Algoritmi za detekciju plagijarizma

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Nemanja Subotić, Igor Rodić
suboticnemanja93@gmail.com, igorrodic@gmail.com

11. april 2016.

Sažetak

Sa pojavom modernih tehnologija, pre svega interneta, svet je postao u velikoj meri izložen raznim oblicima zloupotreba. Jednu od tih zloupotreba predstavljaju plagijarizmi. Osnovnu zaštitu predstavljaju računarski algoritmi za detekciju plagijarizma, jer je , s obzirom na količinu dostupnih podataka, za čoveka u velikom broju slučajeva to nemoguće. Rad prikazuje osnovne ideje i pristupe pri impelementaciji ovih algoritama, kao i detaljni opis i implementaciju algoritma za detekciju plagijarizma tekstova na engleskom. Ovaj algoritam zastupa jedan neobičan pristup i u praksi daje dosta dobre rezultate,dok u nekim slučajevima čak i najbolje.

Sadržaj

1 Uvod

Postoje mnoge definicije plagijarizma, nijedna nije sasvim precizna, pre svega zato što plagijarizam predstavlja nešto drugo za svakoga od nas. Definicija koja stoji u Kembridžovom rečniku (eng. *Cambridge English Dictionary*) glasi:

Definicija 1.1 *Plagirati: koristiti tuđe ideje i praviti se da su vaše.*

Osnovu za detekciju plagijarizama predstavljaju računari i efikasni algoritmi koji su u tu svrhu implementirani. U ovom radu predstavili smo čitaocu osnove pisanja algoritama za njihovu detekciju, kao i detaljniji opis algoritma za detekciju plagijarizma teksta, zasnovanog na n-gramima. Prvo smo opisali ideje i pristupe, kao i klasifikaciju algoritama za detekciju plagijarizama (2). Zatim smo u sledećoj sekciji opisali moguće načine sakrivanja plagijarizma, kao i načine da te metode budu otkrivene (3). Na kraju, implementirali smo algoritam teksta zasnovan na n-gramima (4).

2 Ideje i pristupi

Prva stvar o kojoj treba diskutovati jeste klasifikacija sistema za detekciju plagijarizama. Ona je, prirodno, za svakoga drugačija. Osnovna podela, prema Mozgovoy-u [4] deli pomenute sisteme na dve podgrupe, na sisteme koji prave "otisak prsta"(eng. *fingerprint*) dokumenta i na sisteme koji porede sadržaj.

"Otisak prstađokumenta predstavlja kratku sekvencu bajtova koja karakterizuje duži dokument. On može da se dobije primenom heš (eng. *hash*) funkcije na dokument, ali obično se koristi niz numeričkih atributa (prosečan broj reči po liniji, prosečan broj ključnih reči itd.), i zatim se pomoću funkcije radzdaljine računa stepen različitosti dva dokumenta. Ova tehnika se danas retko koristi jer čak i male promene mogu da rezultuju potpuno drugačijim "otiscima".

Poređenje sadržaja predstavlja osnovu za veliku većinu sistema za detekciju plagijarizama. Ono je generalno zasnovano na Manberovoj definiciji sličnosti i deli se na dve podvrste. One su poredjenje stringova i drveta parsiranja.

Definicija 2.1 *Manberova definiciji sličnosti: Kažemo da su dva fajla slična ako sadrže značajan broj istih podniski koje nisu prekratke.*

Algoritmi za poređenje sadržaja funkcionišu po istom principu:

```
FOR EACH collection file F
  FOR EACH collection file G, F =/= G
    Calculate similarity between F and G
```

dok se funkcija za određivanje sličnosti menja. Kada pričamo o poređenju stringova pod time podrazumevamo poređenje fajlova, koji se tretiraju kao veliki stringovi. Logično, ovaj način poređenja ne čuva strukturu fajlova, kao ni programskog koda. Algoritmi koji koriste ovaj način detekcije su: FPDS [?], koji računa sličnost po formuli:

```
sim(F, G) = MatchedTokens(F, G) / TotalTokens(G)
```

YAP [?] (jedan od prvih algoritama ove vrste, koristi poređenje liniju-po-liniju Levenštajnovim rastojanjem), RKR-GST koji je implementiran u sistemu YAP3 [?] (pravi pokrivač nepreklapajućih stringova koji sadrže

maksimalan mogući broj tokena iz oba fajla, algoritam je NP kompletan pa zavisi od uspešnih heuristika, npr. duže podniske su vrednije od kraćih).

Drveta parsiranja sa druge strane očuvavaju strukturu fajla, kako tekstualnog (poglavlja, pasusi itd.) tako i programskog (klase, funkcije itd.). Prednosti ovog metoda poređenja još nisu u potpunosti iskorišćene, jedan od algoritama koji to pokušava je Sim utility [?]. On koristi poređenje stringova, ali ne nad celim fajlovima, već nad njihovim drvetima parsiranja, tj. parser prethodi algoritmu poređenja stringova.

3 Sakrivanje plagijarizma

U ovoj sekciji biće reči o tehnikama sakrivanja plagijarizama, sa ciljem težeg otkrivanja, kao i tehnikama za njihovo otkrivanje i prevazilaženje.

Kada je u pitanju tekst neke od tehnika za sakrivanje plagijarizma su parafraziranje, korišćenje sinonima, prevođenje na drugi jezik, pa zatim nazad u početni itd. Kada je u pitanju programski kod definisani su nivoi modifikacije koji daju uvid u načine na koje plagijarizam može da bude sakriven [3]:

- Izmena komentara
- Izmena belina
- Preimenovanje identifikatora
- Izmena rasporeda blokova koda
- Izmena rasporeda naredbi unutar blokova
- Izmena redosleda operatora/operanada u izrazima
- Izmena tipova podataka
- Dodavanje redundantnih naredbi ili promenljivih
- Menjanje naredbi kontrole toka ekvivalentnim naredbama kontrole toka (while u do-while itd.)
- Menjanje poziva funkcije sa telom funkcije

Navedene tehnike sakrivanja plagijarizama prevazilaze se pretprocesiranjem. Pretprocesiranje predstavlja modifikovanje fajla pre same primene algoritma za detekciju plagijarizma. U sekcijama 3.1 i 3.2 opisane su tehnike pretprocesiranja teksta, i programskog koda.

3.1 Pretprocesiranje teksta

Tehnike pretprocesiranja teksta su WSD i Thesauri, i parseri. Menjanje reči sinonimima skoro je nemoguće otkriti bez pomoći računara. Na njima se taj problem rešava tokenizacijom, jedan od takvih softvera je i WordNet [?] (Thesaurus). Ali, pre nego što se uradi zamena svih sinonima jednom rečju, prvo mora biti primenjem WSD (eng. *Word Sense Disambiguation*) metod koji određuje značenje reči u zadatom kontekstu, i tek onda može da bude primenjen tokenizator. Još jedna primena WSD metoda je pri otkrivanju plagijarizma ideje, kada se WSD koristi u kombinaciji sa drvetom konceptualnih klasa, ulazni fajl salje se na semantičku analizu i dobija se lista klasa, umesto originalnih reči. Lista se dalje procenjuje pomoću uobičajnih algoritama detekcije plagijarizama.

Parseri se, sa druge strane, koriste kada je izmenjena struktura teksta (izmenjen redosled reči itd.). Oni dele rečenice na manje sekvence

koji oslikavaju strukturu teksta, npr. parser može da prepozna homogene delove rečenice i da ih sortira leksikografski.

3.2 Pretprocesiranje programskog koda

Pretprocesiranje programskog koda deli se na dve tehnike, tokenizaciju i parametrizovano poklapanje. Tokenizacija pretvara programski kod u niz tokena, time ga spuštajući na gradivni nivo (predstavljajući promenljive, pozive funkcija, naredbe itd.) i eliminišući većinu načina skrivanja. Složenost procesa tokenizacije je linearna, stoga ona ne utiče na kompleksnost finalnog algoritma, medjutim, samim procesom mogu da se izgube važne informacije o sličnostima dva programa. U cilju očuvanja te sličnosti tokenizator se koristi zajedno sa parametrizovanim poklapanjem. Parametrizovano poklapanje smatra dva koda identičnim ako jedan može da se dobije iz drugog primenom konačnog broja zamena identifikatora.

4 Implementacija algoritma za detekciju plagijarizama teksta

U ovom delu opisaćemo metod za detekciju pagijarizma tekstova na engleskom jeziku korišćenjem n-grama stopeči (eng. stopwords). Author ovog metoda je grčki profesora sa Egejskog univerziteta, Efstathios Stamatatos i metod je objavljen 2011. godine [5]. Pre nego što krenemo sa opisom algoritma, objasimo prvo osnovne pojmove: n-gram predstavlja niz od n susednih stavki u tekstu, gde stavke mogu biti karakteri ili reči, dok stoprečima smatramo najčešće korišćene reči nekog jezika (u našem slučaju engleskog).

Stopreči su se pokazale veoma korisne u istraživanju teksta u slučajevima kad je bitniji stil pisanja od sardržaja (npr. pripisivanju autorstva i detekcije žanra). Pa tako i Efstathios Stamatatos predstavlja novi metod detekcije plagijarizma koji je zasnovan isključivo na strukturnim informacijama. Umesto da eliminiše stopreči, kao što je uobičajena praksa, iz teksta eliminiše sve ostale tokene i metod bazira na preostalom nizu stopreči. Najčešći način plagiranja je zamena reči i izraza sinonimima. Pošto su stopreči nezavisne od sadržaja i ne nose nikakve semantičke informacije, sekvence stopreči često ostaju nepromenjene prilikom manjih izmena teksta.

4.1 Opis algoritma

Autor je metod podelio na tri dela. Uprvom delu se određuje da li je sumnjivi dokument plagijat ili ne. Nakon toga, trebalo bi pronaći sve plagijarizovane delove, odrediti njihove granice i u sumnjivom i u izvornim dokumentima. Na kraju je pozeljno za svaki pagijarizovani deo koji smo pronasli odrediti koeficijent plagijarizma.[5]

Reprezentacija teksta se zasniva na n-gramima stopreči (SWNG - stopword n-gram). Za dati dokument i listu stopreči tekst dokumenta se transformiše na sledeći nacin: prvo se velika slova prevedu u mala, zatim se vrši tokenizacija i svi tokeni koji se na nalaze u listi stopreči se izbacuju, na kraju se grade n-grami. Ovaj skup n-grama stopreči nazivamo profil dokumenta - $P(n, d)$.

Efstathios Stamatatos kao stopreči koristi pedeset najčešće korišćenih reči engleskog jezika 1 prema Britanskom nacionalnom korpusu.

Tabela 1: 50 najčešćih stopreči u engleskom jeziku.

1. the	11. with	21. are	31. or	41. her
2. of	12. he	22. not	32. an	42. n't
3. and	13. be	23. his	33. were	43. there
4. a	14. on	24. this	34. we	44. can
5. in	15. i	25. from	35. their	45. all
6. to	16. that	26. but	36. been	46. as
7. is	17. by	27. had	37. has	47. if
8. was	18. at	28. which	38. have	48. who
9. it	19. you	29. she	39. will	49. what
10. for	20. 's	30. they	40. would	50. said

4.2 Odredjivanje kandidata

Prvi korak je izdvajanje skupa izvornih dokumenata, čiji elementi imaju delove koji su verovatno plagijarizovani u sumnjivom dokument. Ovo zahteva poređenje sumnjivog dokumenta sa svim izvornim dokumentima koje imamo kako bi se pronašla bilo kakva lokalna sličnost.

Cilj je pronaći zajednički n-gram stopreči sumnjivog i izvornog dokumenta. Najvažnije je odrediti odgovarajuću vrednost za n, tj. odrediti kolika bi trebala da bude dužina niza uzastopnih stopreči da bi se otkrila sličnost između sumnjivog i izvornog dokumenta. Recimo da je to vrednost n1, tada za svaki zajednički n-gram između para dokumenata gde je $n < n_1$ poklapanje se smatra neznačajnim, dok za svaki zajednički n-gram između para dokumenata gde je $n \geq n_1$ sugeriše na plagijarizam. Vredonst n1 bi trebala biti relativno visoka, ali trebalo bi imati u vidu da pri znatim promenama teksta dužina niza uzastopnih reči neće biti velika.

Problem predstavlja slučaj kada se pronađe zajednički n-gram sumnjivog i izvornog dokumenta koji pripadaju delovima koji su semantički različiti, sto uvećava broj lažno pozitivnih plagijarizovanih delova. To je obicno slučaj kada zajednički n-gram sadrži samo veoma česte stopreči, u engleskom jeziku, to su prvih šest najčešćih stopreči(the, of, and, a, in, to) i 's. Da bi izbegao ovaj slucaj, autor metoda, uvodi ograničenje na sadržaj zajedničkog n-grama. Neka je $C = \{the, of, and, a, in, to, `s\}$ skup stopreči koje se obicno pojavljuju u slučajnim pogocima. Neka je D_x skup sumnjivih dokumenata koje zelimo da ispitamo, a D_s skup izvornih dokumenata i da $d_x \in D_x$ i $d_s \in D_s$, dok su $P(n_1, d_x)$ i $P(n_1, d_s)$ odgovarajući profili ovih dokumenata koji sadrže n-grame stopreči dužine n_1. Tada je detektovan plagijarizam ukoliko je zadovoljen sledeći kriterijum :

$$\exists g \in P(n_1, d_x) \cup P(n_1, d_s) : member(g, C) < n_1 - 1 \wedge maxseq(g, C) < n_1 - 2$$

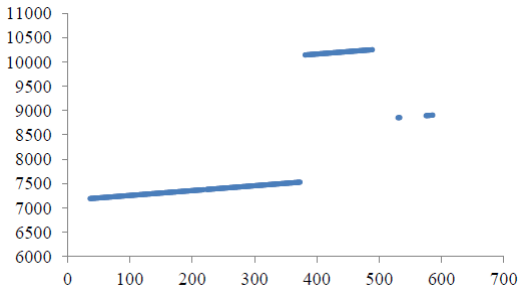
gde je $member(g, C)$ funkcija koja vraća broj stopreči n-grama g koji pripadaju skupu C, a $maxseq(g, C)$ funkcija koja vraća maksimalnu sekvencu stopreči u g, koji pripadaju C.

4.3 Odredjivanje granice delova

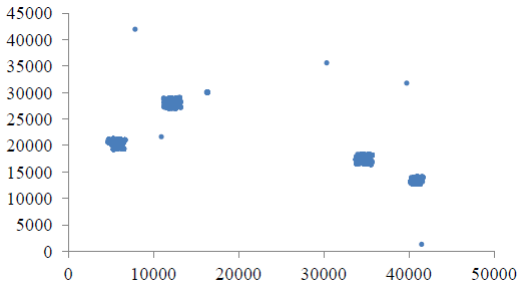
Nakon sto je pronađen skup izvornih dokumenata koji se poklapaju sa sumnjivim, potrebno je odrediti granice sličnih delova teksta u svakom od ovih dokumenata. Neka je $D_{rx} \subseteq D_s$ skup dokumenata izdvojenih u prethodnom koraku. Nas cilj je pronalaženje svih g takvih da je

$g \in P(n, d_x) \cup P(n, d_s)$ pri čemu je $d_s \in D_{rx}$, a nakon toga i građene sekvenci maksimane dužine od pronađenih g,tako da svaka dobijene sekvenca predstavlja plagijarizovani deo teksta. Granice delova nam predstavljaju početni i krajnji n-grami u dobijenim sekvencama.

Poklapanje n-grama sumnjivog i izvornog dokumenta može se predstaviti dijagramom rasipanja. Na slici (1) prikazan je primer doslovnog plagijarizma, kao i primer plagijarizma nastalog izmenama izvornog dokumenta (2). Broj praznina i grupa na dijagramu rasipanja zavisi od vrednosti n, tj. reda n-grama korišćenog ukreiranju profila dokumenata. Što je n veće, imaćemo više praznina i grupa.



Slika 1: Dijagram rasipanja poklopljenih n-grama u slučajevima doslovnog plagijarizma.



Slika 2: Dijagram rasipanja poklopljenih n-grama u slučajevima kada je plagijarizovani pasus pretežno modifikovan.

U prethodnom koraku, da bi utvrdio sličnosti između para dokumenata, author je koristio duže n-grame (reda n_1), ali za određivanje granica potrebni su mu kraći n-grami (reda $n_2 < n_1$) da bi imao detaljnije poređenje. I ovde se dodaje uslov za izbegavanje slučajnih poklapanja n-grama, samo što je uslov malo blaži da bi praznine između zajdničkih n-gram bile manje. Neka su $P(n_2, d_x)$ i $P(n_2, d_s)$ profili sumnjivog i izvornog dokumenta koji sadrže n_2-grame stopreči. Zajdenički n_2-gram g je pronađen ako je zadovoljen sledeći uslov:

$$g \in P(n_2, d_x) \cup P(n_2, d_s) : member(g, C) < n_2$$

Neka je $M(d_x, d_s)$ skup zajedničkih n-grama između profila P(n_2, d_x) i $P(n_2, d_s)$. Na primer, ukoliko je $(17, 9) \in M$, to znaci da sedamnaesti n_2-gram u $d_x$odgovara devetom n_2-gram u d_s. Članovi skupa M su poređani

na osnovi prvog pojavljivanja u tekstu. Skup M možemo podeliti na dva dela M_1 i M_2, tako da odovaraju, redom, sumnjivom i izvornom dokumentu. Uzastopne vrednosti u M_1 su rastuće dok u M_2 mogu biti i opadajuće. Granice plagijarizovanih delova teksta povezane su sa velikim promenama susednih vrednosti u skupovima M_1 i M_2, što se moze videti na slikama 1 2.

Pri određivanja granica problem nam može predstavljati slučaj kada imamo više plagijarizovanih delova teksta koja su relativno blizu, dok su odgovarajući originalni delovi u izvornom dokumentu udaljeni, kao i suprotno. U opisu metoda, predloženo je sledeće rešenje [5] : Prvo odrediti početni skup granica u sumnjivom dokumentu, tj. podeliti skup M na osnovu rezlika susednih vrednosti u M1, pri čemu su dozvoljene manje praznine. Zatim, granice u izvornom dokumentu odreditio tako što se dobijeni skupovi podele na osnovu razlika susednih vrednosti u M2, gde bi takođe trebalo dozvoliti manje praznine. I na kraju konačni skup granica u sumnjivom dokumentu odrediti na osnovu dobijenih granica u izvornom dokumentu.

4.4 Odredjivanje koeficijenta plagijarizma

Svaka detekcija plagijarizma može se predstaviti kao uređena četvroka $< t_x, d_x, t_s, d_s >$, gde t_x predstavlja plagijrarizovan deo teksta u sumnjivom dokument d_x, koji odgovara delu teksta t_s u izvornom dokumentu d_s. Da bi odredio koeficijent plagijarizma, autor metoda, koristi karakterske n-grame. Prvo se vrši normalizacija delova t_x i t_s, tako što se sva velika slova zamene malim i izbace svi interpukcijski znaci. Zatim se sličnost između njih računa po sledećoj formuli:

$$Sim(t_x, t_s) = \frac{|P_c(n_c, t_x) \cap P_c(n_c, t_s)|}{max(|P_c(n_c, t_x)|, |P_c(n_c, t_s)|)}$$

gde su $P_c(n_c, t_x)$ i $P_c(n_c, t_s)$ karakterski n_c-grami profili delova t_x i t_s. Izbor vredonsti za n_c zavisi od toga koliko želimo da merenje bude fleksibilno. Što su duži karakterski n-grami to je veća šansa da se izmene neće detektovati kao plagijarizam.

4.5 Prednosti, rezultati i parametri

Eksperimentalo utrvđene vrednost za veličine parametara, rezultati testiranja...

5 Zaključak

Svakim danom broj dokumenata javno dostupnih je sve veći, a samim tim i broj plagijarizama. Uporedo, razvijaju se i algoritmi za detekciju prlagijarizama kako teksta, tako i programskog koda. U ovom tekstu spomenuli smo neke od tih algoritama i detaljije opisali algoritam grčkog profesora Efstathios Stamatatos zasnovan na n-gramima stopreči za detekciju plagijarizma teksta. Ovaj algoritam je zaniljiv jer uvodi jedan oni pritup, ali i sadrži u sebi neke prethodne. Trebalo bi pokušati modifikovati algoritam i primeniti ga na programskom kodu.

Literatura

[1] Georgina Cosma. *An Approach to Source-Code Plagiarism Detection and Investigation Using Latent Semantic Analysis*. PhD thesis, Coventry, UK, 2008.

[2] A. MacLeod F. Culwin and T. Lancaster. Source code plagiarism in U.K H.E computing schools, issues, attitudes and tools. 2001.

[3] M. Luck M. Joy. Plagiarism in Programming Assignments. 1999.

[4] M. Mozgovoy. *Enhancing Computer-Aided Plagiarism Detection*. PhD thesis, Joensuu, Finland, 2007.

[5] Efstathios Stamatatos. Plagiarism Detection Using Stopword n-grams. 2011.

A Dodatak

Pored ovog rada, u prilogu se nalaze i izvorni kodovi algoritama koji su implementirani.