POROČILO PROJEKTA PRI PREDMETU MATEMATIČNO MODELIRANJE

# ISKANJE PO ZBIRKI DOKUMENTOV

AVTORJI:

Melanija Kraljevska
Tjaša Hočevar
Ajla Hamedović

MENTOR:

Damir Franetič

# 1. OPIS PROBLEMA

Naš problem je bil izdelati iskalnik relevantnih dokumentov po ključnih besedah z metodo latentnega semantičnega indeksiranja (LSI). To je metoda, ki zgradi model, ki združuje več besed v pojme in zato najde tudi dokumente, ki so relevantni pa ne vsebujejo iskalne besede. Naš cilj je bilo izdelati program, ki bo v dani zbirki za dane ključne besede poiskal najbolj relevantne dokumente.

# 2. OPIS REŠITVE

## 2.1 GRADITEV MATRIKE A

Najprej smo zgradili matriko A povezav med besedami in dokumenti:

$$A = [a_{ij}]$$

kjer vsak element $a_{ij}$ predstavlja frekvenco i-te besede v j-tem dokumentu. Torej vrstice matrike A predstavljajo besede v vseh dokumentih, stolpci matrike A pa predstavljajo posamezne dokumente. Ker se vsaka beseda običajno ne pojavi v vsakem dokumentu, je matrika A zelo obširna.

## 2.2 IZBOLJŠAVA MATRIKE A

Za boljše rezultate smo matriko A izboljšali tako, da je vsak element matrike A bil izračunan po postopku:

$$a_{ij} = L_{ij} \cdot G_i$$

kjer sta
$L_{ij}$ lokalna mera za pomembnost besede v posameznem dokumentu,
$G_i$ pa globalna mera pomembnosti posamezne besede.

S tem smo vsaki besedi pripisali lokalno in globalno težo, da smo zvišali oziroma znižali njeno pomembnost v dokumentih.

Lokalna mera ($L_{ij}$) je podana z logaritmom frekvence $f_{ij}$ i-te besede v j-tem dokumentu po naslednji formuli:

$$L_{ij} = \log(f_{ij} + 1)$$

Globalna mera pa je izračunana s pomočjo entropije po formuli:

$$G_i = 1 - \sum_j \frac{p_{ij} \cdot \log(p_{ij})}{logn}$$

kjer so:
$f_{ij}$.....že izračunane vrednosti v prvotni matriki A
(torej frekvence pojavitve i-te besede v j-tem dokumentu)
n .....število dokumentov v zbirki
$p_{ij}$.....razmerje med frekvenco pojavitve besede i v j-tem dokumentu in frekvence besede v celotni zbirki izračunano po formuli:

$$p_{ij} = \frac{f_{ij}}{gf_i}$$

kjer je $gf_i$ frekvenca besede v celotni zbirki.

## 2.3 SVD RAZCEP MATRIKE A

Izboljšano matriko A smo nato razcepili z odrezanim SVD razcepom:

$$A_k = U_k S_k V_k^T$$

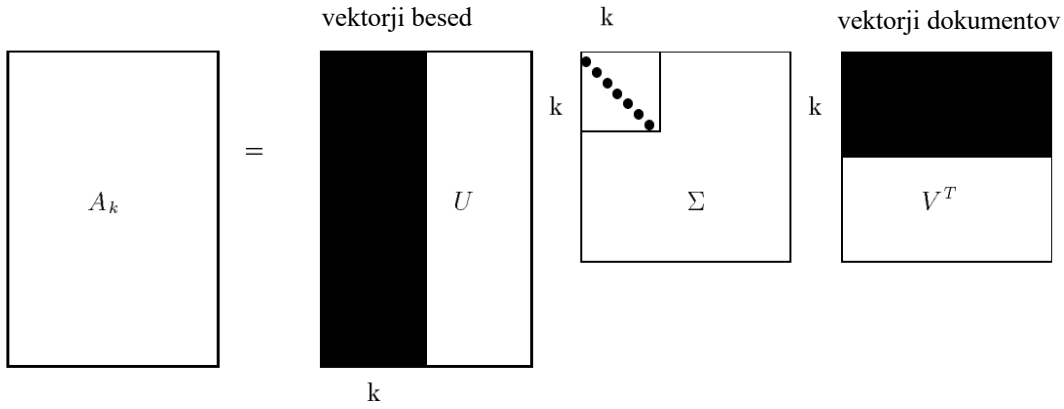
ki obdrži le k največjih singularnih vrednosti.

Stolpci matrike $U_k$ predstavljajo leve lastne vektorje matrike A, stolpci matrike $V_k$ desne lastne vektorje matrike A, diagonalna matrika $S_k$ pa vsebuje prvih k največjih pripadajočih lastnih vrednosti matrike A.

Odrezan SVD razcep zmajša t.i. »overfitting« (preveliko prilagoditev modela podatkom, kar povzroči povečan vpliv šuma). Torej z drugimi besedami odrezan SVD ohrani prvotno osnovno strukturo povezav med besedami in dokumenti, hkrati pa odstrani šume.

Na SVD lahko gledamo tudi kot na tehniko s katero dobimo množico nekoreliranih spremenljivk ali faktorjev, kjer je vsaka beseda in dokument predstavljena s k-dimenzionalnim vektorjem, ki vsebujeta elemente levih in desnih lastnih vektorjev matrike A. Torej matrike, ki jih dobimo s SVD razcepom (U,S in V) predstavljajo razdelitev originalnih povezav (razmerja) na linearno neodvisne vektorje.

Torej stolpci matrike $U_k$ predstavljajo k-dimenzionalne vektorje besed, stolpci matirke ${V_k}^T$ pa k-dimenzionalne vektorje dokumentov.




*Slika 1. SVD razcep matrike A*

Določanje optimalnega k smo ugotovili, da je bolj ''izkustvene'' narave. Torej ni predefiniran ampak je definiran na podlagi večkratnega poizkušanja, testiranja na različnih vrednostih k-ja. Za obsežne primere, torej večje zbirke dokumentov, se je izkazalo, da optimalen k variira med vrednostima 100 in 300. Takšna velika dimenzionalna redukcija dokazuje moč odrezanega SVD razcepa za uporabo pri stiskanju podatkov. Vrednost k pri 100 smo uporabili tudi v naši rešitvi.

## 2.4 POIZVEDOVANJE Z ISKALNIM NIZOM

Da lahko po dokumentih poizvedujemo, mora biti iskalni niz zapisan kot k-dimenzionalni vektor predstavljen na enak način kot vektorji dokumentov, nato pa primerjan z vsakim vektorjem dokumentov.

Iskalni vektor oziroma  vektor v prostoru dokumentov je predstavljen z naslednjo formulo:

$$q_1 = q^T U_k {S_k}^{-1}$$

kjer je $q$ vektor besed, enakih dimenzij kot posamezni stolpec v matriki A.

Za iskanje relevantnih dokumentov smo uporabili postopek primerjanja iskalnega vektorja $q_1$ z vsemi vektorji dokumentov (stolpci matrike ${V_k}^T$). Mera te podobnosti je bil kosinus kota med obema vektorjema ($q_1$in vsakim stolpcem). Ko se je kosinus bližal 1, pomeni da sta si bila vektorja zelo podobna, če pa se je bližal -1, pa pomeni da sta si različna.

Po primerjanju iskalnega vektorja z vsemi vektorji dokumentov poizvedba vrne dokumente, za katere je kosinus večji od izbrane mejne vrednosti. Pri testiranju smo preiskusili različne mejne vrednosti (npr. 0.5, 0.7, in 0.9). Ugotovili smo, da se nam najboljši rezultati zgodijo pri mejni vrednosti 0.5, zato smo to mejno vrednost uporabili tudi pri naši rešitvi.

## 2.5 DODAJANJE DOKUMENTOV IN BESED (brez zagotovitve ortogonalnosti)

Ker je izračun SVD razcepa matrike A zelo zamuden in drag proces, torej uporabi veliko časa in prostora za svoj izračun, si tega ne moremo privoščiti pri vsakokratnem dodajanju novih dokumentov v našo zbirko in pripadajočih novih besed.

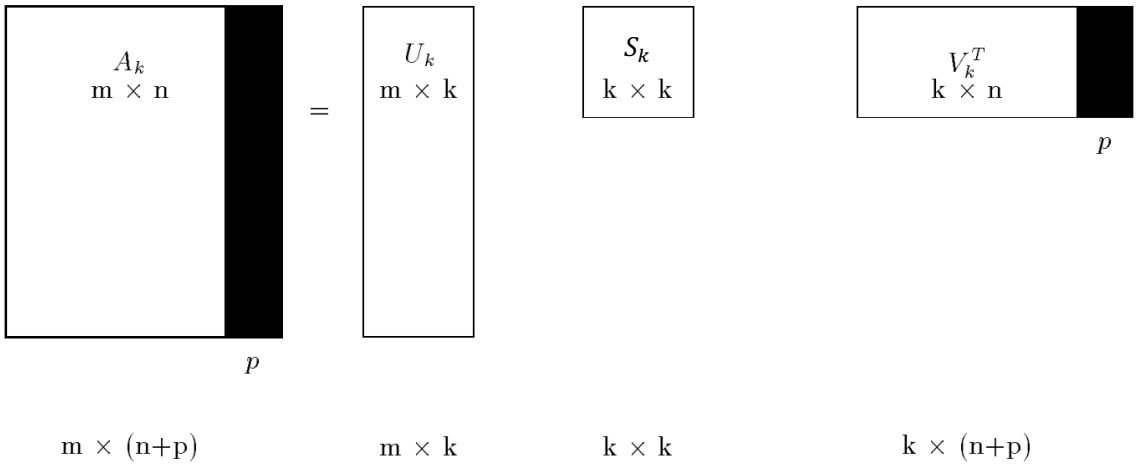
### 2.5.1 DODAJANJE DOKUMENTOV

Recimo, da bi radi dodali p novih dokumentov v našo zbirko. Potem naj bo matrika $D \in \mathbb{R}^{txp}$, kjer je t enak številu vrstic matrike A oziroma številu vseh besed v vseh dokumentih. To matriko novih dokumentov D moramo pretvoriti v k-dimenzionalni prostor po formuli:

$$D_k = D^T U_k S_k^{-1} \qquad D_k \in \mathbb{R}^{pxk}$$

kjer sta $U_k$ in $S_k^{-1}$ matriki, ki jih dobimo z odrezanim SVD razcepom prvotne matrike A (pred dodajanjem dokumentov).

Novo matriko vektorjev dokumentov dobimo tako, da prvotni matriki $V_k$ dodamo na dno matriko $D_k$. S tem ostaneta matriki $U_k$ in $S_k^{-1}$ nespremenjeni. S tem smo zagotovili, da ponoven izračun SVD razcepa ni bil potreben, ampak smo uporabili kar že izračunanega.




*Slika 2. Matematični prikaz dodajanja p dokumentov*

## 2.5.2 DODAJANJE BESED

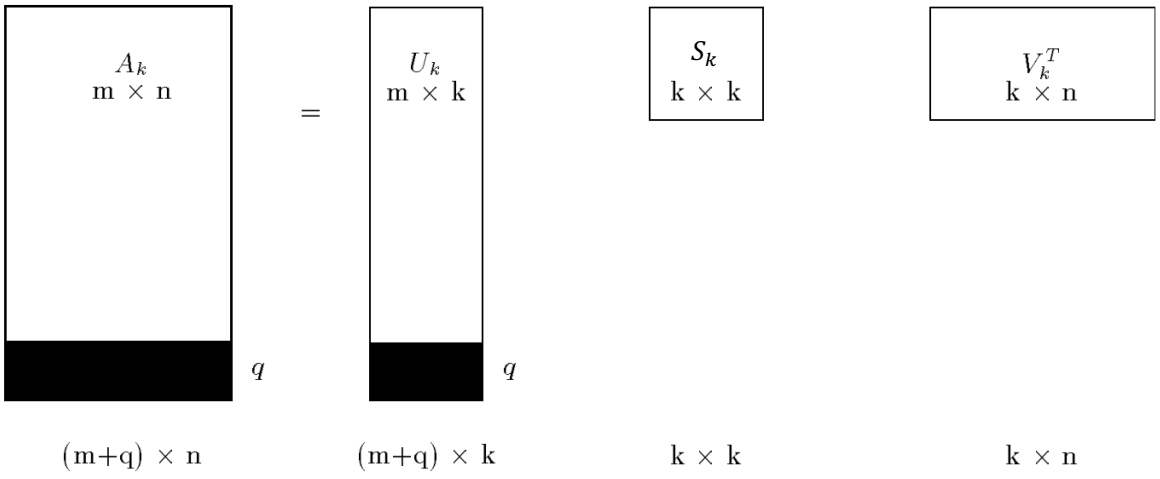
Kaj se zgodi, če novi dodani dokumenti vsebujejo še kakšno besedo več, ki ni vsebovana v matriki A? Potem te nove besede pri poizvedovanju ne bodo upoštevane, zato moramo zagotoviti še zanesljiv postopek dodajanja novih vektorjev besed v matriko $U_k$. To naredimo na enak način kot prej.

Naj bo matrika $T \in \mathbb{R}^{qxd}$, kjer je q število dodanih besed, d pa število dokumentov v naši zbirki. To matriko novih besed T moramo pretvoriti v k-dimenzionalni prostor po formuli:

$$T_k = TV_kS_k^{-1} \qquad T_k \in \mathbb{R}^{qxk}$$

kjer sta $V_k$ in $S_k^{-1}$ matriki, ki jih dobimo z odrezanim SVD razcepom prvotne matrike A (pred dodajanjem dokumentov).

Novo matriko vektorjev besed dobimo tako, da prvotni matriki $U_k$ dodamo na dno matriko $T_k$.
S tem ostaneta matriki $V_k$ in $S_k^{-1}$ nespremenjeni. S tem smo zagotovili, da ponoven izračun SVD razcepa ni bil potreben, ampak smo uporabili kar že izračunanega.




*Slika 3. Matematični prikaz dodajanja q besed*

## 2.6 DODAJANJE DOKUMENTOV IN BESED (z zagotovitvijo ortogonalnosti)

Zgoraj opisani metodi za dodajanje dokumentov in besed sta preprosti in hitri, vendar nam z dodajanjem neortogonalnih podmatrik D in T pokvarita ortogonalnost matrik $U_k$ in $V_k$. To pomeni da bo po vsakem dodajanju dokumentov oziroma besed, rezultat iskanja manj natančen.

Obstaja postopek ki zagotavlja ortogonalnost matrik pri posodabljenju SVD-ja. Ta postopek je sestavljen iz treh korakov: dodajanje novih dokumentov, dodajanje novih besed ter popravljanje uteži besed.

Naj matriki D in T predstavljata isti že opisani matriki v prešnjem postopku.
Definiramo še matriko

$$C = \begin{bmatrix} A_k \\ T \end{bmatrix}$$

ki predstavlja matriko po dodajanju q besed

in matriko

$$B = [A_k \mid D]$$

ki predstavlja matriko po dodajanju p dokumentov.

Če ima matrika A m vrstic in n stolpcev, za spremembo uteži v j izrazih naj bo $Y_j$ matrika dimenzije m x j, sestavljena iz vrstic ničel ali vrstice matrike identitete j-tega reda $I_j$, ki ji rečemo tudi matrika permutacij.
Naj bo $Z_j$ matrika dimenzije n x j, katere stolpci določajo dejanske razlike med starimi in novimi utežmi za vsak izraz j.

Posodabljanje matrike $A_k$ je definirano z naslednjo enačbo, ki opisuje korak popravka:

$$W = A_k + Y_j {Z_j}^T$$

### 2.6.1 POSODABLJANJE DOKUMENTOV

Radi bi izračunali novi SVD matrike B ne da bi ponovno izračunali celotni SVD razcep. Torej iščemo:

$$SVD\ (B) = U_B \Sigma_B {V_B}^T$$

Velja:

$${U_k}^T B \begin{bmatrix} V_k & 0 \\ 0 & I_p \end{bmatrix} = \left[\Sigma_k \mid {U_k}^T D\right]$$

ker je: $A_k = U_k \Sigma_k {V_k}^T$.

Če označimo $F = \left[\Sigma_k \mid {U_k}^T D\right]$ in $SVD\ (F)\ =\ U_F \Sigma_F {V_F}^T$, potem kot rezultat dobimo:

$$U_B = U_k U_F, \qquad V_B = \begin{bmatrix} V_k & 0 \\ 0 & I_p \end{bmatrix} V_F$$

in velja: $\Sigma_B = \Sigma_F$.

### 2.6.2 POSODABLJANJE BESED

Podobno kot pri prešnjem postopku, iščemo:

$$SVD\ (C)\ =\ U_C \Sigma_C {V_C}^T$$

Najprej računamo:

$$\begin{bmatrix} {U_k}^T & 0 \\ 0 & I_q \end{bmatrix} CV_k = \begin{bmatrix} \Sigma_k \\ \overline{TV_k} \end{bmatrix}$$

Naj bo izraz na desni strani matrika H:

$$H = \begin{bmatrix} \Sigma_k \\ \overline{TV_k} \end{bmatrix}$$

potem velja: $SVD\ (H)\ =\ U_H \Sigma_H {V_H}^T$. Novi SVD razcep ki ga iščemo dobimo naslednji način:

$$U_C = \begin{bmatrix} U_k & 0 \\ 0 & I_q \end{bmatrix} U_H \quad V_C = V_k V_H$$

in velja: $\Sigma_H = \Sigma_C$.

# 3. REZULTATI IN ZAKLJUČKI

Za izpeljavo projekta smo uporabili program Octave. In sicer smo napisali funkcijo:

*[datoteke] = iskanje (dir, iskalniNiz)*

ki kot argumenta sprejme pot do datotečne zbirke dokumentov in iskalni niz besed ločenih s presledki, vrne pa tabelo stringov imen dokumentov, ki najverjetneje vsebujejo iskane besede.

Za mejno vrednost kosinusa kota med iskalnim vektorjem in vektorji stolpcev matrike $V_k{}^T$ smo vzeli vrednost 0.5, ker smo ugotovili, da se pri tej mejni vrednosti zgodijo najboljši rezultati.

Testiranje delovanja našega programa smo izvedli predvsem na manjših zbirkah dokumentov, zato se različnih metod optimizacije nismo poslužili. Pri tem mislimo predvsem na to, da bi se lahko izognili vsakokratnemu ponovnemu računanju matrike A in SVD razcepa, ki nam vzame veliko časa, kot to vsakokrat dela naša koda. Prav zaradi tega nismo vključili tudi možnosti dodajanja dokumetov in besed brez računanja ponovnega SVD razcepa, saj naš program predvidi, da bo to naredil v vsakem primeru, tudi če v zbirki zasledi nove dokumente.

Za vrednost k, torej vrednosti pri kateri smo delali odrezani SVD razcep, so vzeli vrednost 100, saj smo po večkratnem testiranju ter iz različnih virov zasledili, da se najboljši rezultati zgodijo ravno pri tej vrednosti.

# 4. BIBLIOGRAFIJA

[1] M. W. Berry, S.T. Dumais, G.W. O'Brien, Michael W. Berry, Susan T. Dumais, and Gavin. Using linear algebra for intelligent information retrieval. SIAM Review, 37:573–595, 1995.

[2] M. W. Berry, S. T. Dumais, and T. A. Letsche. Computational methods for intelligent information access, 1995. Presented at the Proceedings of Supercomputing.

[3] Cornell SMART System ftp://cs.cornell.edu/pub/smart.

[4] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. Journal of the American Society of Information Science, 41(6):391– 407, 1990.

[5] G. W. O'Brien. Information tools for updating an SVD-encoded indexing scheme, 1994. Master's Thesis, The University of Knoxville, Tennessee.

[6] H. Zha and H. D. Simon. On updating problems in latent semantic indexing. SIAM J. Sci. Comput., 21(2):782–791, 1999.