Natural Language Processing 2021

Tutorial 2 — NLP21

Semantische Netze: WordNet, GermaNet

(Syn-)Semantische Netze wie *WordNet* stellen wichtige Ressourcen für NLP dar. In diesem Tutorial werden sie grundlegende Funktionalitäten von WordNet und der deutschen Variante *GermaNet* anwenden. Nutzen Sie die Dokumentationen zu WordNet (nltk.org/howto/wordnet.html) und GermaNet (https://germanetpy.readthedocs.io/en/latest/). Sollten Sie mit der API Dokumentation nicht weiterkommen, nutzen Sie weitere Dokumentationen und Hilfestellungen wie stackoverflow.com oder Tutorials.
Bearbeiten Sie die Aufgaben in einem *Jupyter Notebook*.

1. **Importieren der Module und Daten**.

   (a) **Import der wichtigsten NLP Module**.
   Importieren Sie wie im ersten Tutorial Pandas, Numpy, NLTK und RE und importieren sie WordNet (als wn) von nltk.corpus.

   (b) **Import von "Quality-of-Life Modulen"**.
   Oft sind Module nicht notwendig, erleichtern aber die Arbeit mit größeren Korpora. Installieren sie *pandarallel*. Importieren Sie wie im ersten Tutorial die Methode *pandarallel* (für Parallelization in Pandas) und initialisieren Sie diese mit *pandarallel.initialize()*. Sie können nun bei parallelisierbaren Aufgaben *parallel_apply()* anstelle der Pandas Methode *apply()* verwenden.

2. **Daten importieren.**
   Für das WordNet Tutorial ist ein Datensatz zu *Biased Words* (Wörtern die Voreingenommenheit oder unsachliche Wertung tragen) zur Verfügung gestellt.

   Importieren Sie das als pickle gespeicherte Dataframe "data.pkl".

Fabian Haak: fabian.haak@th-koeln.de

3. **Synsets.**
   Synsets sind die bedeutungsunterscheidbaren Definitionen von Wörtern bzw. Tokens. Ein Anwendungszweck ist es, den Kontext zu Wörtern um bedeutungsgleiche Wörter zu erweitern. Für den gegebenen Datensatz möchten wir die Synonyme zu den Biased Words in Form der *Lemmas* zu den zugehörigen Synsets erfassen.

   (a) **Synsets finden**.
   Extrahieren Sie eine Liste der paarweise verschiedenen Biased Words (Spalte "Label_bias") aus dem Datensatz und speichern Sie diese in einer seperaten Liste "b_words" (nicht im Dataframe!).

   (b) **Synonyme finden**.
   Für jedes der Wörter, bestimmen sie alle Synsets und alle zu den Synsets gehörigen Lemmas. Speichern Sie alle paarweise verschiedenen Lemmas zu allen Biased Words in einer Liste "b_words_synonyms".

   (c) **Kandidaten für neue Biased Words**.
   Die soeben bestimmten Synonyme zu den Biased Words stellen einen guten Startpunkt für die manuelle Bestimmung von weiteren Biased Words dar. Erstellen Sie abschließend eine Liste "new_b_words", in der Sie alle Wörter der Liste "b_words_synonyms" speichern, die nicht in der ursprünglichen Liste von "b_words" enthalten sind. Lassen Sie sich für alle 3 erzeugten Listen die Anzahl der enthaltenen Wörter ausgeben.

4. **GermaNet einrichten**.
   GermaNet ist die an der Uni Tübingen entwickelte Version von Germanet für relationale synsemantische Wortnetze in deutscher Sprache. Obwohl die grundlegenden Funktionalitäten weitestgehend identisch zu WordNet sind, werden sie anders aufgerufen. Da Sie sich im weiteren Verlauf dieser Veranstaltung vermehrt mit den Datensätzen aus dem ESUPOL Projekt beschäftigen werden, könnte GermaNet ein sinnvolles Tool für semantische Analysen darstellen.

   (a) **Germanet installieren und importieren**.
   GermaNet ist nicht öffentlich zugänglich, die TH Köln hat eine Lizenz für die Verwendung im Rahmen von Lehre und Forschung zur Verfügung gestellt bekommen. Füllen Sie zunächst den Geheimhaltungsvertrag (NDA) aus, den Sie im dis25-2021 GitHub finden: https://github.com/irgroup/dis25-2021/blob/main/tutorials/Classroom-Student-Germanet.pdf. Schicken Sie das ausgefüllte NDA an fabian.haak@th-koeln.de. Anschließend werden Sie Zugriff auf die GermaNet Dateien bekommen, die Sie dann unter https://th-koeln.sciebo.de/f/543486731 finden. Kopieren Sie den bereitgestellten Ordner "germanetpy" in ihren Python bzw. Anaconda site-packages Ordner. Vergewissern Sie sich, dass Germanet ordnungsgemäß gefunden wird. Sie können sicher gehen und das Modul noch einmal über pip installieren:

```
import sys
!{sys.executable} -m pip install -U germanetpy


from pathlib import Path
from germanetpy.germanet import Germanet
```

   WordNet nutzt XML für die Relationen und textdateien für Frequencies, die an einem bestimmten Ort abgelegt sein müssen. Folgen sie den Vorgaben der offiziellen API um Germanet richtig eizurichen:

```
data_path = str(Path.home()) + "/germanet/GN_V150/GN_V150_XML"
frequencylist_nouns = str(Path.home())  <ZEILENUMBRUCH>
+"/germanet/GN_V150/FreqLists/noun_freqs_decow14_16.txt"
germanet = Germanet(data_path)
```

   (b) **Datensatz importieren**.
   Importieren sie die Datei "single_term_suggestions.txt" als Dataframe. Die Datei enthält eine Liste von single-Word Query Suggestions aus dem in der Vorlesung vorgestellten Datensatz zur Bundestagswahl 2017. Sie können die Pandas-Methode "read_csv" nutzen. **BONUS**: Unter https://zenodo.org/record/1494858.YHlMnhFCRjE finden Sie den ursprünglichen Datensatz, von dem ausgehend diese Liste erzeugt wurde. Mit dem Wissen aus Tutorial 1 sind Sie in der Lage, sich die Liste selbst aus dem Datensatz aus der Spalte "suggestterm" abzuleiten!

5. **Germanet nutzen**.

   (a) **Synsets**.
   Bestimmen Sie jeweils für alle Suggestions (also jede Zeile der Daten) alle Synsets und speichern Sie die Liste in einer seperaten Spalte.

   (b) **Lemmas**.
   Bestimmen Sie für jede Zeile jeweils alle Lemmas (bzw. *lexikalischen Einheiten*, also *lexunits*) zu allen Synsets und tragen Sie diese als eine Liste in eine neue Spalte ein.

   (c) **Hypernyms**.
   Semantische Netze wie Germanet beschreiben Ist-Beziehungen zwischen Synsets. Hypernyme (Übertypen) und Hyponyme (Untertypen) können hilfreich für die Klassifizierung von Begriffen sein.
   Bestimmen Sie für alle Synsets jeder Suggestion jeweils alle Hypernyme und speichern sie deren Synsets in einer Spalte "hypernyms". Bestimmen Sie anschließend die Lemmas dieser Hypernyme und speichern Sie diese in einer seperaten Spalte.

   (d) **Bonus: Hyponyms**.
   Gehen Sie wie in (c) vor, nur bestimmen Sie dieses Mal die Hyponyme der Suggestions und deren Lemmas. Lassen Sie sich die Anzahl aller paarweise verschiedenen Hypernyme, Hyponyme und Lemmas ausgeben.

   (e) **Klassifizierungs-Tags mit Synsets**.
   Synsemantische Netze können beispielsweise genutzt werden, um Begriffe zu klassifizieren. Nutzen Sie die identifizierten Hypernyme, um alle Städte in dem Dataset zu finden. Klassifizieren Sie in einer Spalte "location" alle Städte, Länder und Orte als "True" und alle anderen Suggestions als "False". Lassen sie sich ein Sub-Dataframe aller Locations im Datensatz ausgeben.

Fabian Haak: fabian.haak@th-koeln.de

6. **BONUSAUFGABE: Semantische Ähnlichkeit und Verwandschaft**.
   Über die relationale Struktur von Synsets lässt sich die Ähnlichkeit bzw. Verwandtschaft zweier Begriffe gleicher Wortart ableiten. Die Ähnlichkeit kann wiederum zum Beispiel zur Beseitigung von Ambiguität verwendet werden. Im Fall des Datensatzes bilden die Terme Suchvorschläge zu personenbezogenen Suchen in Suchmaschienen, denen die Namen von Politikern als Suchterm zugrunde liegen. Der Term "Abbruch" wurde also als Suchvorschlag für mindestens einen Namen eines Politikers vorgeschlagen. Um von den Suggestions nun jeweils das relevante Synset zu identifizieren, kann die Ähnlichkeit zum Synset "Politiker" (Synset(id=s34818, lexunits=Politiker, Politikerin)) bestimmt werden.

   Gehen Sie wie im offiziellen Tutorial zu GermaNet (HIER) erläutert vor, um für alle Synsets aller Suggestions jeweils die Similarity zum Politiker Synset zu berechnen und speichern Sie das Synsets mit der höchsten Ähnlichkeit in einer Spalte "best_syn". Verwenden Sie entweder Path- oder IC- basierte Ähnlichkeit oder führen Sie beides separat durch. Exportieren Sie abschließend ein Sub-Dataframe, in dem nur solche Zeilen enthalten sind, deren Suggestion über mindestens 2 Synsets verfügt.

Fabian Haak: fabian.haak@th-koeln.de