Narzędzie do analizy statystycznej słów i bigramów.

Michał Bobowski, Marcin Cieślikowski

2014-01-16

1 Wstęp

Niniejszy dokument stanowi podsumowanie projektu z przedmiotu WEDT. Zawiera opis interfejsu użytkownika i logiki programu oraz przedstawienie struktury kodu źródłowego.

2 Przypadki użycia

Uznaliśmy, że określenie przypadków użycia jest najprostszym sposobem zapisania wymagań wstępnych. Scenariusze można też traktować jako zwięzłą instrukcję obsługi.

2.1 Przeprowadzenie obliczeń

Podstawowym zadaniem programu jest przeprowadzenie obliczeń i wyświetlenie ich na ekranie. Scenariusz tworzą następujące zdarzenia:

1. Użytkownik wybiera parametry.
2. System przeprowadza obliczenia.
3. System zapisuje wyniki do pliku.
4. System wyświetla wyniki na ekranie.

2.2 Wczytanie wyników

Wysoce prawdopodobnym jest powrót użytkownika do wyników już przeprowadzonych obliczeń. Mając to na uwadze, oraz biorąc pod uwagę czasochłonność symulacji, wprowadziliśmy możliwość wczytania danych z pliku.

1. Użytkownik wybiera ścieżkę do pliku z zapisanymi wynikami.

2. System wyświetla wyniki na ekranie.

2.3 Filtracja

Operacja filtracji staje się dostępna dopiero po wykonaniu któregoś z wcześniejszych przypadków użycia. Aby nie komplikować wyglądu tabel wynikowych wypełnianie filtrów zostało przeniesione do oddzielnego dialogu. Uruchomienie okna filtracji jest możliwe przy pomocy menu kontekstowego.

1. Użytkownik wybiera jedną z tabel wynikowych.
2. Użytkownik otwiera dialog filtracji przy użyciu menu kontekstowego.
3. Użytkownik wypełnia filtry.
4. System wyświetla przefiltrowane wyniki na ekranie.

3 Specyfikacja szczegółowa

W tej części doprecyzowane zostały wymagania dotyczące danych wejściowych i wyjściowych.

3.1 Parametry wejściowe

Przed przeprowadzeniem obliczeń użytkownik może zdefiniować następujące parametry:

- Ścieżka do pliku wejściowego lub katalogu zawierającego wiele plików wejściowych.
- Typ bigramu: obliczany dla kolejnych słów lub wszystkich słów w tekście.
- Części mowy dla słów z bigramu.
- Nazwa pliku wyjściowego.

3.2 Prezentacja danych

Statystyki słów/bigramów są liczone i prezentowane dwa razy - dla słów z odmianą oraz dla formy podstawowej. Statystyki dla słów:

- Liczba wystąpień w zbiorze.

- Liczba zdań w których wystąpiło słowo.
- Liczba dokumentów w których wystąpiło słowo.
- Procent dokumentów w których wystąpiło słowo.
- Miara tf-idf.

Statystyki dla bigramów:

- Liczba wystąpień w zbiorze.
- Liczba zdań w któртch wystąpił bigram.
- Liczba dokumentów w których wystąpił bigram.
- Procent dokumentów w których wystąpił bigram.
- Miara tf-idf.
- Prawdopodobieństwo słowa 1.
- Prawdopodobieństwo słowa 2.
- Prawdopodobieństwo bigramu złożonego ze słów 1 i 2.

3.3 Format danych wyjściowych

Dane wyjściowe są zapisywane do pliku bazy danych sqlite. Baza danych zawiera 8 nastepujących tabel:

1. Tabela words zawiera statystyki słow. Tabela ta zawiera następujące pola:
   (a) word - Nazwa słowa.
   (b) count - Liczba wystąpień słowa w korpusie.
   (c) numberOfSentences - Liczba zdań zawierających słowo.
   (d) numberOfDocuments - Liczba dokumentów zawierających słowo.
   (e) percentOfDocuments - Procent dokumentów zawierających słowo.
2. Tabela Lemmas zawiera statystyki dla słów w formie podstawowej. Pola tej tabeli są identyczne, jak w tabeli words.
3. Tabela WordTFIDF - Zawiera wskaźnik TF-IDF. W tej tabeli znajdują się następujące pola:

(a) word - Słowo, dla którego obliczany jest wskaźnik TF-IDF.

(b) document - Nazwa dokumentu, dla którego oblicznay jest wskaźnik TF-IDF.

(c) TFIDF - Wartość wskaźnika TFIDF

4. Tabela bigrams zawiera statystyki bigramów. Tabela ta zawiera następujące pola:

   (a) word1 - Pierwsze słowo w bigramie.

   (b) word2 - Drugie słowo w bigramie.

   (c) count - Liczba wystąpień bigramu w korpusie.

   (d) numberOfSentences - Liczba zdań zawierające bigram.

   (e) numberOfDocuments - Liczba dokumentów zawierających bigram.

   (f) percentOfDocuments - Procent dokumentów zawierających bigram.

   (g) percentOfSentence - Procent zdań zawierających bigram.

   (h) percentOfSentencew1 - Procent zdań zawierających pierwsze zdanie.

   (i) percentOfSentencew2 - Procent zdań zawierających drugie zdanie.

5. Tabela bigramsLemma zawiera statystyki bgramów w formie podstawowej. Pola tabeli są takie same, jak w tabeli bigrams.

6. Tabela bigramsTFIDF zawiera wskaźnik TF-IDF dla bigramów. Tabela zawiera następujące pola:

   (a) word1 - Pierwsze słowo w bigramie.

   (b) word2 - Drugie słowo w bigramie.

   (c) TFIDF - Wartość wskaźnika TFIDF

7. Tabela bigramsLemmaTFIDF zawiera wskaźnik TF-IDF dla bigramów w formie podstawowej. Pola tabeli pokrywaja się z polami tabel bigramsTFIDF

4 Wybór narzędzi i technologii

Program został zrealizowany w języku Java. Kod tworzyliśmy przy użyciu środowiska Eclipse oraz częściowo Netbeans (edytor interfejsu użytkownika). Do kontroli kodu wykorzystaliśmy system Git.

Do oznaczenia części mowy wykorzystaliśmy bibliotekę Gate. Korzysta ona wewnętrznie z biblioteki TIKA, dzięki czemu uzyskaliśmy wsparcie dla wielu formatów tekstowych m. in. txt, html, odt i doc. Najważniejszym elementem zapożyczonym ze środowiska Gate jest automatyczny POS-tagger dla języka angielskiego.

Dane wyjściowe są przechowywane na dysku w formacie bazy danych SQLite. Jest to efektywne i uniwersalne rozwiązanie.

5 Uruchomienie programu

Do uruchomienia programu niezbędne jest ściągnięcie biblioteki GATE oraz ustawienie zmiennej gate.home. Zmienna jest argumentem maszyny wirtualnej - w środowisku Eclipse należy wejść w menu Run -> RunConfigurations -> Arguments i tam wpisać np. -Dgate.home="/home/preston/GATE_Developer_7.1".

6 Opis kodu źródłowego

Kod programu wraz z historią zmian jest dostępny w repozytorium pod adresem https://github.com/mbobowsk/Bigrams .

Kod źródłowy staraliśmy się rozplanować zgodnie z ideą wzorca MVC. Wyodrębniliśmy cztery pakiety, grupując w nich podobne funkcjonalności.

6.1 Pakiet view

Pakiet view zawiera wszystkie klasy związane bezpośrednio z widokiem. Duża część kodu znajdującego się w tym pakiecie została automatycznie wygenerowana przez edytor formularzy NetBeans. Główne okno programu reprezentuje klasa *AppWindow*. Pozostałe klasy obsługują dialogi oraz logikę w nich zawartą (głównie filtrowanie danych).

6.2 Pakiet model

Pakiet model jest pokłosiem architektury biblioteki Swing. Każdy element wyświetlający grupę danych (listy, tabele) potrzebuje powiązanego modelu. Dodatkowo pakiet zawiera klasę *ModelLogic*, która odpowiada za incjalizację modelu części mowy.

6.3 Pakiet sql

Wszelkie operacje związane z bazą danych są wykonywane w pakiecie sql. Dodatkowo wydzielone zostały oddzielne klasy dla zapisu i odczytu (*SQLWrite* i *SQLRead*).

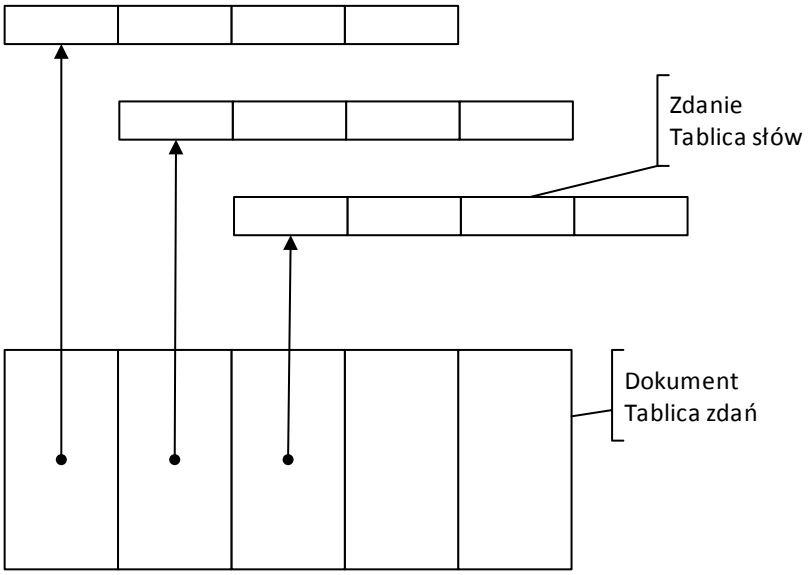
6.4 Pakiet logic

W pakiecie logic znajduje się pozostała częśc logiki oraz typy danych. Klasa *Controller* stanowi warstwę logiki, do której trafiają dane z widoku. *Controller* posiada jedną metodę publiczną, która przyjmuje na wejściu opcje programu (obiekt klasy *Options*), a następnie przeprowadza wszystkie obliczenia. Enkapsuluje ona wywołania biblioteki Gate, poprzez którą przeprowadzany jest podział na tokeny, podział na zdania oraz oznaczanie części mowy.

7 Algorytm obliczania statystyk

Program dla podanych dokumentów uruchamia funkcje bibliotęki GATE, które dzielą tekst na słowa, zdania. Dodatkowo dla słów są określanę części mowy i formy podstawowe słów. Dane zwróconę przez GATE są przetwarzane do formy przedstwionej na rysunku 1. Program przetwarza po kolei termy (słowa i bigramy) w zdaniach. Podczas przetwarzania termów program zlicza statystyki termów.

danych.pdf



Rysunek 1: Struktura danych