Przygotowanie środowiska

Wymagany Python 3.5.

Wirtualne środowisko (opcjonalnie)

```
$ python3 -m venv env               # instalacja venv
$ source env/bin/activate           # aktywacja venv

  ... praca w wirtualnym środowisku ...

$ deactivate                        # deaktywacja venv
```

Instalacja pakietów

Podczas instalacji pakietów pod Linuksem, potrzebny jest kompilator `gcc-fortran`.

```
$ pip3 install -r requirements.txt
```

Instalacja korpusów

Należy uruchomić skrypt instalacyjny, który rozpakowywuje korpus PWr z pliku kpwr-1.2.6-disamb.7z a korpus polski pobiera ze źródła.

```
$ ./installCorpuses.sh
```

Instalacja NLTK

Do tokenizacji słów w podanym tekście użyto NLTK. Dwie opcje instalacji:

1. Interaktywna instalacja w interpreterze:

   ```
   >>> import nltk
   >>> nltk.download()
   ```

2. Instalacja poprzez linię komend:

   ```
   $ sudo python -m nltk.downloader -d /usr/local/share/nltk_data all
   ```

   Dane NLTK zostaną zainstalowane w katalogu `/usr/local/share/nltk_data`.

Instalacja CUDA

1. Pobranie paczki instalującej repozytorium Nvidii ze strony: https://developer.nvidia.com/cuda-downloads, testowana wersja: Linux Ubuntu 14.04, architektura x86_64, paczka deb (local).
2. Instalacja repozytorium w systemie:

   ```
   $ dpkg -i cuda-repo-ubuntu1404-7-5-local_7.5-18_amd64.deb`
   ```

3. Instalacja sterowników i środowiska CUDA:

   ```
   $ apt-get update && apt-get install -y cuda
   ```

4. Restart maszyny w celu załadowania sterowników Nvidii zamiast Nouveau.
5. Instalacja Nvidia cuDNN (biblioteki wspomagające sieci neuronowe): należy umieścić zawartość archiwum `cudnn-7.0-linux-x64-v4.0-prod.tgz` w folderze `/usr/local/cuda`. Do ściągnięcia ze strony https://developer.nvidia.com/rdp/form/cudnn-download-survey.
6. Instalacja modułu tensorflow dla Pythona.

Korzystanie z programu

Konfiguracja środowiska

Należy dodać folder zawierający projekt do zmiennej środowiskowej `PYTHONPATH`. Jeśli akurat się w nim znajdujemy (jest on katalogiem bieżącym), można to zrobić np. poprzez:

```
$ export PYTHONPATH="${PYTHONPATH}:${PWD}"
```

Aby wykonać powyższą komendę wraz z włączeniem `venv`, wystarczy pobrać zawartość pliku `prepare` do shella poprzez:

```
$ source prepare
```

Ustawienia ścieżek

Skrypty korzystają ze ścieżek konfigurowalnych za pomocą pliku src/settings.py.

Konwersja plików xml do csv

Projekt zawiera skrypt umożliwiający konwersję plików xmlowych do formatu csv:

```
$ python3 src/scripts/csv_creator.py -h
usage: csv_creator.py [-h] (--use_pwr | --use_national) [--extract_base_words]
optional arguments:
-h, --help            show this help message and exit
--use_pwr             Use pwr corpus
--use_national        Use national corpus
--extract_base_words  Save words with their base form
```

Tworzenie bazy końcówek

Do stworzenia bazy końcówek służy plik src/scripts/suffix_creator.py.

Tagger - uczenie klasyfikatorów

```
$ python3 src/tagger_trainer.py -h  # więcej o ustawianiu liczby rdzeni
$ python3 src/tagger_trainer.py
```

Domyślnie podczas uczenia używane są wszystkie rdzenie procesora - jeden rdzeń na algorytm. Logi związane z uczeniem zapisywane są do pliku `tagger_factory.log`.

Testowanie taggera

```
$ python3 src/tagger_tester.py  # pojedyncze słowo
$ python3 src/text_tagger.py    # tekst
```

Skrypt zapyta się o słowo/tekst do klasyfikacji.

Uruchomienie benchmarka

```
$ python3 src/benchmark.py
```

Wyniki pomiarów działania programu

Dokładność modeli w bazie

Nazwa algorytmu	Dokładność [%] - korpus uczący PWr	Dokładność [%] - korpus uczący national
Support Vector Machine	56.468	54.949
Decision Trees	57.246	55.671
Stochastic Gradient Descent	53.627	53.121
Logistic Regression	55.571	55.307
Naive Bayes	54.141	52.813
K Neighbors	50.168	50.255
Neural Networks	55.801	54.692

Czas uczenia

Nazwa algorytmu	Korpus uczący PWr `(h:m:s)`	Korpus uczący national `(h:m:s)`
Support Vector Machine	2:05:03	45:46:28
Decision Trees	0:00:30	0:02:24
Stochastic Gradient Descent	0:00:29	0:02:09
Logistic Regression	0:01:02	0:05:41
Naive Bayes	0:00:21	0:01:51
K Neighbors	0:00:23	0:01:49
Neural Networks	0:35:20	2:21:40

Pomiar czasowy wykonany na procesorze Intel(R) Core(TM) i5-2540M CPU @ 2.60GHz.