Rozproszony system eksploracji korpusów tekstu działający w oparciu o ich strukturalną reprezentację

Michał Adamczyk, Dominik Majda

Akademia Górniczo-Hutnicza
Wydział Informatyki, Elektroniki i Telekomunikacji
13.12.2015

Spis treści

Rozdział 1

Cel prac i wizja projektu

W erze informacji przechowywanych w postaci elektronicznej niemożliwym wydaje się sprawna nawigacja pomiędzy różnego rodzaju tekstami bez pomocy różnego rodzaju wyszukiwarek. Umożliwiają one szybkie zawężenie dziedziny z której pochodzą dokumenty wśród których użytkownik wyszukuje te interesujące go, znacznie skracając czas niezbędny do odszukania interesujących treści. Najbardziej powszechną wyszukiwarką jest wyszukiwarka google - dla podanego w postaci słów zapytania stara się ona znaleźć najlepiej dopasowane strony internetowe.

Celem naszego projektu jest stworzenie własnej wyszukiwarki działającej na posiadanym przez nas korpusie tekstów o tematyce finansowej. Budując nasze narzędzie wyszukiwania korzystać będziemy z reprezentacji strukturalnej artykułów. Użytkownik końcowy dostanie do ręki aplikację mobilną na platformę iOS. Artykuły na których operuje aplikacja mają pochodzić z wiodących serwisów informacyjnych o tematyce finansowej jak reuters czy bloomberg. Ze względu na zawężenie dziedziny do konkretnej tematyki możliwe jest przygotowanie narzędzi wyszukiwania bardziej wyspecjalizowanych i dostosowanych do bardziej szczegółowego problemu.

Tworzone oprogramowanie jest oprogramowaniem o architekturze klient-serwer. Wszystkie obliczenia wykonywane być mają po stronie serwerowej a dostarczony klientowi interfejs ma jedynie służyć do przekazywania serwerowi zapytań oraz do wprowadzania ich przez użytkownika.

Rozdział 2

Zrealizowane funkcjonalności

W ramach prac nad systemem zrealizowane zostały wszystkie najważniejsze założone wymagania. Główne funkcjonalności, które udało się zrealizować, to:

- Stworzenie aplikacji mobilnej na platformę iOS, która jest interfejsem użytkownika dla tworzonej wyszukiwarki
- Możliwość wyszukiwania na podstawie wprowadzonego zapytania
- Możliwość wyszukiwania za pomocą podejścia 'drill-down' tj. za pomocą ustawicznego zawężania obszaru zainteresowań
- Stworzenie bazy artykułów spośród których użytkownik może wyszukać interesujące go za pomocą powyższych metod
- Możliwość wybierania metod obliczeniowych użytych po stronie serwerowej w celu redukcji wymiaru artykułów przechowywanych w postaci długich wektorów cech

Rozdział 3

Wybrane aspekty realizacji

W trakcie realizacji projektu zastosowano następujące technologie:

- język Python 2.7 - niewymagający kompilacji język dostępny na każdym systemie operacyjnym. Został wybrany ze względu na doświadczenie autorów, ekspresywność i ogromną liczbę bibliotek oraz rozbudowaną społeczność jego użytkowników.
- biblioteka LDA - biblioteka języka python. Umożliwia wygodny interfejs do obliczania LDA (ang. Latent Dirichlet Allocation) czyli probabilistycznego algorytmu redukcji wymiarów.
- scipy - biblioteka używana by przechowywać wektory lub macierze w postaci rzadkiej. W przypadku artykułów przechowywanych w postaci wektorów cech takie podejście skutkowało znacznym przyspieszeniem obliczeń
- język Swift 2.1 - młody język programowania (zaprezentowany przez Apple w roku 2014), służący m.in. do tworzenia aplikacji mobilnych na platformę iOS. Został wybrany ze względu na doświadczenie autorów oraz z uwagi na zawarte w nim godne uwagi koncepty i rozwiązania.

Rozdział 4

Komponenty systemu

4.1 Aplikacja mobilna

Aplikacja mobilna pełni rolę widoku, to znaczy sama nie wykonuje obliczeń ani nie ma dostępu do bazy danych. Komunikuje się ze stroną serwerową za pomocą zapytań, i prezentuje użytkownikowi ich rezultaty. Dostarcza ona użytkownikowi dwa sposoby na wyszukiwanie interesujących go artykułów z posiadanego ich zbioru.

Wyszukiwanie dla danego zapytania
: Wyszukiwanie pośród potencjalnych artykułów odbywa się na podstawie podanego przez użytkownika zbioru słów. Aktywność w aplikacji ogranicza się więc do wpisania tekstu, dla którego użytkownik dostać chce artykuły z tekstem tym związane tematycznie.

Wyszukiwanie poprzez zawężanie kategorii
: Użytkownik poszukując interesujących artykułów zamiast próbować je znaleźć poprzez podawanie charakterystycznych słów skupia się na generalnych kategoriach do których artykuły te mogą należeć. Aplikacja prezentuje kilkanaście grup (tzw. klastrów), każdą z nich opisując trzema kategoriami które odzwierciedlają tematykę artykułów do kolejnych zbiorów należących. Po wybraniu jednego z klastrów aplikacja prezentuje analogiczny podział o jeden stopień głębiej, biorąc pod uwagę jedynie artykuły z danej grupy. W przeciwieństwie do pierwszego wyboru, grupy od tej pory opisywane będą najczęściej występującymi w artykułach słowami. To schodzenie na coraz głębszy poziom

trwa tak długo jak użytkownik zdecyduje, że chce zobaczyć wszystkie artykuły dla danej grupy, lub aż dany klaster będzie na tyle mały, że jego dalszy podział nie ma sensu.

4.2 Strona serwerowa

Strona serwerowa działa w sposób niewidoczny dla użytkownika. Wykonuje wszystkie niezbędne operacje i obliczenia, a aplikacji wysyła jedynie gotowe odpowiedzi na zadane zapytania. Wystawione przez stronę serwerową API służy do obsługiwania nadchodzących od aplikacji klienckich zapytań. W założeniu strona serwerowa ma działać nieustannie. Spora część obliczeń musi zostać wykonana od nowa w momencie dodawania nowych artykułów do obsługiwanego korpusu, dlatego aktualizacje strony serwerowej planowo odbywać powinny się w nocy.

Czynności wykonywane przez stronę serwerową są dwojakiego rodzaju:

Preprocessing
: Czynności tego typu mogą zostać wykonane jednokrotnie i nie wymagane jest w tym celu posiadanie żadnego rodzaju informacji od użytkownika. Z tego powodu obliczenia te mogą zostać wykonane zawczasu, co powoduje przyspieszenie pracy serwera.

Obliczenia wykonywane podczas wyszukiwania
: Drugim rodzajem czynności jest wykonywanie obliczeń w odpowiedzi na zapytanie użytkownika. Ponieważ w celu ich wykonania konieczna jest znajomość kontekstu, by można było do nich przystąpić niezbędna jest interakcja ze strony użytkownika. Są one więc wykonywane niejako na żądanie. Czas ich wykonania stanowi większość okresu pomiędzy wprowadzeniem do aplikacji klienckiej zapytania a otrzymaniem wyników.

Rozdział 5

Zbiory artykułów

Podczas obliczeń przetwarzane są dwa niezależne zbiory danych:

5.1 Historyczna baza danych artyków reuters

Korpus artykułów serwisu reuters sprzed kilku lat. Artykuły te są zbliżone tematycznie z tymi obsługiwanymi przez nas, dodatkowo są opisane przez kategorie do których należą. Postanowiliśmy więc użyć ich jako wiedzy zewnętrznej i spróbować wykorzystać więdzę którą niesie kombinacja treści artykułów i kategorii do których można je przypisać.

5.2 Baza obsługiwanych artykułów

Ponieważ celem naszej wyszukiwarki jest dostarczenie informacji o tematyce finansowej, posiadane przez nas artykuły są pobrane z kilku serwisów internetowych o tejże tematyce, na przykład: reuters, telegraph czy the street. Artykuły docelowo powinny być także pobierane na bieżąco, by aplikacja była atrakcyjna dla użytkownika.

Rozdział 6

Wyszukiwanie za pomocą łańcucha słów

Pierwszym rodzajem wyszukiwania jest realizowanie zapytań w postaci zbioru słów - analogicznie jak w wyszukiwarce google. Użytkownik wprowadza więc pewne frazy, które jego zdaniem dobrze opisywać będą interesujące go artykuły a rolą aplikacji jest te artykuły odnaleźć. Poniższy rozdział opisuje kolejne wykonane kroki (także zanim użytkownik dane zapytanie wprowadza) które wykonane zostały by aplikacja mogła na zapytania odpowiadać.

6.1 Preprocessing

Preprocessing to wszelkie czynności wykonane w celu ekstrakcji jak największej ilości informacji z posiadanych danych, które nie zależą od wykonanych przez użytkownika czynności (z tego też powodu można je wykonać zawczasu - potrzebne są tylko te dane w których posiadaniu już jesteśmy)

6.1.1 Czyszczenie artykułów

By w sposób wygodny i wydajny pracować z artykułami, niezbędne jest wypracowanie dla nich spójnego i efektywnego sposobu reprezentacji, który równocześnie zachowywałby jak największą dawkę wyekstrahowanych informacji. W celu dobrej ekstrakcji informacji posiadane artykuły są najpierw ćzyszczone- usuwana jest z nich jak największa ilość szumu, czyli zbędnie

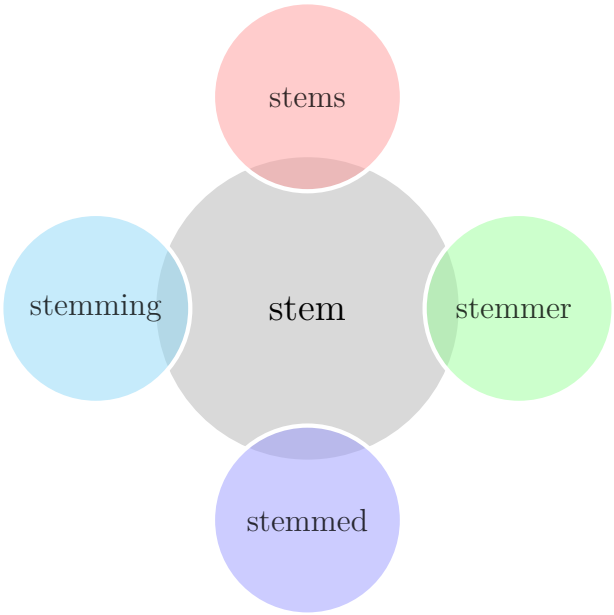
przechowywane informacje. Operacja ta dokonana jest w nastepujących krokach:

Stop words removal

Pierwszym krokiem w procesie czyszczenia artykułów jest usunięcie z niego wszystkich słów klasyfikowanych jako tzw. stop words. Są to słowa takie jak spójniki czy inne słowa które same w sobie nie przenoszą konkretnego znaczenia i z punktu widzenia reprezentacji strukturalnej artykułów nie niosą żadnej wartości. Usunięcie ich powoduje zatem zagęszczenie słów przenoszących informacje w ramach danego artykułu.

Stemming

Jest to proces sprowadzania słowa do jego rdzenia. Operacja ta powoduje, że spokrewnione wyrazy będące różnymi częściami mowy zostają skrócone do jednakowego słowa, dzięki czemu artykuły je zawierające przenoszą teraz taką samą treść i łatwiej będzie odnaleźć pomiędzy nimi semantycze powiązanie.



Rys. 1 - Przykładowe wyniki algorytmu stemmingu. Różne słowa o tym samym rdzeniu sprowadzane są do jednakowego wyrazu.

6.2 Reprezentacja strukturalna

Po dokonaniu czyszczenia plików posiadamy artykuły w postaci maksymalnie skróconych słów przenoszących interesujące nasz znaczenie. Możliwe jest zatem przystąpienie do tworzenia ich reprezentacji innych niż plik zawierający kolejne wyrazy.

6.2.1 Bag of words

Reprezentacją na której będziemy w znacznej części polegać jest reprezentacja bag-of-words czyli wektor cech. Każdy artykuł jest w niej opisany wektorem o długości równej mocy zbioru wszystkich słów występujących w posiadanym korpusie artykułów. Słowa w tym zbiorze zostają ponumerowane i następnie dla każdego artykułu jest tworzony na tej podstawie odpowiedni wektor. Będąc początkowo wektorem zerowym, zostaje on wypełniony poprzez dodawanie wartości jednostkowej na odpowiednim miejscu dla każdego występującego w danym artykule słowa. Po obliczeniu takiego wektora dla każdego posiadanego artykułu, zostają one ułożone obok siebie i uformowana zostaje macierz bag-of-words która następnie zostaje zapisana.

Zaczynamy więc od obliczenia zbioru wystkich słów występujących w artykułach. Przyjmijmy zatem oznaczenie:

$$A \text{ - zbiór wszystkich słów z posiadanych artykułów}$$

Zadajemy następnie odwzorowanie bijektywne:

$$f : A -> X, \text{ gdzie } X = \{x : x \in \mathbb{N}_+ \wedge x \leq \#A\}$$

Odwzorowanie to więc indeksuje słowa kolejnymi liczbami naturalnymi.
Na jego podstawie można skonstruować wektor cech dla kolejnych artykułów. Dla każdego z nich wystarczy rozpatrzeć każde znajdujące się w nim słowo s i dodać 1 do początkowo zerowego wektora na pozycji $f(s)$.

Wartości elementów utworzonej macierzy M skonstruowanej przez ułożenie wektorów cech przetwarzanych artykułów jako kolumny wynoszą więc:

$$A_{ij} = \text{ilość wystąpień słowa } f^{-1}(i) \text{ w artykule numer j.}$$

6.2.2 Modyfikacje wartości macierzy bag-of-words

Proste wyznaczanie wektorów cech posiada pewne isotne wady. Ponieważ ta reprezentacja ma w przyszłości służyć do porównywania artykułów pomiędzy sobą, nadreprezentacja słów często występujących w języku angielskim może zaburzać wyniki, ponieważ wszystkie wyrazy są traktowane jednakowo. Należy także zwrócić uwagę na większe wartości elementów wektorów cech dla dłuższych artykułów, gdyż mają po prostu więcej słów. Te potencjalne problemy zostały zaadresowane na opisane poniżej sposoby.

Inverse document frequency

Dla danego korpusu artykułów, co jest szczególnie widoczne przy ich konkretnej tematyce - u nas to tematyka finansowa - pewne słowa nie będą właściwie niosły żadnej informacji. Przykładowo rozważając artykuły o tematyce motoryzacyjnej, łatwo jest wyobrazić sobie, że słowo ąuto"będzie występować w prawie każdym tekście. Dla naszego zbioru takim wyrazem może być "money". Zastosowana więc została technika osłabienia wpływu często występujących słów na obliczane podobieństwo pomiędzy artykułami, a także pomiędzy zapytaniem użytkownika a artykułami.

Dla każdego wyrazu wystepującego w naszym zbiorze artykułów obliczona zostaje wartość wyrażenia

$$idf_w = log\frac{N}{df_w}, \text{ gdzie}$$

df_w - ilość dokumentów spośród przetwarzanego zbioru które posiadają wyraz w
N - Ilość wszystkich przetwarzanych dokumentów

Następnie wiersze macierzy odpowiadające ilości wystąpień danego słowa w kolejnych artykułach zostają przez wyliczoną dla niego wartość idf. Czynność ta wykonana zostaje dla każdego słowa. Tym sposobem waga słów wystepujących rzadko zostaje zwiększona, a tych bardziej powszechnych zmniejszona.

Normalizacja

Warto zwrócić uwagę, że stosując reprezentację bag-of-words, artykuły dłuższe będą posiadały statystycznie więcej niezerowych elementów, a

wartości dla słów będą większe (gdyż słowo częściej będzie występować tu wielokrotnie). Spowodować to może zaburzenie wyników podczas obliczania podobieństw pomiędzy różnymi artykułami. Dlatego też kolejne wiersze, czyli wartości dla kolejnych słów dla rozważanych artykułów zostają zostają przemnożone przez wartość

$$\frac{1}{\sqrt{\sum_{i=1}^{n} A_{is}}}$$

gdzie:
n - ilość słów
A - macierz bag-of-words
s - indeks rozważanego słowa

6.3 Dekompozycja SVD oraz low-rank approximation

Dekompozycja SVD (ang. Singular Value Decomposition), to pewien rozkład macierzy na iloczyn trzech specyficznych macierzy. Korzystamy z faktu, że każdą macierz rzeczywistą A można przedstawić w postaci rozkładu SVD:

$$A = U\Sigma V^T$$

gdzie

- U i V, macierze ortonormalne
- Σ - macierz diagonalna (przekątniowa), taka że
- $\Sigma = diag(\sigma_i)$, gdzie σ_i - nieujemne wartości osobliwe macierzy A, uporządkowane nierosnąco.

Zostanie teraz przedstawiony problem przybliżenia macierzy.
Dla zadanej macierzy C o wymiarach $M \times N$ i liczby naturalnej k, chcemy

znaleźć macierz C_k o wymiarach $M \times N$ rzędu co najwyżej k taką, by zminimalizować normę Frobeniusa macierzy $X = C - C_k$, zdefiniowaną jako

$$\|x\|_F = \sqrt{\sum_{i=1}^{M}\sum_{j=1}^{N} x_{ij}^2}$$

Dekompozycja SVD może zostać użyta do rozwiązania tego problemu. Po obliczeniu rozkładu SVD dla macierzy C, uzyskujemy macierz Σ_k z macierzy Σ zastąpując w niej $r - k$ najmniejszych wartości własnych na diagonali Σ zerami (r - rząd macierzy C). Rząd macierzy uzyskanej w ten sposób wynosi co najwyżej k. Wynika to z faktu, że macierz Σ_k ma co najwyżej k niezerowych wartości. Twierdzenie Eckarta–Younga–Mirsky'ego mówi, że macierz ta jest macierzą rzędu k najmniejszym możliwym błędem Foreniusa. Intuicyjnie można to wytłumaczyć faktem, że te najmniejsze wartości własne ułożone na diagonali macierzy Σ_k po wymnożeniu macierzy uzyskanych podczas dekompozycji SVD mają najmniejszy wpływ na macierz (kierunki przez nie wyznaczone niosą informację o najmniejszej wariancji).

Nawet dla zbiorów o niewielkim rozmiarze macierz bag-of-words posiadać będzie dziesiątki tysięcy kolumn i wierszy i będzie także rzędu kilku tysięcy. Wyznaczając więc powyżej opisaną macierz następuje znaczna redukcja wymiaru oraz wyeliminowanie szumu. Wektory cech zostają mapowane do przestrzeni k-wymiarowej. Przestrzeń ta jest opisana przez k wektorów własnych odpowiadających wartościom własnym które nie zostały zastąpione zerami. Te wektory własne przenosząc informację o kierunkach w których niesione jest dużo wariancji mogą pomóc znaleźć synonimy oraz słowa zbliżone tematycznie, tym samym zwiększając prawdopodobieństwo dobrego wyboru artykułów dla zadanego zapytania.

6.4 Latent Dirichlet Allocation

LDA to model który stara się opisać zbiór obserwacji za pomocą tzw. ukrytych grup (czyli takich które muszą zostać wywnioskowane z danych), dzięki czemu próbuje odpowiedzieć na pytanie dlaczego pewne elementy danych są podobne. Dla dokumentów zawierających słowa, model ten reprezentuje je [dokumenty] jako miks niewielkiej liczby tematów. Każdy z tematów z kolei zostaje zamodelowany jako pewna dystrybucja (miks) słów wchodzących w

jego skład. Sam algorytm, działając określoną liczbę iteracji stopniowo poprawia jakość przypisania słów do tematów przeliczając ich dystrybucję na nowo. Problemem jest wartość k, czyli ustalona ilość tematów, z których dokumenty się mają składać. Dobry wybór wartości tego argumentu jest kluczowy dla dobrego działania artykułu.

LDA jest traktowane w naszej aplikacji jako alternatywne podejście wobec SVD. Użytkownik ma możliwość w ustawieniach aplikacji wybrać jakiego rodzaju wyszukiwanie preferuje (Szczegóły wyszukiwania opisane zostały poniżej). W trakcie eksperymentów zauważyliśmy znacznie lepsze wyniki stosując wyszukiwanie używając macierzy powstałej przez dekompozycę SVD i low-rank-approximation.

6.5 Wyszukiwanie

Podczas wyszukiwania zapytanie jest traktowane jako artykuł. Zapytanie jest więc czyszczone, następuje stemming oraz stop words removal. Skonstruowany zostaje wektor bag-of-words.

6.5.1 Wyszukiwanie stosując SVD

Obliczając podobieństwo pomiędzy wektorem reprezentującym zapytanie a kolumnami macierzy będącymi wektorami cech artykułów, znajdowane są te z nich dla których miara podobieństwa jest największa i to one stanowią zawartość wyniku dla użytkownika. Miarą podobieństwa pomiędzy dokumentami d1 oraz d2 jest ich podobieństwo wg miary cosinusowej:

$$sim(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)||\vec{V}(d_2)|}$$

gdzie $\vec{V}(d)$ oznacza reprezentację wektorową artykułu d.

Ponieważ na etapie konstrukcji macierzy dokonano jednak normalizacji i każdy artykuł z którego konstruowana była macierz bag-of-words miał długość jednostkową, mianownik jest iloczynem dwóch wartości jednostkowych i nie musi być kalkulowany. Normalizacja nabiera zatem jeszcze większego sensu - by policzyć podobieństwo pomiędzy artykułami wystarczy obliczyć wartość iloczynu skalarnego odpowiednich wektorów.

6.5.2 Wyszukiwanie stosując LDA

W momencie skończenia działania algorytmu LDA dla zadanej macierzy bag-of-words oraz podanej liczby tematów k, otrzymujemy dwa istotne obiekty

Macierz tematy-słowa Macierz ta dla każdego z k tematów opisuje dystrybucję słów w danym temacie. Jest więc wymiarów $k \times S$, gdzie S to liczba słów w rozważanym zbiorze artykułów.

Macierz artykuły-tematy Macierz ta opisuje dystrybucję k tematów w każdym z rozważanych artykułów. Jej wymiary to $N \times k$, gdzie N to liczba dokumentów z przetwarzanym zbiorze.

W celu wyszukania najbardziej zbliżonych artykułow dla danego zapytania musimy uzyskać dystrybucję tematów w wektorze zapytania, a nastepnie porównać go z innymi artykułami, by tym sposobem znaleźć te najbardziej podobne. By znaleźć w jednym kroku podobieństwo z każdym z artykułów wykonywane jest mnożenie

$$\vec{V}(q) M_{ts}^T M_{at}^T$$

gdzie:

- $\vec{V}(q)$ - wektor cech stworzony dla zapytania (ang. query). Jest to więc macierz wymiarów $1 \times S$, S - ilość słów w zbiorze artykułów.
- M_{ts} - Macierz tematy-słowa, opisana powyżej
- M_{at} - Macierz artykuły-tematy, opisana powyżej

Następnie wystarczy wybrać wymaganą ilość najbardziej podobnych artykułów znajdując je po indeksach maksymalnych wartości w wynikowym wektorze.

Rozdział 7

Wyszukiwanie drill-down

Alternatywnym sposobem wyszukiwania dostarczanym przez aplikację klientowi jest wyszukiwanie tzw. drill-down. Polega ona na ciągłym zawężaniu tematyki interesującej użytkownika aż do znalezienia tej interesującej go. Użytkownikowi zaprezentowane zostają klastry, w odpowiedni sposób etykietowane, dzięki czemu może on zorientować się który z nich wydaje się być najbardziej zbliżony do tematyki jego zainteresowań. Po wybraniu tegoż, zostaje mu zaprezentowana kolejna klasteryzacja, tym razem jedynie wewnątrz wybranego fragmentu. Tym sposobem użytkownik w pewnym momencie wybierze grupę tematyczną na tyle małą, że dalszy podział nie będzie miał sensu i zostaną mu przedstawione wybrane artykuły.

7.1 Preprocessing

Analogicznie jak w przypadku przygotowań do zapytań w postaci podania słów, ogromna część obliczeń wykonana zostać może zawczasu. Zanim użytkownik wybierze klastry, zostają one zawczasu odpowiednio policzone oraz wybrane zostają ich etykiety. Jest to możliwe ponieważ dla danego zbioru artykułów klasteryzacja nie zależy od kontekstu.

7.1.1 Bag of words

Analogicznie jak w pierwszym rodzaju wyszukiwania, zostaje zbudowana macierz złożona z wektorów cech artykułów, która następnie zostaje zredukowana w sensie ilości wymiarów za pomocą SVD.

7.1.2 Klasteryzacja

Posiadane artykuły zostają klasteryzowane za pomocą algorytmu k-means. Zostają one za jego pomocą podzielone na 12 odrębnych grup na podstawie odległości pomiędzy sobą. Wyliczone zostają też centroidy (średni reprezentanci) dla grup z pierwszej klasteryzacji. Następnie obliczone i zapisane zostają kolejne klasteryzacje wgłąb każdego z obliczonych klastrów. Te zstępujące rekurencyjne obliczenia kontynuowane są tak długo, jak ilość artykułów należących do danego klastra jest odpowiednio duża.

7.1.3 Użycie wiedzy zewnętrznej

Ponieważ dysponowaliśmy wiedzą zewnętrzną w postaci kilkuset tysięcy artykułów serwisu reuters o tematyce finansowo-ekonomicznej, z których każdy był etykietowany kilkoma kategoriami do których należy, postanowiliśmy jej użyć. Dla każdej kategorii został policzony jej średni reprezentant, przy użyciu tych samych reprezentacji strukturalnych jak dla naszych artykułów, tj. wektor cech oraz niezmienniki grafowe.

Posiadając obliczonych średnich reprezentantów dla naszych klastrów oraz poszczególnych kategorii z bazy artykułów reuters, dla każdej z centroid klastrów naszych artykułów obliczone zostały podobieństwa z każdą z centroid kategorii reuters. Następnie wybrane i zapisane zostały trzy najlepsze dla każdej z naszych grup.

7.1.4 Etykietowanie klastrów

By użytkownik mógł sprawnie wybierać interesujące go klastry i tym samym zawężać przeszukiwane grupy artykułów, grupy te powinny być jak najdokładniej opisane. Służyć temu ma operacja etykietowania. Pierwszą czynnością jest opisanie klastrów najwyższego poziomu (najbardziej ogólnych, pochodzących z pierwszej klasteryzacji) za pomocą nazw przypisanych im kategorii z zewnętrznej bazy artykułów (proces dopasowania opisany został powyżej). Następnie klastry każdego poziomu opisane zostają słowami które przenoszą największą w tym klastrze ilość informacji (przypisana im wartość w macierzy po obliczeniu SVD dla danego artykułu jest największa). Przyjęta zostaje zasada, że etykiety użyte w pewnym klastrze, nie zostaną ponownie użyte do opisania jego klastrów potomnych, by etykiety różnych klastrów jak najbardziej różniły się między sobą.

Rozdział 8

Organizacja pracy

8.1 Przyjęta metodyka

Projekt rozpoczęliśmy od stworzenia prototypu strony serwerowej, wykorzystując najprostszą reprezentację strukturalną dla posiadanych artykułów, czyli wektor słów dla każdego z nich.

Od tego momentu kolejne funkcjonalności dodawane były przyrostowo. Kolejne spotkania z klientem wyznaczały końce kolejnych iteracji. Tym sposobem ograniczone zostało ryzyko niedostarczenia projektu - posiadając pod koniec każdej iteracji działającą stronę serwerową groziło w najgorszym przypadku okrojeniem funkcjonalności, który to scenariusz na szczęscie nie miał miejsca.

8.2 Podział pracy

8.2.1 Michał Adamczyk

- Czyszczenie artykułów - stop words removal, stemming
- Budowa reprezentacji strukturalnej artykułów w postaci wektorów cech
- Redukcja wymiaru macierzy wektorów cech - za pomocą algorytmu SVD lub algorytmu LDA (stosowane wymiennie)
- Dołączenie wiedzy zewnętrznej w postaci kategorii z bazy danych artykułów reutersa

- Implementacja algorytmu k-means++
- Etykietowanie klastrów najbardziej charakterystycznymi dla nich słowami oraz nazwami kategorii z bazy zewnętrznej

8.2.2 Dominik Majda

- Budowa reprezentacji strukturalnej artykułów w postaci engramów oraz wyliczenie na ich podstawie grupy niezmienników grafowych
- Zdobycie za pomocą web crawlingu zbioru artykułów o tematyce finansowej oraz przekształcenie ich z formatu html do formatu czystego tekstu
- Stworzenie aplikacji mobilnej realizującej rolę interfejsu końcowego po stronie użytkownika
- Wystawienie interfejsu serwera w postaci restowego API dostępnego na przykład dla stworzonej przez nas aplikacji mobilnej
- Budowa bazy danych dla artykułów zarówno z naszej bazy jak i bazy zewnętrznej, dzięki czemu możliwe jest przechowywanie dodatkowych o artykułach informacji używanych w trakcie obliczeń (jak link czy tytuł).