Lekturo-skurczator

Dominik Doberski 133207, Michał Kukieła 132265

Cel projektu

Stworzenie klasyfikatora zdolnego podzielić tekst na części będące opisem przyrody i pozostałe.

Umożliwi to na przykład skrócenie szkolnych lektur o zbędne (z technicznego punktu widzenia, pomijając aspekty literackie), nieraz długie i rozbudowane opisy przyrody, przy zachowaniu fabuły i istotnych treści tychże lektur.

Języki i narzędzia

Język: Python

Biblioteka do uczenia maszynowego: Scikit-learn

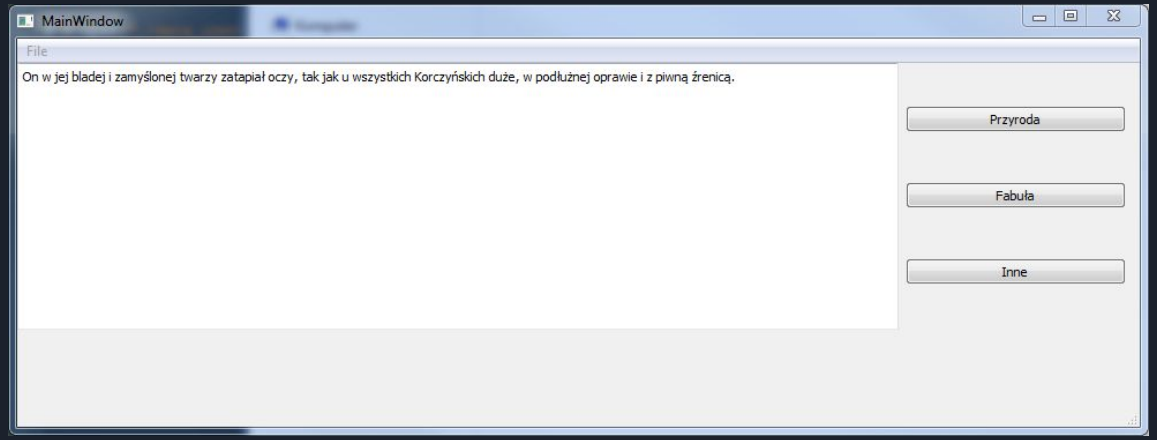
Tworzenie zbioru uczącego

Zbiór uczący utworzyliśmy tagując zdania kategoriami: Opis przyrody (N), Pozostałe (O) (głównie opisy nie będące opisami przyrody i mało istotne treści), Fabuła (P) (wszystko co nie należy do pozostałych kategorii i\lub ma znaczenie fabularne).

Do utworzenia zbioru treningowego skorzystaliśmy z książki “Nad Niemnem” Elizy Orzeszkowej z dodatkiem zdań będących opisem przyrody z “Pana Tadeusza” Adama Mickiewicza. Łącznie otagowaliśmy ponad 1500 zdań, z czego 144 jako opis przyrody - nie jest to dużo, klasyfikator z pewnością działałby lepiej, gdyby przykładów będących opisem przyrody były setki, lub nawet tysiące i pochodziłyby z większej liczby źródeł, lecz nie chcieliśmy się skupiać na wielogodzinnym tagowaniu, a bardziej na utworzeniu działającej aplikacji.

Tworzenie zbioru uczącego

Program wczytujący kolejne zdania i pozwalający je otagować.



Tworzenie zbioru uczącego

Otagowane zdania zostały odpowiednio przetworzone:

- usunięcie znaków interpunkcyjnych, innych znaków specjalnych i błędów (np podwójna spacja)
- usunięcie stop words
- lematyzacja (https://pypi.org/project/pystempel/)

Przetwarzania tego dokonaliśmy osobno, bez pomocy scikit-learn, ponieważ biblioteka ta nie posiada narzędzi do przetwarzania języka polskiego.

Tworzenie klasyfikatora

We wstępnych testach okazało się, że naiwny klasyfikator bayesowski, którego chcieliśmy pierwotnie użyć, wypada gorzej niż klasyfikator stosujący metodę SVM. Dla naiwnego klasyfikatora bayesowskiego co prawda miara precision dla klasy N (opis przyrody) wynosiła 1.0 podczas testów z cross validation, za to recall wynosiło zawsze < 0.2, więc dużo opisów przyrody nie było znajdowanych.

Precision to wskaźnik jak dużo wykrytych zdań miało wykrytą prawidłową klasę, recall to wskaźnik informujący jak dużo zdań zostało poprawnie zaklasyfikowanych spośród wszystkich w danej klasie

Optymalnych parametrów dla klasyfikatora SVM szukaliśmy za pomocą techniki grid search, która jest zaimplementowana w scikit-learn. Ostatecznie klasyfikator przy walidacji krzyżowej (crossvalidation) osiągał zarówno precision jak i recall na poziomie 1.0 dla klasy N (dla pozostałych klas od 0.97 do 1.0).

Tworzenie klasyfikatora

Parametry klasyfikatora:

- użycie n-gramów o wartościach n = {1 ,2} (unigramy i bigramy)
- model binarny wektorów opisujących zdania (0 - token nie występuje w zdaniu, 1 - token występuje w zdaniu co najmniej raz)
- pomijanie tokenów (nie trafiają do słownika\bag of words) występujących tylko 1 raz w całym zbiorze uczącym
- użycie wag tf-idf na wektorach
- funkcja straty: hinge
- waga klas proporcjonalna do częstotliwości ich występowania
- współczynnik alfa = 0.0001

Tworzenie klasyfikatora

Wytrenowany klasyfikator został zapisany do pliku i jest używany w naszej aplikacji obsługiwanej poprzez proste GUI.

GUI

GUI umożliwia wczytanie pliku z tekstem.

Pasek postępu pokazuje postęp klasyfikacji zdań.

Dodatkowo gui informuje o liczbie znalezionych zdań opisów przyrody po zakończeniu przetwarzania.

Wyjściem działania programu jest plik .docx z oryginalnym tekstem oraz zdaniami będącymi opisami przyrody oznaczonymi innym kolorem.