

Metody Obliczeniowe w Nauce i Technice

Laboratorium 6

Wyszukiwarka

Patryk Wojtyczek

1 Opis Implementacji

Do uzyskania zbioru dokumentów przeglądam linki na wikipedii poczynawszy od <https://en.wikipedia.org/wiki/Science> aż do uzyskania 10000 artykułów. Zatrzymałem się na 10 tysiącach, gdyż zseralizowana do pliku macierz rzadka waży już 50Mb. Po sparsowaniu uzyskanych dokumentów html wyciągam z nich słowa (ignorując słowa pomocnicze), a właściwie ich ‘stemy’ do czego wykorzystuję Porter Stemming Algorithm. Otrzymuję w ten sposób ‘bag of words’ który liczy ≈ 44 tysiące słów. Następnie dla każdego dokumentu wyznaczam słownik słów z częstotliwością, z którego tworzę wektor d , który pod i -tą składową zawiera ilość wystąpień i -tego słowa w bag of words.

Z otrzymanych wektorów konstruuje macierz rzadką w taki sposób, że d_j to j -ta kolumna. Aby zredukować znaczenie często występujących słów przemnażam otrzymane wartości przez inverse-document-frequency.

$$IDF(w) = \log \frac{N}{n_w}$$

N - liczba dokumentów,

n_w - liczba dokumentów w którym występuje słowo.

Zatem wartość w i -tym wierszu i j -tej kolumnie zawiera częstość wystąpienia i -tego słowa w j -tym dokumencie przemnożona przez $IDF(i\text{-te słowo})$. Normalizuję wektory d_j i stosując SVD i low-rank-approximation przy eksperymentalnie wybranej wartości $k = 100$ obliczam macierz A_k .

Zapytanie w postaci kilku słów jest wektoryzowane tzn. tworzę znormalizowany wektor q , który zawiera wartości na współrzędnych odpowiadającym słowom z bag of words. Jako miarę podobieństwa między zapytaniem a dokumentem wykorzystuję kąt między wektorem q i d_j

$$\cos(\theta_j) = \frac{q \cdot d_j}{\|q\| \|d_j\|} = q \cdot d_j$$

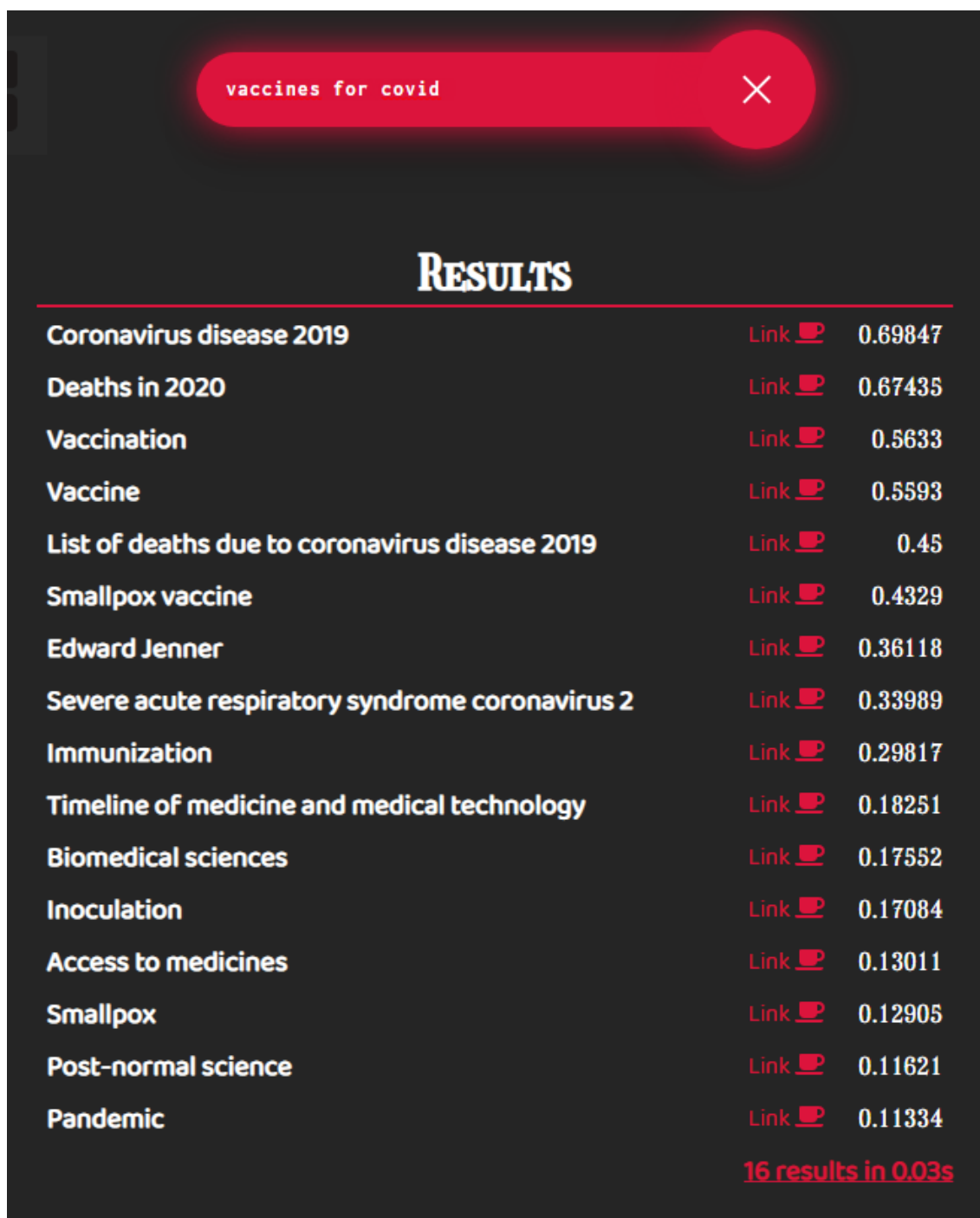
Jako, że wektory q i d_j zostały znormalizowane możemy pominąć mianownik.

Powyższe działanie łatwo uogólnić na całą macierz:

















$$[\cosines] = \mathbf{q}^T \mathbf{A}$$

Jako wynik zwracam k dokumentów z największymi wartościami cosinusa (czyli najmniejszej wartości kąta).

2 Przykład działania



The screenshot shows a search interface with a dark background. At the top, a red pill-shaped button contains the text 'vaccines for covid' and a close icon (X). Below this, the word 'RESULTS' is displayed in a large, white, serif font. A table of results follows, with each row containing a search term, a 'Link' icon (a red cup), and a numerical score. The results are sorted in descending order of score. At the bottom of the table, a red text label indicates '16 results in 0.03s'.

RESULTS		
Coronavirus disease 2019	Link 	0.69847
Deaths in 2020	Link 	0.67435
Vaccination	Link 	0.5633
Vaccine	Link 	0.5593
List of deaths due to coronavirus disease 2019	Link 	0.45
Smallpox vaccine	Link 	0.4329
Edward Jenner	Link 	0.36118
Severe acute respiratory syndrome coronavirus 2	Link 	0.33989
Immunization	Link 	0.29817
Timeline of medicine and medical technology	Link 	0.18251
Biomedical sciences	Link 	0.17552
Inoculation	Link 	0.17084
Access to medicines	Link 	0.13011
Smallpox	Link 	0.12905
Post-normal science	Link 	0.11621
Pandemic	Link 	0.11334
16 results in 0.03s		

















Rysunek 1: Znalezione wyniki dla zapytania 'vaccines for covid'

















Aplikacja umożliwia zaznaczenie czy chcemy aby wyszukiwanie odbywało się z użyciem svd, a także podania liczby rezultatów jaką chcemy otrzymać (wartość pomiędzy 1 a 200). Możemy także obliczyć svd jeszcze raz dla podanej wartości 'k-value-approx'.

3 Porównanie

Używanie svd ma kilka zalet:

1. Znacznie zmniejsza rozmiar naszej macierzy.
2. Usuwamy noise tzn. wystąpienia pojedynczych słów w dokumencie stają się nieistotne.
3. Przez zmniejszenie wymiarowości naszej macierzy na wartość każdej komórki składa się teraz waga z większej ilości słów (przy oryginalnej macierzy na komórkę składało się tylko pojedyncze słowo). Działanie to może połączyć słowa które mają podobne znaczenie - synonimy. Jest to oczywiście jak najbardziej pożądaną cechą bo chcemy dostawać nie tylko dokumenty, które zawierają dokładnie słowa, których wyszukujemy ale także dokumenty zawierające ich synonimy.

RESULTS		
Doctorate	Link 	0.80323
Doctor of Philosophy	Link 	0.67508
Ijaza	Link 	0.5475
Ijazah	Link 	0.5475
Post-graduate	Link 	0.34441
Doctor of Medicine	Link 	0.33849
Academic degree	Link 	0.33266
Doctor-patient relationship	Link 	0.30903
Occamism	Link 	0.21424
Physician	Link 	0.21343
Physicians	Link 	0.21343
Scotism	Link 	0.20923
General practice	Link 	0.18375
Diego de Torres Villarroel	Link 	0.18292
Business Administration	Link 	0.17499
List of medieval European scientists	Link 	0.15188
16 results in 0.344s		

RESULTS		
Medical school	Link 	0.08952
Doctor of Medicine	Link 	0.08787
Medical education	Link 	0.08033
Brooklyn College	Link 	0.07357
Royal College of Surgeons of England	Link 	0.0723
Royal College of Physicians	Link 	0.07162
Oxford University	Link 	0.07095
University of Oxford	Link 	0.07094
Columbia University	Link 	0.06682
Physician	Link 	0.06629
Physicians	Link 	0.06629
Al-Mustansiriya University	Link 	0.0636
Residency (medicine)	Link 	0.06099
Continuing medical education	Link 	0.06082
Medicine	Link 	0.06018
Interdisciplinary sub-specialties of medicine	Link 	0.06012
16 results in 0.096s		

ównanie wyników surowej wyszukiwarki (po lewej) i wyszukiwarki z lsi (po prawej), dla słowa 'doctor'

Możemy porównać zachowanie się wyszukiwarek przez wyszukanie np. słowa 'doctor'. Sprawdzając wartości najlepszego rezultatu wg surowej wyszukiwarki w artykule Doctorate znajduje się około 400-tu wystąpień słowa doctor, natomiast dla wyszukiwarki z svd otrzymujemy medical school gdzie odmian słowa doctor jest tylko około 100. Pokazuje to, że svd wyszukuje nie po prostu na liczbę wystąpień danego słowa natomiast bierze też pod uwagę słowa, które podczas aproksymacji uznało za bliskie semantycznie i najwidoczniej najwięcej takich słów jest w artykule Medical School, co wydaje się skądinąd rozsądnym wynikiem.

4 Detale techniczne

Na całość składa się kilka programów:

1. server.py - plik uruchamia aplikację na porcie 5000. Program zamiast uruchamiać preprocessing, czytuje uprzednio zserializowane do plików macierze. Wysłany projekt zawiera już te pliki więc nie trzeba uruchamiać scraper'a ani SearchEngine.
2. scraper.py - skrypt do scrapowania wikipedii, który pobierze do folderu resources/data zadaną liczbę artykułów. Aby go uruchomić należy wpisać 'scrapy runspider scraper.py'.
3. SearchEngine.py - plik zawiera między innymi skrypt do preprocessingu plików z resources/data. Do jego uruchomienia wystarczy wpisać 'python SearchEngine.py'. Należy zaznaczyć, że jest to dość czasochłonne - preprocessing 10 tys dokumentów zajmuje między 30 minut a 1h.
4. web/static - klient został napisany w angularze. Nie wysłałem kodu źródłowego, a jedynie zbudowane statyczne pliki.

Ostatecznie, żeby zobaczyć działające demo wystarczy w rootcie projektu uruchomić polecenie 'python server.py' i otworzyć przeglądarkę na <http://localhost:5000/>.