ISI
dokumentacja końcowa projektu

Temat: Semantic similarity of texts, how text units are close semantically (scale: 0-4).

Autor: Mateusz Zawiślak

1. Cel projektu.

Celem projektu jest zaproponowanie oraz zaimplementowanie algorytmu, który będzie oceniał podobieństwo dwóch fragmentów tekstu (w języku angielskim) w skali od 0 do 4. Ocena 0 oznacza, że wskazane dwa teksty mówią o dwóch różnych tematach. Natomiast maksymalna możliwa do uzyskania ocena 4 wskazuje na to, że oba teksty mają bardzo podobne semantycznie znaczenie.

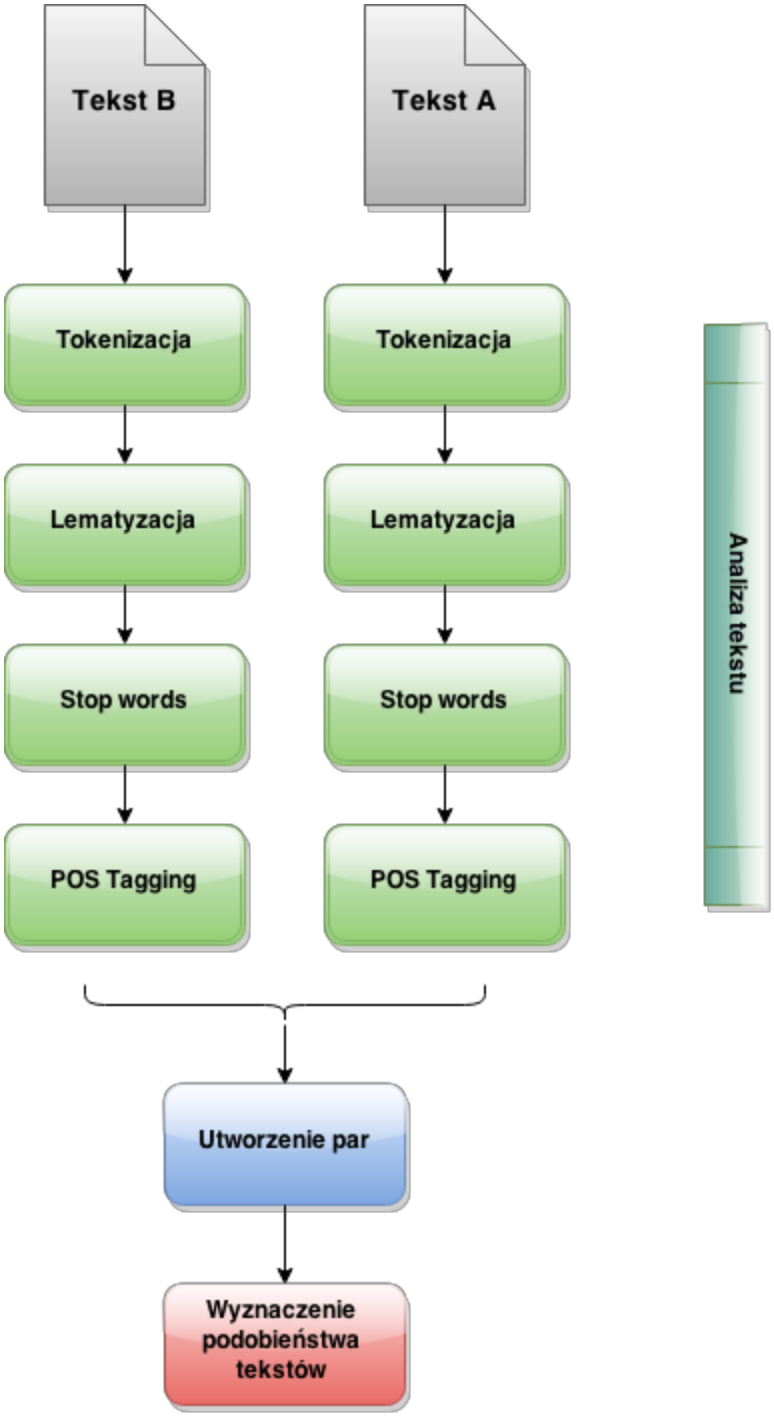
2. Algorytm znajdowania podobieństwa tekstów.

a. Wstęp.

Proponowana metoda znajdowania podobieństwa semantycznego dwóch tekstów opiera się na algorytmie zaproponowanym przez M. Lintean oraz V. Rus w artykule [1]. Ocena podobieństwa oparta jest na jakościowej jak i ilościowej mierze krótkich tekstów. Jakościowa ocena oznacza podobieństwo semantyczne poszczególnych wyrazów oparte na **zewnętrznym słowniku**. Natomiast ocena ilościowa opiera się na mierze *idf* (ang. *inverse document frequency*) informującej o częstości wystąpienia termów (wyrazów) uwzględniając jednocześnie odpowiednie wyważenie znaczenia lokalnego termu i jego znaczenia w kontekście pełnej kolekcji dokumentów.

b. Analizator tekstu.

Zanim porównywane teksty będą mogły być poddane przedstawionemu w dalszej części tego rozdziału algorytmowi konieczna jest ich wstępna analiza. Wczytany tekst był tokenizowany i odrzucane były słowa nie niosące żadnej informacji (ang. *stop words*). Ponadto został on poddany lematyzacji oraz procesowi oznaczania form mowy dla poszczególnych wyrazów (ang. *POS tagging*). Poniższy rysunek przedstawia ogólny przebieg działania programu.



Rysunek 1. Ogólny schemat przebiegu działania programu.

c. Algorytm wyznaczania podobieństwa dwóch tekstów.

Cały algorytm opiera się na mierze **podobieństwa poszczególnych wyrazów**. Oznaczmy podobieństwo wyrazu w_a oraz w_b jako:

$$WordSim(wa, wb)\ \varepsilon < 0, 1 >$$

Podobieństwo tekstu A oraz tekstu B wymaga wskazania listy wykluczających się par wyrazów najlepiej łączących teksty A oraz B. Oczywiście wszystkie możliwe pary jakie można utworzyć są wynikiem iloczynu kartezjańskiego wyrazów z tekstu A oraz wyrazów z tekstu B. Spośród tych par należy wybrać *N* par o największej wartości $WordSim$. Liczba *N* jest równa liczbie wyrazów krótszego tekstu. Dodatkowo żaden wyraz token tekstu A oraz tekstu B nie może wystąpić w więcej niż jednej parze.

Uwaga: jeżeli w którymś z tekstów dany wyraz występuje *k* razy to token zawierający taki wyraz może wystąpić w *k* parach.

Podobieństwo tekstu A do tekstu B oznaczmy będzie liczone według wzoru:

$$Sim(A \rightarrow B) = \sum_{p(wa,wb)\varepsilon S(A \rightarrow B)} WordSim(wa, wb) * Max(idf(wa), idf(wb))$$

gdzie:
$p(wa, wb)\varepsilon S(A \rightarrow B)$ to kolejne pary wyrazów łączących teksty A oraz B.

Zarówno podobieństwo semantyczne wyrazów $WordSim(wa, wb)$ jak również wartości $idf(wa)$ będą musiały pochodzić z zewnętrznej bazy opartej na zdefiniowanym korpusie dokumentów.

Ostateczny wzór na miarę podobieństwa dwóch tekstów wygląda następująco:

$$ParaSim(A, B) = (Sim(A \rightarrow B) + Sim(B \rightarrow A))/(2 * Max(normA, normB))$$

We wzorze pojawiła się jedna nowa zmienna $normA$, która jest sumą wartości *idf* dla poszczególnych wyrazów tekstu A wykorzystanych przy tworzeniu par na początku algorytmu:

$$normA = \sum_{p(wa,wb)\varepsilon S(A \rightarrow B)} idf(wa)$$

d. Dyskretyzacja wyników.

Otrzymane miary $ParaSim(A, B)$ będą należały do zakresu <0,1>. Aby uzyskać ocenę podobieństwa w dyskretnej skali od 0 do 4 konieczne będzie eksperymentalne dobranie granic poszczególnych ocen.

3. Opis implementacji.

a. Język programowania.

Przedstawiony algorytm został zaimplementowany w języku Java. **Wymaganą** wersją do uruchomienia programu jest **Java 8**.

Do początkowego etapu przetwarzania tekstu wykorzystana została zewnętrzna biblioteka Stanford CoreNLP [2].

b. Zewnętrzny słownik.

Do obsługi zewnętrznego słownika wykorzystano bibliotekę DISCO [3]. DISCO to biblioteka języka Java, który pozwala pobrać semantyczne podobieństwo dowolnych słów. Podobieństwa te są oparte na analizie statystycznej dużych zbiorów tekstowych. Wraz z biblioteką DISCO producenci dostarczają zbiór zewnętrznych słowników, na których może działać DISCO. W omawianym programie skorzystanow ze słownika enwiki-20130403-sim-lemma-mwl-lc zbudowanego na podstawie artykułów dostępnych w Wikipedii w kwietniu 2013 roku [4]. W słowniku tym znajduje się 420184 słów.

4. Opis programu.

a. Obsługa programu.

Użycie:

```
java
-Dpl.edu.pw.elka.mzawisl2.semsim.config=C:/path/semsim.properties
-jar semsim.jar -f doc1Path -s doc2Path
```

Dostępne argumenty uruchomienia programu:

`-f doc1Path` – ścieżka do pliku tekstowego z pierwszym tekstem do porównania,
`-s doc2Path` – ścieżka do pliku tekstowego z drugim tekstem do porównania.

Przykładowe uruchomienie programu:

```
java
-Dpl.edu.pw.elka.mzawisl2.semsim.config=C:/Mateusz/semsim.properties
-jar semsim.jar -f C:/Mateusz/doc1.txt -s C:/Mateusz/doc2.txt
```

b. Wczytywanie danych.

Porównywane teksty wczytywane są z dwóch plików tekstowych wskazanych w argumentach wywołania programu.

c. Plik konfiguracyjny.

Plik konfiguracyjny pozwala na określenie wielu parametrów algorytmu mierzenia podobieństwa semantycznego tekstów. Podczas wykonywania programu ścieżka do pliku

konfiguracyjnego musi zostać umieszczona w zmiennej środowiskowej `pl.edu.pw.elka.mzawisl2.semsim.config`.

Poniższa tabela przedstawia wszystkie parametry plliku konfiguracyjnego wraz z ich opisem:

Parametr	Typ	Przykład	Opis
`disco.index.dir`	`String`	`C:/\dictionaries/\wiki`	ścieżka do katalogu słownika używanego przez DISCO
`max.frequent`	`Long`	`3138674`	maksymalna częstość słowa obserwowana w danym słowniku
`pos.accepted`	`Float`	`NN,JJ,NNS,VBN,VBZ,RB,VB,VBG,VBP,JJR,FW,IN`	lista form językowych akceptowanych przez analizator tekstu oddzielonych przecinkami
`stop.words`	`String`	`Are,These,said`	lista słów nie niosących żadnego znaczenia (ang. *stop words*) oddzielonych przecinkami
`similarity.threshold`	`String`	`0.01`	minimalna granica podobieństwa słów powyżej której para słów może wejść do listy par słów łączących teksty
`rate.discretization`	`String`	`0.3,0.4,0.5,0.63`	granice kolejnych poziomów dyskretyzacji oceny podobieństwa

Przykładowy plik konfiguracyjny (*semsim.properties*):

```
#
# semsim configuration
#

# disco dictionary dir
disco.index.dir=C:/\dictionaries/\wiki2013
max.frequent=3138674

# analyzer
pos.accepted=NN,JJ,NNS,VBN,VBZ,RB,VB,VBG,VBP,JJR,FW,IN
stop.words=a,able,about,across,after,all,almost,also,am,among

# similarity
similarity.threshold=0.01

# discretization
rate.discretization=0.3,0.4,0.5,0.63
```

5. Testy.

a. Dane testowe.

Początkowo działanie programu zostało ocenione metodą “ekspercką”, czyli poprzez subiektywną ocenę otrzymanych rezultatów wystawioną przez autora omawianego programu. Na podstawie tych obserwacji dobrane zostały granice poszczególnych poziomów dyskretyzacji ocen podobieństwa tekstów:

```
0 - do 0.3
1 - do 0.4
2 - do 0.5
3 - do 0.6
4 - 0.6 i więcej.
```

Po ustaleniu progów dyskretyzacji przeprowadzono testy na danych pochodząc z zadania 3. konkursu SemEval-2014 [5,6]. Zadanie postawione przed uczestnikami tego konkursu było bardzo zbliżone do omawianego problemu. W danych testowych znajdują się pary tekstów wraz z wystawionymi, subiektywnymi ocenami semantycznego podobieństwa tych tekstów. Niestety wystawiane tam oceny mogły być połowiczne (np. ocena 3.5). Mimo wszystko testy przeprowadzone na tych danych mogą dać ogólny obraz jakości działania niniejszego programu.

b. Wyniki testów.

Nazwa testu	Oczekiwana ocena	Ocena	Różnica ocen
Travel-2	3	3,00	0
Travel-3	0	0,00	0
Travel-4	2,5	4,00	1,5
Travel-5	3	3,00	0
Travel-6	2,5	4,00	1,5
Travel-7	0,25	0,00	0,25
Travel-8	1	0,00	1
Travel-9	0,5	0,00	0,5
Travel-10	0	0,00	0
Travel-11	2,5	4,00	1,5
Travel-12	0,5	0,00	0,5
Travel-13	4	4,00	0

Travel-14	2	4,00	2
Travel-15	4	4,00	0
Travel-16	0	0,00	0
Travel-17	2	3,00	1
Travel-18	0,5	1,00	0,5
Travel-19	2	1,00	1
Travel-20	0,75	0,00	0,75
Travel-21	0	0,00	0
Travel-22	0,5	0,00	0,5
Travel-23	2,5	2,00	0,5
Travel-24	0,5	1,00	0,5
Travel-25	3,5	4,00	0,5
Travel-26	0	0,00	0
Travel-27	2	3,00	1
Travel-28	0	0,00	0
Travel-29	2,5	4,00	1,5
Travel-30	3	3,00	0
Travel-31	3	4,00	1
Travel-32	3,5	3,00	0,5
Travel-33	3	1,00	2
Travel-34	0	1,00	1
Travel-35	3,75	3,00	0,75
Travel-36	2,75	3,00	0,25
Travel-37	3,38	2,00	1,38
Travel-38	3	3,00	0
Travel-39	2,5	2,00	0,5
Travel-40	2	4,00	2
Travel-41	2	4,00	2

6. Wnioski.

Otrzymane rezultaty wskazują na poprawność działania zaproponowanego algorytmu. Program bardzo dobrze rozpoznaje semantyczne podobieństwo dwóch tekstów. Oczywiście niektóre z wyników odbiegają odrobinę od idealnych, oczekiwanych rezultatów. Jednak generalnie rzecz biorąc program poprawnie wskazuje przybliżony stopień podobieństwa (lub jego braku) wskazanych tekstów.

Średnia różnica między otrzymaną oceną a tą oczekiwaną przez organizatorów konkursu wyniosła 0,69. Wynik ten jest bardzo dobry, tym bardziej że oceny organizatorów mogły być dokładniejsze ze względu na dopuszczalność ocen połówkowych (np. ocena 3.5).

7. Podsumowanie.

W przyjętym rozwiązaniu swoją uwagę skupiłem na pojedynczych słowach, a zupełnie nie brałem pod uwagę większych fraz. Takie założenie wynikało z próby jakiegokolwiek uproszczenia i tak skomplikowanego zadania. Na szczęście założenie to nie miało bardzo negatywnego wpływu na wyniki, ponieważ otrzymane rezultaty śmiało można uznać za zadowalające.

Dla uzyskania jeszcze lepszych rezutatów należałoby przetestować rezultaty działania programu dla innych zewnętrznych słowników obsługiwanych przez bibliotekę DISCO.

8. Literatura.

1) Mihai Lintean, Vasile Rus. *Measuring Semantic Similarity in Short Texts through Greedy Pairing and Word Semantics* http://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS12/paper/viewFile/4421/4801
2) Biblioteka Java *Stanford CoreNLP* http://nlp.stanford.edu/software/corenlp.shtml
3) Biblioteka DISCO http://www.linguatools.de/disco/disco_en.html#api
4) Słownik podobieństwa wyrazów dla języka angielskiego http://www.linguatools.de/disco/disco-languagedatapackets_en.html#enwiki13
5) Konkurs SemEval-2014 Zadanie 3, http://alt.qcri.org/semeval2014/task3/
6) Dane do zadanie 3. z konkursu SemEval-2014, http://alt.qcri.org/semeval2014/task3/index.php?id=data-and-tools