Proiect: Clasificarea cuvintelor în funcție de complexitatea lor lexicală

Scopul acestui laborator este familiarizarea cu tema proiectului și implementarea unor prime soluții.

Descrierea proiectului

O aplicație importantă în Procesarea Limbajului Natural o reprezintă simplificarea textului. Ea se bazează pe abilitatea de a identifica cuvinte care sunt considerate greu de înțeles pentru o anumită parte a populației (spre exemplu cei care învață o limbă nouă, vorbitorii nativi cu niveluri scăzute de alfabetizare și persoanele cu dizabilități de citire). Aceste cuvinte au o complexitate lexicală crescută și ar trebui identificate de sistemele care realizează simplificarea textului.

Sistemele cele mai performante abordează problema de identificare a cuvintelor în funcție de complexitatea lor lexicală asociind un scor fiecărui cuvânt, practic rezolvând o problemă de regresie. În proiectul propus vom aborda o problemă mai simplă și anume clasificarea binară a unui cuvânt dintr-o propoziție. În acest scenariu, în funcție de complexitatea lui lexicală un cuvânt poate fi complex (eticheta 1) sau simplu (non-complex, eticheta 0).

Descrierea mult mai detaliată a proiectului o aveți pe pagina proiectului de pe platforma Kaggle:https://www.kaggle.com/t/3dfa2c8369e548d1afa7a7769590284c (https://www.kaggle.com/c/ub-fmi-cti-2021-2022). Pentru a putea participa la concurs trebuie să vă creați un cont pe platforma Kaggle și să dați "Join competition".

Implementarea unei prime soluții

Pentru a putea genera o soluție pentru proiect este necesar să vă descărcați datele puse la dispoziție (vedeți secțiunea Data de pe pagina proiectului de pe Kaggle).

Scriptul *solutie_random.py* constituie un punct de pornire în proiectul vostru. În acest script citim datele de antrenare și testare, afișăm dimensiunile variabilelor citite iar apoi generăm o primă soluție pentru proiect. Soluția propusă nu analizează datele din mulțimea de antrenare și testare ci doar generează predicțiile pentru datele din mulțimea de testare în mod aleator, generând la întâmplare etichetele 0 și 1.

Implementarea unei a doua soluții

A doua soluție propusă folosește modelul "Cei mai apropiați K vecini" (cu care ați lucrat în Laboratorul 3) folosind anumite caracteristice potrivite pentru datele cu care lucrați. Pentru fiecare cuvânt țintă (token) pe care vrem să îl clasificăm vom calcula caracteristici specifice. Caracteristicile includ număr de silabe, lungime, număr de vocale, frecvența cuvântului

într-un anumit corpus, numărul de sinonime (synsets) din baza de date Wordnet, similaritatea între două cuvinte, etc.

1. ***Bibliotecile necesare***
    - numpy (https://pypi.org/project/numpy/)
    - pandas (https://pypi.org/project/pandas/)
    - sklearn (https://pypi.org/project/scikit-learn/)
    - pyphen (https://pypi.org/project/pyphen/)
    - nltk (https://pypi.org/project/nltk/)

2. ***Extragerea caracteristicilor***

    Funcția *featurize_df* apelează funcția *featurize* care, pentru fiecare înregistrare din dataframe, calculează pentru fiecare cuvânt țintă o listă de caracteristici. Vom împărți caracteristicile în trei categorii; caracteristici legate de corpus (contextul în care se găsesc cuvintele), caracteristici legate de structura cuvintelor și caracteristicile obținute folosind biblioteca Wordnet.

    Pentru a obține caracteristicile legate de corpus trebuie să scrieți funcția *corpus_feature* care returnează 0, 1 sau 2 în funcție de categoria de text în care se găsește cuvântul țintă (0 - bible, 1 - biomed, 2 - europarl).

    Pentru a obține caracteristicile legate de structura cuvintelor este nevoie să scrieți toate funcțiile care sunt apelate în funcția *get_word_structure_features.*

    Pentru a obține caracteristicile folosind Wordnet este necesar să completați funcția *get_wordnet_features.* Momentan funcția apelează doar o singură funcție ce returnează numărul de synset-uri. Puteți completa ulterior funcția astfel încât să obțineți mai multe caracteristici. Vă puteți inspira pentru acestea din documentația oficială Wordnet (https://www.nltk.org/howto/wordnet.html).

3. ***Realizarea de predicții***

    Cu ajutorul caracteristicilor extrase vom folosi un clasificator KNN pentru a realiza predicții pe datele de testare. În implementare vom folosi un număr variabil k de vecini. Puteți folosi aceste predicții în submisiile voastre pentru competiția de pe Kaggle. În secțiunea Overview/Evaluation de pe Kaggle este descrisă în detaliu metrica *balanced accuracy* folosită pentru a măsura performanța voastră.