Hochschule RheinMain
Prof. Dr. Adrian Ulges

21.10.2021

Anwendungen der KI (WS 21/22)

Aufgabenblatt 1

zu bearbeiten bis: 28.10.2021

Generelle Anmerkungen für's Praktikum:

- Die Aufgaben werden jeweils im Rahmen des nächsten Praktikums besprochen. Wir kommen hierfür bei den einzelnen Teams vorbei.
- Bereiten Sie sich auf eine kurze Besprechung zu folgenden Punkten vor: (1) Welche Ergebnisse haben Sie erzielt? Bereiten Sie insbesondere Beispielergebnisse vor, die sie zeigen können. (2) Gibt es Bugs und Unklarheiten im Code / in der Fragestellung? (3) Erscheinen Ihnen Ergebnisse unplausibel?

Aufgabe 1.1 (Vorbereitung)
Finden Sie sich zu **Zweier-Teams** zusammen. Wählen Sie einen Team-Namen. Legen Sie sich unter `http://gitlab.cs.hs-rm.de/` ein **Git-Projekt** mit Ihrem Team-Namen an. Machen Sie Prof. Ulges und Johannes Villmow zu Projekt-Mitgliedern.
Hosten Sie Ihren Code während des gesamten Semesters im Git-Projekt und checken Sie Zwischenergebnisse zeitnah ein.

Aufgabe 1.2 (Fragen sammeln)

Wir werden uns im Projekt auf sogenannte "Factoid"-Fragen beschränken, d.h. Fragen, die mit einfachen Fakten (wer? wann? wo? was? etc.) zu beantworten sind. Um einen Eindruck von typischen Factoid-Fragen zu gewinnen, schauen Sie sich den Datensatz von Li und Roth an:

`http://cogcomp.cs.illinois.edu/Data/QA/QC/`

Das generelle Vorgehen unseres Systems wird sein, aus der Wikibase das passende Text-Snippet (typischer Weise eine sogenannte *Named Entity*) herauszusuchen und dem User zu präsentieren. **Jedes Team soll für Evaluationszwecke einen Datensatz von 40 Fragen zur Verfügung stellen.** Die Fragen sollen

- sehr einfach gehalten sein (Fakten-Antworten, keine "warum"-Fragen, etc.),
- sich auf folgende Kategorien (gemäß dem Schema von Li und Roth, siehe Link oben) beschränken: HUM:ind, LOC:other, NUM:count, NUM:date, ENTY:other, ENTY:cremat, HUM:gr, LOC:country, LOC:city, ENTY:animal, ENTY:food.
- mit Snippets aus der Wikibase zu beantworten sein.

Für jede Frage benötigen wir:

- die Frage selbst (z.B. “Who murdered Abraham Lincoln?”)
- das Antwort-Snippet (z.B. “John Wilkes Booth”)
- den Antworttyp (z.B. “HUM:ind”)
- die Dokument ID des Artikels, in dem die Antwort sich befindet (z.B. 307)
- einen Satz aus dem obigen Artikel, in dem sich das Antwort-Snippet befindet (z.B. “Abraham Lincoln was assassinated by John Wilkes Booth on Good Friday, April 14, 1865, while attending a play at Ford’s Theatre as the American Civil War was drawing to a close”).

Sammeln Sie 40 Fragen pro Team. Legen Sie zunächst **bis zum nächsten Praktikum 5 Beispiel-Fragen** vor. Sammeln Sie Ihre Fragen in einer “;”-separierten csv-Datei (ein Beispiel finden Sie im Read.MI (“sample_questions.csv”).

Hinweise: Laden Sie sich zunächst die Dokumentsammlung aus dem Read.MI herunter (“Projekt/wikibase.jsonl”). Es handelt sich ein ein json-lines Dokument (ein json-Dokument mit einem Artikel aus der Wikibase pro Zeile). Am effizientesten gehen Sie hierbei vor, indem Sie zufällige Sätze aus zufälligen Artikeln wählen und sich hierzu passende Fragen überlegen. Mit folgendem Befehl erzeugen Sie sich ein Dokument mit 100 zufällig ausgewählten Artikeln:

```
shuf wikibase.jsonl | head -n 100 > wikibase_rand_100.jsonl
```

Aufgabe 1.3 (Python-Crashkurs))
Machen Sie sich mit Python und Numpy vertraut. Eine sehr gute Quelle hierfür ist das folgende Python/Numpy Tutorial der Stanford University. Arbeiten Sie dieses Tutorial durch.

```
http://cs231n.github.io/python-numpy-tutorial/
```

Aufgabe 1.4 (ElasticSearch einrichten)
Arbeiten Sie sich in ElasticSearch (ES) für Textsuche ein. Grundlagen hierfür finden Sie hier:

- **Die ElasticSearch-Referenz**:
  `https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html`
- Wie man ES aus **Python** benutzt:
  `https://elasticsearch-py.readthedocs.io`

Richten Sie nun Ihre eigene ElasticSearch-Instanz ein:

a) Laden Sie elasticsearch herunter (nutzen Sie am besten die Installation aus einem tar-Archiv):
`https://www.elastic.co/de/downloads/elasticsearch`

b) Erstellen Sie sich eine Kopie des config-Ordners. Tragen Sie in `my_config/elasticsearch.yml` die Ordner ein, in denen ES Ihre Daten und Logs speichern soll, z.B.:

```
path.data:  /home/ulges/my_data/foo/
path.logs:  /home/ulges/my_logs/bar/
```

c) Rufen Sie elasticsearch auf (Sie sollten einige Ausgaben auf der Konsole sehen):

```
ES_PATH_CONF=my_config/ bin/elasticsearch
```

d) Testen Sie ob ES verfügbar ist: Der Aufruf `curl localhost:9200` sollte Ihnen eine Antwort mit verschiedenen Daten (u.a. der Tagline "You know, for Search") liefern.

Aufgabe 1.5 (ElasticSearch mit Python)
Stellen Sie aus Python eine Verbindung zu Ihrer ElasticSearch-Instanz her:

a) Schreiben Sie ein kleines Python-Skript, das (1) fünf Dummy-Dokumente (mit jeweils einem einfachen Titel und einem Satz als Inhalt) indexiert, und (2) mit dem Sie per Text-Query in dem Index suchen können.
Hinweise: Mit indices.create() erstellen Sie einen neuen Index. Mit index() fügen Sie ein Dokument hinzu, mit helpers.bulk() mehrere Dokumente auf einmal (deutlich effizienter). Mit search() suchen Sie nach einem Query. Verwenden Sie einen simplen `MatchQuery`.

b) Prüfen Sie ob Sie die gewünschten Dokumente auch unter Flexion finden. Falls nicht, verwenden Sie einen Analyzer, der ein Stemming der Terme durchführt (so dass Sie z.B. bei der Suche nach `patients` Dokumente mit dem Term `patient` finden). Verifizieren Sie anhand eines kurzen Tests, dass Ihre Änderung erfolgreich war.

c) **(schwierig)**: Wenn Sie beim Aufruf von `search()` den Parameter `explain` verwenden, erhalten Sie eine detailierte Analyse, wie die Scores zustandekommen. Welches Scoring-Verfahren verwendet ElasticSearch per Default?