POLITECHNIKA WROCŁAWSKA
WYDZIAŁ INFORMATYKI I ZARZĄDZANIA

KIERUNEK: INFORMATYKA

Inżyniera języków naturalnych
Projekt

Analiza zastosowania różnych klasyfikatorów do problemu nienadzorowanego WSD dla języka polskiego.

AUTORZY:
Bartosz Cieśla, Bartosz Janusz, Bartosz Kardas

PROWADZĄCY:
dr inż. Maciej Piasecki

OCENA PRACY:

WROCŁAW, 2018

Spis treści

Rozdział 1

Wstęp

1.1. Opis problemu

Ujednoznacznianie znaczeń słów (ang. word sense disambiguation) jest jednym z kluczowych problemów analizy semantycznej. Polega na wyznaczeniu znaczenia danego słowa w kontekście, tak aby cały kontekst był spójny. Znaczenia słów opisane są przez tak zwane synsety (zbiory synonimów, ang. synonyms sets). W tym celu stosuje się algorytmy nadzorowane i nienadzorowane. W algorytmach opartych o uczenie nadzorowane stosowane są metody uczenia maszynowego np. sieci neuronowe lub wektory maszyn wspierających. W uczeniu nienadzorowanym stosowane są zewnętrzne źródła wiedzy np. słowniki, słowosieci, grafy synsetów itp.

1.2. Idea metody

W niniejszej pracy zaprezentowano rozwiązanie oparte o algorytm nienadzorowanym. Wykorzystano słowosieć oraz graf synsetów. W tym celu zaimplementowano rozszerzenia do platformy WoSeDon, która realizuje problem ujednoznaczniania sensów słów za pośrednictwem algorytmów opartych o PageRank. Rozszerzenia skupiają się na wykorzystaniu innych miar centralności w grafach m.in. betweenness, closeness, eigenvector. Poniżej przedstawiono szczegółowy opis metody.

1.3. Centralność w grafach

W teorii grafów wskaźniki centralności informują o najbardziej znaczących wierzchołkach grafu. Ich przykładowymi zastosowaniami mogą być: znalezienie lidera, przywódcy spośród danej grupy osób, ustalenie kluczowego elementu infrastruktury sieciowej lub miejskiej bądź znalezienie osobnika o największym potencjale do roznoszenia choroby. Istnieje wiele odmiennych wskaźników centralności. Zrealizowany projekt implementuje trzy z nich: Closeness Centrality, Betweenness Centrality oraz Pagerank (jedna z odmian Eigenvector Centrality)

1.3.1. Betweenness Centrality

Określa kluczowość wierzchołka w zakresie komunikacji - przechodność, pośredniczenie. Czyli w jakim stopniu dany wierzchołek jest spoiwem dla danej sieci. Jest to miara o bardzo wielkiej wartości, gdyż dzięki niej można znaleźć punkty krytycznej sieci bądź grafu.

Algorytm wyznaczania

1. Wyznaczyć ilość najkrótszych ścieżek między wierzchołkami u i v (d_{uv})
2. Wyznaczyć ilość najkrótszych ścieżek między wierzchołkami u i v, które przechodzą przez wierzchołek w ($d_{uv}(w)$)
3. Suma stosunków oznacza stopień centralności wierzchołka w

$$c_b(w) = \sum_{u \neq v \neq w} \frac{d_{uv}(w)}{d_{uv}}$$

1.3.2. Closeness Centrality

Jest to stopień bliskości. Określa jak blisko (daleko) wierzchołek ma do pozostałych w grafie. Wysoki stopień biskości świadczy o dobrej własności propagacji informacji w grafie - element ten szybko rozprowadzi daną wiadomość (wirusa itp) po całej sieci.

Algorytm wyznaczania

1. Wyznaczyć odległości pomiędzy wierzchołkiem u a pozostałymi wierzchołkami w grafie v (d_{uv})
2. W zależności od rodzaju grafu zsumować otrzymane odległości:
    1. Dla grafów rzadkich
    $$c_c(u) = \frac{1}{\sum d_{uv}}$$
    2. Dla grafów silnie połączonych
    $$c_c(u) = \sum_{u \neq v} \frac{1}{d_{uv}}$$

1.3.3. Eigenvector Centrality - Pagerank

Określa wpływ, oddziaływanie wierzchołka na pozostałe w grafie. Wykorzystuje nie tylko ilość połączeń danego wierzchołka z innymi, a przede wszystkim ich jakość. Wartości przypisane do każdego z wierzchołków bazują na koncepcji w której wysoko ocenione wierzchołki bardziej wpływają na ostateczną ocenę połączonego wierzchołka, niż te, których ocena jest niska. Jedną z odmian Eigenvector Centrality jest algorytm PageRank. Poniżej przedstawiono uproszczony algorytm jego działania.

Algorytm wyznaczania

1. Wyznaczyć ilość wierzchołków w grafie (N)
2. Wyznaczyć stopień każdego z wierzchołków ($l(u)$)
3. Zainicjować wartości początkowe dla każdego wierzchołka wartością początkową ($c_e(u) = 1$)
4. Określić współczynnik tłumienia, zwykle wynosi on około 0.85 ($d = 0.85$)
5. Obliczyć nową wartość PageRank każdego wierzchołka

$$c_e(u) = \frac{1-d}{N} + d \sum_{v \in B_u} \frac{c_e(v)}{l(v)}$$

B_u oznacza zbiór wszystkich wierzchołków, które odnoszą się do wierzchołka u

1.3.4. Pozostałe centralności

Centralność Katz oraz Hits są zbliżone do algorytmów takich jak Pagerank. Określają one w przybliżeniu odziaływanie poszczególnych wierzchołków na pozostałe w grafie. W przypadku Katz mierzona jest nie najkrótsza scieżka pomiędzy wierzchołkami, a względna ilość kroków potrzebnych, aby dostać się do danego celu. W przypadku algorytmu hits analizowane są tzw. autorytety oraz huby. Analiza zależności między nimi pozwala na określenie stopnia ważności poszczególnych wierzchołków.

Rozdział 2

Platforma testowa

2.1. Komponenty

Do realizacji projektu, wykorzystano szerg gotowych rozwiązań, które znacznie przyspieszyły proces implementacji. Poniżej przedstawiono istotne biblioteki, komponenty oraz elementy użyte w projekcie:

- WoSeDon - program bazowy
- Wrocław CFR Tagger - tager
- corpus2 - biblioteka dostarczająca struktur danych i metod do obsługi anotowanych korpusów.
- Korpus PWr - anotowany korpus służący do testów
- docker - środowisko uruchmieniowe WoSeDona

Głównym elementem system jest WoSeDon, program służący do ujdnoznaczniania znaczeń słów za pomocą algorytmu PageRank. Niniejszy projekt sprowadził się do zaimplementowania rozszerzeń do WoSeDona wykorzystujących inne metody pomiaru centralności.

2.2. Docker

Ze względu na dosyć żmudny proces tworzenia środowiska testowego, zadecydowano o użyciu docekra, który pozwoli na szybkie tworzenie homogenicznego środowiska na wielu niezależnych od siebie maszynach. W ramach projektu powstały dwa typy środowisk opisanych przez poniższe dockerfile'e:

- Dockerfile-arch
- Dockerfile-ubuntu

Utworzenie dwóch odrębnych dockerfile'i było wymuszone, ze względu na długi proces budowy środowiska opartego o Linkusa Arch (brak skompilowanych komponentów, długotrwałe kompilacje). Aby ominąć problem czasowy podjęto decyzję o utworzeniu środowiska opartego o Linuksa Ubuntu.

Rozdział 3

Opis metody

3.1. Oryginalna metoda - Personalized PageRank

Słowosieć stosowana jako model języka w programie Wosedon oryginalnie miała postać grafu synsetów. Metoda, która służyła do ujednoznaczniania wyrazów w zdaniu opierała się na zmodyfikowanej wersji PageRank (Personalized PageRank). Można ją w skrócie przedstawić następująco:

1. Każdemu węzłowi w sieci przypisz początkową wagę PageRank równą 0.
2. Dla każdego słowa w tekście wejściowym wylicz początkową wagę i przypisz ją do synsetów zawierających to słowo.
3. Uruchom algorytm PageRank, kryterium stopu - określona liczba iteracji lub brak znaczącej poprawy wartości.
4. Do każdego słowa z tekstu wejściowego przypisz synset z najwyższą wagą.

3.2. Modyfikacja - dodanie alternatywnych miar centralności

Podejście zastosowane przez autorów polegało na zastąpieniu algorytmu PageRank w powyższej metodzie którąś z pozostałych miar centralności. Na pierwszy ogień poszedł betweenness. W tym momencie pojawiły problemy. PageRank dzięki możliwości zdefiniowania kryterium stopu nie posiadał znaczącej złożoności obliczeniowej i wykonywał się dość szybko. Betweenness natomiast takiego kryterium nie posiada. Jak to było opisane wcześniej, wymaga on wyznaczenia najkrótszej ścieżki pomiędzy każdą parą węzłów. W przypadku grafu Słowosieci, która posiada ich około 150 tysięcy, czas wykonywania się algorytmu przekraczał znacząco uzasadnioną praktycznie wartość.

W związku z powyższym konieczne okazało się ograniczenie wielkości grafu. Autorzy po przemyśleniach zdecydowali się na wykorzystanie wyniku algorytmu PageRank. Graf z przypisanymi wartościami PageRank odfiltrowano - pozostawiono tylko te węzły, których wartość była większa od 90 percentyla wszystkich wartości. W praktyce oznaczało to pozostawienie 10% oryginalnych węzłów. Taka ilość okazała się odpowiednia i możliwa do przetworzenia przez betweenness w stosunkowo krótkim czasie.

Pojawił się następny problem. Słowosieć zawiera przede wszystkim związki hiperonimiczne, co oznacza, że graf ten jest właściwie drzewem. Po odfiltrowaniu części węzłów złamana została spójność grafu, czego większość miar centralności nie akceptuje. Autorzy mieli więc dwa wyjścia - albo sztucznie zapewnić spójność grafu (np. budując minimalne drzewo rozpinające) albo spróbować zwiększyć liczbę połączeń poziomych, których w oryginalnym grafie było jak na

lekarstwo. Pierwsze podejście nie zdobyło uznania autorów, ponieważ połączenia utworzone w ten sposób nie niosły by żadnych wartości. Pojawiła się natomiast alternatywa w postaci WordNet Domains. Domeny grupują synsety w zbiory o określonych cechach wspólnych, mogą to być przykładowo instancje obszarów wiedzy (Sport, Sztuka, Ekonomia), w skrócie pewne kategorie semantyczne. Na tej podstawie w grafie powstaje wiele dodatkowych połączeń poziomych, które uzupełniają istniejące pionowe związki hipernonimiczne. Dane o domenach pochodzące z angielskiej wersji WordNet i zmapowane na wersję polską dodano do grafu na którym dokonywano pomiarów. Badania wykazały, że zapewniło to wystarczającą liczbę połączeń i algorytmy miary centralności liczyły się poprawnie.

Na tak odfiltrowanym i wzbogaconym grafie przeprowadzono pomiary skuteczności algorytmów.

Rozdział 4

Eksperymenty

4.1. Metodologia

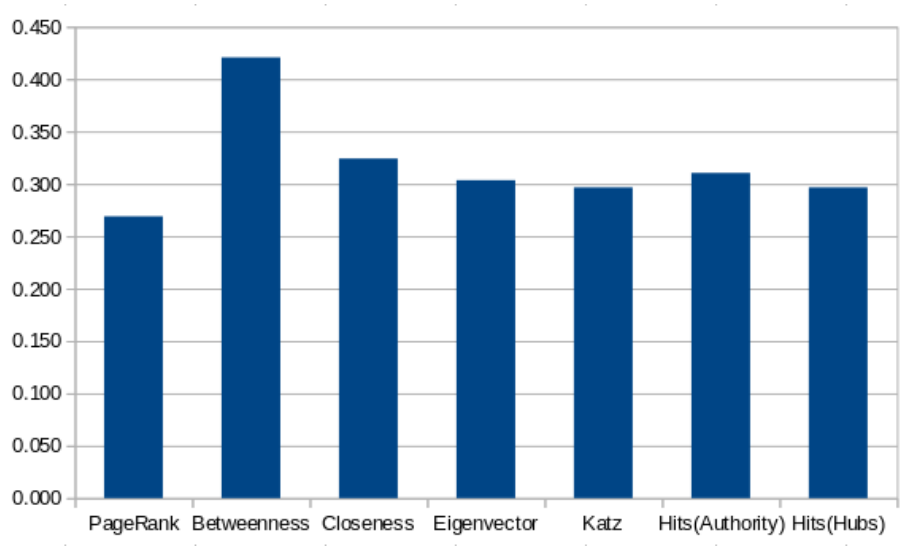
Testy przeprowadzono przy użyciu skryptu zawartego w repozytorium wosedona. Skrypt `evaluation-kpwr.py` wymaga do swojego działania połączenia z bazą danych Słowosieci. Na wejście skryptu podawany jest wcześniej przetworzony przez wosedon, anotowany tekst. Wymóg anotowanego tekstu jest spowodowany tym, że musi istnieć podstawa, na której zostanie określona poprawność ujednoznaczniania. Testy wykonano na wybranych wcześniej pięciu plikach, które zawierają najwięcej wieloznacznych wyrazów, tak aby pliki wynikowe miały jak najwięcej przetworzonych wyrazów. Każdy z siedmiu zaimplementowanych algorytmów został uruchomiony na tych pięciu plikach i już zbiorczo oznaczono ilość poprawnie ujednoznacznionych wyrazów. Do usprawnienia testowania implementacji posłużono się trzema skryptami: `runDocker` - uruchamiający kontener Dockera, `runWosedon` uruchamiający wosedon na kolejnych algorytmach na wszystkich wybranych plikach oraz `runTests`, który przetwarzał wyniki poprzedniego skryptu i oznaczał poprawnie ujednoznacznione wyrazy.

4.2. Wyniki

Poniżej przedstawiono wyniki przeprowadzonych eksperymentów.

Tab. 4.1: My caption

Nazwa	Ilość poprawnych	Ilość niejednoznacznych	Dokładność
PageRank	39	145	0.269
Betweenness	61	145	0.421
Closeness	47	145	0.324
Eigenvector	44	145	0.303
Katz	43	145	0.297
Hits(Authority)	45	145	0.310
Hits(Hubs)	43	145	0.297



Rys. 4.1: Wyniki poszczególnych metod.

Rozdział 5

Podsumowanie

5.1. Wnioski

Na zbadanych plikach można zauważyć, że najlepiej sprawdził się algorytm Betweenness. Zaimplementowany wcześniej Personalized PageRank sprawdził się zdecydowanie najsłabiej. Pozostałe algorytmy sprawowały się w miarę porównywalnie jeśli chodzi o skuteczność ujednoznaczniania. Wspomniana skuteczność to ilość poprawnie ujednoznacznionych wyrazów do wszystkich wyrazów do ujednoznacznienia. Należy wspomieć, że długość wykonania najlepszego algorytmu jest jednak bardzo długa (2,5 raza dłuższa od PageRank), co skutecznie ograniczyło ilość wybranych plików do testów. Mimo zastosowania ograniczenia grafu na którym został uruchomiony Betweenness czas ten był długi. Ograniczenie to polega na wycięciu części grafu na podstawie wyniku PageRank, co gwarantuje dłuższy czas wykonania (muszą zostać uruchomione 2 algorytmy). Z racji braku dostępu do szybszego sprzętu nie można było uzyskać lepszego pokrycia dostarczonych, anotowanych przykładów dokumentów (przykłady obejmowały ok 1600 dokumentów z których wybrano tylko 5).