# Holističko predstavljanje znanja i strukturno učenje

Mladen Stanojević, Mladen Jovanović[1]

***Sadržaj* — Ovaj rad predstavlja inovativni pristup strukturisanoj reprezentaciji podataka u kontekstu obrade prirodnog jezika i sekvencijalnih fenomena. U početnim delovima rada razmatra se hijerarhijska organizacija podataka i mogućnosti koje pruža u analizi i modeliranju semantičkih i sintaktičkih odnosa. Središnji deo rada posvećen je detaljnoj analizi predloženog modela, sa fokusom na njegovu primenu, arhitekturu i algoritme za prepoznavanje i kreiranje hijerarhijskih struktura. Na kraju, kroz poređenje sa postojećim metodama, prikazuju se prednosti i ograničenja modela, dok se u zaključnim delovima iznose pravci za buduća istraživanja, uključujući mogućnosti primene u različitim domenima i optimizacije algoritma za paralelnu obradu i skalabilnost.**

***Ključne reči* — hijerarhijska reprezentacija podataka, jednodimenzionalni prostori, obrada prirodnog jezika, strukturno učenje**

## I. UVOD

Hijerarhijska reprezentacija podataka predstavlja inovativan pristup obradi prirodnog jezika, omogućavajući organizaciju teksta na višestrukim nivoima detalja, od osnovnih jedinica poput karaktera do složenih struktura poput dokumenata. Ovaj pristup otvara nove mogućnosti za analizu, prepoznavanje obrazaca i razumevanje složenih semantičkih odnosa unutar tekstualnih korpusa. Međutim, implementacija ovakvih hijerarhijskih modela suočava se sa brojnim izazovima, uključujući skalabilnost, efikasnost obrade velikih datasetova i precizno očuvanje odnosa među elementima u strukturi.

Obrada prirodnog jezika često se oslanja na distribuisane modele poput *Word2Vec* [1], *BERT* [2] i *Latent Semantic Analysis* [3], [4], koji uspešno rešavaju zadatke semantičkog razumevanja na nivou reči ili rečenica. Ipak, ti modeli nisu prilagođeni za sveobuhvatno prepoznavanje višeslojnih hijerarhijskih obrazaca, što ograničava njihovu primenu u složenijim scenarijima analize i rekonstrukcije teksta. Sa druge strane, potreba za hijerarhijskim modelima postaje sve izraženija u oblastima poput

[1] M. Stanojević, Računarski fakultet, Beograd, Srbija (e-pošta: mstanojević@raf.rs)
M. Jovanović, Računarski fakultet, Beograd, Srbija (e-pošta: mjovanović423m@raf.rs)

bioinformatike, analize vremenskih serija i obrade *IoT* podataka, gde je strukturalna organizacija ključna za interpretaciju i predikciju.

Ovaj rad istražuje mogućnosti unapređenja reprezentacije teksta kroz hijerarhijsku organizaciju podataka, pri čemu se svaki nivo strukture zasniva na jedinstvenim objektima i njihovim međusobnim odnosima. Model omogućava dinamičko kreiranje, prepoznavanje i reorganizaciju objekata, čime se postiže fleksibilnost u prilagođavanju različitim zadacima i domenima. Pored analize performansi predloženog modela, rad se osvrće na ključne izazove poput konkurentne obrade, skalabilnosti i mogućnosti primene u novim kontekstima.

Uvođenje hijerarhijske reprezentacije donosi potencijal za značajne doprinose u analizi tekstualnih podataka, pružajući preciznije rezultate i unapređujući skalabilnost i efikasnost obrade. Cilj ovog rada je da se postave temelji za dalji razvoj ovog pristupa, istovremeno ukazujući na pravce budućih istraživanja koji mogu proširiti mogućnosti primene u širem spektru oblasti.

## II. TEORIJSKE OSNOVE

Razvoj inovativnih modela za hijerarhijsku reprezentaciju podataka zahteva duboko razumevanje teorijskih koncepata koji leže u osnovi ovog pristupa. Hijerarhijska struktura podataka predstavlja način organizacije informacija koji omogućava efikasnu analizu i obradu složenih sistema, čineći je primenjivom u različitim domenima, od obrade prirodnog jezika do analize vremenskih serija (engl. *timeseries*) i bioinformatike. U ovom poglavlju istražuju se ključni teorijski temelji koji podržavaju razvoj predloženog modela.

Prvo, definišu se osnovni pojmovi i koncepti koji su neophodni za razumevanje hijerarhijskog pristupa organizaciji podataka. Naglašava se važnost strukturiranja informacija na način koji omogućava reprezentaciju elemenata na različitim nivoima detalja, od osnovnih jedinica poput karaktera i reči, do složenih jedinica kao što su paragrafi i dokumenti. Ovaj koncept omogućava precizniju analizu i bolje razumevanje odnosa među podacima.

Pored konceptualnih osnova, pažnja se posvećuje modelovanju hijerarhijske strukture. Linearna algebra, teorija grafova i algoritmi za analizu sekvenci igraju ključnu ulogu u formalizaciji i realizaciji ovakvih modela [5]. Posebno se razmatraju matematički pristupi za redukciju dimenzionalnosti i kreiranje reprezentacija koje zadržavaju semantičke i sintaktičke informacije.

Na kraju, poglavlje uključuje pregled algoritama koji se koriste za prepoznavanje obrazaca, kreiranje hijerarhijskih struktura i njihovu optimizaciju. Ovi algoritmi predstavljaju osnovu za implementaciju modela u praktičnim okruženjima i njihovu primenu u rešavanju stvarnih problema.

### *A. Koncept hijerarhijske reprezentacije podataka*

Hijerarhijska reprezentacija podataka omogućava organizaciju elemenata u više nivoa, počevši od osnovnih jedinica kao što su karakteri, do složenijih struktura poput reči, fraza, rečenica, paragrafa i dokumenata. Svaki nivo strukture definiše se kroz jedinstvene objekte [6], gde se međusobni odnosi i sekvencijalna pravila koriste za očuvanje informacija o njihovom redosledu i povezanosti. Ovaj koncept se oslanja na ideju da višeslojna organizacija podataka omogućava efikasniju analizu i rekonstrukciju teksta, posebno u oblastima koje zahtevaju očuvanje semantičkih i sintaksičkih veza. Ključna prednost ovakve reprezentacije leži u njenoj fleksibilnosti, koja omogućava prilagođavanje različitim zadacima analize podataka, od prepoznavanja obrazaca u tekstu do modeliranja složenih struktura.

### *B. Osnovni pojmovi i definicije*

Osnovni pojmovi uključuju elementarne objekte, kao što su slova, cifre ili drugi osnovni simboli, koji čine najniži nivo hijerarhije. Ti objekti se kombinuju u složenije strukture, poput reči, koje dalje grade fraze, rečenice i veće tekstualne jedinice. Ključni atributi objekata uključuju:

- Tip objekta – definicija vrste objekta (npr. slovo, reč, fraza).
- Sekvenca – redosled elemenata unutar objekta.
- Kontekstualni odnosi – veza između objekata na istom ili različitim nivoima hijerarhije. Definišu se i koncepti poput "reprezentacije ponovljenih sekvenci" [7], gde se identifikuju i jedinstveno predstavljaju sekvence koje se pojavljuju u različitim kontekstima, što značajno smanjuje redundantnost i povećava efikasnost analize.

## III. Struktura podataka za objekte

Osnovna struktura podataka za reprezentaciju fenomena u jednodimenzionim prostorima definiše se kao objekat koji integriše osnovne atribute neophodne za opis i povezivanje elemenata u složenim hijerarhijama. Svaki objekat sadrži precizno definisane atribute, uključujući tip (engl. *type*), sekvencu (engl. *sequence*), sledeći objekat (engl. *nextObject*) i opis (engl. *desc*), omogućavajući jedinstvenu reprezentaciju i efikasno upravljanje podacima na svim nivoima strukture.

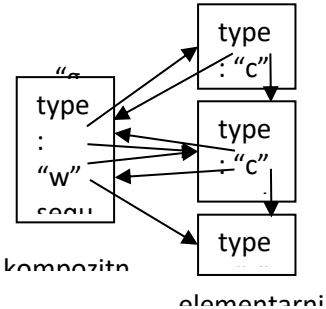
Atribut *type* određuje prirodu objekta, poput slova, brojeva, fraza, rečenica ili dokumenata, čime se omogućava precizna kategorizacija unutar hijerarhijske strukture. Na primer, slova su osnovni objekti, dok su fraze složeniji objekti koji se sastoje od reči ili drugih fraza.

Atribut *sequence* omogućava povezivanje elemenata unutar složenih objekata, osiguravajući njihovu pravilnu organizaciju u sekvencama. Na

primer, reč "*good*" se predstavlja nizom referenci na osnovne objekte koji čine ovu reč – "*g*", "*o*", "*o*" i "*d*" (Sl. 1). Ovaj atribut igra ključnu ulogu u očuvanju semantičkog i sintaktičkog redosleda elemenata unutar složenih objekata.

Atribut *nextObject* koristi heš tabelu kako bi omogućio efikasno povezivanje trenutnog objekta sa sledećim u hijerarhiji, kao i sa njegovim obuhvatajućim objektom. Na primer, slovo "*o*" u reči "*good*" može biti povezano sa sledećim slovom "*d*" ili sa složenim objektom koji predstavlja frazu u kojoj se ova reč pojavljuje. Ovaj atribut olakšava navigaciju kroz hijerarhijsku strukturu i omogućava dinamičko prilagođavanje odnosa između elemenata.

Atribut *desc* prisutan je samo kod elementarnih objekata i služi za opisivanje osnovnih karakteristika objekta, poput UTF8 karaktera za tekst, nukleobaza za DNK sekvence ili fonema za govor.

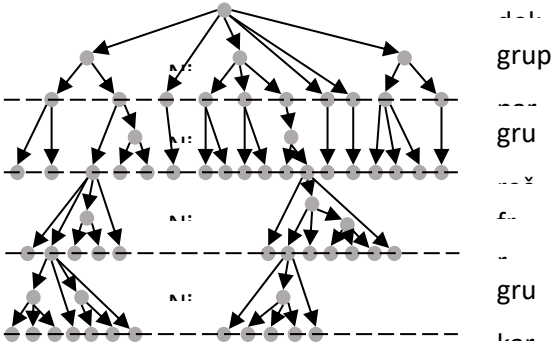



Sl. 1. Atributi objekta.

Dodatno, struktura omogućava jedinstvenu reprezentaciju ponovljenih sekvenci, čime se eliminiše redundancija i optimizuje skladištenje podataka. Na primer, u slučaju reči "*cool*" i "*good*", ponovljeni niz "*oo"* predstavlja se kao jedinstveni objekat koji se koristi u obe reči. Time se smanjuje kompleksnost strukture i obezbeđuje lakša analiza semantičkih i sintaktičkih veza između elemenata.

## IV. KREIRANJE STRUKTURE IZ RAVNOG PRIKAZA

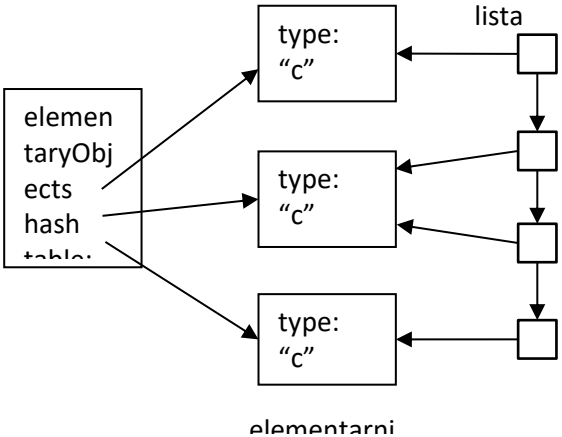
Strukturirana reprezentacija se kreira polazeći od niza karaktera, što predstavlja ravan prikaz. Na primer, dokumenti na prirodnim jezicima predstavljeni su kao fajlovi, a naš zadatak je da kreiramo strukture gde će se na vrhu nalaziti objekti koji predstavljaju te dokumente. Ispod njih će biti sloj koji se sastoji od grupa paragrafa i samih paragrafa. Paragrafi se dalje sastoje od grupa rečenica i rečenica (predstavljajući još jedan sloj), dok će se na dnu strukture nalaziti grupe karaktera i karakteri (Sl. 2).




Sl. 2. Nivoi i strukturna reprezentacija prirodnog jezika dokumenata.

Proces kreiranja objekata unutar strukture može se posmatrati kao proces učenja, gde se kreiraju novi objekti i dodaju odgovarajući prostorni odnosi u jednoj dimenziji. Ovaj proces učenja počinje učenjem elementarnih objekata [8]. U slučaju tekstova na prirodnim jezicima, ovo predstavlja učenje abecede.

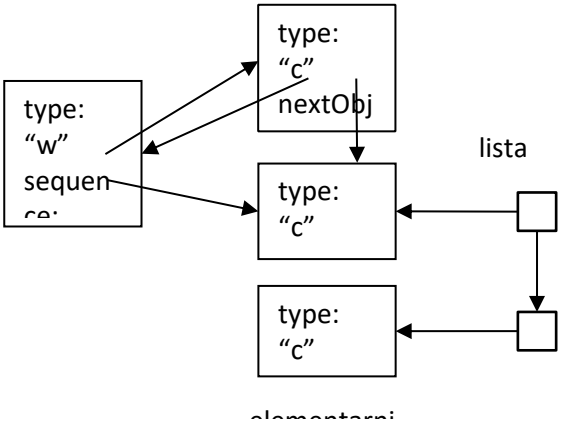
Pretpostavimo da želimo da naučimo reč "*good*" i da prethodno nisu naučene nijedne reči (struktura je prazna). Kao ulaz imamo niz karaktera. Prvo slovo u ovom nizu je slovo "*g*". Svi elementarni objekti (karakteri) čuvaju se u heš tabeli *elementaryObjects*. Ključ je slovo iz niza, a vrednost je referenca na objekat koji predstavlja to slovo. Ako slovo još nije naučeno, heš tabela će vratiti null. Pošto slovo "*g*" nije još naučeno, dobićemo *null*, nakon čega ćemo kreirati novi objekat koji predstavlja slovo "*g*" i napraviti odgovarajući unos u heš tabelu, tako da sledeći put kada naiđemo na "*g*", možemo ponovo iskoristiti isti objekat. Objekat koji predstavlja "*g*" dodaje se u listu *characters*, koja će sadržati sva slova sve dok se ne naiđe na neki karakter koji nije alfanumerički ili numerički, što označava kraj reči. Isti proces se ponavlja za slovo "*o*". Napomena: kada se obrađuje drugo slovo "*o*", ponovo se koristi objekat koji predstavlja prvi slučaj, i ne kreira se novi objekat. Kada se obradi poslednji karakter – slovo "*d*", lista characters će sadržati reference na četiri objekta koji predstavljaju do tada pročitane karaktere (Sl. 3).




Sl. 3. Obrađivanje karaktera unutar reči.

Kada se naiđe na prvi karakter koji nije slovo, to se interpretira kao signal za kreiranje objekta koji predstavlja reč od objekata slova sadržanih u listi *characters*. Uzima se prvi objekat iz liste, a zatim i drugi. Proverava se da li drugi objekat sledi prvi tako što se referenca drugog objekta prosledi heš tabeli *nextObject* koja je sadržana u prvom objektu. Ako rezultat bude *null*, znamo da ne postoji reč koja počinje sa “*go*”, pa ćemo kreirati novi objekat gde će vektor *sequence* sadržati referencu na objekat koji predstavlja “*g*” na prvoj poziciji i referencu na slovo “o” na drugoj poziciji (Sl. 4).

Nakon što su sva četiri karaktera obrađena, lista characters će biti prazna, a imaćemo reprezentaciju reči “*good*”, kao što je prikazano na Sl. 1. Referenca na objekat koji predstavlja reč “*good*” će zatim biti dodata kao prvi element u listu *words*. Ova lista će se proširivati drugim rečima sve dok ne naiđemo na neki od karaktera “.”, “?”, “!”, itd., koji označavaju kraj rečenice. Ovi objekti će se koristiti za kreiranje novog objekta koji predstavlja rečenicu, a taj objekat će se zatim postaviti kao prvi element u listu *sentences*. Kada naiđemo na karaktere <*nl*><*cr*>, kreiraćemo novi objekat koji predstavlja paragraf i postaviti ga na početak liste *paragraphs*. Na kraju, kada naiđemo na <*eof*>, kreiraćemo objekat koji predstavlja ceo dokument u strukturisanom obliku.



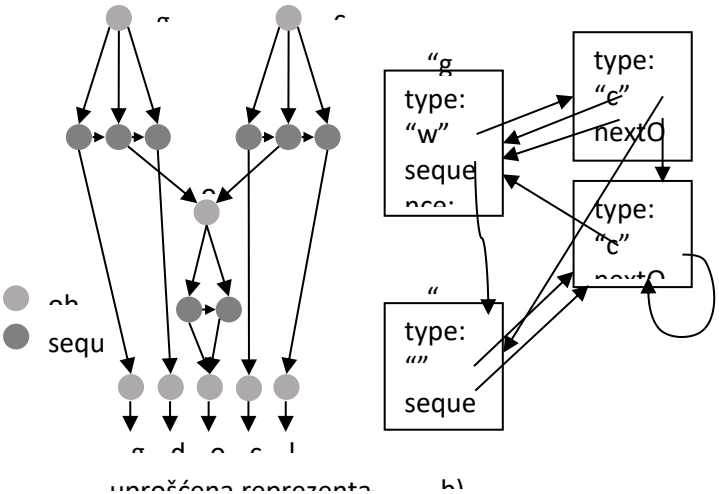

Sl. 4. Stanje objekta reči nakon što su obrađena dva karaktera.

Vraćamo se primeru kreiranja reči. Pretpostavimo da ponovo naiđemo na reč “*good*”. Ponovo ćemo imati četiri objekta koji predstavljaju odgovarajuća slova u listi characters. Međutim, ovog puta ćemo pronaći objekat koji predstavlja “*g*” u heš tabeli *elementaryObjects*. Takođe ćemo pronaći odgovarajući objekat za “*o*” i u heš tabeli *nextObject* objekta koji predstavlja “*g*”, pronaći ćemo referencu na objekat koji predstavlja reč *“good*”. Dakle, ovde nemamo ništa da radimo jer objekat koji predstavlja “*o*“ sledi objekat koji predstavlja “*g*” unutar objekta koji predstavlja reč “*good*”. Isti proces se ponavlja za drugo slovo “*o*” i “*d*”, i zaključujemo da je odgovarajuća reč već naučena i da samo treba dodati referencu na ovu reč u listu *words*.

Pretpostavimo da nakon obrade reči “*good*” naiđemo na reč “*cool*”. Možemo primetiti da se niz slova “*oo*” ponavlja u ove dve reči. Ponovljeni nizovi koji se javljaju u različitim kontekstima treba da budu predstavljeni jedinstveno, kreiranjem novog složenog objekta. Na Sl. 5. a) možemo videti pojednostavljenu reprezentaciju u kojoj su prikazani samo objekti i nizovi unutar njih nakon obrade reči “*cool*”. Kreirali smo novu složenu grupu tipa “” koja jedinstveno predstavlja ponovljeni niz slova – “*oo*” u ove dve reči. Na Sl. 5. b), atribut *nextObject* objekta koji predstavlja slovo “*d*” sada sadrži unos koji označava da objekat koji predstavlja “*oo*” sledi slovo “*g*”, ali i dalje postoji unos koji opisuje da slovo “*o*” i dalje sledi slovo “*g*”. Slovo “o” sledi slovo “*g*”, iako ne unutar istog niza referenci u okviru reči “*good*”, već unutar objekta koji sledi “*g*”. Važno je predstaviti ovaj odnos jer, kada proveravamo da li jedan objekat sledi drugi kako bismo odlučili da li je potrebno kreirati novi obuhvatajući objekat ili ne, moramo biti u mogućnosti da proverimo objekte na dnu odgovarajućeg sloja. U slučaju reči proveravamo da li su dva slova

susedi u istom nizu, u slučaju rečenica proveravamo to za reči, u slučaju paragrafa to proveravamo za rečenice itd.

Na kraju, pretpostavimo da dodamo reč "*gold*". Sada imamo ponovljene nizove: "go" u rečima "*gold*" i "*good*", i "*oo*" u rečima "*good*" i "*cool*". Kao što je očigledno, ova dva niza ne mogu istovremeno biti predstavljena u objektu koji predstavlja reč "*good*". Ovde niz koji prethodi ima prednost, tako da će biti predstavljen samo niz "*go*", a pojednostavljena reprezentacija prikazana je na Slici 6.




Sl. 5. Stanje objekta reči nakon što su obrađena dva karaktera.

*A. Razdvajanje objekta*

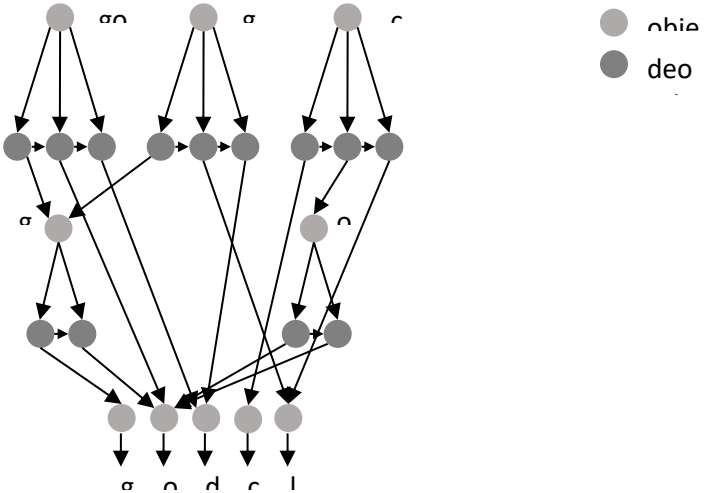
Razdvajanje objekata predstavlja proces u kojem se ponovljeni nizovi elemenata unutar različitih konteksta prepoznaju, izdvajaju i predstavljaju jedinstveno u hijerarhijskoj strukturi. Ovaj proces omogućava optimizaciju reprezentacije podataka i smanjenje redundancije, dok se zadržava semantička i strukturna konzistencija.

Na primeru reči "*gold*", možemo uočiti ponovljene nizove "*go*" u rečima "*gold*" i "*good*", kao i niz "*oo*" u rečima "*good*" i "*cool*". Ovi ponovljeni nizovi ne mogu istovremeno biti pravilno predstavljeni u okviru postojećeg objekta za reč "*good*". U ovakvim slučajevima, prioritet se daje nizovima koji prethode u procesu obrade, pa se niz "*go*" izdvaja kao jedinstven, dok se drugačiji ponovljeni nizovi prepoznaju i reorganizuju u zasebne objekte.

Kada se obrađuje novi niz, poput reči "*gold*", proces počinje identifikacijom ponovljenih sekvenci. Na primer, kada se dolazi do karaktera "*o*" koji sledi "*g*", prepoznaje se niz "*go*". Međutim, pošto "*o*" pripada prethodno

definisanom nizu “*oo*”, neophodno je izvršiti razdvajanje objekata. Niz “*oo*” postaje referenciran samo u kontekstu reči “*cool*”, dok se kreira novi objekat koji predstavlja niz “*go*”. Time se omogućava jedinstvena reprezentacija oba niza unutar strukture.

Različite reči, kao što je “*good*”, mogu imati više mogućih strukturnih oblika. Svaki oblik zavisi od redosleda reči koje su već obrađene i dodate u strukturu. Na primer, za reč od četiri karaktera moguće je kreirati jedanaest različitih strukturnih formi. Ipak, krajnja forma nije presudna, već je važno da svaka reč bude jedinstveno predstavljena, uz očuvanje veza i odnosa između njenih delova.




Sl. 6. Razdvajanje objekta.

Uopšteno, proces razdvajanja može obuhvatiti mnoge objekte, a ne samo jedan, kao što je prikazano na Sl. 6. Ovo se može desiti kada se reči obrađuju radi kreiranja odgovarajuće rečenice, a dugačak niz reči u toj rečenici delimično se preklapa sa istim nizom reči u drugoj rečenici (Sl. 7).

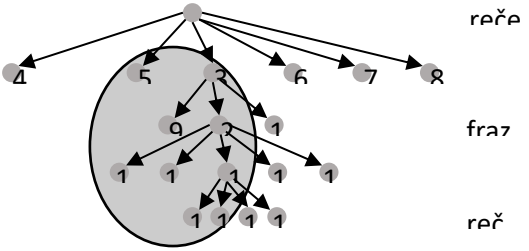
Ovde moramo razdvojiti tri objekta označena kao 1, 2 i 3. Primetite da imamo isti niz reči predstavljen frazama 5, 9, 11, 12 i rečima 15 i 16. Cilj je kreirati novu frazu koja će jedinstveno predstavljati ovaj niz reči u dve rečenice.

Počećemo proces razdvajanja od fraze na dnu (fraza broj 1). Prvo ćemo kreirati dve nove fraze 1’ (koja sadrži reči 15 i 16) i 1” (koja sadrži reči 17 i 18). Tako će fraza 1 sada sadržavati niz koji se sastoji od fraza 1’ i 1”.

Zatim ćemo razdvojiti frazu broj 2. Kreiraćemo frazu 2’ koja predstavlja niz sastavljen od fraza 11, 12 i 1’. Pre nego što kreiramo frazu 2“, moramo

proveriti da li već postoji fraza koja predstavlja isti niz reči kao fraze 1' i 13. Ako postoji (npr. fraza 19), fraza 2" će predstavljati niz sastavljen od fraze 19 i fraze 14. Ako ne postoji, fraza 2" će predstavljati niz sastavljen od fraza 1", 12 i 14.

Na kraju, razdvojićemo frazu broj 3 na sličan način kao što smo razdvojili frazu broj 2. Kreiraćemo frazu 3' koja predstavlja niz sastavljen od fraza 9 i 2'. U slučaju fraze 3" takođe imamo dve mogućnosti. U prvom slučaju, već postoji fraza 20 koja predstavlja niz fraza 2" i 10. U tom slučaju, fraza 3" će biti predstavljena frazom 20. U suprotnom, fraza 3" će sadržati fraze 2" i 10.
Fraza 21 je ona koja nam je potrebna i ona će se sastojati od fraza 5 i 3'.
Moramo proveriti da li postoji fraza koja predstavlja fraze 3" i 6. Ako takva fraza postoji (npr. fraza 22), stara rečenica će se sastojati od fraza 4, 21, 22, 7 i 8. Ako takva fraza ne postoji, stara rečenica će biti predstavljena nizom fraza 4, 21, 3", 6, 7 i 8. Proces razdvajanja je ovde završen.
U slučaju kada objekti x' ili x" sadrže samo jedan objekat, tada nije potrebno kreirati nove objekte, već samo ponovo iskoristiti isti objekat za kreiranje y' ili y" na nivou iznad.




Sl. 7. Razdvajanje objekta u opštem primeru.

## V. PREGLED POSTOJEĆIH MODELA

U ovom poglavlju razmatraju se tri istaknuta modela za analizu i reprezentaciju hijerarhijskih struktura podataka, uključujući njihove osnovne karakteristike, prednosti i ograničenja. Nakon detaljnog pregleda, predloženi model će biti upoređen sa njima kako bi se istakle inovacije i doprinosi u odnosu na postojeće pristupe.

### *A. Word2Vec*

*Word2Vec* predstavlja jedan od najpoznatijih modela za distribuisanu reprezentaciju reči. Zasniva se na dve arhitekture: *Continuous Bag-of-Words* (engl. skraćeno *CBOW*) i *Skip-Gram*. *CBOW* model predviđa ciljnu reč na osnovu njenog konteksta, dok *Skip-Gram* predviđa kontekstualne reči na osnovu ciljne reči. Ovaj model omogućava otkrivanje semantičkih i sintaktičkih odnosa između reči kroz njihove vektorske reprezentacije. Iako je

*Word2Vec* efikasan za analizu odnosa na nivou reči, on ne nudi eksplicitnu podršku za hijerarhijske strukture višeg nivoa, poput fraza, rečenica ili dokumenata.

*B. Latent Semantic Analysis*

*Latent Semantic Analysis (*engl. skraćeno *LSA)* se oslanja na linearnu algebru, konkretno *Singular Value Decomposition* (engl. skraćeno *SVD*), kako bi smanjio dimenzionalnost podataka i identifikovao latentne semantičke strukture u tekstualnim korpusima. Model je koristan za otkrivanje sinonimije i polisemije, ali je njegov primarni fokus na dokumentima i rečima kao celinama. Nedostatak ovog pristupa je ograničena fleksibilnost u analizi sekvencijalnih podataka i nemogućnost dinamičkog prilagođavanja hijerarhijskim strukturama.

*C. BERT*

Bidirectional Encoder Representations from Transformers (engl. skraćeno *BERT*) koristi dvosmerne transformere za kreiranje kontekstualizovanih reprezentacija reči [9]. Njegova glavna snaga leži u sposobnosti da simultano analizira veze između reči na levoj i desnoj strani konteksta. *BERT* je izuzetno efikasan za zadatke poput klasifikacije teksta i odgovaranja na pitanja, ali njegova primena na strukturisane hijerarhijske reprezentacije podataka je ograničena. Model nema ugrađenu podršku za više nivoe hijerarhije, kao što su fraze, rečenice i paragrafi, već se fokusira na analizu pojedinačnih sekvenci reči.

*D. Uporedna analiza sa predloženim modelom*

Predloženi model pruža značajne inovacije u odnosu na navedene modele. Dok *Word2Vec* i *BERT* nude izuzetnu semantičku analizu na nivou reči, dati model proširuje tu mogućnost na više nivoe hijerarhije, uključujući fraze, rečenice, paragrafe i dokumente. Za razliku od *LSA*, koji se oslanja na statičnu analizu, predloženi model omogućava dinamičko kreiranje i prilagođavanje struktura u realnom vremenu.

Jedna od ključnih prednosti predloženog modela u odnosu na *BERT* jeste njegova sposobnost da rekonstruše tekst iz hijerarhijskih struktura, što je značajno u domenima gde je očuvanje originalnih sekvenci ključno, poput analize prirodnog jezika ili bioinformatike. Takođe, model eliminiše redundantnost kroz proces čepanja objekata, što omogućava efikasnu obradu velikih skupova podataka, dok *Word2Vec* i *LSA* ne nude sličnu funkcionalnost.

Što se tiče obrade sekvenci, predloženi model koristi algoritme koji eksplicitno prepoznaju i reprezentuju odnose između elemenata na svim nivoima hijerarhije. Ova fleksibilnost omogućava njegovu primenu u širokom spektru domena, uključujući prirodne jezike, analizu vremenskih serija i

bioinformatiku, dok su *Word2Vec*, *LSA* i *BERT* optimizovani za specifične zadatke.

## V. ZAKLJUČAK

Predloženi model za hijerarhijsku reprezentaciju podataka donosi značajne mogućnosti za unapređenje analize sekvencijalnih fenomena u različitim oblastima. Fokusiran na dinamičku prilagodljivost, eliminaciju suvišnih podataka i očuvanje semantičkih odnosa, model pruža osnovu za dalji razvoj metoda strukturisane obrade podataka.

Jedan od ključnih pravaca budućih istraživanja odnosi se na unapređenje performansi kroz paralelnu i distribuiranu obradu. Upotrebom višenitnog programiranja i ubrzanja putem grafičkih procesora, može se značajno povećati brzina analize velikih skupova podataka. Dodatno, primena distribuiranih sistema za obradu podataka, poput Sparka ili Hadupa, otvara mogućnosti za efikasniju analizu velikih korpusa informacija u realnom vremenu.

Dalje istraživanje treba da obuhvati primenu modela u specifičnim domenima kao što su biološke nauke, analiza vremenskih nizova (engl. *timeslots)* i mreže senzora [10]. Na primer, u biološkim istraživanjima model bi mogao doprineti otkrivanju obrazaca u genetskim podacima, dok bi u analizi vremenskih nizova bio koristan za predikciju promena u klimatskim uslovima ili ekonomskim tokovima. U oblasti mreža senzora, model može omogućiti efikasniju obradu podataka prikupljenih u stvarnom vremenu.

Poseban pravac razvoja može biti proširenje modela ka dubljem razumevanju semantičkih i sintaksičkih odnosa u tekstovima. Povećanjem preciznosti u prepoznavanju odnosa među elementima, model može biti prilagođen za zadatke poput analize sadržaja, generisanja prilagođenih preporuka ili automatizovanog prevođenja.

Na kraju, potrebno je istražiti načine za optimizaciju memorijskih i računarskih resursa potrebnih za implementaciju modela. Razvoj specijalizovanih hardverskih rešenja, poput memorijskih sistema sa direktnom obradom podataka, mogao bi dodatno unaprediti efikasnost i proširiti primenu modela na još širi spektar problema. Ova unapređenja postavljaju temelje za razvoj novih tehnologija i alata koji mogu doneti značajan doprinos naučnim i industrijskim aplikacijama.

## LITERATURA

[1] Xin Rong, “word2vec Parameter Learning Explained”, 2014.

[2] A. Vaswani, N. Shazeer, N. Parmar, L. Jones, J. Uszkoreit, A.N. Gomez, L. Kaiser, “Attention is all you need”, Proc. of NeurIPS, 2017.

[3] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, R.A. Harshman, "Indexing by latent semantic analysis", Japan Analytical & Scientific Instruments Show, vol. 41, no. 6, pp. 391–407, 1990.
[4] Rehder, B., Schreiner, M. E., Wolfe, M. B. W., Laham, D., Landauer, T. K., & Kintsch, W, "Using latent semantic analysis to assess knowledge: Some technical considerations. Discourse Processes", 25(2–3), 337–354. 1998.
[5] S. Hochreiter, J. Schmidhuber, "Long short-term memory", Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[6] Y. Ji, T. Cohn, L. Kong, C. Dyer, J. Eisenstein, "Document context language models", arXiv:1511.03962, 2015.
[7] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations", 2020.
[8] N. Kalchbrenner, E. Grefenstette, P. Blunsom, "A convolutional neural network for modelling sentences", Proc. of ACL, 2014.
[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, "Language Models are Unsupervised Multitask Learners", 2019.
[10] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P.J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer", 2020.

ABSTRACT

This paper addresses the challenge of holistic knowledge representation and structural learning through the concept of structured representation in one-dimensional spaces. Unlike traditional natural language processing (NLP) techniques, which focus on distributed representations and partial consideration of semantics, the proposed approach enables comprehensive text analysis by integrating hierarchical data organization.

The proposed method involves creating dynamic structures for representing textual corpora, where each level of the structure is based on unique objects that represent elementary and complex units, such as words, phrases, sentences, and paragraphs. By employing algorithms for object recognition, creation, and reorganization, the method facilitates efficient management of large volumes of textual data.

The results demonstrate that the proposed model overcomes the limitations of existing methods, such as scalability and precision in text analysis, while maintaining flexibility in modeling new information. The conclusions highlight the significant potential of structured representation in tasks such as text classification, sentiment analysis, and language generation.

## HOLISTIC KNOWLEDGE REPRESENTATION AND STRUCTURE LEARNING

Mladen Stanojević, Mladen Jovanović