Primjena Blumovog filtera u algoritmima detekcije noviteta

Damir Delijic 1/21 C

Contents

[Definicija problema 3](#_Toc138854850)

[Koncept 3](#_Toc138854851)

[Predstavljanje skupova tekstualnih dokumenata 3](#_Toc138854852)

[Mjera slicnosti tekstualnih dokumenata 3](#_Toc138854853)

[Hes funkcija koja cuva slicnost 4](#_Toc138854854)

[Formiranje matrice potpisa 4](#_Toc138854855)

[Blumov filter 4](#_Toc138854856)

[Banding tehnika 4](#_Toc138854857)

[Realizacija algoritma 5](#_Toc138854858)

[Trening 5](#_Toc138854859)

[Detekcija 6](#_Toc138854860)

[Testiranje 7](#_Toc138854861)

[Literatura 7](#_Toc138854862)

# Definicija problema

Zadatak se sastoji u dizajniranju i implementaciji algoritma za detekciju noviteta u skupu tekstualnih dokumenata pomocu Blumovog filtera.

# Koncept

Obzirom da detekcija noviteta podrazumijeva odredjivanje statisticke pripadnosti elementa nekoj distribuciji, nije moguce primjeniti Blumov filter na konvencionalan nacin koriscenjem bilo kakve hes funkcije. Ideja je da se nad trening skupom izvrsi odredjeni broj hesiranja koja cuvaju dio statistickih osobina dokumenata cak i nakon kompresije. A tek nakon toga vrsiti upis u Blumov filter. Prilikom detekcije izvrsiti istu vrstu kompresije nad podacima I provjeriti da li pripadaju Blumovom filteru. Ovim nacinom je moguce veoma brzo testirati pripadnost distribuciji na racun preciznosti.

## Predstavljanje skupova tekstualnih dokumenata

Dokumenti su predstavljeni karakteristicnom matricom. Kolone matrice odgovaraju dokumentima, a redovi odgovaraju šinglovima koji su izvuceni iz poznatog skupa dokumenata. U -tom redu i -toj koloni stoji jedinica ako je taj šinlg prisutan u tom dokumentu, u suprotnom se u toj celiji nalazi nula.

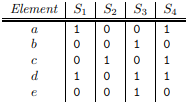


Figure 1: Matricna reprezentacija skupova

Definicija -šingla je sekvenca od tokena koji se pojavljuju u dokumentu. Tokeni mogu biti karakteri, rijeci ili nesto trece. Izbor tokena zavisi od primjene do primjene.

## Mjera slicnosti tekstualnih dokumenata

Nad dokumentima se definise Jaccard-ova mjera slicnosti.

su vektorske reprezentacije dokumenata uz pomoc šinglova . Odnosno . Gdje su:

* - broj koordinata koje su jedinice kod oba vektora
* - broj koordinata koje su jedinice kod a nule kod
* - broj koordinata koje su nule kod a jedinice kod

## Hes funkcija koja cuva slicnost

Hes funkcija, definisana za Jaccard-ovu mjeru slicnosti, koja cuva “udaljenost” dva dokumenta je *minhash*. Heš vrijednost dokumenta je pozicija prve jedinice u njegovom vektoru karakteristicne matrice. Veza izmedju Jaccard-ove slicnosti i *minhasing*-a je ta da je vjerovatnoca da je heš vrijednost dva dokumenta nakon nasumicne permutacije redova ista jednaka njihovoj Jaccard-ovoj slicnosti. Dokaz se moze naci u knjizi [3] u poglavlju 3.3.3.

## Formiranje matrice potpisa

Bira se broj koji predstavlja broj permutacija koji treba da se izvrsi nad karakteristicnom matricom kako bismo pronasli *minhash* vrijednosti za svaki od dokumenata. Nakon toga se dobija matrica kod koje je vrijednost *minhash-a* za tu permutaciju tog dokumenta.

Matricu potpisa nije izvodljivo eksplicitno permutovati zbog njene velicine. Zbog toga se vrsi implicitno permutovanje redova pomocu hes funkcije definisane nad rednim brojem reda. Red se za hes funkciju permutuje u . Konstrukcija matrice potpisa se vrsi na nacin koji je opisan u knjizi [3] u poglavlju 3.3.5.

## Blumov filter

Blumov filter je prostorno efikasna probabilisticka struktura podataka koja se koristi u svrhe testiranja pripadnosti elementa nekom skupu. Lazno pozitivni rezultati su moguci, dok lazno negativni nisu. Ideja je da se elementi mapiraju u koristeci neku heš funkciju , i onda testirati pripadnost provjeravanjem da li vazi

## Banding tehnika

Za dalju kompresiju se koristi banding tehnika opisana u knjizi [3] u poglavlju 3.4. Heš vrijednosti dobijene heširanjem dokumenata kao sto je opisano u banding tehnici se upisuju u Blumov filter. Odnosno redni brojeva baketa u koje se salju dokumetni se cuvaju u filteru. Prilikom detekcije se vrsi ista transformacija podataka i provjerava se da li dokument upada u neki od poznatih baketa. Moguca je greska sa odredjenom vjerovatnocom. Analiza ove tehnike se moze naci u [3] u poglavlju 3.4.2.

# Realizacija algoritma

Algoritam je opisan kroz dva koraka, trening I detekciju.

## Trening

Ulaz programa je skup dokumenata predstavljen karakteristicnom matricom u tekstualnom fajlu. Na osnovu nje se vrsi konstrukcija matrice potpisa. Kao sto je navedeno prije, matricu potpisa nije izvodljivo eksplicitno permutovati vec se koristi hes funkcija koja vrsi preslikavanje rednog broja red u neki drugi. Sto dovodi do toga da matrica bude permutovana implicitno.

Prvo se izabere , odnosno broj permutacija koji treba izvrsiti nad matricom. Broj permutacija utice I na samu preciznost algoritma, sto je veci broj permutacija veci se broj ispitivanja moze izvrsiti nad podacima I manja je vjerovatnoca slucajnih kolizija. Nakon toga se vrsi konstrukcija hes funkcija. Nije lak posao konstruisati razlicitih hes funkcija sa malim brojem kolizija, pa se pravi jedna I generise se seed-ova za tu hes funkciju.



Figure 2: Hes funkcija koja permutuje redni broj reda

Kada bi modulator bio prost broj, permutacija bi bila prava, bez kolizija. Ovakva situacija bi bila idealna, ali obzirom da je modulator u stvari broj singlova u karakteristicnoj matrici, u opstem slucaju ce doci do odredjenog broja kolizija. Potencijalno rjesenje bi bilo da se za dati broj singlova doda odredjeni broj nula-redova karakteristicnoj matrici dok ukupan broj ne bi postao prost. Jer nula-redovi nemaju uticaja na slicnost dokumenata.

Kalkulacija matrice potpisa se vrsi na isti nacin kao u [3] u 3.3.5.

Nakon toga se redovi matrice potpisa grupise u pojaseve (band-ove) od po r redova. Analiza same tehnike I zavistnost slicnosti dokumenata od izabranih b I r su opisani u [3] u poglavlju 3.4.2.

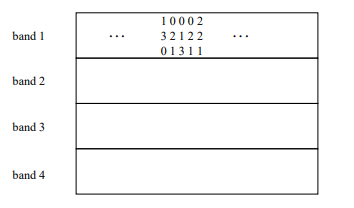


Figure 3: Grupisanje u pojaseve

Formira se matrica pojaseva gdje je hes vrijednost vektora iz -tog pojasa, za -ti dokument.

Vektor se formira iz sledecih komponenti:

* Rednog broja pojasa,
* – vektora dokumenta dobijenog razdvajanjem matrice na pojaseve

A njegova hes vrijednost se dobija pomocu:

* – velika konstanta
* – veliki prosti broj
* Hash = sum

Da bi hesiranje imalo najbolju distribuciju, prime\_factor bi trebao znacajno velik prost broj. U praksi je to dosta tesko izvesti, pa i ovdje dolazi do gubitka na preciznosti.

Nakon izracunavanja matrice pojaseva sa hes vrijednostima dokumenata za svaki pojas, vrsi se popunjavanje Blumovog filtera tim vrijednostima. Hes vrijednosti dobijene u matrici su zapravi redni brojevi baketa u koje su se preslikavali elementi pojasa. A na svaki -vektor je dodat I jos I redni broj pojasa kako bi se izbjeglo poklapanje dva vektora iz razlicith pojaseva.

Za potrebe detekcije se dobijeni Blumov filter sacuva u tekstualni fajl zajedno sa seed-ovima koji su koristeni za hes funkcije.

## Detekcija

Prilikom detekcije, neophodno je koristiti isti broj pojaseva , permutacija (sa istim seed-ovima) , veliki prosti broj koji predstavlja velicinu Blumovog filtera, kao I konstantu Test dokumenti moraju imati isti broj singlova kao I trening dokumenti.

Prvi korak je citanje sacuvanog trening modela. Detekcija se vrsi provjerom da li se u Blomovom filter nalazi jedinica na pozicijama dobijenim nakon vrsenja istih transformacija nad test skupom kao I nad trening skupom.

Dakle, detekcije se preadje input tekstualni fajl koje je karakteristicna matrica test skupa. Vrsi se kalkulacija matrice potpisa. Nakon toga se nad matricom potpisa vrsi dijeljenje u pojaseve a zatim se provjerava Blumov filter za dobijene vrijednosti.

# Testiranje

Za potrebe testiranja napisan je program *generator.py* koji generise trening I test podatke u formi tekstualnih fajlova koji sadrze karakteristicne matrice.

## Prvi test

Trening podaci su generisani funkcijom prikazanom na slici 4.

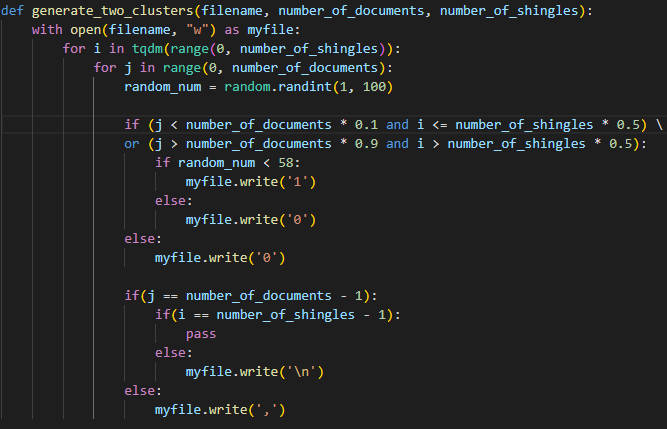


Figure 4: Generator dva gusta klastera

Dok su testni podaci generisani kaon a slici 5.

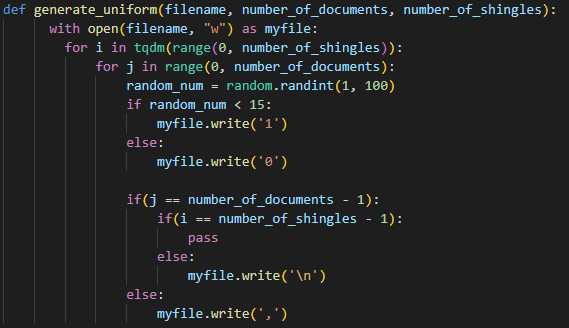


Figure 5:Generator testnih podatka

Zadati parametri su bili:

* Broj trening dokumenata – 500
* Broj test dokumenata – 50
* Broj šinglova – nasumican broj izmedju 2000 I 4000

Glavni program se nalazi u skripti *main.py* koja izgleda kao na slici 6, sa koje se vidi zadati broj permutacija, pojaseva I broja redova u pojasu.

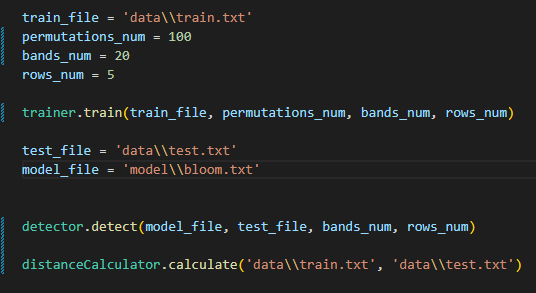


Figure 6: Glavni program

Skripta *distanceCalculator.py* sluzi za racunanje stvarne Jaccard-ove udaljenosti dokumenata. Napisana je tako da za zadati prag slicnosti *s* vrati vektor *b.* Vektor *b* ima coordinate koje predstavljaju broj trening dokumenata koji su imaju slicnost vecu od *s* sa *i*-tim test dokumentom*.*

Prvi test izveden nad dokumentima koji su generisani uniformno daje ocekivane rezultate, a to je da ni jedan od testnih dokumenata ne pripada distribuciji.

## Drugi test

Drugi test se razlikuje od prvog u tome sto ce sada testni podaci biti generisani na isti nacin kao I trening podaci, odnosno generisace se dva klastera.

# Literatura

[1] <https://scikit-learn.org/stable/modules/outlier_detection.html>

[2] <https://en.wikipedia.org/wiki/Bloom_filter>

[3] http://infolab.stanford.edu/~ullman/mmds/book.pdf