Text mining
Pracownia 2
Zajęcia 4 i 5

Uwaga: Jedno zadanie z tej listy można przełożyć na pracownię 3.

Zadanie 1. (8p) Napisz program MCR (Mistrz Ciętej Riposty), który umożliwia przeprowadzenia dialogu człowieka z komputerem, w którym wypowiedzi komputera są

a) wzięte z wikicytatów,

b) pasujące w pewnym stopniu do wypowiedzi przedmówcy.

Należy zaproponować podział dłuższych cytatów na „minicytaty", które będą potencjalnymi odpowiedziami. Program dodatkowo powinien:

1. tworzyć sobie wewnętrzny ranking pasujących wypowiedzi do danego pytania,
2. losować odpowiedź w ten sposób, by odpowiedzi na szczycie rankingu miały dużo większą szansę być wylosowanymi (losowanie powinno być możliwe do wyłączenia, w tej wersji odpowiedź jest po prostu liderem rankingu)
3. korzystać z prostego stemmera lub lematyzacji,
4. (generalnie) nie powtarzać wypowiedzi w ramach jednego dialogu,
5. pilnować, by wypowiedź nie była nigdy kopią (lub prawie kopią) wypowiedzi poprzedzającej
6. reagować jakimś ogólnym cytatem w przypadku nieznalezienia niczego pasującego
7. unikać takich słów jak *on*, *go*, *ich*, których znaczenie zależy od kontekstu.

Przykładowe dialogi hipotetycznego MCR-a znajdują się na SKOS-ie. Do tworzenia Mistrza Ciętej Riposty jeszcze wrócimy na kolejnych listach.

Zadanie 2. (8p) W zadaniu tym powinieneś napisać wyszukiwarkę dla Wikipedii. Wymagamy tu oddzielenia procesu indeksowania treści od wyszukiwania, czyli podczas indeksowania powinny powstać listy postingowe, które następnie należy zapisać do bazy danych (lub do pliku). Proces wyszukiwania z kolei powinien startować szybko i wczytywać listy postingowe do pamięci w sposób leniwy (czyli tylko wtedy, gdy któraś jest potrzebna).

Wyszukiwarka powinna prezentować wyniki w kolejności uwzględniającej następujące rzeczy:

a) Trafienia w tytule są cenniejsze od trafienia poza tytułem

b) Trafienia dokładne (*kotami-kotami*) są cenniejsze od trafienia niedokładnego (czyli zgodności lematu, np. (*kotami-kotu*).

c) Lekko preferowane są dokumenty o mniejszych identyfikatorach (czyli występujące wcześniej w pliku z Wikipedyjką)

Do prezentacji wyników wykorzystaj kolory, w celu zwiększenia czytelności. Wyświetlaj zarówno tytuł artykułu, jak i fragment jego treści, zawierający termy z zapytania. Wystarczy, że Twój program zadziała dla zmniejszonej Wikipedii (patrz SKOS).

Zadanie 3. (8p) W tym zadaniu zajmiemy się odpowiadaniem na pytania z teleturnieju 'Jeden z 10'. Twoje główne zadanie składa się z dwóch punktów:

a) Zaadaptuj rozwiązanie bazowe z konkursu PolEval 2021 (program `baseline.py`) w taki sposób, by korzystał z Twojej wyszukiwarki z poprzedniego zadania. Sprawdź, jak zmienia się skuteczność programu w zależności od parametrów wyszukiwarki (na przykład wagi trafień dokładnych, czy premii za kolejność). Uwaga: zwróć uwagę, że Twój system odpowiadania powinien inaczej traktować trafienia w tytule, niż wyszukiwarka z zadania 1. Sporządź raport z wynikami programu dla różnych parametrów (uwaga: nie wymagamy tu bardzo systematycznych badań, wystarczy sprawdzić parę ustawień).

b) Dodając proste heurystyki i różne traktowanie pytań różnych typów postaraj się powiększyć wynik z poprzedniego zadania o kilka punktów procentowych[1] (na kolejnych listach będziemy dalej pracować nad odpowiadaniem na pytania, kolejne zadania będą miały punktację zależną od liczby trafień).

[1] Wskazówka: posortuj pytania za pomocą systemowego polecenia sort i popatrz w jakie „klastry" się układają. Niektóre z nich są bardzo łatwe do obsłużenia w sposób dający konkretny przyrost jakości