

Streszczenie

Rekomendacje artykułów opisujących produkty w serwisach e-commerce

Tematyka niniejszej pracy skupia się wokół zagadnień określania podobieństwa semantycznego pomiędzy dokumentami tekstowymi i rekomendacji dokumentów podobnych do danego. Szczegółowy problem pochodzi z internetowego serwisu aukcyjnego Allegro, który posiada dział artykułów opisujących produkty dostępne w serwisie. W dziale tym funkcjonuje system rekomendacji podobnych artykułów tekstowych w oparciu o ich treść. Celem pracy jest zbadanie możliwości usprawnienia działania istniejącego systemu rekomendacji wykorzystując metody semantycznej analizy tekstu.

W niniejszej pracy adaptuję dostępne metody określania podobieństwa pomiędzy dokumentami tekstowymi do powyższego problemu, wprowadzam miary umożliwiające ocenę działania tych metod oraz dokonuję analizy możliwości ich wykorzystania w rzeczywistym systemie.

Słowa kluczowe: rekomendacje, przetwarzanie języka naturalnego, osadzanie słów, semantyka, allegro

Abstract

Content-based recommendations in e-commerce services

The subject of this paper focuses on the issues of determining the semantic similarity between text documents and the recommendation of documents similar to a given. A detailed problem comes from the Allegro on-line auction site, which has a section of articles describing the products available on the site. This section offers a recommendation system for similar textual articles based on their content. The aim of this paper is to investigate the possibility of improving the existing recommendation system using semantic text analysis methods.

In this paper, I adapt the available methods for determining the similarity between text documents to the above problem, I introduce measures to evaluate the performance of these methods and analyze the possibilities of using them in the real system.

Keywords: recommendations, natural language processing, word embedding, semantics, allegro

Łukasz Dragan

Warszawa, dnia

Nr albumu 254179

Oświadczenie

Oświadczam, że pracę magisterską pod tytułem „Rekomendacje artykułów opisujących produkty w serwisach e-commerce”, której promotorem jest dr inż. Anna Wróblewska wykonałem samodzielnie, co poświadczam własnoręcznym podpisem.

.....

Łukasz Dragan

Spis treści

| | |
|---|-----------|
| Wstęp | 11 |
| 0.1. Rekomendacje artykułów tekstowych w Allegro | 11 |
| 0.2. Struktura pracy | 13 |
| 0.3. Uwagi | 14 |
| 1. Przegląd wiedzy z zakresu tematyki pracy | 15 |
| 1.1. Systemy rekomendacji | 15 |
| 1.1.1. Filtrowanie kolaboratywne (collaborative filtering) | 16 |
| 1.1.2. Filtrowanie oparte na treści (content-based filtering) | 16 |
| 1.2. Silnik Elasticsearch | 16 |
| 1.3. Techniki przetwarzania języka naturalnego | 17 |
| 1.3.1. Bag-of-words | 17 |
| 1.3.2. Term frequency - inverted document frequency | 18 |
| 1.3.3. Distributional semantics | 19 |
| 1.3.4. Latent Semantic Analysis | 19 |
| 1.3.5. Latent Dirichlet Allocation | 20 |
| 1.3.6. Word embeddings | 21 |
| 1.3.7. Podejścia deep learningowe | 22 |
| 1.3.8. Word2vec | 23 |
| 1.3.9. FastText | 25 |
| 1.3.10. GloVe | 26 |
| 1.3.11. Odległość między dokumentami | 26 |
| 1.4. Miary oceny wyszukiwania | 28 |
| 1.4.1. nDCG | 28 |
| 2. Dane | 30 |
| 2.1. Opis danych | 30 |
| 2.1.1. Treść artykułu | 30 |
| 2.1.2. Kategoria | 31 |

| | |
|--|-----------|
| 2.1.3. Słowa kluczowe | 32 |
| 2.2. Wstępne przetwarzanie danych | 33 |
| 2.3. Opis danych po wstępnym przetwarzaniu | 34 |
| 3. Metody ewaluacji | 36 |
| 3.1. Miara 1: Dystans oparty na metadanych | 37 |
| 3.1.1. Kategorie | 37 |
| 3.1.2. Słowa kluczowe | 38 |
| 3.2. Miara 2: Historyczna aktywność użytkowników serwisu | 39 |
| 3.3. Miara 3: Ocena przez użytkowników offline | 40 |
| 4. Opis testów | 41 |
| 4.1. Testowane metody generowania rekomendacji | 41 |
| 4.1.1. Metody oparte o modelowanie tematu | 41 |
| 4.1.2. Metody oparte o word embedding | 41 |
| 4.1.3. Elasticsearch | 43 |
| 4.1.4. Metoda losowa | 43 |
| 4.2. Metody ewaluacji | 43 |
| 5. Wyniki badań | 45 |
| 5.0.1. LSI w zależności od liczby tematów | 45 |
| 5.0.2. LDA w zależności od liczby tematów | 46 |
| 5.1. Word2vec w zależności od korpusu | 47 |
| 5.2. Word2vec w zależności od metody porównywania dokumentów | 49 |
| 5.2.1. Wymiar wektorów: 100 | 49 |
| 5.2.2. Wymiar wektorów: 300 | 50 |
| 5.3. Metody word embedding w zależności od wymiarowości wektorów | 52 |
| 5.3.1. Word2vec | 52 |
| 5.3.2. GloVe | 53 |
| 5.3.3. FastText | 54 |
| 5.4. Ocena jakości wybranych metod przez użytkowników | 55 |
| 5.4.1. Zestawienie wyników testów automatycznych | 55 |
| 5.4.2. Wyniki oceny eksperckiej | 56 |
| 6. Wnioski i podsumowanie | 58 |
| Bibliografia | 59 |
| Wykaz symboli i skrótów | 62 |
| Spis rysunków | 63 |

| | |
|---|-----------|
| Spis tabel | 64 |
| Spis załączników | 65 |
| A. Technologie i narzędzia | 66 |

Wstęp

Systemy rekomendacji są powszechnym elementem wielu serwisów internetowych. Sprawdzają się na takich polach, jak polecanie produktów w sklepie czy rekomendacje ofert pracy. Dają użytkownikowi poczucie indywidualnego traktowania przez serwis internetowy dopasowujący niejako zawartość swoich stron do konkretnego użytkownika. Pozwala to użytkownikowi na bardziej efektywne korzystanie z serwisu oraz może prowadzić do większego zaangażowania ze strony użytkownika i przywiązania do serwisu. Systemy rekomendacji dają obopólną korzyść zarówno użytkownikowi jak i właścicielowi serwisu internetowego.

Celem niniejszej pracy magisterskiej jest analiza możliwości usprawnienia istniejącego systemu rekomendacji o oparciu o adaptację istniejących metod wyszukiwania semantycznego podobieństwa pomiędzy dokumentami tekstowymi. Rzeczony system rekomendacji istnieje w internetowym serwisie e-commerce Allegro w dziale artykułów tekstowych o tematyce związanej z produktami dostępnymi za pośrednictwem serwisu. System ma na celu zarekomendowanie użytkownikowi artykułów o tematyce podobnej do tego, który znajduje się na stronie aktualnie odwiedzanej przez użytkownika.

W swojej pracy badam możliwość użycia istniejących metod semantycznej analizy tekstu w odniesieniu do opisanego problemu. Badane metody to: Latent Semantic Analysis, Latent Dirichlet Allocation, Word2vec, GloVe oraz FastText. Jakość działania tych metod porównuję poprzez samodzielnie opracowane metody ewaluacji.

Podczas prowadzenia badań stworzyłem szereg skryptów przetwarzających dane i wykorzystujących implementacje opisywanych w tej pracy metod. Opis użytych narzędzi programistycznych i bibliotek zawarłem w dodatku A do niniejszej pracy.


Rekomendacje artykułów tekstowych w Allegro

Allegro jest największą[1] działającą na rynku polskim platformą aukcyjną on-line. Posiada ponad 20 mln zarejestrowanych klientów. Każdego dnia na Allegro sprzedaje się ponad 870 tysięcy

przedmiotów. Zatrudnia 1300 pracowników.[2] Serwis umożliwia użytkownikom wystawianie na sprzedaż oraz kupno przedmiotów poprzez mechanizm licytacji lub natychmiastowego zakupu.

Oprócz głównej części serwisu odpowiedzialnej za transakcje Allegro posiada dział zajmujący się publikacją artykułów opisujących produkty wystawiane za pośrednictwem serwisu. Ma to na celu pomoc użytkownikom przy wyborze interesującego ich produktu.

Po to, aby zachęcić użytkowników do zapoznania się z treścią kolejnych artykułów, zastosowany został tu system rekomendacji przyporządkowujący danemu artykułowi listę powiązanych artykułów. Kryterium mówiącym, czy artykuły są powiązane jest tutaj jedynie treść samych artykułów, a nie wcześniejsze zachowanie użytkownika.



The screenshot shows the Allegro website interface. At the top, there's a navigation bar with links like 'Strefa Marek', 'Inspiracje', 'Poradniki', 'wystaw przedmiot', 'obserwowane', 'moje allegro', 'załóż konto', and 'załóżuj'. Below this is the Allegro logo and a search bar. The main content area displays an article titled 'Jaka farba dla alergika?' (Which paint for allergies?). The article is by Ewelina Wojtunik, published on 23-04-2015. The article text discusses the need for fresh paint and the health benefits of low-VOC paints for allergy sufferers. The sidebar features a profile of Ewelina Wojtunik and a recommendation for 'Wnętrzarski hit – ściany ombre' (Interior hit – ombre walls).

Jaka farba dla alergika?

autor: Ewelina Wojtunik, data publikacji: 23-04-2015

Za chwilę wiosna, a wraz z nią potrzeba porządków i odświeżenia ścian. Jak co roku będziemy sprzątać, wietrzyć i wymiatać zimę z kątów mieszkania. Zaraz po tym zaczną się pierwsze remonty.

Czy wiecie, że nawet do 30% populacji to alergicy? Uczulają nas pyłki, kurz, sierść zwierząt, pokarmy, ale przede wszystkim chemikalia i detergenty. Znajdujące się w nich alergeny mogą być powodem problemów zdrowotnych, a także nasilać objawy nadwrażliwości takie jak łzawienie oczu, zapalenie skóry czy kaszel astmatyczny.

Szkodliwe związki lotne

W styczniu 2010 roku Unia Europejska wprowadziła normę, która reguluje zawartość szkodliwych lotnych związków organicznych tak zwanych LZO (VOCs, ang. volatile organic compounds) w trafiających do sprzedaży farbach i lakierach. Warto wiedzieć, że lotne związki lubią pozostawać aktywne pomimo wyschnięcia farby i starannego wywietrzenia mieszkania. Co więcej, mogą uwalniać się ze ścian całymi latami, nasilając objawy alergiczne i pogarszając samopoczucie mieszkańców. Im mniej ich w składzie, tym lepiej dla nas.

Ewelina Wojtunik

Zawodowo związana z Social Media, pisała m.in. do Aktivist.pl. Prywatnie pasjonatka projektowania wnętrz, zdrowego stylu życia i roślin doniczkowych. Podróże i kuchnie świata są dla niej inspiracją. W wolnym czasie spełnia się jako mama i uczy języków.

może Cię również zainteresować

Wnętrzarski hit – ściany ombre

Ombre stało się hitem w wizażu i modzie już kilka sezonów temu! Chętnie rozjaśniamy końcówki włosów, cieniujemy kolory na paznokciach, a także nosimy ubrania w przenikających si...

Rysunek 0.1: Widok strony internetowej zawierającej jeden z artykułów serwisu Allegro. [3]

Od serwisu Allegro otrzymałem zserializowaną kopię 20000 artykułów dostępnych na stronach serwisu. Pojedynczy artykuł składa się z głównej zawartości tekstowej oraz metadanych. W celu otrzymania wszelkich danych od firmy Allegro wynagane było, abym podpisał umowę, w której zobowiązuję się do nieujawniania żadnych danych, które otrzymałem. Stąd opisy danych, na których pracuję, zawarte w tej pracy nie wnikają w ich szczegóły i nieodbiegają od informacji publicznie dostępnych za pośrednictwem strony pod adresem <https://allegro.pl/artykuly>.

Aktualnie w rzeczonym dziale serwisu Allegro istnieje system rekomendacyjny, który opiera się o wyszukiwanie podobnych artykułów tekstowych za pomocą silnika Elasticsearch[7]. Metoda ta wykorzystuje słowa kluczowe przypisane do każdego artykułu przez autora. W swojej pracy staram się porównać wyniki działania dotychczasowej metody z metodami semantycznej analizy tekstu, które potrafią wykryć podobieństwo pomiędzy artykułami bazując jedynie na ich treści, bez potrzeby dołączania żadnych metadanych. Pomyślna próba zastosowania metod semantycznych pozwoliłaby na dokładniejsze dopasowanie podobnych artykułów w oparciu być może o pewne ukryte cechy semantyczne nieosiągalne dla silnika wyszukiwania tekstowego, jakim jest Elasticsearch. Bardziej szczegółowego opisu silnika Elasticsearch dokonuję w kolejnym rozdziale.

Struktura pracy

W rozdziale 1 wprowadzam do zagadnienia rekomendacji oraz dokonuję przeglądu metod semantycznej analizy tekstu, które mogą zostać zastosowane w celu określenia podobieństwa pomiędzy dokumentami tekstowymi.

Następnie w rozdziale 2 dokonuję opisu konkretnego problemu, jakim jest generacja rekomendacji artykułów tekstowych w serwisie Allegro. Opisuję dane otrzymane z serwisu oraz kolejne etapy ich wstępnego przetwarzania, aby nadawały się do zaaplikowania do nich wybranych metod.

Dalej, w rozdziale 3 opisuję stworzone i zastosowane później metody ewaluacji wyników.

Następnie w rozdziale 4 dokonuję opisu testów: jakie metody i w jako sposób testuję.

W rozdziale 5 opisuję wyniki przeprowadzonych eksperymentów.

Ostatecznie w rozdziale 6 dokonuję podsumowania przeprowadzonych badań i rozważam kierunki dalszych prac w tej dziedzinie.

Załącznik A zawiera opis narzędzi programistycznych i bibliotek wykorzystanych przeze mnie podczas prowadzenia badań.

Uwagi

W celu uniknięcia nieporozumień należy podkreślić różnicę pomiędzy znaczeniami słowa „artykuł”, które może oznaczać zarówno tekst publicystyczny, literacki lub naukowy jak i rzecz, która jest przedmiotem handlu.[4] W niniejszej pracy skupiam się na rekomendacjach artykułów tekstowych, stąd używam pierwszego znaczenia (chyba, że inne znaczenie jest wyraźnie zaznaczone).

Przegląd wiedzy z zakresu tematyki pracy

W swojej pracy dokonuję adaptacji metod przetwarzania języka naturalnego na potrzeby generowania rekomendacji artykułów tekstowych w oparciu o ich treść. W niniejszym rozdziale dokonuję przeglądu znanych metod z obszaru tematyki pracy dyplomowej, skupiając się szczególnie na nowo powstałych metodach wektorowej reprezentacji słów, które cieszą się obecnie dużym zainteresowaniem środowisk naukowych oraz firm komercyjnych.

Dokonuję krótkiego wprowadzenia do zagadnienia generowania rekomendacji, którego głęboka analiza nie jest konieczna z punktu widzenia niniejszej pracy. Następnie wykonuję chronologiczny przegląd metod ciągłej reprezentacji słów zaczynając od trywialnych metod zliczania słów (bag-of-words, tf-idf), przechodząc przez metody wykorzystujące koncepcję tematów (Latent Semantic Analysis, Latent Dirichlet Allocation) i kończąc na głośnych ostatnio metodach osadzania słów w przestrzeni wektorowej (Word2vec, GloVe, FastText). Przy zarysie historycznym opieram się w dużej mierze na artykule[5].

Systemy rekomendacji

Systemy rekomendacji to narzędzia i techniki mające na celu zasugerować użytkownikowi przedmioty. Sugestie te odnoszą się do różnych procesów podejmowania decyzji takich jak np. które artykuły kupić, jakiej muzyki słuchać czy też które wiadomości czytać. „Przedmiot” jest tutaj ogólnym pojęciem oznaczającym coś, co system poleca użytkownikowi. [6]

Przy wciąż wzrastającej ilości danych użytkownicy serwisów internetowych często nie są w stanie dotrzeć do informacji, która ich interesuje. Jest to pole do rozwoju zautomatyzowanych systemów rekomendacyjnych polecających użytkownikom treści, które mogą ich zainteresować. Działalność takiego systemu daje zysk zarówno użytkownikowi, pozwalając mu dotrzeć do informacji, której mógłby samodzielnie nie odszukać, albo wręcz nie wiedzieć, iż taka informacja istnieje, jak i właścicielowi serwisu internetowego, któremu zależy, by przyciągnąć do siebie użytkowników, aby ci w jak największym stopniu korzystali z ich usług.

Sposoby działania systemów rekomendacji można podzielić na różne warianty, spośród których wyodrębnić można dwa najszerzej używane. Są to: filtrowanie kolaboratywne (collaborative filtering) i filtrowanie oparte na treści (content-based filtering).

Filtrowanie kolaboratywne (collaborative filtering)

Technika ta opiera się na spostrzeżeniu, iż użytkownicy o podobnych preferencjach zachowują się podobnie. Stąd jeżeli użytkownik zachowuje się podobnie do zaobserwowanej wcześniej grupy użytkowników, można przewidzieć jego preferencje na podstawie zachowań ów grupy. Istotną zaletą tej metody jest fakt, iż nie zależy ona od dziedziny, w której ulokowany jest system rekomendacji (w przeciwieństwie do rekomendacji opartych na treści), a jedynie od zachowań użytkowników.

Filtrowanie oparte na treści (content-based filtering)

W technice tej przedmioty polecane użytkownikowi zależą od innych przedmiotów, na temat których stwierdzono, że użytkownik się nimi interesuje. Mogą się one opierać np. na podobieństwie przedmiotów: jeżeli użytkownik „lubi” przedmiot A, który jest podobny do przedmiotu „B” to można spodziewać się, że również przedmiot B zainteresuje użytkownika. Technika ta jest mocno zależna od dziedziny rekomendowanych przedmiotów, gdyż wymaga wprowadzenia pewnej miary podobieństwa między nimi. Stąd jest trudniejsza do zastosowania, ale daje też możliwości nieosiągalne dla filtrowania kolaboratywnego.

Celem niniejszej pracy jest zbadanie metod sugerujących użytkownikowi artykuły podobne do aktualnie odwiedzanego, co wprost wiąże się z metodami używanymi w technice filtrowania opartego na treści.

Silnik Elasticsearch

Obecnie wykorzystywana przez Allegro metoda generowania rekomendacji artykułów opiera się o zapytanie do usługi Elasticsearch[7] wykorzystujące słowa kluczowe dołączone do artykułów. Elasticsearch jest popularnym silnikiem wyszukiwania tekstu opartym o indeks Lucene[8]. Działa w architekturze rozproszonej a komunikacja z nim następuje poprzez protokół HTTP i format JSON. Umożliwia on efektywne przechowywanie dokumentów tekstowych oraz efektywne ich wyszukiwanie.

Apache Lucene jest biblioteką napisaną w języku Java służącą do wyszukiwania tekstu, która w tym celu wykorzystuje mechanizm odwróconego indeksu. Zasada działania biblioteki polega

na stworzeniu słownika ze wszystkich (odpowiednio wstępnie przetworzonych) słów dokumentów przeznaczonych do wyszukiwania. Następnie na bazie ów słownika tworzony jest odwrócony indeks: każdemu ze słów przypisywana jest lista dokumentów, które zawierają to słowo. Pozwala to przyspieszyć proces wyszukiwania, gdyż w poszukiwaniu pojedynczego słowa biblioteka nie przeszukuje całego zbioru dokumentów, a jedynie słownik, który na ogół jest wielokrotnie krótszy.

Zaletą silnika Elasticsearch są jego wydajność, skalowalność, niezawodność i prostota użytkowania, co przekłada się na jego dużą popularność wśród np. serwisów internetowych[9].

Wadą metody jest to, że ogranicza się ona do wyszukiwania tekstowego pomijając aspekt semantyczny. Stwarza to trudności przy wyszukiwaniu synonimów lub homonimów.

Techniki przetwarzania języka naturalnego

Oparcie rekomendacji jedynie na treści artykułu wymaga zagłębienia się w tematykę analizy i przetwarzania języka naturalnego, wszak właśnie w języku naturalnym, zrozumiałym dla człowieka (polskim) pisane są owe artykuły. Język naturalny z powodu swojego niskiego stopnia sformalizowania nie jest niestety wprost zrozumiały dla maszyn. W związku z tym koniecznym staje się tu użycie technik przetwarzania języka naturalnego (natural language processing), które to pozwalają wyodrębnić z tekstu pewne cechy, na bazie których maszyna obliczeniowa przy pomocy pewnych algorytmów jest w stanie określić podobieństwo pomiędzy dokumentami. W poniższych paragrafach dokonuję przeglądu technik matematycznej reprezentacji dokumentów pisanych w języku naturalnym. Warto wspomnieć, iż dziedzina ta bardzo dynamicznie się rozwija a część z opisywanych metod zostało stworzonych na przestrzeni ostatnich kilku lat, czy wręcz miesięcy.

W celu formalizacji w dalszych opisach stosowanych metod stosuję następujące oznaczenia:

- Korpus C : zbiór dokumentów d_i ,
- Dokument d : skończony ciąg zdań s_i ,
- Słowo w : skończony ciąg znaków c_i ,
- Słownik zbudowany na korpusie C : $V = \{w \mid \exists d \in C \ w \in d\}$.

Bag-of-words

Bag-of-words (worek słów)[10] jest jedną z pierwszych koncepcji reprezentacji tekstu jako zbioru zawartych w nim słów w postaci wektorów. Metoda nie zachowuje kolejności słów w

tekście, lecz liczbę ich wystąpień. Istotną zaletą reprezentacji wektorowej dokumentów jest możliwość zdefiniowania miary odległości pomiędzy dokumentami (np. miara kosinusowa opisana później) odzwierciedlającej ich podobieństwo. Technikę tę można opisać jako przekształcenie z korpusu w przestrzeń wektorów $bow : C \rightarrow \mathbb{R}^n$ gdzie:

C : korpus

$m = |C|$: liczba dokumentów w korpusie C

V : słownik zbudowany na C

$n = |V|$: liczba słów w V

$v_i \in \mathbb{R}^n$, gdzie $i \in 1, 2, \dots, n$ wektor reprezentujący dokument $d_i \in C$

v_{ij} , gdzie $j \in 1, 2, \dots, m$: liczba wystąpień w dokumencie $d_i \in C$ słowa $w_j \in V$

Technika ta jest stosunkowo prosta, lecz jej wadą jest traktowanie każdego słowa z jednakową wagą. Pewne słowa (np. „i”, „lub”, „o”) występują bardzo często, lecz ich wkład w znaczenie całego dokumentu jest marginalny. Stąd powstały bardziej zaawansowane techniki uwzględniające istotność słów dla znaczenia całego dokumentu. Mimo to metoda BOW jest często wykorzystywana w bardziej zaawansowanych technikach NLP.

Term frequency - inverted document frequency

TF-IDF[11] (ważenie częstością termów - odwrotna częstość w dokumentach) jest metodą reprezentacji tekstu jako zbioru słów przy jednoczesnym uwzględnieniu wagi słów, która zależy od częstości występowania słowa w korpusie. Oznaczenia formalne takie same jak w przypadku BOW. $v_{ij} = tf_{ij}idf_{ij} = tf_{ij} * idf_i$, gdzie:

$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}$, „term frequency” to liczba wystąpień słowa w_i w dokumencie d_j podzielona przez liczbę słów dokumentu d_j ,

$idf_i = \log \frac{|D|}{|d: w_i \in d|}$, „inversed document frequency” to liczba dokumentów w korpusie podzielona przez liczbę dokumentów zawierających przynajmniej jedno wystąpienie słowa w_i ,

Dokumenty reprezentowane są tu jako wektory, składające się z wag słów występujących w każdym z nich. TF-IDF przechowuje informację o częstotliwości występowania słów biorąc przy tym pod uwagę istotność znaczenia słowa lokalnego w stosunku do jego znaczenia w kontekście całego zbioru dokumentów. W tej technice słowa występujące rzadko są premiowane względem słów pospolitych. Wadą metody tej i poprzedniej jest postać wektorów reprezentujących słowa: są to na ogół rzadkie wektory dużej wymiarowości.

Distributional semantics

Kolejne, bardziej zaawansowane, omawiane tu metody opierają się na tzw. „distributional hypothesis” - hipotezie zakładającej, że słowa występujące w tym samym kontekście niosą ze sobą podobne znaczenie[10][12]. Sprzyja to zastosowaniu metod algebry liniowej jako narzędzia obliczeniowego oraz sposobu reprezentacji tekstu. Podstawowe podejście polega na zgromadzeniu informacji o rozkładzie słów w dokumentach w postaci wielowymiarowych wektorów a następnie wyodrębnieniu podobieństw pomiędzy tymi wektorami, które świadczyłyby o pewnych powiązaniach między reprezentowanymi słowami.

Latent Semantic Analysis

Analiza rozkładu słów w dokumentach tekstowych pozwala na wyodrębnienie podobieństw między słowami pod kątem: ich znaczenia (podobieństwo tematu słowa), ich osadzenia w stosunku do innych typów słów czy też ich struktury wewnętrznej. Dwie istotne metody: Latent Semantic Analysis[13] oraz Latent Dirichlet Allocation[14] zakładają istnienie abstrakcyjnych niejawnych (latent) tematów, do których można przydzielić słowa wchodzące w skład korpusu.

LSA (1990) [13], znane również jako Latent Semantic Indexing (LSI) dokonuje transformacji każdego dokumentu w wektor dł. $|V|$ posiadający na i -tym miejscu wagę TF-IDF i -tego słowa ze słownika. W ten sposób tworzona jest rzadka macierz: kolumny reprezentują dokumenty a wiersze reprezentują unikalne słowa. W celu identyfikacji istotnych cech tej macierzy dokonuje się rozkładu według wartości osobliwych (Singular Value Decomposition[15], SVD), który jest techniką redukcji wymiarowości. Celem użycia SVD jest redukcja liczby wierszy macierzy dla wydajniejszych dalszych obliczeń numerycznych oraz pozbycie się szumów, utrzymując jednocześnie podobieństwa pomiędzy kolumnami. Ostatecznie uzyskuje się macierz przynależności tematów do dokumentów, gdzie wiersze odpowiadające tematom można interpretować jako kombinacje pierwotnych wierszy-słów o podobnym znaczeniu. Np. $\{(samochod), (ciagnik), (jezdnia)\} \rightarrow \{(1.3452 * samochod + 0.2828 * ciagnik + 0.3 * jezdnia)\}$. Wymiar uzyskiwanej macierzy jest ustalany za pomocą hiperparamtru, który oznacza liczbę tematów. Używając uzyskanej macierzy, podobieństwo pomiędzy kolumnami-dokumentami obliczane jest wykorzystując odległość kosinusową (opisaną później w tym rozdziale). Metoda LSA łagodzi problem synonimów poprzez scalanie podobnych słów w jeden temat. Niweluje również problem homonimów, włączając je częściowo w skład różnych tematów. Niemniej jednak poprzez arbitralne ustalanie hiperparametru odpowiedzialnego za liczbę tematów część semantycznie odrębnych tematów może zostać wchłonięta przez inne lub też rozbite na tematy może być zbyt „drobne” nie wykorzystując w pełni semantycznych powiązań.

Latent Dirichlet Allocation

LDA[14] jest techniką automatycznego wykrywania niejawnych (latent) tematów zawartych w dokumentach przy użyciu uczenia nienadzorowanego. LDA reprezentuje dokumenty jako mieszanki tematów, które z kolei reprezentowane są jako rozkłady prawdopodobieństwa na zbiorze słów. Liczba tematów ustalana jest za pomocą hiperparametru. LDA jest modelem statystycznym, który wykorzystuje m.in. rozkład Dirichleta.

Rozkład Dirichleta to ciągły rozkład prawdopodobieństwa parametryzowany przez wektor α K dodatnich liczb rzeczywistych. Wymiar wektora α określa wymiar rozkładu. Gęstość rozkładu Dirichleta wyraża się wzorem: $f(x_1, \dots, x_{K-1}; \alpha_1, \dots, \alpha_K) = \frac{1}{B(\alpha)} \prod_{i=1}^K x_i^{\alpha_i-1}$, gdzie $x_1, \dots, x_{K-1} > 0$, $x_1 + \dots + x_{K-1} < 1$, $x_K = 1 - x_1 - \dots - x_{K-1}$, a B to stała normalizująca. Nośnikiem gęstości rozkładu jest $K - 1$ wymiarowy sympleks.

W LDA rozkład Dirichleta jest wykorzystany w celu nadania początkowych wartości przynależności tematów do dokumentów oraz słów do tematów. Cechy rozkładu sprawiają, że tak dobrane początkowe wartości parametrów modelu są zgodne z intuicją, że dokument pokrywa jedynie mały zestaw tematów, a temat zawiera najczęściej tylko mały zestaw słów. Wykorzystanie rozkładu Dirichleta do określenia wartości początkowych skutkuje w rezultacie lepszym dopasowaniem dokumentów i tematów.

Przypuśćmy, że mamy zestaw dokumentów. Wybieramy ustaloną liczbę T tematów, które zamierzamy wykryć. Chcemy użyć LDA w celu wyznaczenia reprezentacji każdego dokumentu jako mieszanki tematów oraz słów powiązanych z każdym tematem. Jednym ze sposobów, aby osiągnąć ten cel jest wnioskowanie oparte na próbkowaniu Gibbsa. Metoda ta działa zgodnie z następującymi krokami.

1. Przejdź przez każdy dokument i losowo (zgodnie z rozkładem Dirichleta) przypisz każde słowo dokumentu do jednego z T tematów. Warto zauważyć, iż etap ten daje pierwsze przybliżenie docelowej reprezentacji. W kolejnych krokach należy poprawiać to przybliżenie.
2. Dla każdego dokumentu d , dla każdego słowa w należącego do d , dla każdego tematu t oblicz: $p(t|d)$, czyli odsetek liczby słów w d , które są aktualnie przypisane do tematu t oraz oblicz $p(w|t)$, czyli odsetek liczby wystąpień słowa w , które są przypisane do tematu t w skali całego korpusu. Przypisz słowu w nowy temat poprzez losowanie z prawdopodobieństwem $p(t_i|d) * p(w|t)$ dla każdego tematu t_i .

Ciągłe wykonywanie powyższych kroków doprowadzi do stabilnej sytuacji, w której przestaną następować zmiany przypisań słów do tematów. Wtedy należy zakończyć działanie algorytmu.

Zaletą LDA jest jego interpretowalność: do każdego tematu przypisane są z pewną wagą prawdziwe słowa pochodzące z przetwarzanego korpusu. Metoda ta może być traktowana jako technika redukcji wymiarowości, gdyż dopasowuje dokumentowi składającemu się z wielu słów reprezentację złożoną z małej liczby tematów.

Word embeddings

Od 2013r., wraz z wprowadzeniem przez T. Mikolova metody word2vec[16] nastąpił gwałtowny rozwój i niewątpliwy sukces metod „word embeddings”. Określenie „word embeddings” oznacza osadzanie słów w przestrzeni wektorowej przy pomocy uczenia nienadzorowanego i zostało po raz pierwszy użyte 2003r. w pracy Y. Bengio[17], gdzie wektory słów generowane są przez głęboką sieć neuronową. Ogół technik zaliczanych obecnie do „word embeddings” cechuje się usiłowaniami reprezentacji słów wraz z zależnościami pomiędzy nimi w postaci wektorów o stosunkowo niskiej wymiarowości. Dzieje się to w opozycji do wcześniejszych podejść podobnych do Bag of words - produkującego ogromne, rzadkie wektory, których wymiary równają się rozmiarowi słownika, o który oparty jest model (rzędu setek tysięcy). Ważną własnością metod osadzania słów jest zachowanie przez wektory semantycznych i syntaktycznych właściwości słów, co pozwala wykonywać na nich operacje arytmetyczne na wektorach odwzorowujące cechy tychże słów np. $vector("king") - vector("man") + vector("woman") \approx vector("queen")$

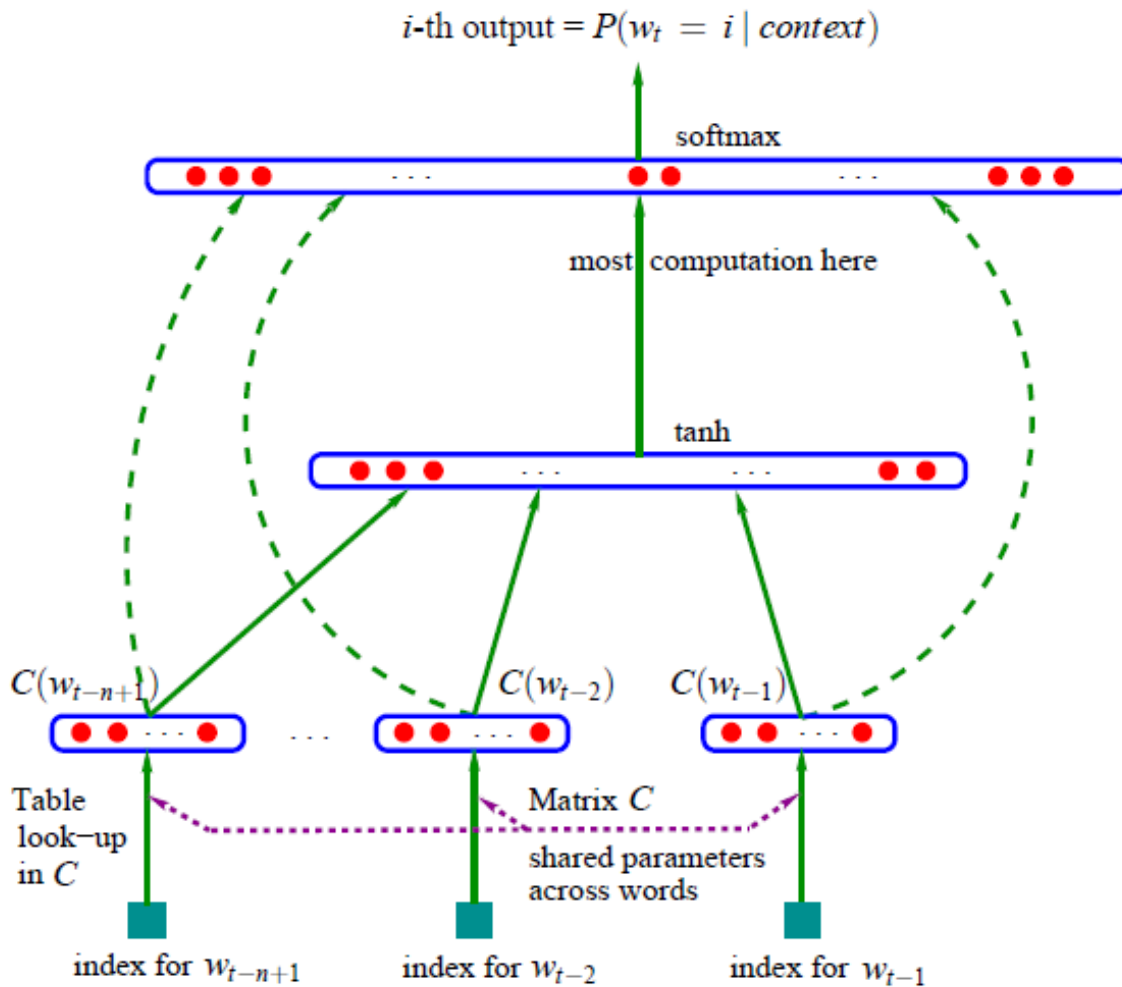
Stosowane obecnie podejścia generowania wektorowych reprezentacji słów można podzielić na dwa typy:

1. Modele predykcyjne: uczą się wektorowych reprezentacji słów poprzez zmniejszenie błędu predykcji słów należących do lokalnego kontekstu słowa i . Poniżej opisuję sztandarowy przykład takiego modelu - word2vec, gdzie sposobem na optymalizację funkcji celu jest zastosowanie płytkiej sieci neuronowej typu feed-forward optymalizowanej za pomocą metody stochastic gradient descent.
2. Metody oparte o zliczanie: generują wektory słów poprzez redukcję wymiarowości w globalnej macierzy współwystąpień słów. Jako pierwszy etap konstruują one ogromną (wymiar równa się liczbie słów w słowniku korpusu) macierz, która (podobnie, jak metodzie LSI) następnie ulega faktoryzacji, aby uzyskać macierz o mniejszym wymiarze, lecz nadal zachowującą powiązania pomiędzy słowami. Przykładem jest tu opisana poniżej metoda Global Vectors - GloVe.

Jedną z szerokiego wachlarza możliwości, jakie dają tego typu techniki jest określanie podobieństwa pomiędzy całymi dokumentami, wykorzystując dodatkowe metody pozwalające przenieść zależności między poszczególnymi słowami dokumentu na zależności między całymi zbiorami słów, co jest istotne z punktu widzenia tematu niniejszej pracy. Dwie z nich: metodę centroidu oraz Word Mover's Distance opisuję później w tym rozdziale. Warto zauważyć, iż posiadając wektorową reprezentację słów można wyznaczyć „odległość” pomiędzy dwoma dokumentami nawet, jeżeli nie posiadają one wspólnych słów.

Podejścia deep learningowe

Wspomniane podejście Bengio oparte jest o sieć neuronową typu feed-forward o jednej warstwie ukrytej zgodnie z architekturą z poniższego rysunku.



Rysunek 1.1: Neuronowy model języka. Źródło: [17].

Celem działania sieci jest maksymalizacja funkcji celu $J_\theta = \frac{1}{T} \sum_{t=1}^T \log f(w_t, w_{t-1}, \dots, w_{t-n+1})$, gdzie $f(w_t, w_{t-1}, \dots, w_{t-n+1})$ odpowiada prawdopodobieństwu $p(w_t | w_{t-1}, \dots, w_{t-n+1})$ wystąpienia słowa w_t bezpośrednio po sekwencji słów $w_{t-1}, \dots, w_{t-n+1}$. Wektorowa reprezentacja słowa uzyskiwana jest tu przez przemnożenie wejściowego wektora (wektor zer z jedynką na i -tym miejscu reprezentujący i -te słowo, „one-hot-vector”) z macierzą wag pierwszej warstwy sieci.

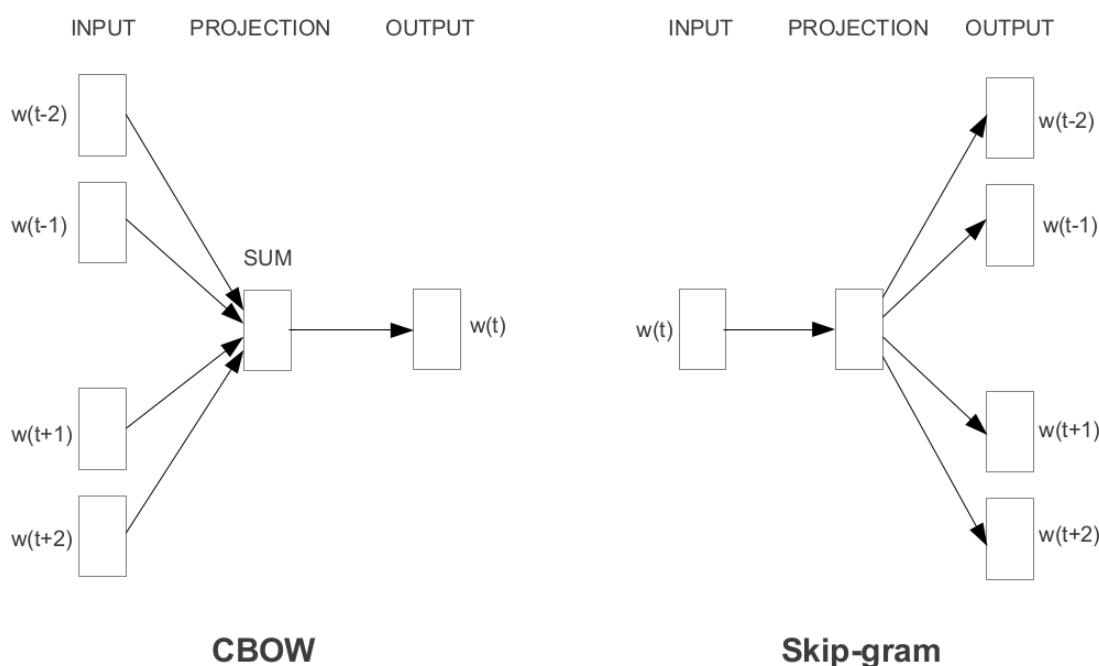
Podejście to jak i kolejne ([18]) wykorzystujące głębokie sieci neuronowe nie znalazły zastosowań komercyjnych, ponieważ ich wydajność nauki jest na tyle niska, że niemożliwe jest użycie przy ogromnych zbiorach danych wykorzystywanych w środowiskach produkcyjnych.

Rozwiązaniem tego problemu wydają się być nowe metody wektorowej reprezentacji słów powstałe na przestrzeni ostatnich lat. W odróżnieniu do metod deep learningowych opierają się one o metody szybkiej nauki, np. o płytkie sieci neuronowe, które uczą się na tyle krótko, że sprawdzają się one w zastosowaniach komercyjnych.

Word2vec

Metoda word2vec wprowadzona w 2013r. m.in. przez T. Mikolova w [16] odniosła niewątpliwy sukces w porównaniu z wcześniejszymi metodami osadzania słów w przestrzeni wektorowej. Autorzy metody proponują sieć neuronową, która podobnie jak wcześniejsze podejścia ma za zadanie odtworzyć kontekst danego słowa i na tej podstawie dokonać reprezentacji słowa jako wektora liczb rzeczywistych. Różnica jest taka, iż sieć ta ani nie jest głęboka, ani też nie zawiera nieliniowych funkcji aktywacji wykorzystywanej orzy warstwie we wcześniejszych modelach. Wyróżnia się dwie odwrotne architektury sieci:

- CBOW (continuous bag of words): na podstawie okna N sąsiednich słów sieć przewiduje słowo, którego z największym prawdopodobieństwem te N słów jest sąsiedztwem. W tym modelu funkcja celu przyjmuje postać $J_\theta = \frac{1}{T} \sum_{t=1}^T \sum_{-n \leq j \leq n, j \neq 0} \log p(w_{t+j} | w_t)$.
- skip-gram: na podstawie słowa sieć dokonuje predykcji N sąsiednich słów. Zadaniem sieci neuronowej jest wtedy optymalizacja funkcji celu postaci $J_\theta = \frac{1}{T} \sum_{t=1}^T \log p(w_t | w_{t-n}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+n})$.



Rysunek 1.2: Schemat sieci wykorzystującej podejście skip-gram i CBOW. Źródło: [16].

Używając tej stosunkowo prostej architektury można wykonać proces nauki używając milionów słów, których powiązania między sobą zostaną zachowane w systemie wag sieci neuronowej. Wady i zalety obu podejść są wymienione w [19].

W celu dalszego opisu metody word2vec wprowadzam pojęcie funkcji softmax użytej w warstwie wyjściowej sieci neuronowej. Softmax jest generalizacją funkcji logistycznej, zamieniającą K -wymiarowy wektor z dowolnych liczb rzeczywistych na K -wymiarowy wektor liczb rzeczywistych z zakresu $(0, 1]$, które sumują się do 1. Funkcja wyraża się wzorem $\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$ dla $j = 1, \dots, K$. Wyjście funkcji można traktować jako pewien rozkład prawdopodobieństwa.

W metodzie word2vec nauka polega na trenowaniu sieci neuronowej. Jednakże w odróżnieniu od innych metod wykorzystujących sieci neuronowe, Word2vec nie używa później wytrenowanej sieci jako takiej, a jedynie otrzymanych w wyniku nauki wag warstwy ukrytej sieci, które faktycznie są wynikowymi wektorami słów.

W dalszym opisie metody szczegółowo skupiam się na podejściu CBOW, lecz podejście skip-gram wygląda analogicznie.

Sieć neuronowa będąca wynikiem nauki przyjmuje na wejściu wektor binarny długości odpowiadającej liczbie słów w słowniku V zbudowanym na korpusie treningowym. Wektor ten wypełniony jest wartościami 0 oraz jedną wartością 1 na i -tej pozycji. Taki wektor odpowiada

i -temu słowu ze słownika V . Wejściem sieci są kolejne słowa z korpusu w tej właśnie reprezentacji. Wyjściem sieci jest wektor tej samej długości o wartościach rzeczywistych z zakresu $[0,1]$, w którym wartość na i -tej pozycji odpowiada prawdopodobieństwu, że i -te słowo ze słownika znajduje się w sąsiedztwie słowa wejściowego. Za „sąsiedztwo” wielkości x należy tu rozumieć zbiór złożony z x słów występujących przed danym słowem w korpusie i x słów położonych za danym słowem. Wartość x może być tu ograniczona przez początek/koniec zdania, które ograniczają kontekst danego słowa.

Jako efekt należy się spodziewać, że dla słowa wejściowego „Brytania” otrzymamy na wyjściu wysoką wartość prawdopodobieństwa dla słowa „Wielka”, a niską np. dla słowa „skoroszyt”.

Jednym z parametrów metody Word2vec jest wymiarowość przestrzeni, w której znajdują się otrzymane wektory odpowiadające słowom z korpusu. Liczba ta ma swoje źródło z wielkości warstwy ukrytej sieci neuronowej. Wagi warstwy ukrytej można interpretować jako macierz $M \times N$, gdzie M to liczba słów słownika V - wielkość wektowa wejściowego, a N to liczba neuronów w warstwie ukrytej. Po przeprowadzeniu nauki i -ty wiersz tej macierzy odpowiada wektorowi długości N , który reprezentuje i -te słowo ze słownika V .

W sieci nie jest używana funkcja aktywacji, ale prawdopodobieństwa na wyjściu są efektem działania funkcji softmax. Funkcja ta ma za zadanie sprowadzić wyjściowe wartości warstwy ukrytej do postaci rozkładu prawdopodobieństwa.

FastText

FastText[21] to biblioteka stworzona w 2016 w celu wydajnego uczenia wektorowej reprezentacji słów oraz klasyfikacji zdań. Od „klasycznego” word2vec różni się stopniem szczegółowości analizy słów. Word2vec traktuje słowo jaką najmniejszą, niepodzielną jednostkę, której wektorową reprezentację usiłuje wyznaczyć. FastText natomiast dokonuje analizy również wewnętrznej analizy słów. Wykorzystuje w tym celu rozbięcie słowa na podsłowa - ciągi znaków o określonej długości n , „character n -grams”. Np. słowo „pokój” składa się następujących 3-gramów: [pok], [okó], [kój]. Podejście takie daje szereg nowych możliwości. Pomaga wyznaczyć reprezentację wektorową rzadkich słów, które być może mają wspólny rdzeń (i znaczenie) z innymi, częściej występującymi słowami. Metoda pozwala również nadać wektorową reprezentację słowom, których w ogóle nie ma w słowniku, jako że ich podsłowa mogą należeć do słów w słowniku się znajdujących. Zalety te wydają się być szczególnie obiecujące w przypadku bogatych morfologicznie języków, np. języka polskiego, tureckiego, czy fińskiego.

Zasada działania metody bazuje na word2vec. Jednakże oprócz predykcji tylko całych słów następuje tu również predykcja n -gramów słowa a w otoczeniu słowa a . Ostatecznie słowu a

zostaje przypisany wektor składający się ze średniej oryginalnej reprezentacji wektorowej słowa oraz reprezentacji jego n-gramów.

Jak pokazuje badanie[?] metoda ta sprawdza się lepiej od word2vec w wykrywaniu syntaktycznych podobieństw pomiędzy słowami.

GloVe

GloVe[22] (GLObal Vectors) jest kolejną wartą uwagi metodą word embedding powstałą na przestrzeni ostatnich lat. Algorytm GloVe różni się od word2vec w sposobie uzyskania wektorowej reprezentacji słów. Word2vec jest modelem predykcyjnym, natomiast trening w GloVe opiera się na globalnej macierzy współwystąpień słów. Ponadto w porównaniu do word2vec GloVe stara się wyznaczyć reprezentacje wektorowe wprost, podczas gdy w word2vec dzieje się „przy okazji” - szkoli się sieć neuronową nie w celu jej dalszego wykorzystania w celu predykcji, a jedynie dla jej macierzy wag.

Algorytm GloVe składa się z następujących kroków[23]:

1. Zgromadź współwystąpienia słów w formie macierzy X . Każdy element X_{ij} takiej macierzy reprezentuje jak często słowo i występuje w pobliżu słowa j . Zazwyczaj macierz buduje się poprzez skanowanie bazowego korpusu oknem o ustalonej szerokości, w obrębie którego centralne słowo leży w kontekście słów je otaczających. Dodatkowo można tu wprowadzić wagi dla słów malejące wraz ze wzrostem dystansu od słowa centralnego.
2. Zdefiniuj ograniczenie dla każdej pary słów: $w_i^T w_j + b_i + b_j = \log(X_{ij})$, gdzie w_i oznacza wektor głównego słowa, w_j słowa leżącego w pobliżu i , b_i i b_j to skalary.
3. Zdefiniuj funkcję kosztu $J = \sum_{i=1}^V \sum_{j=1}^V f(X_{ij})(w_i^T w_j + b_i + b_j - \log X_{ij})^2$, gdzie f jest funkcją ważącą, która pomaga zapobiec uczeniu tylko na podstawie najbardziej popularnych par słów. Autorzy proponują funkcję postaci:
$$f(X_{ij}) = \begin{cases} (\frac{X_{ij}}{x_{max}})^\alpha & \text{if } X_{ij} < XMAX \\ 1 & \text{otherwise} \end{cases}$$

Celem funkcji optymalizacji funkcji kosztu jest minimalizacja różnicy pomiędzy iloczynami skalarnymi wektorów współwystępujących słów.
4. Dokonaj minimalizacji funkcji kosztu poprzez stopniową aktualizację wektorów w_i i w_j .

Odległość między dokumentami

W celu wykorzystania omówionych metod osadzania słów należy wybrać metodę obliczania odległości między całymi dokumentami, których słowa potrafimy reprezentować jako wektory. Zakładamy, że jeżeli dystans pomiędzy dokumentami jest mały, to ich tematyka jest podobna.

Centroid

Najprostszą i najbardziej intuicyjną metodą obliczenia odległości pomiędzy wektorową reprezentacją dokumentów jest wykonanie dwóch prostych kroków:

1. Uśrednienie wektorów wchodzących w skład każdego z dokumentów. Powstały w ten sposób wektor jest centroidem reprezentującym dokument w przestrzeni wektorowej.
2. Obliczenie dystansu między wektorami. Powszechnie przyjętą praktyką jest stosowanie tzw. odległości kosinusowej - znormalizowanego iloczynu skalarnego wektorów A i B . Jest to kosinus kąta pomiędzy dwoma wektorami reprezentującymi dokumenty. Zaletą tej metody jest natychmiastowa normalizacja wyniku do zakresu $(0, 1)$. Odległość $sim = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2} =$

$$\frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}, \text{ gdzie } A_i \text{ i } B_i \text{ są składowymi wektorów odpowiednio } A \text{ i } B$$

Wadą opisanej powyżej metody jest utrata potencjalnie użytecznych zależności pomiędzy poszczególnymi wektorami wchodzącymi w skład dokumentu.

W kontrze to tego prezentuję metodę obliczania dystansu między dokumentami uwzględniającą rozkład wektorów wewnątrz dokumentu.

Word Mover's Distance

Word Mover's Distance[24] to rozwiązanie zwracające odległość między dokumentami tekstowymi. W tym celu adaptuje algorytm Earth Mover's Distance[25] oraz wektorową reprezentację słów dokumentu. WMD mierzy odległość między dokumentami jako minimalny dystans jaki wektory słów pierwszego dokumentu muszą „pokonać” aby osiągnąć wartości wektorów z drugiego dokumentu.

EMD jest metodą mierzenia odległości pomiędzy dwoma rozkładami, która opiera się na minimalnym koszcie, jaki musi zostać poniesiony, aby dokonać transformacji jednego rozkładu w drugi. Problem można sformalizować jako problem programowania liniowego, gdzie: $P = \{f(p_1, w_{p_1}) \dots (p_m, w_{p_m})\}$, $Q = \{f(q_1, w_{q_1}) \dots (q_n, w_{q_n})\}$ są danymi rozkładami o m (odpowiednio n) klastrach p_i (q_j), a w_{p_i} (w_{q_j}) jest masą klastra. $D = [d_{ij}]$ jest macierzą odległości, w której d_{ij} reprezentuje odległość pomiędzy klastrami p_i i q_j . Celem jest znaleźć taki przepływ $F = [f_{ij}]$, gdzie f_{ij} to przepływ pomiędzy p_i i q_j , który minimalizuje całościowy koszt $Work(P, Q, F) = \sum_{i=1}^m \sum_{j=1}^n d_{ij} f_{ij}$ przy odpowiednich ograniczeniach[25]. EMD jest to dobrze zbadanym problemem transportowym[25], dla którego powstały efektywne metody rozwiązania[26].

Przypuśćmy, że dzięki metodzie Word2vec dla słownika V o n słowach otrzymujemy macierz $X \in \mathbb{R}^{d \times n}$. i -ta kolumna tej macierzy reprezentuje i -te słowo ze słownika V . Odległości pomiędzy wektorami reprezentującymi semantycznie zbliżone słowa są relatywnie mniejsze od odległości dla słów niezwiązanych ze sobą. Celem WMD jest zawrzeć semantyczne podobieństwo pomiędzy poszczególnymi parami słów w dystans pomiędzy całymi dokumentami. Aby to osiągnąć metoda traktuje dokument jako rozkład, którego i -tym elementem jest liczba wystąpień i -tego słowa w tym dokumencie, a następnie stosuje metodę EMD do obliczenia dystansu między tymi rozkładami. Macierz odległości D używana w metodzie EMD jest zbudowana na bazie odległości między wektorami Word2vec reprezentującymi słowa dokumentów. $d_{ij} = \|x_i - x_j\|$, gdzie i i j to indeksy słów ze słownika V a x_{ij} to element macierzy X . Autorzy metody określają złożoność metody jako $O(p^3 \log p)$, gdzie p to wielkość słownika V .

Miary oceny wyszukiwania

Razem z rozwojem systemów wyszukiwania informacji powstały miary pozwalające ocenić wyniki działania tych systemów. Do najbardziej popularnych należą „precyzja” i „zwrot”.

Precyzja to odsetek wyszukanych dokumentów, które są relewantne do zapytania; $precision = \frac{|\{relevant\} \cap \{retrieved\}|}{|\{retrieved\}|}$.

Zwrot natomiast jest liczbą wyszukanych relewantnych dokumentów w stosunku do wszystkich relewantnych dokumentów (również tych niewyszukanych); $recall = \frac{|\{relevant\} \cap \{retrieved\}|}{|\{relevant\}|}$.

Bardziej złożoną miarą lepiej charakteryzującą jakość procesu wyszukiwania jest F-miara. W celu obliczenia wyniku uwzględnia ona zarówno precyzję jak i zwrot. Miara wyraża się wzorem: $F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$. Rezultat miary można interpretować jako ważoną średnią precyzji i zwrotu.

nDCG

Powyższe miary dokonują binarnego podziału wyników wyszukiwania: na relewantne i nierelewantne. Jednakże część rekomendowanych przedmiotów może być mniej lub bardziej od innych istotnych dla użytkownika. W tym świetle listę rekomendacji można uznać za pewien ranking, którego elementy mogą być uszeregowane lepiej lub gorzej. Ocena, czy dane uszeregowanie jest dobre, jest istotnym problemem z punktu widzenia ewaluacji metod generujących elementy rankingu. Stąd zachodzi potrzeba wprowadzenia formalnej miary jakości uszeregowania elementów rankingu. Można osiągnąć poprzez rozszerzenie podstawowych metod ewaluacji opartych na binarnej ocenie relewantności, jak recall i precision.

Miara nDCG[27] (Normalized Discounted Cumulative Gain) jest miarą jakości rankingu, która opiera się na założeniu, że im bardziej relewantne wyniki, tym wyżej powinny być w rankingu, aby ranking był najbardziej wartościowy. Miara ta mierzy skumulowany „zysk” powstały poprzez umieszczenie poszczególnych przedmiotów na określonych pozycjach rankingu. Miara nDCG jest znormalizowaną wersją miary DCG (Discounted Cumulative Gain), która wyraża się wzorem: $DCG_p = \sum_{i=1}^p \frac{2^{rel_i} - 1}{\log_2(i+1)}$, gdzie p to liczba elementów rankingu, i to miejsce przedmiotu w rankingu, a rel to poziom relewantności elementu. DCG premiuje relewantne przedmioty, które są wysoko w rankingu oraz karze za relewantne przedmioty w dole rankingu. W wariacie nDCG następuje jeszcze normalizacja przez podzielenie wartości DCG rzeczywistego rankingu przez DCG idealnego rankingu zbudowanego na elementach korpusu ułożonych malejąco pod kątem relewantności: $nDCG_p = \frac{DCG_p}{IDCG_p}$, $IDCG_p = \sum_{i=1}^{|REL|} \frac{2^{rel_i} - 1}{\log_2(i+1)}$, gdzie $|REL|$ oznacza listę relewantnych przedmiotów z całego zbioru podlegającego szeregowaniu.

Dane

Dane, na których testowane były opisywane w niniejszej pracy metody otrzymałem dzięki życzliwości serwisu Allegro. Jednak, by dane te otrzymać, zobowiązany zostałem po podpisaniu umowy o poufności. Stąd, w niniejszej pracy brak jakichkolwiek przykładów danych poza tymi dostępnymi za pośrednictwem strony internetowej Allegro. W niniejszym rozdziale opisuję strukturę i stan pozyskanych danych oraz proces ich wstępnego przetwarzania i nanalizy w celu zastosowania na nich zaadaptowanych metod opisanych w rozdziale poprzednim.

Opis danych

Otrzymane dane to baza ok. 20000 artykułów tekstowych w formacie JSON. Są to te same artykuły, które są dostępne dla użytkowników poprzez serwis internetowy (stan na styczeń 2017). Pojedynczy rekord danych składa się z głównej treści artykułu oraz z metadanych, z których za istotne z punktu widzenia tematu pracy uznałem pola: id, kategoria i słowa kluczowe.

Treść artykułu

Treść każdego artykułu składa się z trzech pól: „zawartość”, „nagłówek” i „tytuł”. Średnia długość artykułu to 821 słów, w tym nagłówek to jednozdaniowy wstęp. Średnią tę estymuję na podstawie średniej liczby znaków artykułu i średniej długości słowa w języku polskim. Dokładne statystyki tekstu będą dostępne dopiero po wstępnym przetwarzaniu.

Wszystkie artykuły napisane są w języku polskim, w nielicznych przypadkach wykryłem błędy, tzw. „literówki”. Jako, że artykuły ze zbioru dotyczą produktów sprzedawanych za pośrednictwem serwisu Allegro, w skład słownika zbudowanego na ich bazie wchodzi wiele słów specyficznych dla różnych branż. Są to m.in. nazwy modeli aparatów (np. „Sony Alpha 77 II”), samochodów, gier komputerowych, a także nazwy techniczne: „sprężarka”, „hipertoniczny”, „autofocus”.

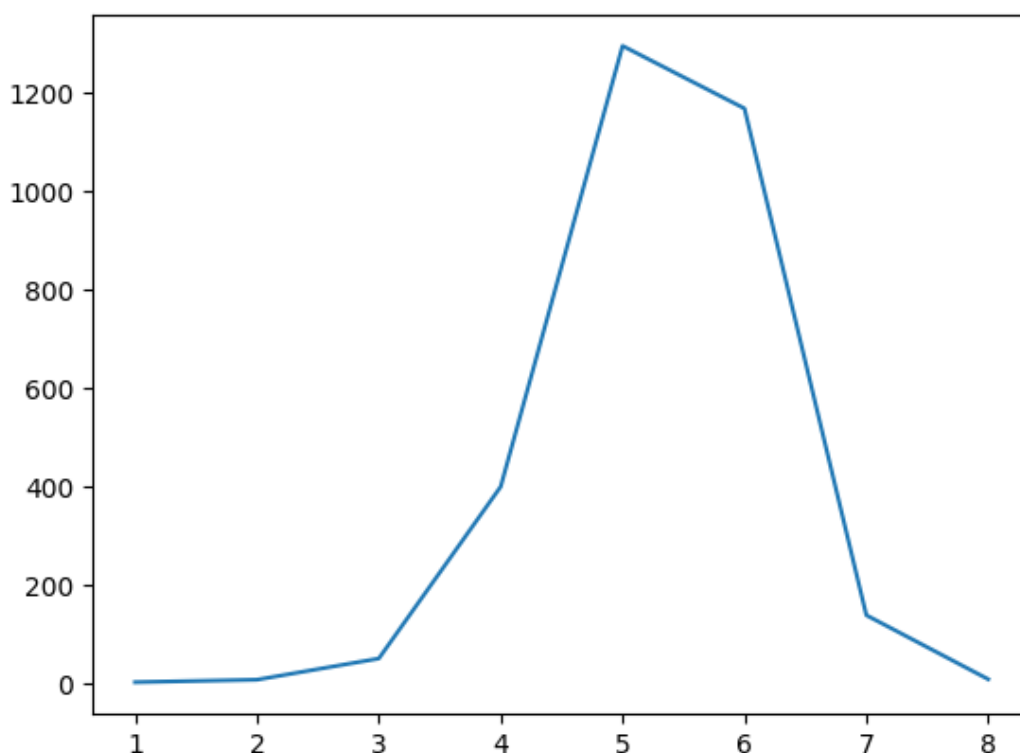
Artykuły posiadają w swej treści wiele znaczników interpretowanych przez system, na podstawie których wzbogacana jest warstwa wizualna strony internetowej zawierającej artykuł, np. obrazki czy łącza do ofert związanych z tematem artykułu.

2.1. OPIS DANYCH

Spójność danych oceniam na wysoką, tj. każde pole zawarte w strukturze dokumentu jest zawsze wypełnione - brak jest wartości typu NULL.

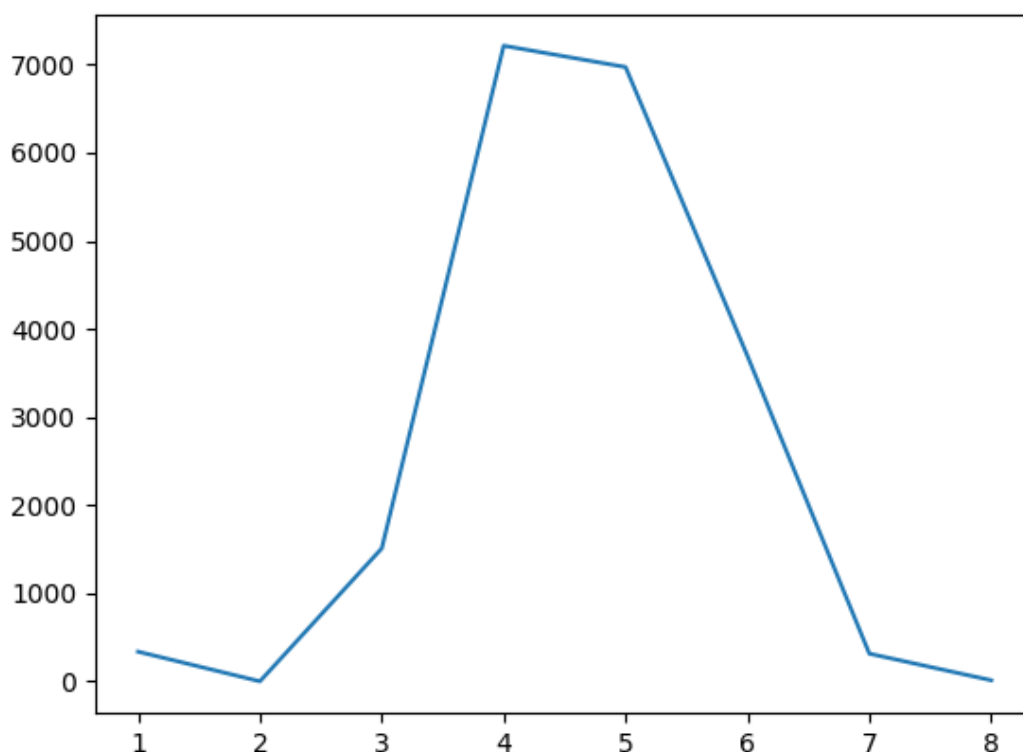
Kategoria

Każdy artykuł został przez autora przydzielony do pewnej kategorii, która odpowiada tematyce artykułu, np. „Aparaty cyfrowe” czy „Przyprawy i zioła”. W skład pola „kategoria” wchodzi również lista kategorii nadrzędnych, a cała hierarchia kategorii ma strukturę drzewiastą. Np. kategoria nadrzędna dla kat. „Przyprawy i zioła” to „Delikatesy”, a dla kat. „Delikatesy” to „Dom i zdrowie”. Każdy artykuł należy do tylko jednej kategorii będącej dowolnym węzłem w drzewie (nie tylko liściem). Drzewo kategorii posiada 8 poziomów, a najwięcej węzłów znajduje się na poziomach 5 i 6.



Rysunek 2.1: Liczba kategorii na poszczególnych poziomach drzewa.

Najwięcej artykułów z kolei jest przypisanych do kategorii z poziomów 4 i 5, średnio artykuł jest przypisany do kategorii z poziomu 4,63. Średnio na kategorię będącą liściem w drzewie przypada 7,71 artykułu.



Rysunek 2.2: Liczba artykułów, dla których kategoria na danym poziomie drzewa jest tą najbardziej szczegółową.

Po wstępnej analizie system kategorii oceniam jako spójny i rzetelny. Uważam, że można użyć go jako punkt odniesienia przy konstrukcji metod ewaluacji testowanych technik określania podobieństwa między artykułami.

Słowa kluczowe

Do każdego artykułu dołączona jest lista słów kluczowych. Są to wyrażenia złożone z jednego lub kilku słów, które mają za zadanie scharakteryzować w skrócie jego zawartość, np. „aparaty”, „aparaty cyfrowe”, „lustrzanki”, „Sony”. Pole to jest wykorzystywane w dotychczasowym mechanizmie generowania rekomendacji - artykuły podobne do danego są wyszukiwane na podstawie jego słów kluczowych. Przyglądając się wszystkim słowom kluczowym zestawionym razem zauważyłem pewne niespójności: część słów kluczowych o tej samej treści pisana jest w inny sposób, np. różną wielkością liter, czy używając myślnika zamiast spacji. Wynika to zapewne z faktu przypisywania słów kluczowych samodzielnie przez autorów w oderwaniu od słów przypisanych

do reszty artykułów. Średnio każdy artykuł ma przypisane 5,8 słów kluczowych, natomiast unikalnych słów kluczowych w skali całego korpusu jest 60403.

Wstępne przetwarzanie danych

W celu zwiększenia skuteczności metod analizy tekstu stosuje się wstępne przetwarzanie danych. Ma ono na celu takie przygotowanie tekstu, aby zmaksymalizować jakość wyników operujących na nim później algorytmów. Techniki wstępnego przetwarzania tekstu nie wchodzą w skład żadnego standardu - dobieram je indywidualnie do konkretnego przypadku, zgodnie z intuicją.

Niżej opisuję kolejne kroki wstępnego przetwarzania tekstu, które wykonuję na posiadanym zbiorze artykułów.

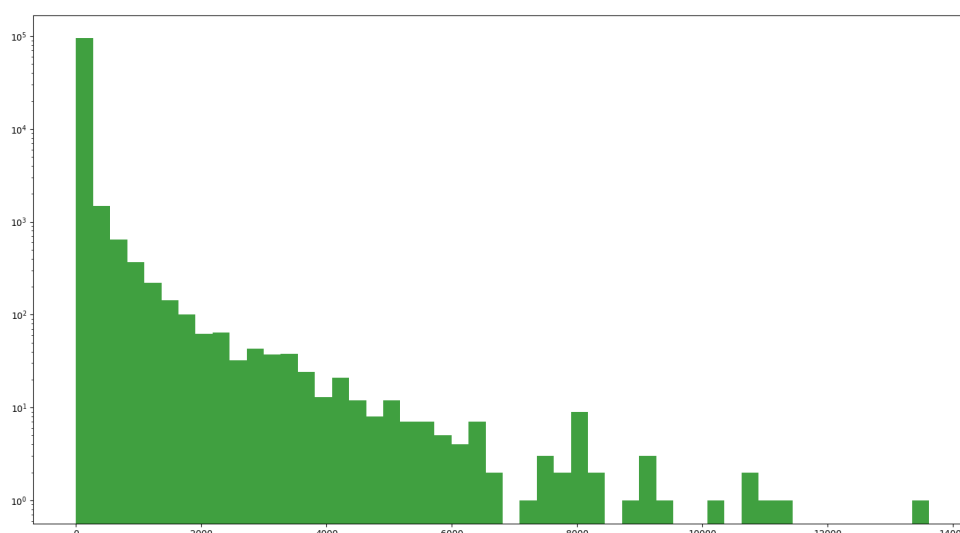
1. Oczyszczanie tekstu ze zbędnych, wspomnianych wcześniej znaczników. Z punktu widzenia semantycznej analizy tekstu są one bezużyteczne, czy wręcz szkodliwe (powodują pewne „zanieczyszczenie” tekstu). Stąd usuwam je wykorzystując odpowiednio skonstruowane wyrażenia regularne.
2. Usunięcie „słów stopu”(ang. stopwords) - na ogół krótkich słów nie wnoszących nic do znaczenia całości artykułu. Są to np. „w”, „z”, „ponieważ”. Ich usunięcie zmniejsza liczbę słów dokumentu skracając tym samym czas jego przetwarzania. Jako że słowa te występują często, usunięcie ich daje możliwość uwypuklenia znaczenia innych słów mających wpływ na rzeczywiste znaczenie całego artykułu. Zbiór słów stopu czerpię z [28].
3. Sprowadzenie wszystkich słów dokumentu do małych liter. Pomaga to ujednolicić postać części słów o tym samym znaczeniu, wśród których jedno występuje na początku zdania a inne w środku.
4. Rozbicie słów połączonych myślnikiem. Doświadczenie w późniejszym etapie (tokenizacji) pokazuje, że narzędzie jej dokonujące (Morfologik[29]) nie radzi sobie z tego typu słowami (np. „biało-czerwony”) i zostania je w niezminionej postaci gramatycznej (np. „biało-czerwonego”). Stąd konieczność ręcznego wykoania przeze mnie mechanizmu rozbijającego takie słowa do postaci kompatybilnej z tokenizerem. Do wykonania odpowiedniej funkcji potrzebna była wcześniejsza analiza tego typu słów pod kątem zachowania obu członów w zależności od ich rodzaju, przypadku i występowania konkretnych liter w sufiksach słów składowych. Zależało mi także, aby nie rozбивać słów będących nazwami własnymi, czy symbolami urzędów.

5. Tokenizacja. Jest to najistotniejszy element całego procesu. Polega na sprowadzaniu słów o tym samym znaczeniu, a różnej formie gramatycznej do tej samej postaci. Sporym utrudnieniem jest tutaj stopień skomplikowania języka polskiego oraz liczba wyjątków, jaką ten język posiada. Za przykład może posłużyć słowo „mieć”, którego jedna z form to „ma”, kolejna to „miej”. Celem etapu jest sprowadzenie każdego z tych wyrazów do formy podstawowej „mieć”. Do przeprowadzenia tej operacji stosuję narzędzie Morfologik[29].

Użycie wymienionych technik nie jest jedynym standardem a wynikiem analizy przetwarzanych danych i techniki te zostały dobrane dla tego konkretnego przypadku

Opis danych po wstępnym przetwarzaniu

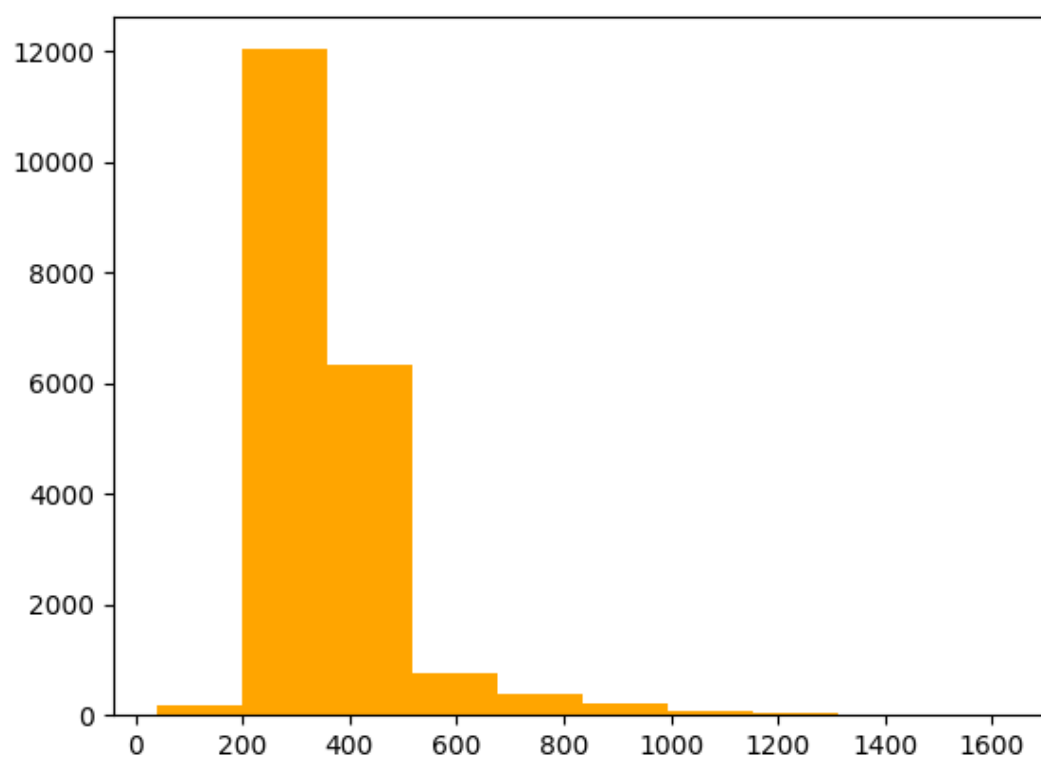
Powyższe kroki doprowadzają dane do stanu, w którym można zastosować techniki semantycznej analizy tekstu. Słownik zbudowany na wstępnie przetworzonym korpusie zawiera 98174 unikalnych słów, oraz 7409145 wszystkich słów (z powtórzeniami).



Rysunek 2.3: Histogram liczby wystąpień słów w korpusie w skali logarytmicznej.

Większość artykułów okazała się być podobnej długości, średnia długość artykułu to 370 słów.

2.3. OPIS DANYCH PO WSTĘPNYM PRZETWARZANIU



Rysunek 2.4: Histogram długości artykułów.

Metody ewaluacji

W celu porównania stosowanych metod wyznaczania podobieństwa między artykułami konieczna jest formalizacja pewnych miar tego podobieństwa. W opisie znanych ogólnych miar posłużyłem się pojęciem „relewantności” — formalną wartością wyrażoną za pomocą liczb rzeczywistych. Jednakże w praktyce rzadko dysponuje się wartością, na ile dany element rankingu jest adekwatny do zapytania, generującego ów ranking.

Ewaluacja rankingu, w którym trafność wyników zależy od zachowania realnych użytkowników jest zadaniem nietrywialnym. Podobieństwo artykułów napisanych w języku naturalnym jest rzeczą subiektywną. W sytuacji idealnej dysponowałbym obiektywną miarą podobieństwa pomiędzy parami N artykułów (np. wyznaczoną wcześniej przez miarodajną grupę użytkowników), które to N artykułów stanowiłoby zbiór testowy. Uzyskanie takich danych wiąże się jednak z dużymi kosztami i leży poza moimi możliwościami.

Inną praktyką umożliwiającą obiektywną ocenę, wykorzystywaną w działających systemach są tzw. testy A/B polegające na podziale użytkowników na grupy i zaaplikowaniu każdej grupie innego rozwiązania. Następnie mierzone są pewne wskaźniki wśród każdej grupy (w naszym przypadku np. liczba „kliknięć” prawdziwych użytkowników w artykuły rekomendowane) i spośród zgromadzonych wyników wybierane jest rozwiązanie najlepsze.

Z powodu braku możliwości wykorzystania do ewaluacji rozwiązań rzeczywistych użytkowników serwisu internetowego jestem zmuszony wprowadzić własne miary oparte na dostępnych danych. Należy tu zaznaczyć niedoskonałość wprowadzanych miar, ponieważ każda z nich opiera się na pewnych założeniach, od których prawdziwości zależy jakość całej miary.

Testowane metody adaptuję tak, aby na podstawie pewnego artykułu bazowego otrzymywać listę artykułów podobnych do niego uszeregowanych pod kątem relewantności malejąco. Takie działanie można sformalizować w postaci funkcji $S_p : a_j \rightarrow \{a_i\}_{i < p}$, gdzie a to artykuł, a p to liczba elementów zwracanego ciągu. Funkcja S przyjmuje artykuł tekstowy i zwraca skończony ciąg artykułów do niego podobnych zgodnie ze stopniem dopasowania (najlepsze na początku). Celem działania niżej opisanych miar jest każdej parze postaci: artykuł wejściowy-jeden z wyznaczonych artykułów podobnych przypisać ocenę tego podobieństwa — relewantność. Np. metoda

3.1. MIARA 1: DYSTANS OPARTY NA METADANYCH

S dla artykułu X zwraca ciąg Y_1, Y_2, \dots . Dla każdej z par za pomocą poniższych metod ewaluacji można określić relewantność np. $rel(X, Y_1) = a_1, rel(X, Y_2) = a_2$ itd. Następnie wyniki dla metody S i wejścia X należy zagregować. Dokonuję tego na dwa sposoby.

- Obliczenie średniej $\frac{1}{p} \sum_{i=1}^p relevance(X, Y_i)$.
- Użycie metody nDCG, gdzie relewantność to wynik poszczególnych ewaluacji podobieństwa artykułów.

Oceny dla konkretnej metody, dla ustalonej próby par artykułów są następnie wykorzystane do wyznaczenia wartości średniej miary nDCG tej metody.

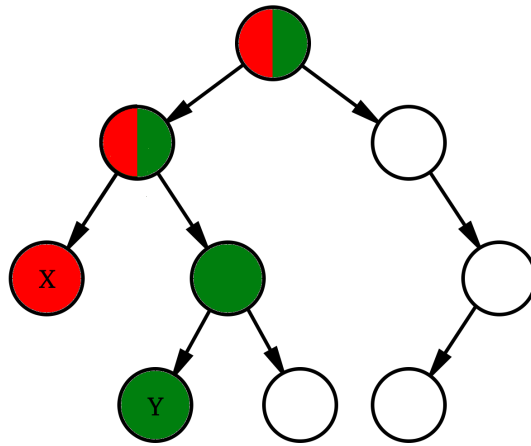
Miara 1: Dystans oparty na metadanych

Jak wspomniałem wcześniej dane prócz treści artykułów zawierają również pewne metadane, a wśród nich umożliwiające tworzenie powiązań między artykułami. Skupiam się tu na polach: „słowa kluczowe” i „kategoria”.

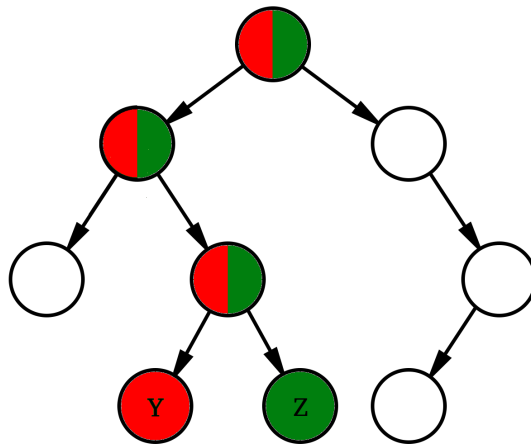
Kategorie

Pierwszą zastosowaną miarą, pozwalającą ocenić jakość dopasowania podobnych artykułów jest ich odległość we wcześniej wspomnianym drzewie kategorii. Formalnie wartość miary to długość części wspólnej ścieżek od korzenia drzewa kategorii do węzłów reprezentujących artykuły. Im więcej wspólnych przodków w drzewie, tym bardziej podobne do siebie są artykuły reprezentowane przez węzły drzewa. Zaletą miary jest fakt, iż przypisanie artykułu do kategorii zostało wykonane przez autora, którego można określić ekspertem w dziedzinie tematyki artykułu. Stąd przynależność artykułu do danej kategorii jest mocno uzasadniona. Kolejną zaletą tej miary jest fakt, iż można ją zastosować automatycznie - wiedza ekspercka jest już zapisana w metadanych artykułów. Należy zaznaczyć tu jednak, że miara nie jest idealna - każdy artykuł należy do tylko jednego liścia drzewa kategorii. Stąd artykuł poruszający zagadnienia z różnych obszarów, który można by przypisać dwóm stosunkowo odległym kategoriom A i B , zostanie przypisany tylko do jednej kategorii, np. A . Miara pokaże wtedy dużą odległość od artykułów z kategorii B , co nie jest prawdą.

Za przykład mogą posłużyć drzewa na poniższych rysunkach. W drzewie 1. artykuły X i Y mają dwie wspólne kategorie, stąd $relevance(X, Y) = 2$. Natomiast artykuły Y i Z z drugiego drzewa mają trzech wspólnych przodków, stąd $relevance(Y, Z) = 3$. Miara wskazuje, że artykuły X i Y są do siebie mniej podobne, niż artykuły Y i Z .



Rysunek 3.1: Drzewo kategorii dla przykładu 1.



Rysunek 3.2: Drzewo kategorii dla przykładu 2.

Słowa kluczowe

Kolejna miara oparta na metadanych artykułów korzysta ze słów kluczowych. Wśród ogółu słów kluczowych występują niespójności, których większość zlikwidowałem poprzez sprowadzenie słów od małych liter oraz usunięciu „słów stopu”. Po tej unifikacji liczba unikalnych słów kluczowych to 58565.

Niniejsza miara działa podobnie do poprzedniej opartej o kategorie. Para artykułów otrzymuje 1 punkt za każdą wspólną parę posiadanych kategorii. Np. dla artykułu X o słowach kluczowych {„kosmetyki”, „kremy”, „zmarszczki”} oraz Y o słowach kluczowych {„kosmetyki”, „zeledowlosow”, „kremy”, „szampony”} $relevance(X, Y) = 2$, ponieważ oba artykuły mają dwa wspólne słowa kluczowe („kosmetyki” i „kremy”).

Miara 2: Historyczna aktywność użytkowników serwisu

Zbieranie a następnie przechowywanie informacji o aktywności użytkownika w ramach serwisu internetowego jest powszechną praktyką. Proces ten pozwala na analizę zachowania użytkowników co może doprowadzić do wniosków, jakie usprawnienia należy przedsięwziąć, aby spełnić cele biznesowe. Jednym z przykładów aktywności użytkownika zapisywanej przez serwis Allegro są kliknięcia w linki znajdujące się na stronie internetowej. Informacja ta pozwala sporządzić jeszcze jedną miarę jakości dopasowania podobnych do siebie artykułów. Postać danych, jakie udało mi się uzyskać z serwisu to tabela o polach: adres strony, na której nastąpiło kliknięcie, adres strony, na którą prowadzi link, data kliknięcia.

Jak już zostało opisane powyżej strona z artykułem tekstowym zawiera odnośniki do innych artykułów poruszających tematykę podobną do danego. Skoro zapisywana jest informacja o przejściach pomiędzy podstronami serwisu, to można obliczyć ile razy z artykułu X dokonano przejścia na rekomendowany do niego artykuł Y_1 , a ile razy na rekomendowany artykuł Y_2 . Jeżeli liczba przejść na artykuł Y_1 jest większa niż na Y_2 , można wnioskować, iż Y_1 wydaje się być bardziej relevantną rekomendacją dla artykułu X .

Posługując się powyższym założeniem, można zaproponować miarę jakości rekomendacji generowanych przez testowane metody w odniesieniu do popularności rzeczywistych rekomendacji wyekstrahowanej z danych serwisu o aktywności użytkowników.

W tym celu dokonuję adaptacji miary nDCG. Załóżmy, że testowana metoda A zwraca pewien ciąg p artykułów $c_A = a_1, a_2, \dots, a_p$ podobnych do danego artykułu x , w kolejności od najbardziej relevantnego. Załóżmy również, część elementów ciągu c_B artykułów rekomendowanych w serwisie dla x (używaną dotychczas w serwisie metodą B) znajduje się również w ciągu c_A , tj. $(\exists a_i, a_j \in c_A) a_i \in c_B \wedge a_j \in c_B$, gdzie i, j to indeksy w ciągu c_A . Załóżmy ponadto, że z danych o kliknięciach użytkowników w linki w ramach serwisu wiadomo, że przejście z x na a_i jest bardziej popularne niż przejście z x na a_j . Stąd jeżeli $i < j$ ($i > j$), to jakość działania metody A jest dobra (zła), bo metoda ta generuje podobne artykuły w kolejności zgodnej ze stopniem podobieństwa z artykułem bazowym, opartym o częstość przejść użytkowników między artykułami.

Za miarę relevantności używaną przez nDCG przyjmuję liczby rzeczywistych przejść pomiędzy artykułami, a samą metodę stosuję tylko do przecięcia zbioru artykułów podobnych do danego generowanych przez daną metodę ze zborem artykułów rekomendowanych do danego przez dotychczasową metodę działającą w serwisie.

Zaletą metody jest, iż można ją zastosować automatycznie, lecz jest zależna od danych analitycznych pochodzących z serwisu, które są niedoskonałe.

Miara 3: Ocena przez użytkowników offline

Ostatnią opracowaną przez mnie miarą jest subiektywna ocena ekspercka. W celu obiektywizacji oceny, ewaluacja powinna być dokonana przez reprezentatywną grupę T osób operujących na tych samych danych. W metodzie tej grupa użytkowników dokonuje oceny podobieństwa par artykułów generowanych przez poszczególne testowane metody genreowania rekomendacji.

1. Wybieram n losowych artykułów bazowych $b_j, j = 1, \dots, n$.
2. Testowaną metodą M generuję p artykułów $s_{jk}, k = 1, \dots, p$ podobnych do każdego b_j z n wcześniej wylosowanych.
3. Grupuję artykuły w pary: artykuł bazowy b_j — artykuł podobny s_{jk} .
4. Za pomocą stworzonego interfejsu prezentuję każdemu z użytkowników kolejno wygenerowane pary w postaci rzeczywistych stron z artykułami dostępnymi przez przeglądarkę internetową. Co ważne każdy użytkownik otrzymuje ten sam zestaw par. Podczas prezentacji pary użytkownik za pomocą interfejsu webowego ocenia podobieństwo pomiędzy artykułami $rel(b_j, s_{jk})$ w skali 1-10.
5. Dla każdej pary obliczam średnią ocen wszystkich T użytkowników $avg(j, k) = \frac{1}{T} \sum_t rel_t(b_j, s_{jk})$, gdzie rel_t to ocena t -go użytkownika.
6. Dla każdego artykułu bazowego b_j obliczam $m(j) = \frac{1}{\sum_{i=1}^p i} \sum_{i=0}^k avg(j, k) * (p - i)$ będące średnią ważoną ocen, gdzie każdy kolejny artykuł podobny otrzymuje coraz mniejszą wagę.
7. Oceną testowanej metody jest średnia $\frac{1}{n} \sum_j m(j)$.

Wadą tej metody jest jej powolność i potrzeba zaangażowania dodatkowych osób dokonujących ewaluacji. Niemożliwym wydaje się przeprowadzenie badania dla wszystkich artykułów, stąd konieczny jest wybór losowej próby artykułów, które parami poddane zostaną ocenie pod kątem podobieństwa.

Opis testów

W przeprowadzanych testach staram się dokonać porównania pomiędzy różnymi konfiguracjami tej samej metody oraz pomiędzy najlepszymi wariantami różnych metod. W tym celu wykorzystuję wprowadzone miary ewaluacji. W większości przypadków używam miar automatycznych bazujących na metadanych artykułów oraz historycznej aktywności użytkowników serwisu. Do ostatecznego porównania pomiędzy najlepszymi wariantami testowanych metod stosuję również miarę opartą na ocenie eksperckiej. Miara ta daje najbardziej rzetelne wyniki, jednakże jej użycie jest wyjątkowo kosztowne - wymaga zaangażowania osób testujących oraz sporych nakładów czasowych. Stąd zdecydowałem na użycie jej tylko w jednym przypadku.

Testowane metody generowania rekomendacji

Wykonuję szereg testów adaptacji metod semantycznej analizy języka naturalnego o różnych nazwach, w różnych konfiguracjach. Stąd dla czytelności wprowadzam niekiedy w nawiasach skrócone sygnatury tych metod, które stosuję w dalszej części pracy zamiast pełnych opisów.

Metody oparte o modelowanie tematu

Najważniejszym hiperparametrem metod tej grupy: LSI oraz LDA jest liczba tematów, stąd dokonuję testu jak zmiana tego parametru wpływa na jakość modelu.

Metody oparte o word embedding

Wykonuję testy adaptacji metod wektorowej reprezentacji słów dla wielu konfiguracji. Poniżej obszary, na których dokonuję zmian konfiguracji.

Korpus bazowy

Pierwszy model word2vec, z jakim miałem styczność to model[30] stworzony m.in. przez dr inż. M. Piaseckiego z Politechniki Wrocławskiej dostępny poprzez stronę internetową. Model ten był

uczony na korpusie Słownosieci ver. 10[31]. Zgodnie z opisem autorów dane przed uczeniem przeszły segmentację, lematyzację i ujednolicanie morfosyntaktyczne. Użyte parametry uczenia word2Vec: metoda skip gram, wektry długości 100, okno kontekstu wielkości 5.

Model ten zawiera 73875 spośród 98174 (75%) unikalnych słów korpusu artykułów Allegro oraz 7313915 z 7409145 (99%) wszystkich słów korpusu artykułów. Wskazuje to, iż słowa nieobecne w modelu są bardzo mało popularne w korpusie artykułów (stanowią ok 1% całości). Po samodzielnym sprawdzeniu stwierdzam, że słowa nieobecne w modelu to: „literówki” lub słowa niepoprawnie stokenizowane (np. „urządzeia”), symbole marek produktów (np. „ux305fa”, „i7-4700qm”), żargon branżowy (np. „bootsów”), złożenia wyrazów (np. „kurzoodporne”), wyrazy obce lub ich spolszczenia (np. „thermoprotect”). Uważam, iż mimo niewielkiej liczby tych słów w stosunku mogą mieć one znaczący wpływ na semantykę artykułów.

W związku z powyższym stwierdzeniem wykonuję naukę modelu word2vec również na własnym korpusie artykułów w celu zawarcia brakujących w poprzednim modelu słów. Najrozsądniejszym postępowaniem byłoby tutaj rozszerzenie modelu opartego na korpusie Słownosieci również o brakujące słowa, jednak metoda word2vec nie pozwala na dodanie nowych słów do słownika istniejącego modelu, a jedynie na dalszą naukę w oparciu o słowa już istniejące w słowniku. Siłą rzeczy model zbudowany na zbiorze artykułów zawiera wszystkie słowa występujące w artykułach słowa. Korpusu tego używam do tworzenia modeli w oparciu o wszystkie pozostałe sposoby.

Dokonuję porównania jakości modelu uczonego na korpusie Słownosieci z modelem uczonym na korpusie artykułów Allegro.

Sposób generowania wektorów i długość wektorów

Do wygenerowania wektorów reprezentujących słowa używam metod wektorowej reprezentacji tekstu: word2vec, GloVe oraz fastText. Każdą z nich testuję pod kątem różnej wymiarowości generowanych wektorów.

Metoda określenia podobieństwa między wektorowymi reprezentacjami dokumentów

Zadałem metod „word embedding” jest wygenerowanie wektorowej reprezentacji słowa. W celu porównania całych dokumentów i wyznaczenia podobieństw między nimi należy użyć dodatkowych środków. W tym celu korzystam z poniższych metod:

1. odległość cosinusowa pomiędzy centroidami wektorowej reprezentacji tekstów (w skrócie metoda centroidu),
2. Word Mover’s Distance,

3. autorska metoda wykorzystująca odległość cosinusową pomiędzy wektorową reprezentacją słów kluczowych wyznaczonych metodą LDA. Celem metody jest zmniejszenie wymiarowości artykułu oraz użycie tylko znaczących słów przy obliczaniu centroidu artykułu. W tej metodzie za pomocą LDA wyznaczam słowa kluczowe, które z największą wagą przynależą do danego artykułu. Nauczony model LDA każdemu dokumentowi i przypisuje zestaw tematów z określonymi wagami przynależności t_{ij} oraz każdemu tematowi j przypisuje zestaw słów wraz z wagami w_{jk} . Na tej podstawie dla każdego artykułu i mogę wybrać p słów, które posiadają najwyższą wagę przynależności do artykułu liczoną wg. wzoru $x(i, k) = \sum_j t_{ij} * w_{jk}$, gdzie k to dane słowo. Ostatecznie artykuł reprezentuję jako centroid n słów o najwyższym x dla danego słowa. W dalszych testach przyjmuję n jako 10.

Powyższe metody stosuję w celu wyznaczenia dla danego artykułu a p artykułów najbardziej do niego podobnych uszeregowanych mając pod względem relewantności do artykułu bazowego a .

Elasticsearch

W celu porównania z metodami semantycznymi testuję również dotychczasowe zapytanie do silnika Elasticsearch używane dotychczas w Allegro. Zapytanie to odpowiada listą p artykułów najbardziej zbliżonych do artykułu wejściowego w kolejności od najbardziej podobnego

Metoda losowa

W celu posiadania punktu odniesienia dokonuję testów dla metody losowo wybierającej p podobnych do danego artykułów. Wybór następuje zgodnie z rozkładem jednostajnym ze zbioru wszystkich artykułów.

Metody ewaluacji

W celu wykonania oceny testowanych metod wykorzystuję wszystkie metody ewaluacji opisane w rozdziale 4, które opatruję skróconymi sygnaturami.

1. *clicks* - ocena na podstawie historycznej aktywności użytkowników mierzona na podstawie liczby kliknięć w odnośniki.
2. *mut_cat[_ndcg]* - relewantność wyszukanych artykułów liczona na podstawie liczby wspólnych kategorii z artykułem bazowym. Stosuję dwa warianty: średnia relewantność wyszukiwanych artykułów oraz miara nDCG.

3. *mut_kw[_ndcg]* - relewantność wyszukanych artykułów liczona na podstawie liczby wspólnych słów kluczowych z artykułem bazowym. Również stosuję dwa warianty: średnia relewantność wyszukanych artykułów oraz miara nDCG.
4. *users* - ocena na podstawie eksperckiej oceny użytkowników. W badaniu wykorzystałem 5 użytkowników operujących każdy na tym samym zbiorze par testowych. Pary zostały wygenerowane (zgodnie z wcześniejszym opisem metody) na podstawie 50 artykułów bazowych wylosowanych spośród wszystkich artykułów udostępnionych mi przez Allegro.

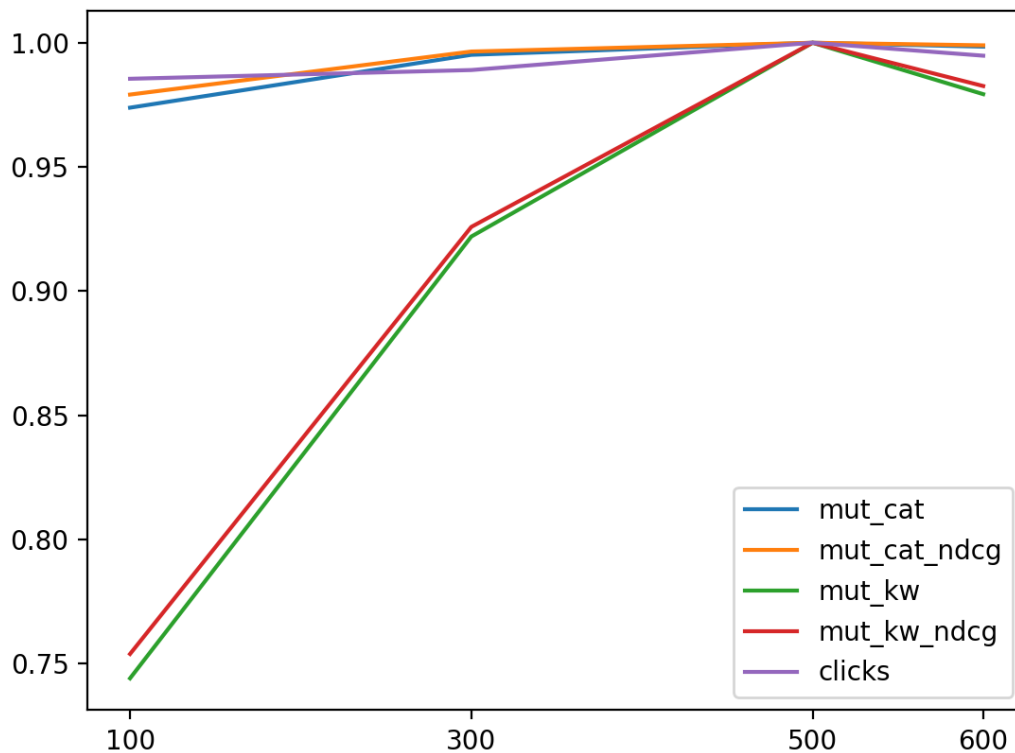
Wyniki badań

W rozdziale tym przedstawiam wyniki oceny jakości poszczególnych metod w zależności od ich hiperparametrów. Oceny dokonuję na podstawie opisanych uprzednio miar. Do wyników każdego testu dołączam ich interpretację oraz wnioski.

LSI w zależności od liczby tematów

Dokonuję porównania jakości modelu LSI w zależności od wartości hiperparametru - liczby tematów.

| | 100 | 300 | 500 | 600 |
|--------------|-----------|----------|----------|----------|
| mut_cat | 3.43131 | 3.50634 | 3.52346 | 3.51793 |
| mut_kw | 1.77666 | 2.20166 | 2.38808 | 2.33861 |
| mut_cat_ndcg | 0.507238 | 0.516217 | 0.518053 | 0.517515 |
| mut_kw_ndcg | 0.0972985 | 0.119504 | 0.129079 | 0.126829 |
| clicks | 0.865496 | 0.868586 | 0.878243 | 0.87366 |



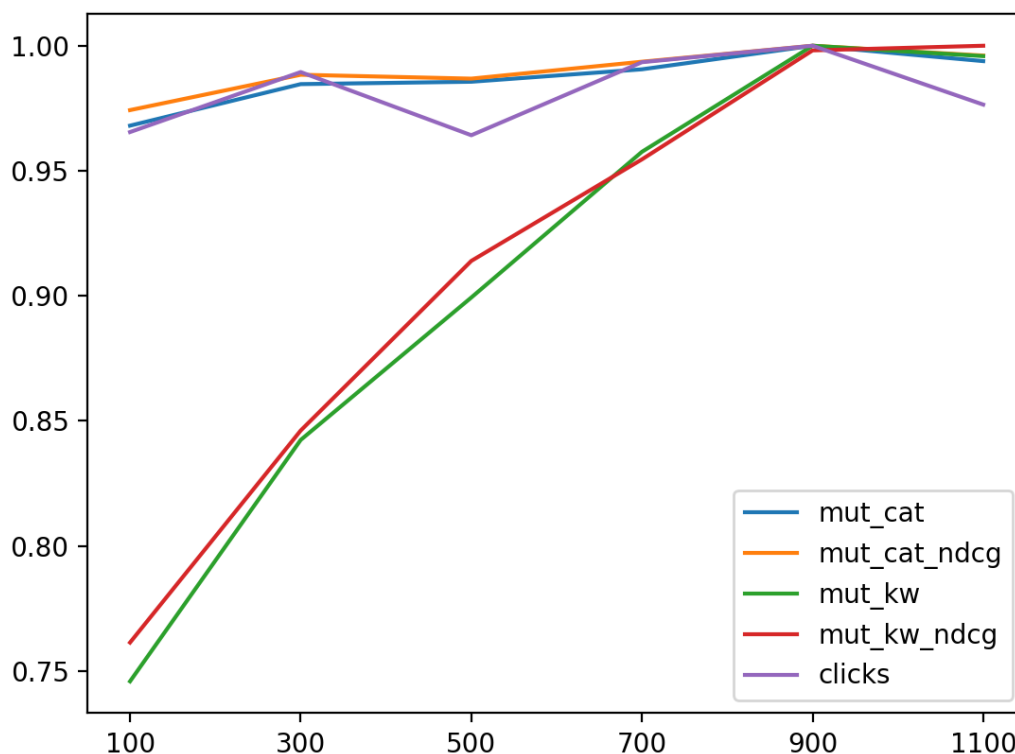
Rysunek 5.1: Porównanie znormalizowanych wyników dla metody LSI.

Miary oparte na liczbie wspólnych słów kluczowych (*mut_kw...*) wyraźnie pokazują wzrost jakości modelu przy wzroście liczby tematów. Średnia ze wszystkich miar pokazuje, że najwyższą jakość model osiąga dla liczby tematów 500.

LDA w zależności od liczby tematów

Podobnie, jak w poprzednim punkcie dokonuję porównania jakości modelu LDA w zależności od wartości hiperparametru - liczby tematów.

| | 100 | 300 | 500 | 700 | 900 | 1100 |
|--------------|-----------|-----------|-----------|-----------|-----------|-----------|
| mut_cat | 3.29108 | 3.34755 | 3.35077 | 3.36767 | 3.39972 | 3.3788 |
| mut_cat_ndcg | 0.48793 | 0.495011 | 0.494236 | 0.497623 | 0.500823 | 0.498854 |
| mut_kw | 1.31741 | 1.48792 | 1.58854 | 1.69146 | 1.76637 | 1.75904 |
| mut_kw_ndcg | 0.0733675 | 0.0815366 | 0.0880742 | 0.0919873 | 0.0961874 | 0.0963659 |
| clicks | 0.880891 | 0.90278 | 0.879733 | 0.906384 | 0.912387 | 0.890913 |



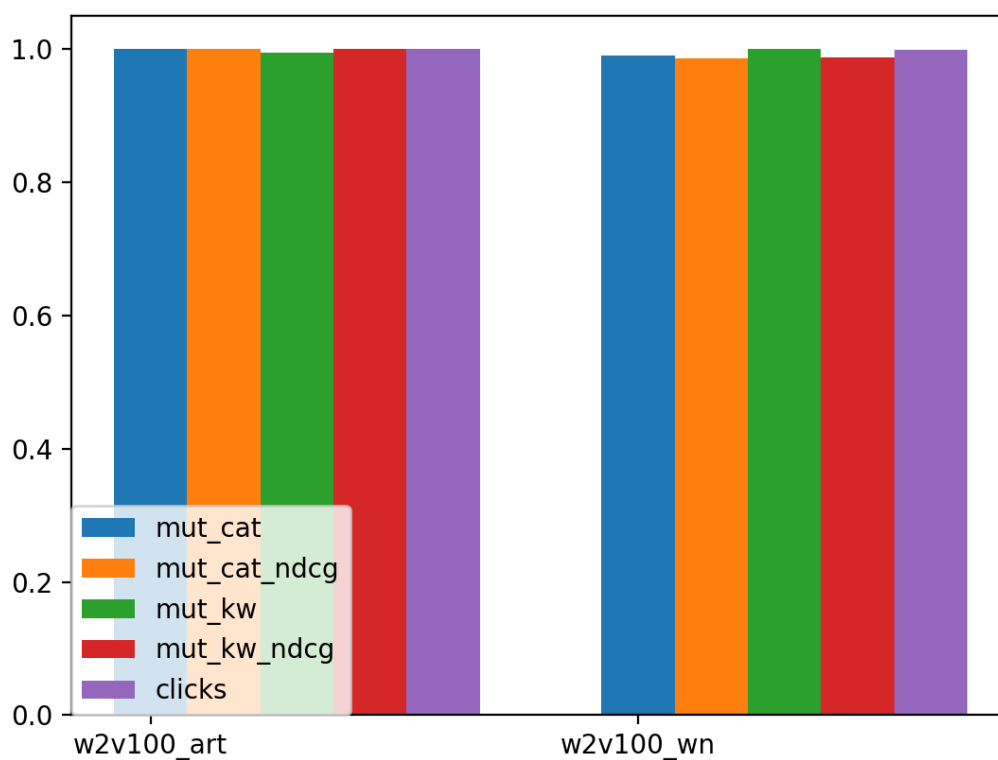
Rysunek 5.2: Porównanie znormalizowanych wyników dla metody LDA.

Podobnie jak w przypadku modelu LSI miary oparte na liczbie wspólnych słów kluczowych (mut_kw) pokazują wzrost jakości modelu przy wzroście liczby tematów. Średnia ze wszystkich miar pokazuje, że najwyższą jakość model osiąga dla liczby tematów 900.

Word2vec w zależności od korpusu

Poniżej zestawiam wyniki dla modeli word2vec o wektorach długości 100 uczonych: na korpusie artykułów z Allegro ($w2v100_art$) oraz na korpusie Słownosieci ($w2v100_wn$). Dokumenty porównywane są tu metodą centroidu.

| | w2v100_art | w2v100_wn |
|--------------|------------|-----------|
| mut_cat | 3.41328 | 3.38265 |
| mut_cat_ndcg | 0.504018 | 0.496797 |
| mut_kw | 1.66191 | 1.67062 |
| mut_kw_ndcg | 0.0896029 | 0.0885063 |
| clicks | 0.946487 | 0.946191 |



Rysunek 5.3: Porównanie znormalizowanych wyników dla metody word2vec w zależności od bazowego korpusu.

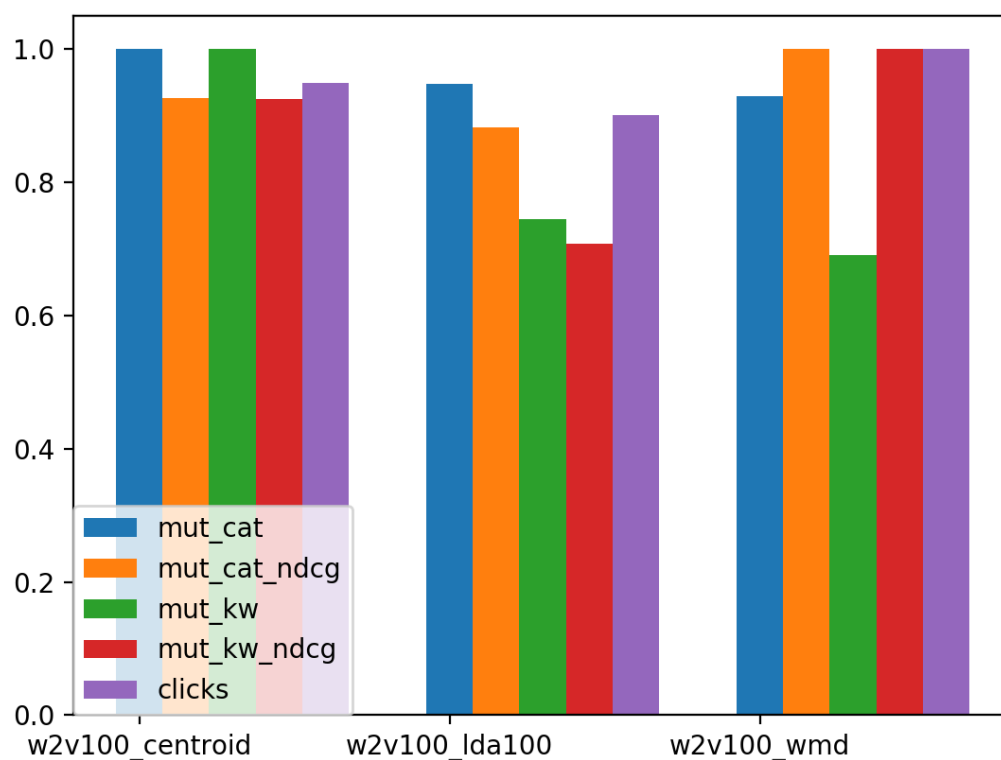
Maksymalna różnica między najlepszym i najgorszym wynikiem nie przekracza 2%. Stąd trudno jest stwierdzić, która metoda daje tu lepsze wyniki. Na korzyść modelu word2vec uczonego na korpusie artykułów przemawia jego mały rozmiar, a co za tym idzie szybkość nauki. Z przyczyn braku wyraźnych zalet korzystania tu z modelu uczonego na korpusie słowosieci korzystam dalej jedynie z modeli opartych o korpus artykułów Allegro.

Word2vec w zależności od metody porównywania dokumentów

W niniejszym punkcie dokonuję porównania jakości metod określania podobieństwa między wektorowymi reprezentacjami dokumentów tekstowych wyznaczonymi metodą word2vec dla wektorów długości 100 oraz 300. Biorę tu pod uwagę metody: centroidu całości dokumentu ($w2v100_{centroid}$), centroidu słów kluczowych wyznaczonych metodą LDA dla 100 tematów ($w2v100_{lda100}$) oraz Word Mover's Distance ($w2v100_{wmd}$).

Wymiar wektorów: 100

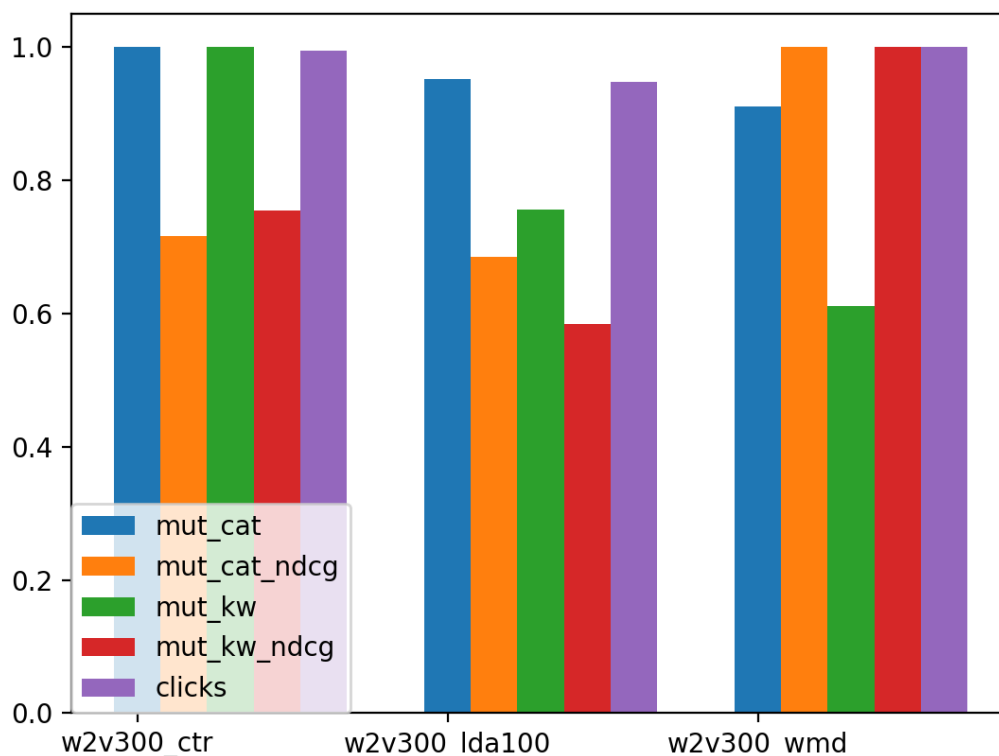
| | w2v100_centroid | w2v100_lda100 | w2v100_wmd |
|--------------|-----------------|---------------|------------|
| mut_cat | 3.41763 | 3.25386 | 3.17404 |
| mut_cat_ndcg | 0.504596 | 0.482482 | 0.54389 |
| mut_kw | 1.66962 | 1.2749 | 1.14807 |
| mut_kw_ndcg | 0.0900018 | 0.0701703 | 0.096873 |
| clicks | 0.938998 | 0.888249 | 0.996717 |



Rysunek 5.4: Porównanie znormalizowanych wyników dla metody word2vec w zależności od sposobu wyznaczania podobieństwa między wektorowymi reprezentacjami dokumentów.

Wymiar wektorów: 300

| | w2v300_ctr | w2v300_lda100 | w2v300_wmd |
|--------------|------------|---------------|------------|
| mut_cat | 3.42132 | 3.25724 | 3.1182 |
| mut_cat_ndcg | 0.505457 | 0.482896 | 0.705453 |
| mut_kw | 1.68395 | 1.27438 | 1.02838 |
| mut_kw_ndcg | 0.0906939 | 0.0701459 | 0.120046 |
| clicks | 0.936386 | 0.891596 | 0.940603 |



Rysunek 5.5: Porównanie znormalizowanych wyników dla metody word2vec w zależności od sposobu wyznaczania podobieństwa między wektorowymi reprezentacjami dokumentów.

Wyniki dla metody opartej o tematy LDA są jednoznacznie gorsze od wyników dla metody centroidu.

Dla obu konfiguracji metody word2vec metoda Word Mover's Distance wykazuje się największą zdolnością do poprawnego szeregowania dokumentów (wysoki wskaźnik metod nDCG w porównaniu z metodami wykorzystującymi średnią).

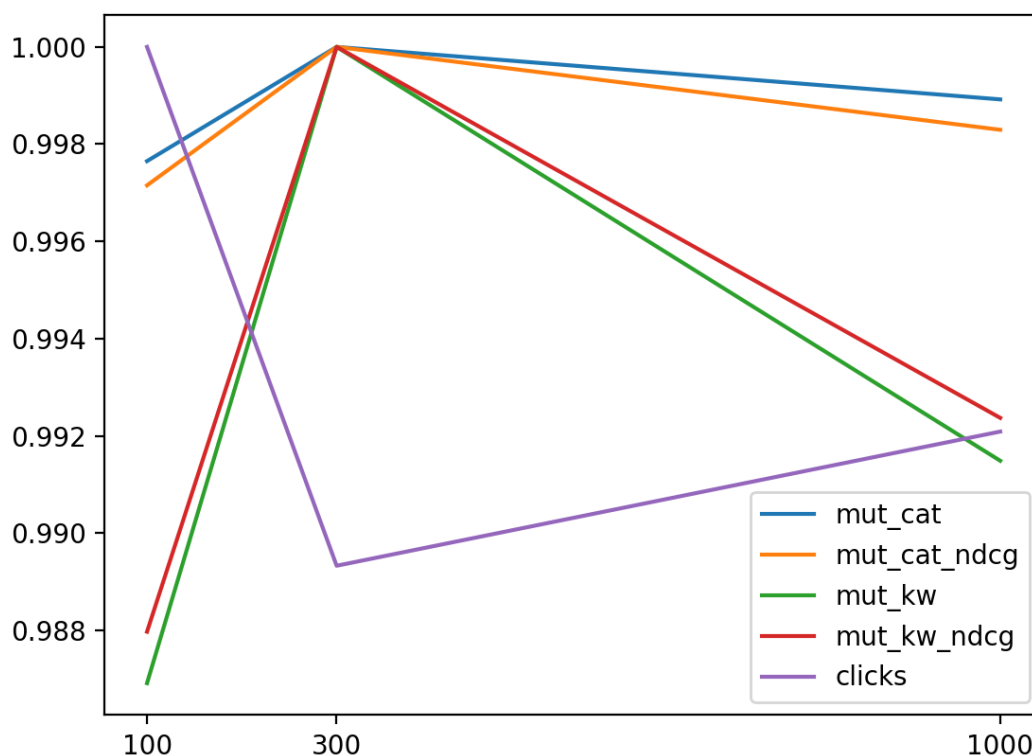
Wadą metody WMD wykluczającą ją z użycia w tym przypadku jest jej powolność. Dla pierwszego przypadku obliczenia trwały 55 minut, co przy czasie < 1 sek dla centroidu jest wartością niedopuszczalną. Proporcjonalnie użycie tej metody dla całego korpusu trwałoby ok. 183 godzin.

Metody word embedding w zależności od wymiarowości wektorów

W niniejszym punkcie porównuję jakość metod word embedding (word2vec, GloVe i fastText) w zależności od wymiarowości wektorów: 100, 300 i 1000. Użyta metoda wyznaczania podobieństwa dokumentów to metoda centroidu.

Word2vec

| | 100 | 300 | 1000 |
|--------------|-----------|-----------|-----------|
| mut_cat | 3.41328 | 3.42132 | 3.41763 |
| mut_cat_ndcg | 0.504018 | 0.505457 | 0.504596 |
| mut_kw | 1.66191 | 1.68395 | 1.66962 |
| mut_kw_ndcg | 0.0896029 | 0.0906939 | 0.0900018 |
| clicks | 0.946487 | 0.936386 | 0.938998 |



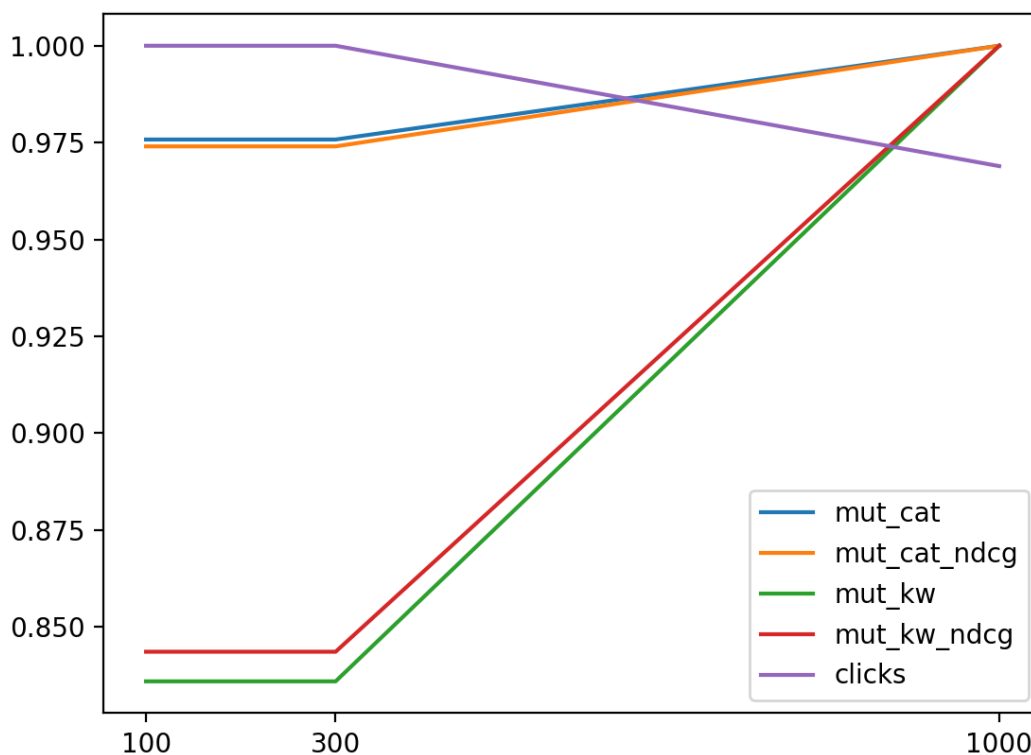
Rysunek 5.6: Porównanie znormalizowanych wyników dla metody word2vec w zależności od bazowego korpusu.

5.3. METODY WORD EMBEDDING W ZALEŻNOŚCI OD WYMIAROWOŚCI WEKTORÓW

Dla różnej wymiarowości wektorów wyjściowych metoda word2vec daje bardzo podobne rezultaty. Różnica między najlepszym i najgorszym wynikiem wynosi jedynie ok 1,5 pkt. proc. Mimo to najlepszą wartością hiperparametru wydaje się być 300.

GloVe

| | 100 | 300 | 1000 |
|--------------|----------|----------|----------|
| mut_cat | 3.4595 | 3.4595 | 3.54532 |
| mut_cat_ndcg | 0.510077 | 0.510077 | 0.523677 |
| mut_kw | 1.79875 | 1.79875 | 2.15176 |
| mut_kw_ndcg | 0.097138 | 0.097138 | 0.115149 |
| clicks | 0.907997 | 0.907997 | 0.879786 |

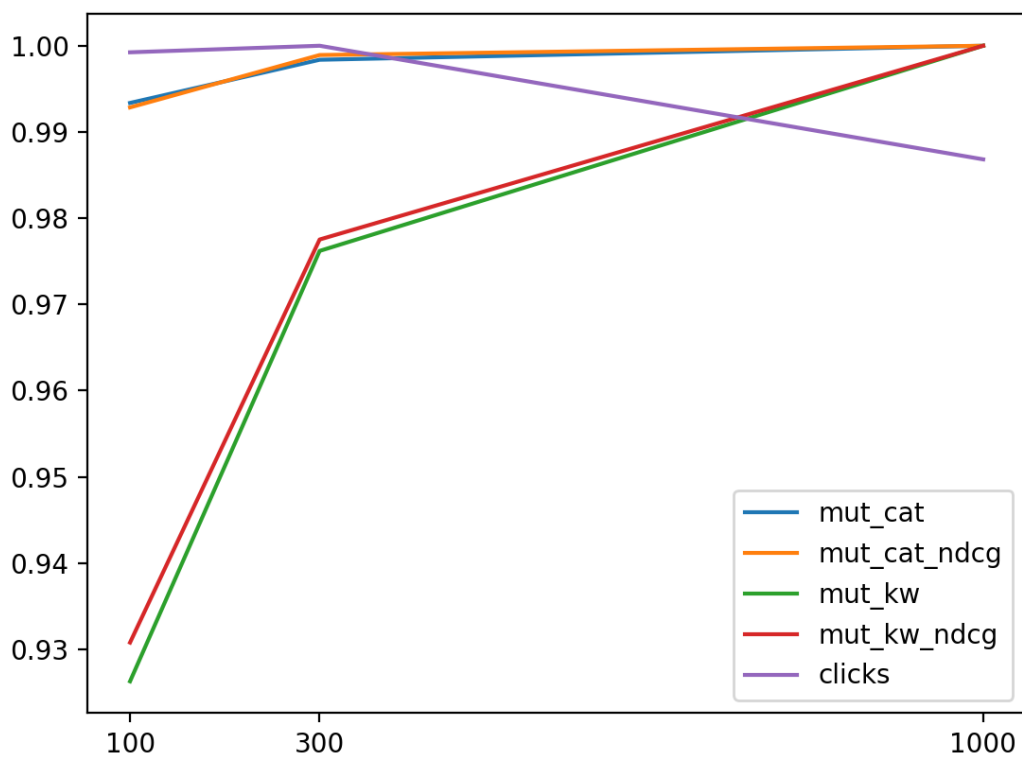


Rysunek 5.7: Porównanie znormalizowanych wyników dla metody GloVe w zależności od bazowego korpusu.

Ewidentnie najlepsze wyniki metoda osiąga dla wektorów wyjściowych długości 1000.

FastText

| | 100 | 300 | 1000 |
|--------------|----------|----------|----------|
| mut_cat | 3.55494 | 3.57287 | 3.57869 |
| mut_cat_ndcg | 0.525164 | 0.528378 | 0.528948 |
| mut_kw | 2.00994 | 2.11828 | 2.16992 |
| mut_kw_ndcg | 0.108461 | 0.113908 | 0.116528 |
| clicks | 0.896766 | 0.89745 | 0.885622 |



Rysunek 5.8: Porównanie znormalizowanych wyników dla metody fastText w zależności od bazowego korpusu.

Również jak w przypadku metody GloVe metoda fastText osiąga najlepsze rezultaty dla wektorów długości 1000.

Ocena jakości wybranych metod przez użytkowników

W niniejszym punkcie dokonuję porównania najlepszych konfiguracji testowanych wcześniej metod. Zestawiam z nimi wyniki dla metody używanej dotychczas w Allegro oraz metodę losową.

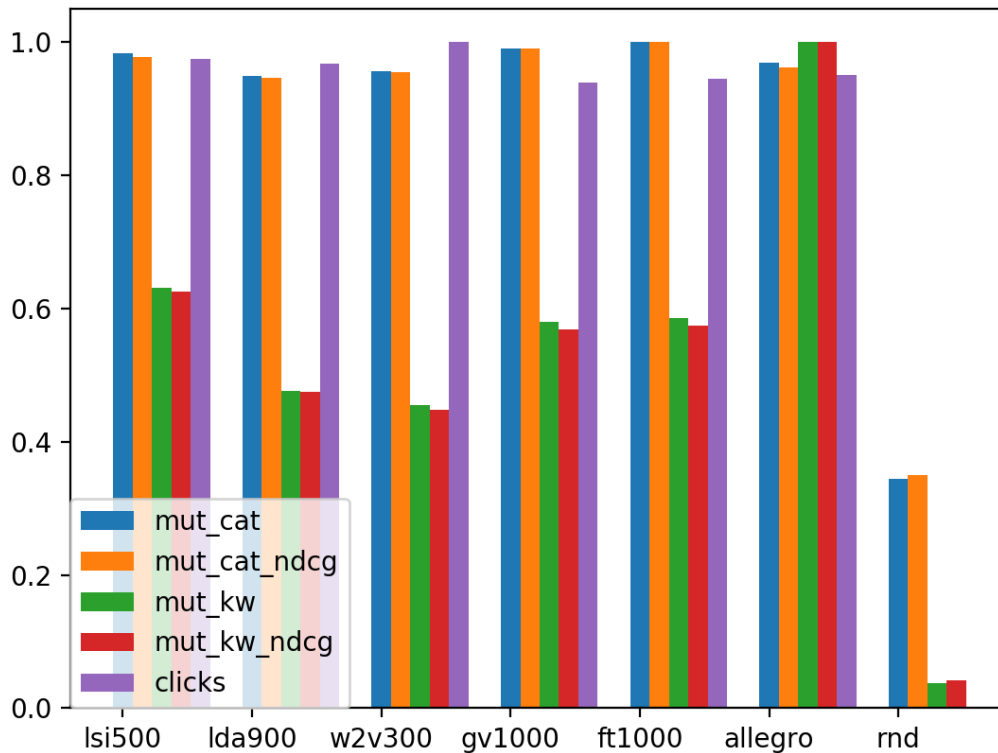
Testowane warianty:

- *lsi500* — Latent Semantic Indexing dla 500 tematów,
- *lda900* — Latent Dirichlet Allocation dla 900 tematów,
- *w2v300* — word2vec dla wektorów dł. 300 i przy użytej metodzie centroidu,
- *w2v300* — GloVe dla wektorów dł. 1000 i przy użytej metodzie centroidu,
- *w2v300* — fastText dla wektorów dł. 1000 i przy użytej metodzie centroidu,
- *allegro* — używane dotychczas w Allegro zapytanie do silnika elasticsearch,
- *rnd* — metoda losowa.

Zestawienie wyników testów automatycznych

Poniżej zestawienie wyników miar automatycznych dla najlepszych wariantów testowanych wcześniej metod.

| | lsi500 | lda900 | w2v300 | gv1000 | ft1000 | allegro | rnd |
|--------------|--------|--------|--------|--------|--------|---------|-------|
| mut_cat | 3.523 | 3.400 | 3.421 | 3.545 | 3.579 | 3.471 | 1.233 |
| mut_cat_ndcg | 0.518 | 0.501 | 0.505 | 0.524 | 0.529 | 0.509 | 0.185 |
| mut_kw | 2.388 | 1.766 | 1.684 | 2.152 | 2.170 | 3.705 | 0.137 |
| mut_kw_ndcg | 0.129 | 0.096 | 0.091 | 0.115 | 0.116 | 0.203 | 0.008 |
| clicks | 0.913 | 0.9064 | 0.936 | 0.880 | 0.886 | 0.890 | — |



Rysunek 5.9: Porównanie znormalizowanych wyników dla wybranych metod.

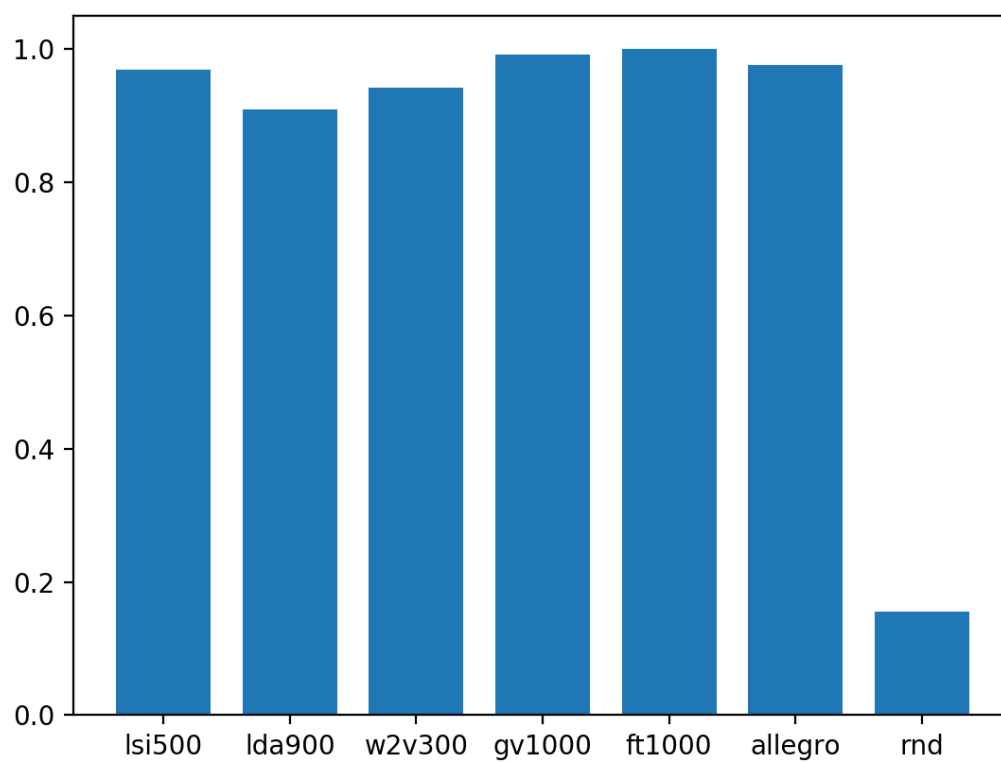
Wszystkie testowane metody dają dużo lepsze rezultaty od metody losowej. Stosunkowo wysokie wskaźniki *mut_cat* dla metody losowej wynikają z dużo większej szansy, że dwa artykuły mają wspólnych przodków w drzewie kategorii niż, że mają wspólne słowa kluczowe.

Najwyższe wyniki wszystkich miar uzyskała metoda *allegro*. Jej wysokie wskaźniki dla metody *mut_kw* wynikają wprost ze struktury zapytania do silnika elasticsearch, które to jest oparte właśnie o słowa kluczowe dołączone do artykułu.

Wyniki oceny eksperckiej

| | lsi500 | lda900 | w2v300 | gv1000 | ft1000 | allegro | rnd |
|-------|--------|--------|--------|--------|--------|---------|-------|
| users | 8.415 | 7.895 | 8.185 | 8.61 | 8.68 | 8.47 | 1.345 |

5.4. OCENA JAKOŚCI WYBRANYCH METOD PRZEZ UŻYTKOWNIKÓW



Rysunek 5.10: Porównanie znormalizowanych wyników dla wybranych metod.

Wnioski i podsumowanie

Istotną zaletą dotychczasowego rozwiązania stosowanego w Allegro jest uniwersalność silnika elasticsearch oraz to, że pozwala edytować indeksowane dane w locie, bez konieczności przebudowy systemu. Metody semantycznej analizy tekstu potrzebują przebudowania modelu przy każdej zmianie korpusu, na którym się opierają.

Wyniki metody *clicks* odstają niekiedy od innych wyników. Może się to wiązać z faktem, iż liczba przejść dokonanych przez użytkowników między parami artykułów niekoniecznie odzwierciedla podobieństwo między nimi. Fakt, iż użytkownik przechodzi z artykułu *A* do rekomendowanego artykułu *B* może również wynikać z faktu, iż artykuł *B* porusza inną tematykę niż artykuł *A*, która to jednak interesuje użytkownika. Wprowadzenie do rekomendacji małego odsetka niepowiązanych przedmiotów jest znaną praktyką przy tworzeniu systemów rekomendacji i ma na celu zainteresowanie użytkownika przedmiotami spoza kręgu jego dotychczasowych zainteresowań.

korelacja między metodami ewaluacji

Dalsze badania.

Niniejsza praca nie wyczerpuje sposobów wyboru artykułów podobnych.

Nie wszystkie pola zawarte w strukturze zostały wykorzystane. Pozostają np. „autor”.

Przed zastosowaniem metod wyznaczania podobieństwa wykonałem przetwarzanie wstępne dokumentów, które można przeprowadzić również na inne sposoby. Jest to temat osobnych badań.

Zdaję sobie sprawę z niedoskonałości zastosowanych miar.

Tematem niniejszej pracy jest przypisanie danemu artykułowi artykułów najbardziej podobnych. Warto tutaj zaznaczyć różnicę pomiędzy tematyką pracy a komercyjnym zagadnieniem najlepszych rekomendacji. Artykuły, które można uznać za dobre rekomendacje, tj. takie, które przynoszą przedsiębiorstwu największy zysk, wcale nie muszą być podobne do danego. Powszechnym zjawiskiem jest wzbogacanie rekomendacji o przedmioty niepodobne do danego, a pozwalające użytkownikowi na poznanie osobnej kategorii przedmiotów, która może go zainteresować a tym samym przyciągnąć do serwisu.

Bibliografia

- [1] <http://gadzetomania.pl/11824,zakupy-w-sieci-porownanie-najwiekszych-polskich-serwisow-aukcyjnych-2> (09.08.17)
- [2] <https://magazyn.allegro.pl/3333-serwis-allegro-to-nasz-sposob-na-wasze-szybkie-i-wygodne-zakupy-przez-internet> (07.05.2017)
- [3] <https://allegro.pl/arttykul/jaka-farba-dla-alergika-55917/> (26.06.2017)
- [4] Słownik Języka Polskiego PWN <http://sjp.pwn.pl/sjp/arttykul;2441396.html> (07.05.2017)
- [5] <http://blog.aylien.com/overview-word-embeddings-history-word2vec-cbow-glove/> (18.08.2017)
- [6] F. Ricci, L. Rokach, B. Shapira, *Introduction to Recommender Systems Handbook*, Springer, 2011
- [7] <https://www.elastic.co/> (18.08.2017)
- [8] <https://lucene.apache.org/> (18.08.2017)
- [9] <https://www.elastic.co/use-cases> (10.08.17)
- [10] Z. S. Harris, *Distributional Structure*, WORD, tom 10, num. 2-3, 1954
- [11] G. Salton and M. McGill, *Introduction to modern information retrieval*, McGraw-Hill, 1983
- [12] J.R. Firth, *A synopsis of linguistic theory 1930-1955*, Oxford: Philological Society, 1957
- [13] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, *Indexing by latent semantic analysis*, Journal of the American Society for Information Science, tom 41, num. 6, 1990
- [14] David M. Blei, Andrew Y. Ng, Michael I. Jordan, *Latent Dirichlet Allocation*, Journal of Machine Learning Research, tom 3 num. 4-5, 2003

- [15] G. H. Golub, W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis. 2 (2), 1965
- [16] T. Mikolov, K. Chen, G. Corrado, J. Dean, *Efficient Estimation of Word Representations in Vector Space*, International Conference on Machine Learning (ICML), 2013
- [17] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, *A Neural Probabilistic Language Model*, Journal of Machine Learning Research 3 1137–1155, 2003
- [18] R. Collobert, J. Weston, *A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning*, NEC Labs America, 2008
- [19] <https://code.google.com/archive/p/word2vec/> (26.05.2017)
- [20] <http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/> (26.05.2017)
- [21] A. Joulin, E. Grave, P. Bojanowski T. Mikolov, *Bag of Tricks for Efficient Text Classification*, Facebook AI Research, 2016
- [22] Jeffrey Pennington, Richard Socher, Christopher D. Manning, *GloVe: Global Vectors for Word Representation*, Computer Science Department, Stanford University, Stanford, CA 94305, 2014
- [23] <https://cran.r-project.org/web/packages/text2vec/vignettes/glove.html> (30.08.2017)
- [24] M. J. Kusner, Y. Sun, N. I. Kolkin, K. Q. Weinberger, *From Word Embeddings To Document Distances*, International Conference on Machine Learning (ICML), 2015
- [25] Y. Rubner, C. Tomasi, L. J. Guibas, *The Earth Mover's Distance as a Metric for Image Retrieval*, str. 1, Computer Science Department, Stanford University, 2000
- [26] O. Pele, M. Werman, *Fast and robust earth mover's distances*, ICCV, 2009
- [27] K. Jarvelin, J. Kekalainen, *Cumulated gain-based evaluation of IR techniques*, University of Tampere, 2002
- [28] <https://pl.wikipedia.org/wiki/Wikipedia:Stopwords> (15.04.2017)
- [29] <http://morfologik.blogspot.com/> (07.05.2017)

- [30] P. Kędzia, G. Czachor, M. Piasecki, J. Kocoń *Vector representations of polish words (Word2Vec method)*, Wrocław University of Technology, 2016, <https://clarin-pl.eu/dspace/handle/11321/327> (26.06.2017)
- [31] <http://plwordnet.pwr.wroc.pl/wordnet/> (28.06.2017)

Wykaz symboli i skrótów

| | |
|------|-------------------|
| nzw. | nadzwyczajny |
| * | operator gwiazdka |
| ~ | tylda |

Spis rysunków

| | | |
|------|---|----|
| 0.1 | Widok strony internetowej zawierającej jeden z artykułów serwisu Allegro. [3] | 12 |
| 1.1 | Neuronowy model języka. Źródło: [17]. | 22 |
| 1.2 | Schemat sieci wykorzystującej podejście skip-gram i CBOW. Źródło: [16]. | 24 |
| 2.1 | Liczba kategorii na poszczególnych poziomach drzewa. | 31 |
| 2.2 | Liczba artykułów, dla których kategoria na danym poziomie drzewa jest tą najbliższą szczegółową. | 32 |
| 2.3 | Histogram liczby wystąpień słów w korpusie w skali logarytmicznej. | 34 |
| 2.4 | Histogram długości artykułów. | 35 |
| 3.1 | Drzewo kategorii dla przykładu 1. | 38 |
| 3.2 | Drzewo kategorii dla przykładu 2. | 38 |
| 5.1 | Porównanie znormalizowanych wyników dla metody LSI. | 46 |
| 5.2 | Porównanie znormalizowanych wyników dla metody LDA. | 47 |
| 5.3 | Porównanie znormalizowanych wyników dla metody word2vec w zależności od bazowego korpusu. | 48 |
| 5.4 | Porównanie znormalizowanych wyników dla metody word2vec w zależności od sposobu wyznaczania podobieństwa między wektorowymi reprezentacjami dokumentów. | 50 |
| 5.5 | Porównanie znormalizowanych wyników dla metody word2vec w zależności od sposobu wyznaczania podobieństwa między wektorowymi reprezentacjami dokumentów. | 51 |
| 5.6 | Porównanie znormalizowanych wyników dla metody word2vec w zależności od bazowego korpusu. | 52 |
| 5.7 | Porównanie znormalizowanych wyników dla metody GloVe w zależności od bazowego korpusu. | 53 |
| 5.8 | Porównanie znormalizowanych wyników dla metody fastText w zależności od bazowego korpusu. | 54 |
| 5.9 | Porównanie znormalizowanych wyników dla wybranych metod. | 56 |
| 5.10 | Porównanie znormalizowanych wyników dla wybranych metod. | 57 |

Spis tabel

Spis załączników

1. Załącznik 1
2. Załącznik 2

Technologie i narzędzia

Analizę danych, ich wstępne przetworzenie a następnie przeprowadzenie docelowych eksperymentów wykonałem korzystając głównie z języka Python i szeregu skryptów napisanych w nim własnoręcznie, wykorzystujących istniejące specjalistyczne biblioteki posiadające interfejs w tymże języku.

Wykorzystane narzędzia:

- Elasticsearch — silnik wyszukiwania tekstowego. Używam go do przechowywania bazy artykułów oraz ich przetworzonych wersji.
- MongoDB — nierelacyjna baza danych, której używam do przechowywania wyników generowanych przez testowane algorytmy.
- GloVe — implementacja metody GloVe. Generuje wektory słów, które następnie mogą być wykorzystane np. przez bibliotekę gensim.
- fastText — implementacja metody fastText wykonana przez zespół Facebook Research. Generuje wektory słów, które następnie mogą być wykorzystane np. przez bibliotekę gensim.

Poniższa lista zawiera wykorzystane biblioteki języka Python:

- Gensim — rozbudowana biblioteka służąca do przetwarzania języka naturalnego; zawiera implementację metod word2Vec, LSI, LDA, TF-IDF i inne.
- Morfologik — tokenizer języka polskiego.
- Numpy — pozwala wydajnie wykonywać obliczenia numeryczne.
- Pyemd — implementacja algorytmu Earth Mover's Distance.
- Elasticsearch — ułatwia wykonywanie zapytań do silnika Elasticsearch wprost z kodu Pythona.
- Matplotlib — biblioteka służąca do wykonywania wykresów.
- Pymongo — umożliwia wykonywanie zapytań do bazy MongoDB wprost z kodu Pythona.