SWAT - PyNER Intercompatibility

Konrad Delong Antek Piechnik

6 lutego 2009

Spis treści

1 Wstep

W ponizszym dokumencie opisane zostaly narzedzia (wraz z przykladowym kodem) z których korzystaliśmy podczas tworzenia projektu oraz przy tworzeniu pełnej komunikacji kodu w Pythonie z klasami oraz metodami Javy stosowanymi w projekcie SWAT.

2 Struktura projektu

Projekt sklada sie z nastepujacych czesci:

2.1 Tagger tekstow do testow

Aplikacja majaca na celu skanowanie przykladowych tekstow a nastepnie generowania specyficznych raportow odnosnie ilosci wystapien poszczegolnych encji w tekscie. Raporty generowane będą w formie sprzyjającej reszcie systemu efektywne ich użytkowanie. Wykorzystywane one będą następnie do badania sprawności wykorzystywanego algorytmu ewolucyjnego oraz trafnosci pojawiajacych się nowych reguł. Tagger bazować będzie na liście danych encji, z której nie będziemy korzystać w pracy Rdzenia aplikacji.

2.2 Generator reguł n-gramowych

W związku z chęcią implementacji reguł bazujących na najpopularniejszych n-gramach utworzony zostanie generator reguł który zbierze najczęściej występujące n 3,4,5 gramy występujące w bazie nazwisk przygotowanej wcześniej (do celów tagowania tekstow testowych).
Generator, w celu uniknięcia zbierania nic nie znaczących n-gramów porównywał będzie nie tylko jego zawartość ale również pozycję w słowie.

2.3 Rdzen

W tej części aplikacji znajdować się będzie zaimplementowany algorytm który przekazywal będzie zestawy regul, otrzymywal raporty dotyczace ich sprawnosci a następnie tworzyl kolejne zestawy regul na podstawie otrzymanych wyników.
W celu zarówno uproszczenia implementacji jak i ulatwienia dalszej rozbudowy systemu chcemy odizolowac rdzeń aplikacji od całej reszty i nie powierzać mu zadań takich jak skanowanie tekstu. Pozwoli to zarazem na większe pole manewru przy

wyborze języka implementacji.
Rdzeń aplikacji uruchamiany będzie w celu przygotowania optymalnego zestawu reguł, a więc za każdym razem gdy dodane zostaną nowe reguły tudzież gdy dodane zostaną do systemu nowe teksty testowe, mające zwiększyć odporność systemu na coraz bardziej różnorodne dane wejściowe.
Skaner będzie starał się dopasowywać kombinacje słów w tekscie do przedstawionych reguł a następnie starał się zapisać wyniki dla poszczególnych reguł (tak, aby algorytm ewolucyjny mógł sprawnie dobierać nowe reguły na podstawie wyników indywidualnych kombinacji zastosowanych wcześniej.

3 Zastosowane technologie

Do implementacji znacznej czesci aplikacji pragniemy wykorzystac jezyk Python, ze wzgledu na jego perfekcyjne przystosowanie do prac nad prztwarzaniem jezyka naturalnego. Oczywiście za pomocą pewnych narzędzi będziemy się starali powiązać go z obecną architekturą systemu, napisanego w dużej części w języku Java.

3.1 Tagger

Wykonano wstepny prototyp taggera w Pythonie na potrzeby testowe.

3.2 Rdzen

Rozpoczeto prace w Pythonie. Dodatkowo będziemy jak najszybciej starali się skorzystać z Jythona.

4 Zastosowane narzedzia

4.1 Jython

Bridge pozwalajacy na transparentna komunikacje miedzy Pythonem a klasami Javy. Będzie niezmiernie przydatny do komunikacji Rdzenia aplikacji napisanego w pythonie z obecna architektura systemu.

4.2 Wykorzystane biblioteki

- Egothor

- Morfologik - Stemming

5 Opis powiazania aplikacji z Interfejsem IQuantumDetector z projektem SWAT

5.1 Header w pliku Java

```
import jyinterface.factory.JythonFactory;
```

5.2 Kod aplikacji w Javie

```
String shortName =
    "org.ppbw.agh.swat.hoover.smith.quantum.detection.IQuantumDetector";
Object obj = JythonFactory.getJythonObject(shortName,
                                           "pyner/ngrams_detector.py",
                                           "NgramsDetector");
```

5.3 Przykładowe importy w aplikacji pythonowskiej

```
from org.ppbw.agh.swat.hoover.smith.quantum import QuantumType
...
from org.ppbw.agh.swat.hoover.smith.stemmer import StemmerPL
```

5.4 Przykladowe użycie klas Javy w pythonie (za pomoca jythona

```
if ngram in prev_word:
  dq[word_id] =
    DetectedQuantum(leafSegment.getWordToken(word_id), QuantumType.SURNAME)
```