Beschreibung des Codes
Zu den Akteur-Embeddings

Florian Omiecienski

1 Beschreibung

In diesem Ordner befindet sich der Code zum bootstrapen der Akteur-Embeddings. Das Verfahren ist in der Bachelor Arbeit in Abschnitt 4.5 Akteur-Repräsentationen beschrieben. In diesem Ordner liegen eine Reihe von Programmen. Der Code für diese Programme liegt in dem Order *./code/*.
Um die Akteur-Embeddings aus gegebenen FastText-Embeddings zu bootstrapen werden folgende Schritte durchgeführt:

1 Downloaden der Wikipedia-Seiten

2 Vorbereiten der Daten

3 Trainieren der neuen Akteur-Embeddings

Diese drei Schritte korrespondieren mit den Programmen *crawl.py*, *prepare.py*, *bootstrap.py*. Der erste Schritt benötigt eine Liste von Wikidata-Entitäten als Eingabe und erstellt eine Ordnerstruktur, die alle gesuchten Wikipedia-Artikel enthält. Der zweite Schritt vorverarbeitet die Wikipedia-Seiten (tokenisieren, Kontexte extrahieren, etc.). Der dritte Schritt trainiert die Akteur-Embeddings für alle Wikidata-Entitäten für die Daten zu Verfügung standen.
Der Code verwendet im wesentlichen die folgenden Klassen:

- DataExtractor - Stellt die Methoden für *prepare.py* zu Verfügung
- DataManager - Wird von allen Programmen verwendet, managt die Pfade.
- Downloader - Stellt die Methoden für *crawl.py* zu Verfügung
- EmbeddingBootstrapper - Stellt die Methoden zum trainieren der Embeddings zu Verfügung [1]

[1] Verwendet das pytorch-Modell in *bootstrap_model.py*

2 Die Programme

crawl.py
Für alle gegebenen Wikidata-Entitäten werden ihre Wikipedia-Artikel heruntergeladen. Weiterhin werden alle Wikipedia-Seiten gedownloaded, die auf eine der bereits erwähnten Seiten verlinken. Um die verlinkenden Seiten zu identifizieren, wird die *backlinks*-Funktion der Wikipedia-API[2] verwendet. Da zwischen jeder Anfrage an die Wikipedia-API, sowie zwischen jedem Download mindestens 1 Sekunde Wartezeit eingehalten werden muss, kann dieser Schritt bis zu einem halben Tag benötigen. Alle gedownloadeten Artikel werden in dem Ordner *.../database/* abgespeichert. Neben diesem Ordner wird eine Job-Datei und eine Index-Datei abgelegt, welche Meta-Daten über die zu herunterzuladenden Daten enthalten. So kann der Download auch unterbrochen und fortgesetzt werden. In *.../database/* wird für jede gedownloadete Seite der Download-Zeitpunkt in der Datei *download_times.meta* vermerkt.

prepare.py
Für alle gegeben Wikidata-Entitäten wird eine Datei angefertigt, welche den Text der Wikipedia-Seite der Entität enthält. Zusätzlich enthält die Datei alle Kontext-Worte der Links auf diese Entität. Die Texte werden mit Spacy tokenisiert und normalisiert.

bootstrap.py
Für alle mittels prepare.py erstellten Dateien in einem Ordner, werden Embeddings aus den spezifizierten FastText-Vektoren generiert. Der Ablauf dieses Verfahrens ist in der Bachelor-Arbeit beschrieben. Die Ausgabe besteht in einer .vec Datei, welche die erstellten Vektoren im Textformat enthält.

word_count.py
Erstellt eine Liste mit Wörtern und ihren Häufigkeiten in den heruntergeladenen Wikipedia-Seiten. Wird benutzt um in bootstrap.py einen globalen Wort-Prior zu berechnen.

create_random_embeddings.py
Erstellt zufällige Vektoren für alle angegebenen Akteur-Entitäten. Jede Komponente jedes Vektors wird gleichverteilt zwischen 0 und 1 gesampelt. Die Ausgabe besteht in einer .vec Datei, welche die erstellten Vektoren im Textformat enthält.

find_knns.py
Such für eine Menge an zufälligen Entitäten entweder die nächsten Entitäts-Nachbarn im Vektor-Raum oder die nächsten FastText-Wort-Nachbarn. Die Ausgabe erfolgt auf der Konsole.

[2] `https://de.wikipedia.org/w/api.php`

3 Installations-Hinweise

Es werden folgende python3 Pakete benötigt:

- urllib
- argparse
- glob
- json
- math
- matplotlib
- numpy
- os
- re
- requests
- shutil
- spacy[3]
- sys
- time
- torch[4]
- bs4

4 Benutzungs-Hinweis

Zum durchführen der Experimente wie in der Bachelor-Arbeit beschrieben, muss nur das Skript *./entity_bootstrap.sh* ausgeführt werden. Die Pfade sind in dieser Datei beschrieben und können dort auch geändert werden, wenn gewünscht. Bitte beachten Sie, dass der resultierende Ordner um die 3 GB Speicherplatz benötigt.
Bitte beachten sie, dass mindestens 15-20 GB Arbeitsspeicher vorhanden sein müssen, da die FastText-Embeddings ca. 10 GB benötigen.

[3] Das modell *de_core_news_sm* wird benötigt.
[4] CUDA-Support empfohlen, aber nicht notwendig. Ohne kann die Laufzeit sehr lang werden.