog članka kao iz prethodnog modela. Kada se usporede sa sažecima pomoću NLTK biblioteke, može se vidjeti da ova metoda daje vidljivo drugačije rezultate. Još uvijek postoje nedostaci u tekstu na hrvatskom jeziku, ali lakše je doći do subjekta nego u prethodnom modelu. Rečenice djeluju kao da su odsječene i krivo poredane, ali uz poboljšanja algoritma u vidu usporedbe mjesta rečenice u tekstu i premještanja rangiranih rečenica na adekvatnu poziciju, ovaj algoritam može davati vrlo dobre rezultate.



Slika 10 Prikaz rezultata sažimanja teksta na hrvatskom jeziku temeljenog na grafovima



Slika 11 Prikaz rezultata sažimanja teksta na engleskom jeziku temeljenog na grafovima

Što se tiče sažetka na engleskom, on se sastoji od samo jedne rečenice. Ovo je za sad prvi primjer da je subjekt eksplicitno naveden, a u usporedbi s izvornim tekstom, prva rečenica je uključena u sažetak. Rezultat je vrlo kratak, ali i vrlo precizan, bez detalja o subjektu. Greška koja se pojavljuje je iz razloga što se algoritam vrti još 7 puta (jer traži zadanih 8 rečenica), no sažetak, iako štur, je dobar.

4.3. Implementacija sažimanja temeljenog na spaCy neuronskoj mreži

Poput prijašnjih modela i ovaj je ekstraktivni. SpaCy biblioteka koristi statistički model hrvatskog jezika temeljen na korpusu. Korpus koji se koristi je najveći zadani korpus iz spaCy okruženja baziran na vijestima (*hr_core_news_lg)* iako se može koristiti proizvoljan.

4.3.1. Model 3

Ovo rješenje koristi model kako slijedi:

Unos članka → tokenizacija (podjela na rečenice) → izrada rječnika → težinsko rangiranje riječi u rečenicama → rangiranje rečenica prema zbroju težinskog ranga riječi → odabir top N rečenica za sažetak

Na početku se učitava hrvatski statistički model (Kodni blok 16, red [3]) te se ulazni tekst obrađuje prema njemu (Kodni blok 16, red [4]). Varijabla `doc` nije ništa drugo do tokenizirani tekst, tj podjela na rečenice. Nakon toga stvara se rječnik `word_dict` u koji se stavljaju riječi te im se povećava vrijednost svaki puta kada se riječ pojavi u rječniku (Kodni blok 16, red [9]).

```
[1] import spacy
[2] nlp = spacy.load("hr_core_news_lg")
[3] text = pd.read("ruza_pospis_hr.txt")
[4] doc = nlp(text)

[5] word_dict = {}
[6] for word in doc:
[7]     word = word.text.lower()
[8]     if word in word_dict:
[9]         word_dict[word] += 1
[10]     else:
[11]         word_dict[word] = 1
```

Kodni blok 16.

Temeljem rezultata u rječniku, u kodnom bloku 17 se radi lista rečenica, gdje se svakoj rečenici pridaje vrijednost svake riječi koja se u njoj nalazi. Ujedno se u varijablu `sents` (Kodni blok 17, red[18]) pridaju novokreirane rečenice.

```
[12] sents = []
[13] sent_score = 0
[14] for index, sent in enumerate(doc.sents):
[15]     for word in sent:
[16]         word = word.text.lower()
[17]         sent_score += word_dict[word]
[18]     sents.append((sent.text.replace("\n"," "),
sent_score/len(sent), index))
```

Kodni blok 17.

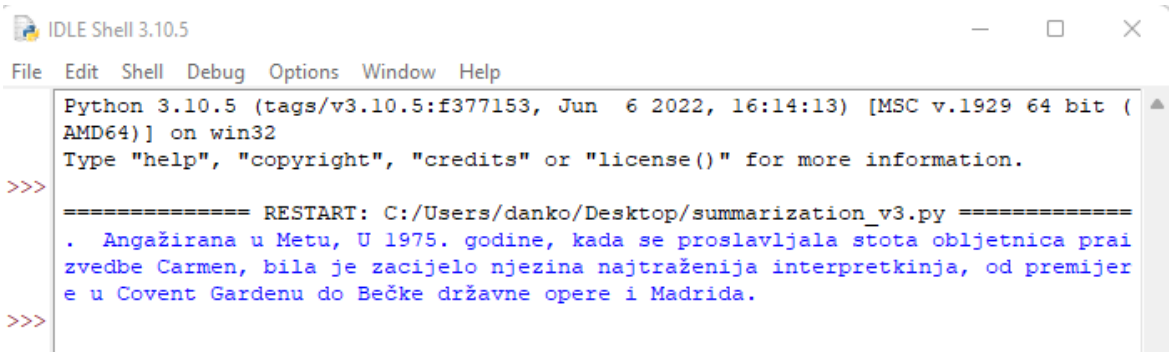
Na kraju se u kodnom bloku 18 rečenice sortiraju prema dobivenom rezultatu (Kodni blok 18, red [19]) te se odabiru prve 3 rečenice (Kodni blok 18, red [20]) za izlazni tekst i stavljaju u varijablu `summary_text` (Kodni blok 18, red [23]) koja se zatim ispisuje. Broj rečenica za izlazni tekst je odabran zbog kratkoće ulaznog teksta.

```
[19] sents = sorted(sents, key=lambda x: -x[1])
[20] sents = sorted(sents[:3], key=lambda x: x[2])
[21] summary_text = ""
[22] for sent in sents:
[23]     summary_text += sent[0] + " "
[24] print(summary_text)
```

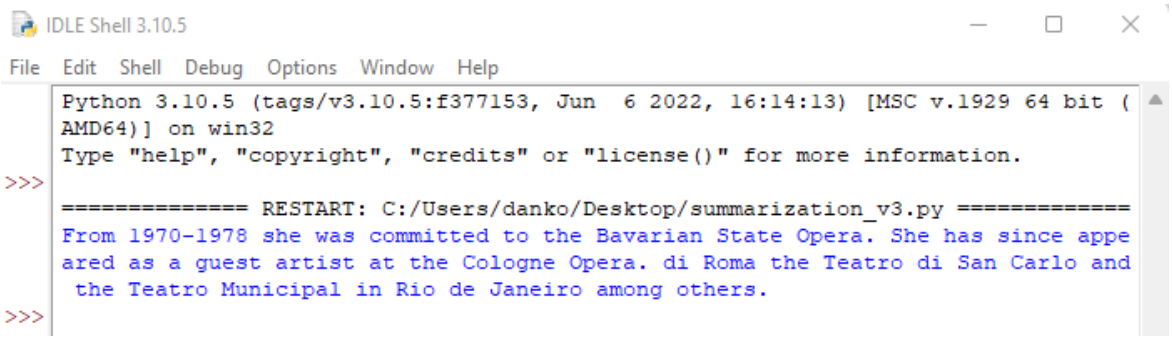
Kodni blok 18.

4.3.2. Rezultat 3

Slike 12 i 13 prikazuju rezultate sažimanja na hrvatskom i engleskom jeziku pomoću spaCy biblioteke. Potrebno je napomenuti da je engleski korpus gotovo dvostruko veći od hrvatskog. Rezultati su dosta različiti, a također se razlikuju i od rezultata u prva dva modela.



Slika 12 Prikaz rezultata sažimanja teksta na hrvatskom temeljenog na spaCy neuronskoj mreži



Slika 13 Prikaz rezultata sažimanja teksta na engleskom temeljenog na spaCy neuronskoj mreži

Tablica 7 prikazuje rezultate prema kriterijima određenim na početku poglavlja za svaki model i za oba jezika. Prvi model na engleskom jeziku uključuje sve rečenice iz izvornog teksta u sažetak i prema Hovy, to se ne može smatrati sažetkom. Sažetak teksta na hrvatskom sadrži 6 rečenica, što je granično, obzirom da izvorni tekst sadrži ukupno 11 rečenica. S druge strane, izvorni tekst je kratak te se može zanemariti kriterij gornje granice od 50 % izvornog teksta u sažetku. Subjekt je vidljiv samo u engleskoj verziji, ali na kraju sažetka. Tekst je koherentan također samo u engleskoj verziji, dok je u hrvatskoj verziji već prva sažetka rečenica

zbunjujuća. U drugom modelu je i hrvatski sažetak bolji te je vidljiv subjekt (u sredini teksta). Broj rečenica hrvatskog sažetka je isti kao i u prvom modelu, ali rečenice nisu iste. U engleskoj verziji sažetak je samo jedna rečenica, ali vrlo precizno odgovara na pitanje tko je subjekt. Samim time engleska verzija je koherentna, dok u hrvatskoj zbunjuju česti ulomci teksta. Posljednji model ni u hrvatskoj ni u engleskoj verziji ne otkriva subjekt, a oba sažetka su vrlo kratka (hrvatski samo jedna rečenica, engleski tri). Ni u ovom modelu hrvatski tekst nije koherentan, počinje zarezom, dok je engleski čitljiv bez problema (iako ne otkriva o kome je riječ).

Kada bi se trebao odrediti «pobjednik» ovih usporedbi, prema rezultatima i u hrvatskoj i u engleskoj verziji najbolji se pokazao model 2, iako sva tri modela ostavljaju dosta prostora za poboljšanja.

Tablica 7 Tablična usporedba modela

	Vidljivost subjekta		**Duljina sažetka**		**Koherentnost teksta**	
	hr	***en***	***hr***	***en***	***hr***	***en***
Model 1	ne	da, na kraju	6 rečenica	7 rečenica	ne	da
Model 2	da, u sredini	da	6 rečenica	1 rečenica	ne	da
Model 3	ne	ne	1 rečenica	3 rečenice	ne	da

4.4. Usporedba s web aplikacijama za sažimanje teksta

Kako bi se utvrdilo je li alat dobar ili loš, potrebno ga je usporediti s nekim drugim alatom iste namjene. Stoga, isti ulazni tekst (prilog 4) korišten je u dvije različite web aplikacije za sažimanje teksta.

4.4.1. Autosummarizer.com

Ova aplikacija se nalazi na web adresi https://autosummarizer.com/. Autor(i) su ostavili adresu virtualnog ureda u Reykjaviku prilikom registriranja domene, dok na e-poštu i poruke preko društvenih mreža do završetka ovog rada nije pristigao nikakav odgovor. Time se, na žalost, gubi znanje o pozadini sustava i načinu na koji on funkcionira, tako da na usporedbu rada treba staviti zvijezdicu. Od dostupnih informacija, vidljivo je da aplikacija koristi neku od ekstraktivnih metoda za sažimanje teksta. Od postavki, može se mijenjati samo broj rečenica sažetka koje se želi dobiti, a minimalno pet. Dobiveni sažetak je:

> *Pripadao je studentici Muzičke akademije Ruži Pospiš, poslije udanoj Baldani. Ruža Pospiš za čitave je svoje duge četrdesetogodišnje karijere od 1961. do 2001. ostala vjerna Zagrebačkoj operi i usporedno stjecala svjetski ugled. Počelo je to već dvije godine nakon debuta, u napuljskome San Carlu u ulozi Venere u Offenbachovu Orfeju u podzemlju, nastavilo se 1964. godine na Holland festivalu s Marinom u Borisu Godunovu i Dubrovačkim ljetnim igrama s Oktavijom u Krunidbi Popeje pod ravnanjem Lovre pl. U 1975. godine, kada se proslavljala stota obljetnica praizvedbe Carmen, bila je zacijelo njezina najtraženija interpretkinja, od premijere u Covent Gardenu do Bečke državne opere i Madrida. Njezin jedinstveno lijep glas, velika muzikalnost i prekrasan legato uz atraktivnu scensku pojavu činili su je raskošnom Carmen, Dalilom i Amneris, golem glasovni potencijal izražajnom Azucenom, uznositost interpretacije primjerenim Orfejom, zrelost psihološkoga poniranja u lik izvrsnom Marfom u Hovanščini, proglašenom u njemačkom časopisu Opernwelt najboljom kreacijom u 2000., a uzvišena mirnoća fraze idealnom interpretkinjom velikih vokalno-instrumentalnih djela Bacha, Beethovena, Verdija, Händela.*

Ovaj sažetak je usporediv s prvim modelom koji koristi NLTK biblioteku. Slične su duljine te je uključeno nabrajanje umjetničkog stvaralaštva. Model sažimanja aplikacije nije poznat, ali temeljem sličnosti s prvim modelom iz ovog rada te dostupnost NLTK biblioteke, nije nemoguće pretpostaviti da se NLTK biblioteka koristi i u web aplikaciji u prilagođenom izdanju.

4.4.2. Quillbot.com

Aplikacija se može naći na web adresi https://quillbot.com/summarize i prema autorima je nastala kako bi pomogla studentima i profesionalcima u pisanju tematskih tekstova. Sažimanje je samo jedna od opcija koje se nude, uz parafraziranje, provjeru pravopisa, provjeru plagijata, generiranje citata te skupno pisanje. Sažimanje se može prilagoditi prema duljini, ali za povećanje objektivnosti ostavljena je pretpostavljena duljina prilikom učitavanja stranice. Dobiveni sažetak je:

> *Ruža Pospiš Baldani (Varaždinske Toplice, 25. srpnja 1942) je hrvatska operna pjevačica (mezzosopran). Od 2010. dopisna je članica Razreda za*

> *glazbenu umjetnost i muzikologiju. Pozornost privukao glas orguljske ljepote, baršunaste mekoće i bogatih preljeva.*

Rezultat je usporediv s drugom metodom iz ovog rada na što posebno ukazuje jedna zanimljivost. Iako je posljednja rečenica muškog roda a radi se o ženskoj osobi, ona ujedno ukazuje da algoritam ne tokenizira cijele rečenice, već riječi ili skupine riječi. Rečenica u originalu glasi:

> *Na prvoj izvedbi u Hrvatskoj opere Rat i mi“ Sergeja Prokofjeva u HNK u Zagrebu (20. studenog 1961.) posebnu je pozornost privukao glas orguljske ljepote, baršunaste mekoće i bogatih preljeva u maloj ulozi Muratova ađutanta. Sažetak iz ove aplikacije je počeo rečenicu tek od Posebnu je pozornost... jednako kao i druga metoda iz ovog rada.*

Ono što ova web aplikacija radi, a čime se u ovom radu ne bavi, jest post obrada teksta na način da rečenica počinje velikim tiskanim slovom. Ova značajka se može staviti u budućem razvoju kao poboljšanje korisničkog iskustva. Nigdje nije navedeno koju metodu ili metode aplikacija koristi za sažimanje, no rezultat je svakako usporediv sa sažimanjem temeljeno na grafovima iz ovog rada.

Jednostavnim pretraživanjem nađe se više alata za sažimanje[24] teksta. Usporedba sa svakim od alata nije cilj ovog rada te rezultati svake usporedbe nisu opisani, no svakako su usporedivi sa sve tri izvršene metode kao i s rezultatima opisanih web aplikacija.

[24] https://www.textcompactor.com/
https://resoomer.com/
https://tldrthis.com/

5. Prijedlog poboljšanja

U ovom poglavlju predstavljaju se moguća poboljšanja alata u budućnosti, počevši od korisničkog sučelja koje bi bilo prihvatljivije većoj skupini korisnika.

Tablica 8 pokazuje o kakvim se mogućim poboljšanjima radi. Navedeno je područje koje se treba poboljšati te opis na koji način je potrebno to izvesti.

Tablica 8 Moguća poboljšanja alata za sažimanje teksta

Poboljšanje	Opis
Korisničko sučelje	Prvo moguće poboljšanje, bilo prvog primjera s NLTK bibliotekom ili drugog temeljenog na grafovima je izrada grafičkog sučelja (engl. GUI) za odabir teksta. Prema trenutnim postavkama, korisnik treba otvoriti programski kod i mijenjati link na Wikipediju ili na željenu ulaznu datoteku. To je moguće ostvariti na više načina, jedan od kojih je PyQt[25] biblioteka. Time bi se postigla određena razina tzv. user-friendliness, tj. olakšavanje korisnicima rada s programima.
Post obrada teksta	Osim samog grafičkog sučelja, izlazni tekst bi trebalo obraditi prije ispisa tako da odgovara jezičnim pravilima i normama. Očito u sažecima rečenice počinju malim slovom te su redovi povremeno čudno odijeljeni. Relativno jednostavna dodatna obrada na kraju se može provesti kako bi se ispravili takvi nedostatci.
Zaustavne riječi	Sljedeće, i vjerojatno značajnije poboljšanje je izrada zaustavnih riječi (engl. *stopwords*) hrvatskog jezika za NLTK biblioteku. Zbog svoje važnosti u pročišćavanju teksta na riječi sa značenjem, pretpostavka je da bi se ovime povećala preciznost sažimanja tekstova na hrvatskom jeziku. Osim zaustavnih riječi, u izvedbi programa temeljenog na grafovima, dodatno poboljšanje bi bilo da program «pamti» naučene vektore, kako ne bi svaki puta morao prolaziti kroz GloVe i «učiti» ispočetka. Sama izvedba toga nije kompliciranа, dokle god se temelji na GloVe Wikipedija korpusu. Problem se javlja ukoliko se želi promijeniti korpus, jer je tada potrebno pronaći način kako integrirati (i da li uopće) postojeće podatke s novima.

[25] PyQT je biblioteka za Python pomoću koje se mogu programirati grafička sučelja (GUI). Za više vidi: https://en.wikipedia.org/wiki/PyQt

Poboljšanje	Opis
Korpus	Kako spaCy biblioteka koristi korpus temeljem kojeg neuronska mreža uči i pravi statistički model, očekivano poboljšanje bi bilo koristiti veći ili tematski određeni korpus. Kako je spaCy biblioteka otvorenog koda, na web stranicama se mogu naći sve upute kako napraviti novi ili poboljšati postojeće korpuse. To je dosta iscrpljujuć i dugotrajan proces, ali svakako bi se dugoročno isplatio.
Automatsko sažimanje više tekstova	U ovom slučaju, poboljšanje je dodavanje mogućnosti odabira od strane korisnka, hoće li sažeti jedan tekst (dokument) ili više njih. Problem nije u programskoj izvedbi, već konceptualnoj. Sažimanje više tematski povezanih tekstova nije lak zadatak iz razloga što uvijek postoji mogućnost redundancije informacija. Sustav treba prepoznati i locirati preklapanje tema te odlučiti što uključiti u sažetak, a da pritom taj sažetak bude konzistentan u svom sadržaju i vremenu (jedan dokument može biti napisan u prošlom vremenu, drugi u sadašnjem, treći u budućem ili se može preklapati više vremena u istom dokumentu). Dosadašnja istraživanja u ovom području imala su najviše uspjeha u domeni sažimanja novinskih članaka (Marcu i Gerber, 2001). Jedna od predloženih tehnika je tzv. SUMMONS, koju su predložili Radev i McKeown (1998). SUMMONS koristi ekstraktivne metode za generiranje sažetaka i pretpostavlja predobradu teksta, kako bi svi ulazni podaci bili standardizirani. Drugi takav sustav je Newsblaster [26]sa Sveučilišta u Kolumbiji (McKeown *et al.,* 2004) u radu *Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster*.

[26] Na web stranicama se može vidjeti rad sustava koji sažima vijesti 24 sata dnevno: http://newsblaster.cs.columbia.edu

6. Zaključak

Postoje različite metode sažimanja i principi generiranja sažetog teksta. U ovom se radu daje pregled područja sažimanja, definicije, postignuća i otvorena pitanja. Većina autora daje uglavnom slične definicije, a u literaturi nalazimo i uvjet, kojeg smo uzeli u obzir u provedenom istraživanju, da sažetak ne smije biti veći od polovice izvornog teksta. Uz pregled sažetka s aspekta njegove svrhe, dana su dva pogleda evaluacije sažetka te je predstavljen problem subjektivnosti na koji se nadovezuje i problem određivanja relevantnosti neke rečenice za ulazak u sažetak.

Predstavljaju se i detaljno prikazuju koraci sažimanja uzimajući u obzir ekstraktivni i apstraktivni pristup stvaranju sažetka. Ekstraktivne metode odabiru važne rečenice iz ulaznog teksta i slažu ih u sažetak pri čemu se važnost neke rečenice određuje temeljem njezinih statističkih i lingvističkih značajki. Bez upotrebe obrade prirodnog jezika, sažetak može izgubiti koheziju i semantičko značenje. Ukoliko izvorni tekst sadrži više tema, iz sažetka se može izgubiti ono o čemu tekst govori. Apstraktivne metode, za razliku od ekstraktivnih, ne uzimaju postojeće rečenice, već temeljem «iskustva», tj. korpusa na kojem uče, stvaraju nove rečenice i konstrukcije koje se ne nalaze nužno u izvornom tekstu. Zbog jednostavnijih izvedbi, do sada je istraženo mnogo više ekstraktivnih nego apstraktivnih metoda sažimanja teksta.

U drugom dijelu rada predstavljaju se tri modela sažimanja istog ulaznog teksta na hrvatskom i engleskom jeziku te se predstavljaju rezultati. Modeli su odabrani zbog rasprostranjenosti alata i biblioteka pomoću kojih se ostvaruju. Hrvatski jezik je prema morfološkoj podjeli flektivni jezik – sklanja se po padežima, što uvelike otežava održavanje koherencije sažetka, pogotovo kad se k tomu doda mogućnost muškog, ženskog i srednjeg roda. Ekstraktivne metode u tim modelima ne znaju prepoznati ni rod ni broj ni padež te je potrebno uložiti više vremena kako bi se u algoritme dodale razne klauzule i provjere kako bi sažetak poštivao pravopis i sintaksu hrvatskog jezika. Engleski jezik nema taj problem ili barem ne u tolikoj mjeri, jer je blago sintetički jezik te se riječi mijenjaju vrlo rijetko i iste su neovisno o rodu (npr. prilikom stvaranja množine se dodaje -s, neovisno o rodu imenice). Dapače, sam rod imenice u engleskom jeziku je nekad teško utvrditi, no za sažimanje je nevažan. Samim time engleski je jednostavniji za generiranje sažetka jer se ne treba paziti na toliko elemenata kao u hrvatskom.

Prilikom pisanja ovog rada i istraživanja područja, većina znanstvenih članaka o sažimanju je bila upravo o sažimanju engleskog, iako u posljednjih par godina autori počinju istraživati i druge jezike poput kineskog, njemačkog pa čak i filipinskog.

Rezultati izvedbi u ovom radu imaju premali uzorak da bi se sa sigurnošću moglo utvrditi koliko su dobri ili loši. U usporedbi s dvije web aplikacije, vidi se da su itekako relevantni i s nekim poboljšanjima, naročito u post procesiranju, mogli bi se koristiti i za mnogo veće tekstove i u izazovnijim primjenama.

Na samom kraju rada, predstavljaju se moguća poboljšanja alata dodavanjem grafičkog sučelja pomoću PyQt biblioteke, dodavanjem zaustavnih riječi za hrvatski jezik u NLTK biblioteku, korištenje tematskog korpusa ili povećanje korpusa na kojem algoritam uči i mogućnost sažimanja više tekstova odjednom.

Tema automatskog sažimanja teksta je opširna te postoji mnoštvo literature čiji autori predstavljaju različite metode generiranja sažetka. Više su istražene ekstraktivne metode jer ih je jednostavnije izvesti, no s pojavom umjetne inteligencije i neuralnih mreža, istražuju se i apstraktivne metode te se pojavljuju i praktične izvedbe koje će s vremenom davati sve bolje rezultate. Vjerojatno najveći izazov u sažimanju je taj što ne postoji jedinstven sažetak koji bi se mogao opisati kao savršen, za bilo koji tekst. Velika je vjerojatnost da će dvije osobe isti tekst sažeti različito. Sažeci će možda biti vrlo slični, ali neće biti isti. To ne znači da je jedan ili drugi pogrešan, već da se u tom smislu ne može postići jedinstveni matematički model sažimanja. Kao zaključak može se reći da alati za sažimanje teksta trebaju raditi sažetke u razumnom vremenu, s najmanje mogućeg ponavljanja informacija, moraju biti koherentni i držati semantiku jezika na kojem je izvorni tekst koji se sažima. Iako u literaturi nalazimo dosta dobrih rješenja za jezike poput engleskog, hrvatski jezik sa svojim specifičnostima i dalje ostaje nedovoljno istraženo područje s mnogo potencijalnih rješenja.

7. Literatura

1. Aggrawal, G.; Sumbaly, R.; Sinha, S. (2009) Update Summarization
2. Edmundson, H.P. (1969) New methods in automatic extracting, Advances in Automatic Text Summarization, str. 23-42
3. Ganesan, K.; Zhai, C. X.; Han, J. (2010) Opinosis: A Graph-Based Approah to Abstractive Summarization of Highly Redundant Opinions, Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), str. 340-348
4. García-Hernández, R.A.; Ledeneva, Y. (2009) Word Sequence Models for Single Text Summarization, Second International Conferences on Advances in Computer-Human Interactions
5. Gholamrezazadeh, S.; Salehi, M.A.; Gholamzadeh, B. (2009) A comprehensive survey on Text Summarization Systems, 2nd International Conference on Computer Science and its Applications
6. Gong, Y.; Liu, X. (2001) Generic text summarization using relevance measure and latent semantic analysis, Proceedings of the 24th annual international ACM SIGIR conference on Reasearch and development in information retreival, str. 19-25
7. Gupta, V.; Lehal, G. (2010) A Survey of Text Summarization Extractive Techniques, Journal of emerging technologies in web intelligence, vol.2, no.3, str. 258-268
8. Hassel, M. (2004) Evaluation of Automatic Text Summarization, Universitetsservice US AB
9. Hovy, E.; Chin, Y.L. (1999) Automated text summarization in SUMMARIST, MIT Press, str. 81-94
10. Hovy, E. (2005) Text Summarization, The Oxford Handbook of Computational Linguistics, str. 583-598
11. Jaya Kumar, Y.; Salim, N. (2012) Automatic Multi Document Summarization Approaches, Journal of Computer Science, str. 133-140
12. Jaya Kumar, Y.; Sing Goh, O.; Basiron, H.; Hea Choon, N.; Suppiah, P. C. (2016) A Review on Automatic Text Summarization Approaches, Journal of Computer Science, str. 178-190

13. Kaikhah, K. (2004) Text Summarization Using Neural Networks, 2nd International IEEE Conference on „Intelligent Systems“
14. Kan, M.-Y. (2003) Automatic text summarization as applied to information retrieval: Using indicative and informative summaries
15. Kan, M.-Y.; R. McKeown, K. (2002) Corpus-trained text generation for summarization, Department of Computer Science Columbia University
16. Kisiček, I. (2018) Postupci mjerenja semantičke sličnosti tekstova, Sveučilište u Rijeci, odjel za informatiku
17. Kubo, T.; Mamdapure, A. (2017) Awesome Text Summarization. Preuzeto sa: https://github.com/icoxfog417/awesome-text-summarization
18. Kupiec, J.; Pedersen, J.; Chen, F. (1995) A Trainable Document Summerizer, Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, str. 68-73
19. Kyoomarsi, F.; Khosravi, H.; Eslami, E.; Khosravyan Dehkordy, P. (2008) Optimizing Text Summarization Based on Fuzzy Logic, Proceedings of 7th IEEE/ACIS International Conference on Computer and Information Science, str. 347-352
20. Lauc, T.; Mikelić, N.; Boras, D. (2005) Croatian Text Summarizer, Proceedings of the ITI 2005, 27th International Conference of Information Technology Interfaces
21. Lin, C-Y.; Hovy, E. (2002) From Single to Multi-document Summarization: A Prototype System and its Evaluation, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, str. 457-464
22. Lin, C-Y. (2004) ROUGE: A Package for Automatic Evaluation of Summaries, Association for Computational Linguistics, str. 74-81
23. Lloret, E.; Ferrández, O.; Muñoz, R.; Palomar, M. (2008) A Text Summarization Approach Under the Influence of Textual Entailment, Proceedings of the 5th International Workshop on Natural Language Processing and Cognitive Science, str. 22-31
24. Lloret, E.; Palomar, M. (2010) Challenging Issues of Automatic Summarization: Relevance Detection and Quality-based Evaluation, Informatica 34, str. 29-35

25. Luhn, H.P. (1958) The automatic creation of literature abstracts, Advances in Automatic Text Summarization, str. 15-22

26. Mani, I. (1999) Advances in Automatic Text Summarization, MIT Press Cambridge

27. Mani, I. (2001a) Automatic Summarization, John Benjamins Publishing Company, Amsterdam/Philadelphia

28. Mani, I. (2001b) Summarization evaluation: An overview, Proceedings of the North American chapter of the Association for Computational Linguistics

29. Maybury, M.T. (1998) Tools for the knowledge artist: An information superiority visionary demonstration, Computer Software and Applications Conference

30. McKeown, K.; Barzilay, R.; Evans, D.; Hatzivassiloglou, V.; Klavans, J.; Nenkova, A.; Sable, C.; Schiffman, B.; Sigelman, S. (2002) Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster, Proceedings of HLT 2002, Second International Conference on Human Language Technology, str. 280-285

31. Mikelić Preradović, N.; Ljubešić, N.; Boras, D. (2010) Croatian web text summarizer, Proceedings of the ITI 2010, 32nd International Conference of Information Technology Interfaces

32. Mikelić Preradović, N.; Boras, D.; Vlainić, M. (2014) Importance of Surface Methods in Human and Automatic Text Summarization, Journal of Computers 8, str. 9-16

33. Prabhakaran, S. (2018) Cosine Similarity – Understanding the math and how it works. Preuzeto sa: https://www.machinelearningplus.com/nlp/cosine-similarity/

34. Radev, D. R.; McKeown, K (1998) Generating Natural Language Summaries from Multiple On-Line Sources, Association for Computational Linguistics, Vol.24, No.3, str. 469-500

35. Radev, D. R.; Blair-Goldensohn, S.; Zhang, Z. (2001) Experiments in single and multi-document summarization using mead, First Document Understanding Conference, str. 1-7

36. Radev, D. R.; Hovy, E.; McKeown, K (2002) Introduction to the Special Issue on Summarization, Computational Linguistics, Volume 28-4, str. 399-408

37. Rush, A.M.; Chopra, S.; Weston, J. (2015) A Neural Attention Model for Abstractive Sentence Summarization, arXiv:1509.00685v2 [cs.CL]

38. Salton, G.; Buckley, C. (1987) Term Weighting Approaches in Automatic Text Retreival, Cornell University

39. Torres-Moreno, J.M. (2014) Automatic Text Summarization, John Wiley & Sons, Inc.

40. Wang, M.; Wang, X.; Xu, C. (2005) An approach to concept-obtained text summarization, IEEE International Symposium on Communications and Information Technology, str. 1337-1340

41. Zhou, L.; Hovy, E. (2003) Headline Summarization at ISI, USC Information Sciences Institute

Prilozi

Prilog 1 – Program za sažimanje pomoću NLTK biblioteke

```
import bs4 as bs
import urllib.request
import re
import nltk
from stop_words import get_stop_words

scraped_data =
urllib.request.urlopen('https://hr.wikipedia.org/wiki/Ru%C5%BEa_Posp
i%C5%A1_Baldani')
article = scraped_data.read()

parsed_article = bs.BeautifulSoup(article,'lxml')

paragraphs = parsed_article.find_all('p')

article_text = ""

for p in paragraphs:
    article_text += p.text

article_text = re.sub(r'\[[0-9]*\]', ' ', article_text)
article_text = re.sub(r'\s+', ' ', article_text)

formatted_article_text = re.sub('[^a-zA-Z]', ' ', article_text )
formatted_article_text = re.sub(r'\s+', ' ', formatted_article_text)

sentence_list = nltk.sent_tokenize(article_text)

stopwords = get_stop_words('english') #czech, polish, slovak

word_frequencies = {}
for word in nltk.word_tokenize(formatted_article_text):
    if word not in stopwords:
        if word not in word_frequencies.keys():
            word_frequencies[word] = 1
        else:
            word_frequencies[word] += 1

maximum_frequncy = max(word_frequencies.values())

for word in word_frequencies.keys():
    word_frequencies[word] =
(word_frequencies[word]/maximum_frequncy)

sentence_scores = {}
for sent in sentence_list:
    for word in nltk.word_tokenize(sent.lower()):
        if word in word_frequencies.keys():
```

```
            if len(sent.split(' ')) < 45:
                if sent not in sentence_scores.keys():
                    sentence_scores[sent] = word_frequencies[word]
                else:
                    sentence_scores[sent] += word_frequencies[word]

import heapq
summary_sentences = heapq.nlargest(8, sentence_scores,
key=sentence_scores.get)

summary = ' '.join(summary_sentences)
print(summary)
```

Prilog 2 – Program za sažimanje pomoću metode temeljene na grafovima

```
import numpy as np
import pandas as pd
import nltk
from nltk.tokenize import sent_tokenize
from nltk.corpus import stopwords
#nltk.download('punkt')
import re
from sklearn.metrics.pairwise import cosine_similarity
import networkx as nx
#from stop_words import get_stop_words

df = pd.read_csv("ruza_pospis_hr.txt")

sentences = []
for s in df:
    sentences.append(sent_tokenize(s))

sentences = [y for x in sentences for y in x]

clean_sentences = pd.Series(sentences).str.replace("[^a-zA-Z]", " ")

clean_sentences = [s.lower() for s in clean_sentences]

#stop_words = get_stop_words('czech') #czech, polish, slovak
stop_words = stopwords.words('english')

def remove_stopwords(sen):
    sen_new = " ".join([i for i in sen if i not in stop_words])
    return sen_new

clean_sentences = [remove_stopwords(r.split()) for r in
clean_sentences]

word_embeddings = {}
f = open('glove.6B/glove.6B.100d.txt', encoding='utf-8')
for line in f:
    values = line.split()
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    word_embeddings[word] = coefs
f.close()

sentence_vectors = []
for i in clean_sentences:
    if len(i) != 0:
        v = sum([word_embeddings.get(w, np.zeros((100,))) for w in
i.split()])/(len(i.split())+0.001)
    else:
        v = np.zeros((100,))
    sentence_vectors.append(v)

sim_mat = np.zeros([len(sentences), len(sentences)])
```

```
for i in range(len(sentences)):
    for j in range(len(sentences)):
        if i != j:
            sim_mat[i][j] =
cosine_similarity(sentence_vectors[i].reshape(1,100),
sentence_vectors[j].reshape(1,100))[0,0]

nx_graph = nx.from_numpy_array(sim_mat)
scores = nx.pagerank(nx_graph)

ranked_sentences = sorted(((scores[i],s) for i,s in
enumerate(sentences)), reverse=True)

sn = 8

for i in range(sn):
    print(ranked_sentences[i][1])
```

Prilog 3 – Program za sažimanje pomoću spaCy neuronske mreže

```
import spacy
nlp = spacy.load("hr_core_news_lg")
text = pd.read("ruza_pospis_hr.txt")

doc = nlp(text)

word_dict = {}

for word in doc:
    word = word.text.lower()
    if word in word_dict:
        word_dict[word] += 1
    else:
        word_dict[word] = 1

sents = []
sent_score = 0
for index, sent in enumerate(doc.sents):
    for word in sent:
        word = word.text.lower()
        sent_score += word_dict[word]
    sents.append((sent.text.replace("\n"," "), sent_score/len(sent),
index))

sents = sorted(sents, key=lambda x: -x[1])
sents = sorted(sents[:3], key=lambda x: x[2])

summary_text = ""
for sent in sents:
    summary_text += sent[0] + " "

print(summary_text)
```

Prilog 4 – Ulazni tekst na hrvatskom jeziku

Ruža Pospiš Baldani (Varaždinske Toplice, 25. srpnja 1942.), hrvatska operna pjevačica (mezzosopran). Od 2010. dopisna je članica Razreda za glazbenu umjetnost i muzikologiju Hrvatske akademije znanosti i umjetnosti. Na prvoj izvedbi u Hrvatskoj opere Rat i mir Sergeja Prokofjeva u HNK u Zagrebu (20. studenog 1961.) posebnu je pozornost privukao glas orguljske ljepote, baršunaste mekoće i bogatih preljeva u maloj ulozi Muratova ađutanta. Pripadao je studentici Muzičke akademije Ruži Pospiš, poslije udanoj Baldani.

Ruža Pospiš za čitave je svoje duge četrdesetogodišnje karijere od 1961. do 2001. ostala vjerna Zagrebačkoj operi i usporedno stjecala svjetski ugled. Počelo je to već dvije godine nakon debuta, u napuljskome San Carlu u ulozi Venere u Offenbachovu Orfeju u podzemlju, nastavilo se 1964. godine na Holland festivalu s Marinom u Borisu Godunovu i Dubrovačkim ljetnim igrama s Oktavijom u Krunidbi Popeje pod ravnanjem Lovre pl. Matačića, 1965. nastupom na festivalu u Edinburghu u Haydnovoj operi Ribarice i 16. veljače 1966. godine s prvim nastupom u Metropolitanu kao Maddalena iznad prosjeka u Rigolettu.

Ruža Pospiš Baldani je u Metropolitan operi nastupila 57 puta s mnogim opernim velikanima među kojima su: Birgit Nilsson Alfredo Kraus Regine Crespin Franco Corelli Robert Merrill Carlo Bergonzi Sherrill Milnes Nicolai Gedda Renata Tebaldi Leontyne Price Richard Tucker Montserrat Caballe Placido Domingo i Jon Vickers.

Angažirana u Metu, bila je 1970. njegova Carmen u režiji Jean-Louisa Barraulta koju je upoznala i široka publika u tri radijska prijenosa from coast to coast (od obale do obale). U 1975. godine, kada se proslavljala stota obljetnica praizvedbe Carmen, bila je zacijelo njezina najtraženija interpretkinja, od premijere u Covent Gardenu do Bečke državne opere i Madrida.

Suradnja s velikim dirigentima Karajanom Karlom Richterom Claudiom Abbadom vodila ju je na Salzburške svečane igre i Uskrsne svečane igre u istome gradu u Scalu Bavarsku državnu operu u Muenchenu Teatro del Liceo u Barceloni u najveće koncertne dvorane Berlinske i Bečke filharmonije Carnegie Hall, u Vatikan pred Papu Pavla VI.

Njezin jedinstveno lijep glas velika muzikalnost i prekrasan legato uz atraktivnu scensku pojavu činili su je raskošnom Carmen Dalilom i Amneris golem glasovni potencijal izražajnom Azucenom uznositost interpretacije primjerenim Orfejom zrelost psihološkoga poniranja u lik izvrsnom Marfom u Hovanščini proglašenom u njemačkom časopisu Opernwelt najboljom kreacijom u 2000. a uzvišena mirnoća fraze idealnom interpretkinjom velikih vokalno-instrumentalnih djela Bacha Beethovena Verdija Haendela.

Prilog 5 – Ulazni tekst na engleskom jeziku

Ruža Pospiš-Baldani (born 25 July 1942) is a Croatian operatic mezzo-soprano. Baldani was born in Varaždinske Toplice and made her professional opera debut in 1961 at the Croatian National Theatre in Zagreb as Konchakovna in Alexander Borodin's Prince Igor. She remained active at that theatre and at the National Theatre in Belgrade throughout the 1960s.

In 1965 she made her debut at the Metropolitan Opera in New York City as Maddalena in Giuseppe Verdi's Rigoletto.

From 1970-1978 she was committed to the Bavarian State Opera. Between 1973 and 1987 she was a frequent guest artist at the Vienna State Opera; drawing particular acclaim there as Brangäne in Richard Wagner's Tristan und Isolde.

In 1976 she made her debut at the Paris Opera as Amneris in Verdi's Aida and made her first appearance at the Opéra de Monte-Carlo in the title role of Georges Bizet's Carmen.

She has since appeared as a guest artist at the Cologne Opera. the Edinburgh Festival the Greek National Opera the Hamburg State Opera the Houston Grand Opera the Hungarian State Opera House La Scala the Liceu the Lyric Opera of Chicago the National Opera of Sofia the Salzburg Festival the San Francisco Opera the Savonlinna Opera Festival the Teatro dell'Opera di Roma the Teatro di San Carlo and the Teatro Municipal in Rio de Janeiro among others.

Izrada alata za automatsko sažimanje teksta

Sažetak

Cilj ovog rada je dati pregled teorije automatskog sažimanja teksta, te primjerom prikazati praktičnu izvedba programskog rješenja sažimanja teksta. Uspoređuju se rezultati tih rješenja temeljem tekstova na hrvatskom i engleskom jeziku te se ukazuje na njihove prednosti i mane.

Ključne riječi: *obrada prirodnog jezika, automatsko sažimanje teksta, python, ekstraktivne metode sažimanja, apstraktivne metode sažimanja, sažimanje temeljeno na grafovima, sažimanje temeljeno na neuronskoj mreži*

Creating a tool for automatic text summarization

Summary

The goal of this theses is going trough the theory and practical implementations for automatic text summarization. Results of those implementations are shown for english and croatian text with a quick look at their faults and advantages.

Key words: *natural language processing, automatic text summarization, python, extractive summarization, abstractive summarization, graph-based summarization, neural network summarization*