



Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych



Automatyczna kategoryzacja tematyczna tekstów przy użyciu metryk w przestrzeni ciągów znaków

Natalia Potocka
Warszawa, 21.04.2014

- Cel pracy
- O metrykach słów kilka
- Postęp prac
- Co dalej?

Celem pracy jest skategoryzowanie tekstów z polskiej Wikipedii pod względem tematu na podstawie liczności słów występujących w tekście. Można się spodziewać, że jeśli w dwóch tekstach występuje dużo podobnych do siebie słów, to pochodzą one z tej samej kategorii tematycznej.

Celem pracy jest skategoryzowanie tekstów z polskiej Wikipedii pod względem tematu na podstawie liczności słów występujących w tekście. Można się spodziewać, że jeśli w dwóch tekstach występuje dużo podobnych do siebie słów, to pochodzą one z tej samej kategorii tematycznej.

| A | | B | | C | | D | |
|-------------|----|-------------|----|-------------|----|-------------|---|
| całka | 10 | całka | 5 | niewłaściwe | 3 | ułamek | 4 |
| po pochodna | 5 | po pochodna | 15 | powieść | 7 | mianownik | 5 |
| niewłaściwa | 4 | granica | 7 | granica | 15 | niewłaściwy | 6 |

Co ze słowami podobnymi?

Przykładowo słowa *niewłaściwy* i *niewłaściwa* mają ten sam temat, różnią się tylko rodzajem (męski / żeński). W tekstach mogą też występować błędy ortograficzne, błędy spowodowane brakami znaków diaktrycznych (ą, ę, ł, ...) itd. Takie słowa również chcielibyśmy traktować jako „podobne”. W celu określenia jak bardzo dwa słowa są do siebie podobne, posłużą *metryki określone na napisach*.

Metryki oparte na operacjach edytowania zliczają liczbę operacji potrzebnych do przetworzenia jednego napisu w drugi. Najczęściej wymienianymi operacjami są:

- zamiana znaku, np. $'ala' \rightarrow 'ela'$
- usunięcie znaku, np. $'ala' \rightarrow 'aa'$
- wstawienie znaku, np. $'ala' \rightarrow 'alka'$
- transpozycja dwóch przylegających znaków, np. $'ala' \rightarrow 'laa'$

Metryki oparte na operacjach edytowania zliczają liczbę operacji potrzebnych do przetworzenia jednego napisu w drugi. Najczęściej wymienianymi operacjami są:

- zamiana znaku, np. $'ala' \rightarrow 'ela'$
- usunięcie znaku, np. $'ala' \rightarrow 'aa'$
- wstawienie znaku, np. $'ala' \rightarrow 'alka'$
- transpozycja dwóch przylegających znaków, np. $'ala' \rightarrow 'laa'$

Przykładowe metryki: Hamminga, najdłuższego wspólnego podnapisu (*longest common substring*), Levenshteina, optymalnego dopasowania napisów (*optimal string alignment*), Damareu-Levenshteina.

Metryka **najdłuższego wspólnego podnapisu**, ozn. d_{lcs} , zlicza liczbę usunięć i wstawień, potrzebnych do przetworzenia jednego napisu w drugi.

Np. $d_{lsc}('leia', 'leela') = 3$, bo $leela \xrightarrow{us. e} lela \xrightarrow{us. l} lea \xrightarrow{wst. i} leia$.

Metryka **najdłuższego wspólnego podnapisu**, ozn. d_{lcs} , zlicza liczbę usunięć i wstawień, potrzebnych do przetworzenia jednego napisu w drugi.

Np. $d_{lsc}('leia', 'leela') = 3$, bo $leela \xrightarrow{us. e} lela \xrightarrow{us. l} lea \xrightarrow{wst. i} leia$.

Odległość Levenshteina, ozn. d_{lv} zlicza sumę usunięć, wstawień oraz zamian znaków, potrzebnych do przetworzenia jednego napisu w drugi.

Metryka **najdłuższego wspólnego podnapisu**, ozn. d_{lcs} , zlicza liczbę usunięć i wstawień, potrzebnych do przetworzenia jednego napisu w drugi.

Np. $d_{lsc}('leia', 'leela') = 3$, bo $leela \xrightarrow{us. e} lela \xrightarrow{us. l} lea \xrightarrow{wst. i} leia$.

Odległość Levenshteina, ozn. d_{lv} zlicza sumę usunięć, wstawień oraz zamian znaków, potrzebnych do przetworzenia jednego napisu w drugi.

np. $d_{lv}('leia', 'leela') = 2$, bo $leela \xrightarrow{us. e} lela \xrightarrow{zm. l \text{ na } i} leia$.

Metryka **optymalnego dopasowania napisów**, ozn. d_{osa} , zlicza liczbę usunięć, wstawień, zamian oraz transpozycji przylegających znaków, potrzebnych do przetworzenia jednego napisu w drugi. Np.

$d_{osa}('leia', 'leela') = 2$, bo $leela \xrightarrow{us. e} lela \xrightarrow{zm. l na i} leia$.

Co zostało zrobione?

- wczytano 1 075 568 artykułów z polskiej Wikipedii

Co zostało zrobione?

- wczytano 1 075 568 artykułów z polskiej Wikipedii
- razem to 2 806 765 różnych słów...

Co zostało zrobione?

- wczytano 1 075 568 artykułów z polskiej Wikipedii
- razem to 2 806 765 różnych słów...
- ... z czego 49% wystąpiło tylko w **jednym** tekście
- ... a 44% wystąpiło tylko **jeden raz** we wszystkich tekstach

Co zostało zrobione?

- wczytano 1 075 568 artykułów z polskiej Wikipedii
- razem to 2 806 765 różnych słów...
- ... z czego 49% wystąpiło tylko w **jednym** tekście
- ... a 44% wystąpiło tylko **jeden raz** we wszystkich tekstach

Po usunięciu tzw. *stopwords*, czyli słów nieistotnych w kontekście analizy, jak np. *a, bo, co, jak, to, w, z, że*, słów jednoliterowych oraz słów w językach obcych z niełacińskiego alfabetu, pozostało 2 805 858 słów do analizy.

Z powodu słabej jakości grupowania oraz braku możliwości obliczeniowych dokonano grupowania przy pomocy tzw. *stemmingu*. Polega on na przyporządkowaniu do słowa jego rdzenia, a więc takiej jego części, która jest odporna na odmiany przez rodzaje, przyimki, przypadki itd. Przykładowo dla słowa *używająca* rdzeniem jest *żyw*.

Z powodu słabej jakości grupowania oraz braku możliwości obliczeniowych dokonano grupowania przy pomocy tzw. *stemmingu*. Polega on na przyporządkowaniu do słowa jego rdzenia, a więc takiej jego części, która jest odporna na odmiany przez rodzaje, przyimki, przypadki itd.

Przykładowo dla słowa *używająca* rdzeniem jest *żyw*.

Do stemmingu użyto narzędzia Hunspell, które sprawdza pisownię dla wielu programów, takich jak: OpenOffice, Mozilla Firefox, Thunderbird czy Google Chrome.

Dzięki niemu udało się poklastrować 733 828 słów ($\approx 26\%$ wszystkich) z czego 89% stanowiły polskie słowa 5,5% - słowa angielskie, a po ponad 2% - słowa francuskie i niemieckie. Innych języków nie sprawdzano. Liczba uzyskanów grup (klastrów) to 186 942.

Co z pozostałymi słowami?

Słowa, które wystąpiły więcej niż raz we wszystkich tekstach, dołączono do już istniejących grup przy pomocy metryk. Takich słów było 973 855, co dało łącznie pogrupowanych słów w liczbie 1 707 683. Co więcej, grupy, które miało mało słów (między 1 a 5) połączono lub dołączono do innych zbiorów. W ten sposób uzyskano 13 różnych zbiorów grup słów.

Następnie dla próbki tekstów z trzech kategorii: matematyka, historia sztuki oraz wojny, dokonano grupowania artykułów. Kryterium była liczność **grup słów** występujących w danym tekście. Do grupowania użyto metody *sferycznych k -średnich*.

Opierając się na kategoriach z Wikipedii, poprawnie sklasyfikowanych zostało 61% z 59 403 artykułów.

| tytuł | kat | id_kat | kl |
|-------------------------------------|-----|--------|----|
| kościół św. rocha w poznaniu | szt | 1 | 1 |
| portret | szt | 1 | 2 |
| quantum of solace (gra komputerowa) | szt | 1 | 2 |
| kurka wodna (seria gier) | szt | 1 | 2 |
| technika macierzy rzadkich | mat | 2 | 2 |
| kryterium walda | mat | 2 | 2 |
| generalized markup language | mat | 2 | 2 |
| czesław falkiewicz | woj | 3 | 3 |
| william goodenough | woj | 3 | 3 |
| kazimierz gallas | woj | 3 | 3 |
| wacław krzywiec | woj | 3 | 3 |
| fabian aleksandrowicz | woj | 3 | 3 |

Metoda ta dała dość dobre rezultaty dla małej próbki i małej liczby tematów. Dla większej liczby tematów i większej próbki, R miał problemy z pamięcią...

Metoda ta dała dość dobre rezultaty dla małej próbki i małej liczby tematów. Dla większej liczby tematów i większej próbki, R miał problemy z pamięcią...

Stąd pomysł użycia metody mini-batch kmeans w pythonie. Dla 20 482 artykułów i 981 tematów, obliczenia trwały ok. 70s. Wyniki dokładności były trochę gorsze, tzn. ok. 59% artykułów z tego samego tematu znalazło się w tej samej grupie.

POSTĘPY PRAC

Metoda ta dała dość dobre rezultaty dla małej próbki i małej liczby tematów. Dla większej liczby tematów i większej próbki, R miał problemy z pamięcią...

Stąd pomysł użycia metody mini-batch kmeans w pythonie. Dla 20 482 artykułów i 981 tematów, obliczenia trwały ok. 70s. Wyniki dokładności były trochę gorsze, tzn. ok. 59% artykułów z tego samego tematu znalazło się w tej samej grupie.

```
Loading 3 groups dataset for categories:
['sztuka', 'matma', 'wojna']
done in 19.075230s
n_samples: 20482, n_features: 96022

Clustering sparse data with MiniBatchKMeans(batch_size=1000, compute_labels=True,
init=k-means++,
    init_size=1000, max_iter=100, max_no_improvement=10,
    n_clusters=981, n_init=1, random_state=None,
    reassignment_ratio=0.01, tol=0.0, verbose=False)
done in 70.564s

Homogeneity: 0.464
Completeness: 0.592
V-measure: 0.520
Adjusted Rand-Index: 0.068
```

```
natalia@natalia:~/Text-clustering-basing-on-string-metrics/Python/python_mini_b
```

Postanowiono puścić obliczenia na jednym ze zbiorów na wszystkich artykułach (1 075 464), dla 67 969 tematów. Same dane wczytywały się 3 godziny i zajęły...

Postanowiono puścić obliczenia na jednym ze zbiorów na wszystkich artykułach (1 075 464), dla 67 969 tematów. Same dane wczytywały się 3 godziny i zajęły... 100 GB RAMu (+swap). I algorytm liczył... i liczył...

POSTĘPY PRAC

Postanowiono puścić obliczenia na jednym ze zbiorów na wszystkich artykułach (1 075 464), dla 67 969 tematów. Same dane wczytywały się 3 godziny i zajęły... 100 GB RAMu (+swap). I algorytm liczył... i liczył... Po 22 dniach stwierdziłam, że czas przerwać obliczenia.

```
natalia@natalia: ~/Text/Classification/basing on string metrics/Python/python_mini_batch 79x20
```

```
1  [|||||||||||||||||100.0%]
2  [|||||||||||||||||100.0%]
3  [|||||||||||||||||100.0%]
4  [|||||||||||||||||100.0%]
Mem[|||||||||||||||||7694/7868MB]
Swp[|||||||||||||||||99515/117658MB]
5  [|||||||||||||||||100.0%]
6  [|||||||||||||||||100.0%]
7  [|||||||||||||||||100.0%]
8  [|||||||||||||||||100.0%]
Tasks: 127, 265 thr; 10 running
Load average: 4.10 2.99 2.74
Uptime: 3 days, 07:00:40
```

| PID | USER | PRI | NI | VIRT | RES | SHR | S | CPU% | MEM% | TIME+ | Command |
|------|--------|-----|----|-------|------|------|---|------|------|---------|--------------|
| 1 | root | 20 | 0 | 33876 | 1736 | 620 | S | 0.0 | 0.0 | 2:29.90 | /sbin/init |
| 6997 | root | 20 | 0 | 77052 | 1332 | 1000 | S | 0.0 | 0.0 | 0:09.67 | /usr/sbin/cu |
| 7001 | lp | 20 | 0 | 63156 | 480 | 480 | S | 0.0 | 0.0 | 0:00.00 | /usr/lib/ |
| 2225 | rtkit | 21 | 1 | 164M | 356 | 328 | S | 0.0 | 0.0 | 0:01.21 | /usr/lib/rtk |
| 2232 | rtkit | RT | 1 | 164M | 356 | 328 | S | 0.0 | 0.0 | 0:00.43 | /usr/lib/ |
| 2231 | rtkit | 20 | 0 | 164M | 356 | 328 | S | 0.0 | 0.0 | 0:00.77 | /usr/lib/ |
| 2008 | root | 20 | 0 | 173M | 204 | 204 | S | 0.0 | 0.0 | 0:00.00 | /usr/lib/x86 |
| 2010 | root | 20 | 0 | 173M | 204 | 204 | S | 0.0 | 0.0 | 0:00.00 | /usr/lib/ |
| 1998 | colord | 20 | 0 | 294M | 648 | 648 | S | 0.0 | 0.0 | 0:00.16 | /usr/lib/col |

```
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice +F9Kill F10Quit
```

Kolejne testy wykazały, że bardzo duża liczba tematów powoduje, że obliczenia przetwarzają się w nieskończoność. W tej chwili jestem na etapie zmniejszenia liczby tematów i rozproszenia obliczeń na kilka(-naście) komputerów.

- [1] Martin Kober Christian Buchta Kurt Hornik, Ingo Feinerer. Spherical k-means clustering. *Journal of Statistical Software*, 50(10):1–22, 9 2012.
- [2] Mark P. J. van der Loo. The stringdist Package for Approximate String Matching. *The R Journal*, 6:111–122, 2014.
- [3] Stefan Wild. *Seeding Non-Negative Matrix Factorizations with the Spherical K-Means Clustering*. University of Colorado, Colorado, 2002.

Dziękuję za uwagę.