Zastosowanie technologii internetowych - Projekt

Task 2: Broader Named Entity Identification and Linking

Prowadzący:
dr Jarosław Bąk

Autorzy:
Adam Dębczak
Olaf Bergmann
Piotr Mielcarzewicz
Filip Krzemień

23 stycznia 2019

1. Charakterystyka projektu

1.1 Założenia

Celem projektu było stworzenie rozwiązania dla problemu konkursowego - OKE2018 Challenge: Broader Named Entity Identification and Linking. Zadanie to jest rozszerzeniem zadania pierwszego Focused Named Entity Identification and Linking, oprócz trzech klas w pierwszym zadaniu system może wymagać identyfikacji innych klas podmiotów. Poniższa tabela zawiera w pierwszej kolumnie pełną listę rozważanych superklas. Kolumna w środku zawiera niekompletną listę podklas (jeśli występuje), a ostatnia przykładową instancję.

super class	sub class examples	instance examples
Activity	Game, Sport	Chess, Baseball
Agent	Employer, Organisation, Person	Leipzig_University, Angela_Merkel
Award	Decoration, NobelPrize	Humanitas_Prize
Disease		Diabetes_mellitus_type_2
EthnicGroup		Javanese_people
Event	Competition, PersonalEvent	Extended_Semantic_Web_Conference
Language	ProgrammingLanguage	English_language, Scala_(programming_language)
MeanOfTransportation	Aircraft, Train	Airbus_A300
PersonFunction	PoliticalFunction, Profession	
Place		Leipzig
Species	Animal	Cat
Work	Artwork	Debian

1.2 Podział prac

Adam Dębczak:

- Praca nad sprawozdaniem/dokumentacją
- Projekt i integracja całego systemu

Olaf Bergmann:

- Praca nad sprawozdaniem/dokumentacją
- Wykorzystanie PYQT5 do stworzenia interfejsu użytkownika

Piotr Mielcarzewicz:

- Praca nad sprawozdaniem/dokumentacją
- Zapytania do DBpedii w języku SPARQL

Filip Krzemień:

- Praca nad sprawozdaniem/dokumentacją
- Parsowanie zbioru dostarczonego przez OKE
- Przygotowanie zbioru treningowego oraz szkolenie taggera

2. Użyte narzędzia

2.1 NLTK

NLTK jest wiodącą platformą do budowania programów w języku Python. Służy do przetwarzania danych języka naturalnego. Zapewnia łatwe w użyciu interfejsy do ponad 50 korpusów i zasobów leksykalnych, takich jak np. WordNet. Zawiera zestaw bibliotek przetwarzania tekstu klasyfikacji, tokenizacji, tłumaczenia, tagowania, parsowania i rozumowania semantycznego. Ważną zaletą jest też bardzo aktywnie prowadzone forum dyskusyjne użytkowników gdzie można znaleźć rozwiązania występujących problemów: https://groups.google.com/forum/#!forum/nltk-users

2.2 SPARQLWrapper

Pakiet ułatwiający wysyłanie zapytań do serwisu SPARQL. Pomaga tworzyć zapytania URI oraz konwertować ich wyniki do przystępniejszego formatu danych - słowników w Pythonie.

2.3 RDFLib

Biblioteka dla języka Python służąca do pracy ze zasobami zapisanymi w formacie RDF. Zawiera parsery/serializery plików RDF/XML oraz jej interfejs grafowy może służyć jako backend. Stworzona i wspierana przez Daniela Krecha.

2.4 PYQT5

PyQt to zbiór bibliotek Pythona tworzonych przez Riverbank Computing umożliwiających szybkie projektowanie interfejsów aplikacji okienkowych opartych o międzyplatformowy framework Qt dostępny w wersji Open Source na licencji GNU LGPL . Działa na wielu platformach i systemach operacyjnych. Użyliśmy go do stworzenia GUI, ponieważ pozwala na wygodne i szybkie dodawanie elementów w postaci widgetów do programu.

2.5 Stanford NER tagger

Stanford NER to narzędzie zaimplementowane w Javie służące do oznaczania słów (Named Entity Recognition - NER). NER oznacza “byty nazwane” słowa w tekście (najczęściej rzeczowniki lub wyrażenia rzeczownikowe), które oznaczają np. osoby lub nazwy firm, czy też zdarzenia. Paczka zawiera kilka wyszkolonych taggerów do najbardziej klasyczny problemów (np. 3-klasowy - geolocation, person, organisation). Jednak jest możliwe wytrenowanie własnego wyspecjalizowanego modelu dostosowanego do danego problemu poprzez przygotowanie korpusu danych treningowych.

3. Realizacja

Realizację zadania konkursowego rozpoczęliśmy od stworzenia wstępnego szkicu projektu, a następnie podzielenia się zadaniami po których każdy z grupy miał zastanowić się nad problematyką projektu i przedstawić na forum grupy proponowane rozwiązania oraz potencjalnie użyteczne narzędzia. Po wstępie rozdzieliliśmy szczegółowo zadania, jednak poszczególne elementy ostatecznie staraliśmy się wykonać podczas wspólnych spotkań i konsultacji. Co spowodowało, że każdy wniósł do poszczególnych etapów projektu jakiś wkład.

3.1 Działanie

Aplikacja korzysta z taggera NER w celu oznaczania kategorii słów. Pierwszym krokiem było parsowanie zbioru treningowego dostarczonego przez OKE dla zadania drugiego(https://project-hobbit.eu/challenges/oke2018-challenge-eswc-2018/). do formatu używanego w szkoleniu NER taggera. Przy pomocy biblioteki

RDFlib dokonano ekstrakcji interesujących danych oraz skonwertowano je do postaci (słowo - klasa). Kategorie słów danych treningowych uzyskaliśmy z dbpedii za pomocą odpowiednich zapytań SPARQL, a następnie można było rozpocząć szkolenie.

Po przygotowaniu taggera aplikacja jest gotowa do oznaczania słów w używanych tekstach. Byty jednakże mogą składać się z wielu słów, a tagger oznacza je pojedynczo. Musieliśmy więc napisać moduł łączący pojedyncze słowa w odpowiednie byty, oprócz tego znajduje on indeksy startowe i końcowe bytów w tekście wejściowym. URL do dbpedii na podstawie wykrytych bytów pobierane są przy użyciu odpowiedniego zapytania SPARQL. Wszystkie te informacje wyświetlane są następnie w GUI w postaci tabeli oraz tekstu z podświetlonym występowaniem bytów.

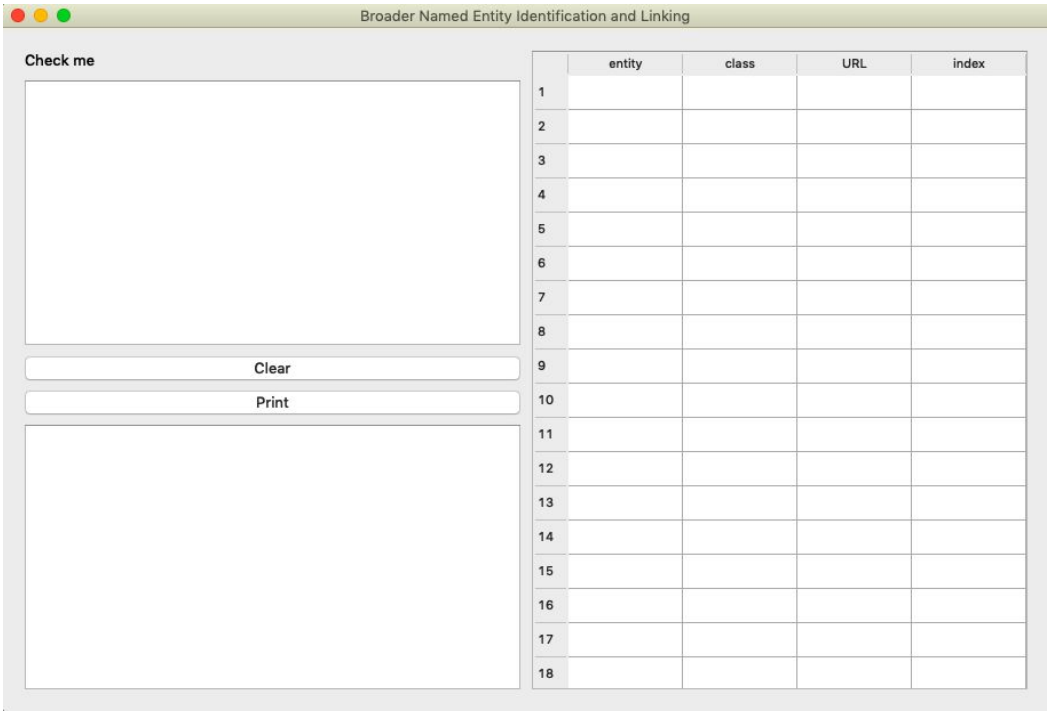
3.2 Instrukcja

Obsługa programu odbywa się poprzez interfejs przedstawiony poniżej na grafice numer 1. Użytkownik chcący go przetestować wpisuje lub wkleja wybrany tekst w górne pole tekstowe, po czym naciska przycisk print w przypadku jeżeli chce otrzymać przetworzony tekst lub przycisk clear jeżeli chce wykasować cały wpisany tekst. Po naciśnięciu przycisku Print tekst zostaje poddany szeregowi operacji opisanych w powyższym podrozdziale 3.1, aby następnie zostać wyświetlony w odpowiedni sposób w dolnym polu tekstowym. Przez odpowiedni sposób rozumiemy podświetlenie odpowiednio wyrażeń na odpowiadający im kolor klasy do której zostały zaklasyfikowane. W ten sposób następujące klasy są podświetlane na kolory czcionek tak jak wyświetlone poniżej:

- **Activity**
- **Agent**
- **Award**
- **Disease**
- **EthnicGroup**
- **Event**
- **Language**
- **MeanOfTransportation**
- **PersonFunction**
- **Place**
- **Species**
- **Work**

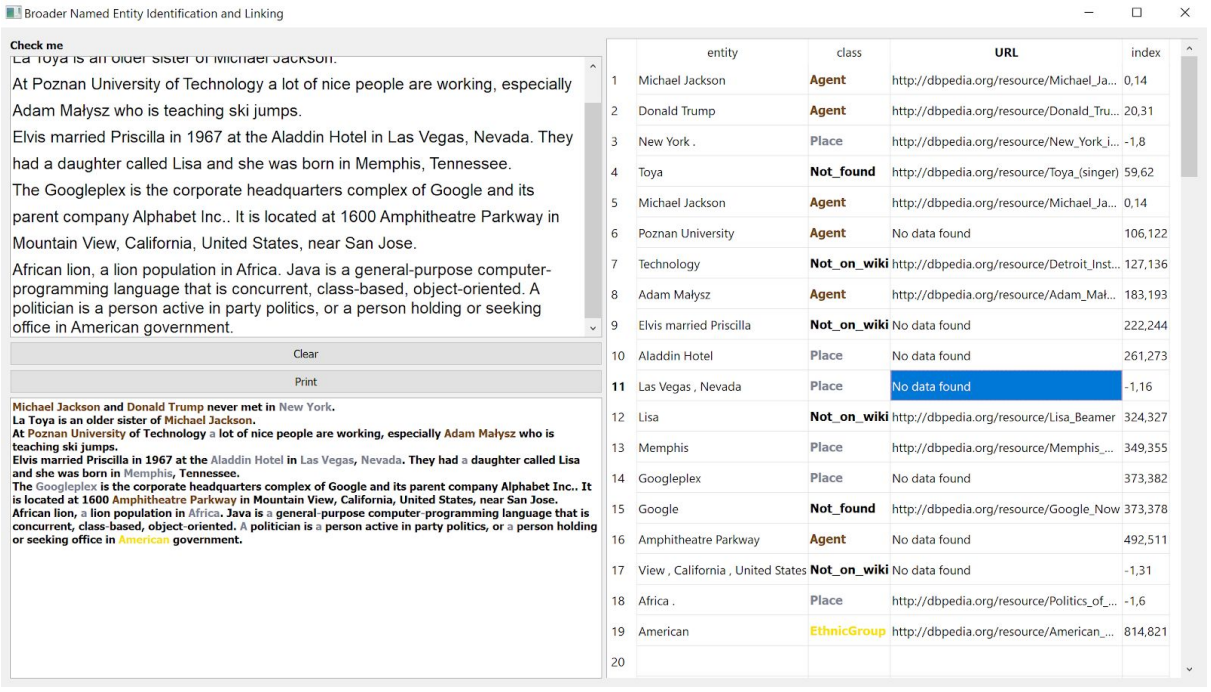
po prawej stronie w tabeli wyświetlane są natomiast przyporządkowania po kolei: pierwsza kolumna entity, druga kolumna class(tutaj również wykonywane jest powyższe podświetlenie tekstu na określony kolor klasy),

trzecia kolumna URL i czwarta kolumna zawierająca index. W przypadku dużej liczby wyników tabela pozwala na przesuwanie.

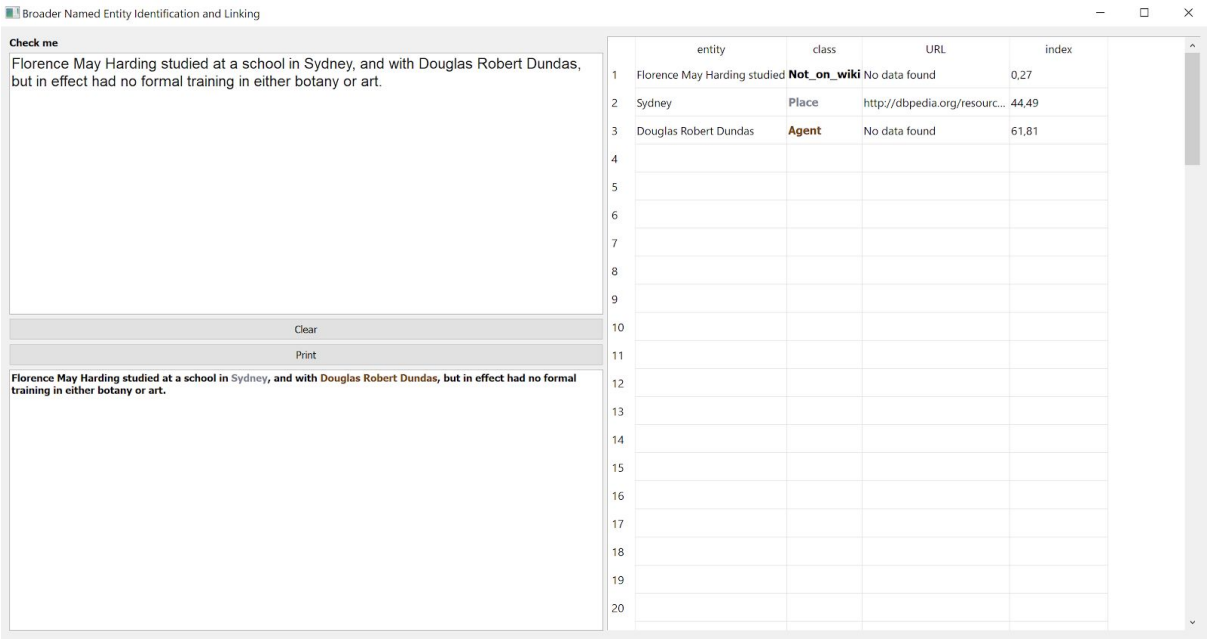


Grafika 1: GUI

3.3 Testy



Grafika 2: Przykładowe wyniki



Grafika 3: Przykładowe wyniki

4. Podsumowanie

Realizacja projektu umożliwiła wstępne zapoznanie się z dziedziną problemów jaką jest Named Entity Recognition, mechanizmami używanymi w rozwiązaniach NLP, oraz formatami nieodłącznie związanymi z tą dziedziną, czyli NIF i RDF.
Przypomnieliśmy sobie również strukturę zapytań SPARQL oraz przetestowaliśmy Stanford Tagger. Zbudowanie graficznego interfejsu pozwoliło nam zapoznanie się z PYQT. Praca grupowa skłoniła nas do przyjęcia modularnej architektury, gdzie każda osoba zajmowała się osobną funkcjonalnością, aby potem wspólnie przedyskutować działanie i zająć się integracją rozwiązania. Ważną kwestią we współpracy było również dbanie o przejrzystość modułów i czytelność kodu. Ostateczne rozwiązanie dało satysfakcjonujące rezultaty, biorąc pod uwagę małą liczność zbioru treningowego. Możliwą drogą rozwoju byłoby ponowne szkolenie na większej ilości przypadków z nowego korpusu oraz usprawnienie mechanizmów korekty i podziału bytów.