Ausgewählte Kapitel sozialer Webtechnologien - Neuronale Netze

Kursprojekt Exposé

Trainieren eines Word2Vec Modells und Darstellung von Wort- und Dokumentenvektoren anhand von Anfragetexten des FragDenStaat-Projektes

Bearbeitet von:
Sebastian Jüngling (558556)
Konstantin Bruckert (558290)

Datum:
10.05.2019

Prüfer:
Benjamin Voigt

Einleitung

Als Teil der Werkstudententätigkeit von Sebastian am Fraunhofer FOKUS (Abteilung: Digital Public Services) wurde ihm die Analyse des Datenbestandes von FragDenStaat.de anvertraut. Im Zuge dessen ist es auch möglich, themenverwandte Hochschulprojekte zu verbinden. Als Projektmitglied wird Konstantin die Arbeit unterstützen.

Grundlage des Kursprojektes bilden die frei verfügbaren Anfragen auf der Plattform FragDenStaat[1]. Dieses Projekt des Open Knowledge Foundation Deutschland e.V. bietet Nutzern, unter Berufung auf das Informationsfreiheitsgesetz, die Möglichkeit Anfragen an Verwaltungsorgane der Bundesregierung zu stellen. Die FragDenStaat-Anwendung dient hier als eine Art Vermittler zwischen Bürger und Behörde. Im Gegenzug erklärt sich der Nutzer bereit, dass jegliche Kommunikation mit den Behörden frei zugänglich ist.

Anhand der Anfragetexte soll ein Word2Vec Modell mithilfe der Python Library Tensorflow trainiert werden, um Word-Embeddings zu erhalten. Unter Verwendung dieser Word-Embeddings, werden je Anfragetext die Document-Embeddings abgeleitet, welche schließlich mithilfe des Nearest Neighbor Algorithmus t-SNE visualisiert werden sollen.
Ziel unseres Projektes ist es dabei die Semantik der zugrunde liegenden Anfragen automatisiert zu erörtern, wobei mithilfe des t-SNE Algorithmus Anfragen ähnlicher Bedeutung geballt werden können. Idealerweise entsteht dadurch eine eindrucksvolle Visualisierung der Thematiken, die viel von den Nutzern des FragDenStaat-Projektes angefragt werden und die Menschen der Plattform bewegt.

Problembeschreibung

Über das FragDenStaat-Portal wurden mittlerweile mehr als 100.000 Anfragen an Behörden gestellt[1]. Die sich im Rohzustand befindlichen Daten bieten eine große Vielfalt an Informationsgehalt, der auf der Webseite bisher nicht vollumfänglich ausgeschöpft wird. Ziel ist es, diese Lücke zu verringern und aus den bestehenden Daten mehr Informationen zu gewinnen. Bei der Zielsetzung ist es der besondere Fokus des Projekts, im Bereich semantischer Abhängigkeiten zwischen den Anfragen, automatisiert Zusammenhänge zu erkennen, diese aufzubereiten und entsprechend zu visualisieren.

Die Rohdaten sind über die frei zugängliche API des FragDenStaat-Projektes verfügbar und wurden bereits im Zuge der Werkstudententätigkeit von Sebastian bezogen und aufbereitet. Dieser Datensatz bietet verschiedenste Attribute in Zusammenhang mit den von Nutzern gestellten Anfragen an Behörden, wobei im Kontext des Projekts nur die Anfragetexte relevant sind.
Lange Zeit war es schwierig die Semantik von Wörtern bzw. Texten maschinenlesbar und automatisiert zu identifizieren. Mit der Erfindung der Word2Vec Architektur von Tomas Mikolov (Google) im Jahr 2013 entstand der Ansatz, Wörter als maschinenlesbare

[1] (n.d.). FragDenStaat. Abgerufen am Mai 6, 2019, von https://fragdenstaat.de/

Wortvektoren, sogenannte Word-Embeddings, darzustellen und zu vergleichen[2]. Die Semantik eines Wortes wird dabei durch die Verortung eines Wortvektors innerhalb eines n-Dimensionalen Raumes beschrieben. Ähnlichkeiten zwischen Wortvektoren, also der Bedeutung von Wörtern, können daraufhin mit hilfe der Cosine Similarity berechnet werden.

Aktuelle Entwicklungen im Bereich der Generierung von Sprachmodellen schreiten in großer Geschwindigkeit voran. Modelle wie FastText, ELMO, BERT oder GPT2 erreichen vorher nicht da gewesene Fähigkeiten und Präzision[3] und es erscheint schwer, mit dem neuesten Stand der Technik Schritt zu halten. Auch wenn es sich beim Word2Vec Verfahren vermutlich nicht mehr um den neuesten Stand der Entwicklung handelt[3], bietet dieses einen fundierten Einstieg und wird deshalb im Projekt angewandt und mithilfe der Python Library Tensorflow implementiert.

Die Modellarchitektur von Word2Vec kann in zwei Architekturstile unterschieden werden: CBOW und Skipgram. Diese neuronalen Netze lernen anhand großer Textmengen die Gewichtungen (Weights), welche dann als einige hundert dimensionale Vektordarstellung repräsentiert werden können.
Mithilfe der CBOW-Architektur (Continous Bag of Words) wird unter Eingabe eines Kontextes von Wörtern (Input) ein in diesen Kontext passendes Wort (Label) vorhergesagt. Nach Eingabe des Kontextes “Die Katze _ auf den Baum” könnte beispielsweise das Wort “klettert” vorhergesagt werden.
Die Skipgram-Architektur funktioniert auf ähnliche Weise, jedoch invers. Um die Semantik im Satzkontext erfassen zu können, werden die umgebenden Wörter des betrachteten Wortes mit sogenannten Context-Windows in den Lernprozess miteinbezogen. Hierbei gilt das betrachtete Wort als Input und die umgebenden Worte als Label. Somit kann anhand der Eingabe eines Wortes, dessen Kontext vorhergesagt werden. Um die semantische Ähnlichkeit von Wörtern vergleichen zu können hat sich die Skipgram-Architektur bei der Bearbeitung größerer Datensätze als effizienter herausgestellt[4], weshalb diese Architektur im alleinigen Fokus des Projektes steht.

Aus den trainierten Word-Embeddings können nun Document-Embeddings abgeleitet werden. Darauffolgend kann die semantische Ähnlichkeit der Anfragetexte, repräsentiert als n-dimensionale Document-Embeddings, mithilfe der Cosine Simililarity ermittelt werden. Dabei gilt der Abstandswinkel zwischen den Dokumentenvektoren der jeweiligen Anfragen, als Gradmesser der Ähnlichkeit[5].

Um die Vielfalt der semantischen Ähnlichkeit der Anfragetexte zu visualisieren bietet es sich an, diese in einem 2- oder 3D-Raum zu verorten. Da jedoch Word- oder

---

[2] (n.d.). tmikolov/word2vec - GitHub. Abgerufen am Mai 8, 2019, von https://github.com/tmikolov/word2vec
[3] (2018, Dezember 3). The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer .... Abgerufen am Mai 8, 2019, von http://jalammar.github.io/illustrated-bert/
[4] (n.d.). Vector Representations of Words | TensorFlow Core | TensorFlow. Abgerufen am Mai 8, 2019, von https://www.tensorflow.org/tutorials/representation/word2vec
[5] (n.d.). A Beginner's Guide to Word2Vec and Neural Word ... - Skymind. Abgerufen am Mai 8, 2019, von https://skymind.ai/wiki/word2vec

Document-Embeddings in der Regel mehrere hundert Dimensionen besitzen, ist es nötig, die hohe Dimensionalität dieser Vektoren zu reduzieren. Dank spezieller Verfahren kann diese Transformation mit möglichst geringem Informationsverlust bewerkstelligt werden. Neben PCA (Principal Component Analysis) bietet der Nearest Neighbor Algorithmus t-SNE (t-Distributed Stochastic Neighbor Embedding) die Möglichkeit, die Dimension von Vektoren zu reduzieren. Eine Darstellung mittels t-SNE ist besonders geeignet um Strukturen im visualisierten Datensatz, wie lokale Nachbarschaften oder Wort-/Dokumenten-Gruppierung, zu erkennen[6]. Ähnliche Anfragen werden sich somit voraussichtlich clustern und es entsteht eine dreidimensionale Visualisierung, die einen ersten Überblick über die semantischen Zusammenhänge zwischen den Anfragetexten gibt. Geballte Anfragen (Document-Embeddings) werden mit hoher Wahrscheinlichkeit über das gleiche Thema handeln.

Lernziele

Neben dem allgemeinen Vertiefen der Fähigkeiten im Bereich maschinelles Lernen, Mathematik und Phyton, möchte das Team den Einstieg in **TensorFlow** meistern. Rudimentäre Berührungspunkte in vergangenen Semestern sollen aufgefrischt und die Grundlagen gefestigt werden. Überdies soll der Schwerpunkt der Fortbildung auf TensorFlow im Kontext von **Word2Vec** liegen. Bei letzterem Thema gibt es je nach Gruppenmitglied unterschiedliche Erfahrungshorizonte, die im Rahmen des Kursprojekts idealerweise angeglichen und gemeinsam vertieft werden sollen. Zusätzlich handelt es sich bei der **Visualisierung von Embeddings** um ein gemeinsames Interesse des Teams. Im Projektkontext soll diesem anhand des Beispiels mit t-SNE nachgegangen werden, um ein besseres Grundverständnis in diesem (bildlich) abstrakten Thema zu erlangen.

Team-Organisation

Im Rahmen der Projektplanung wurden die folgenden Projektorganisationsstrukturen festgelegt:

- Versionsverwaltung mit GitHub Repository
- Projekt- und Fehlermanagement mit GitHub Projects
- Projektleistung und Dokumentation als Jupyter Notebook(s)
- Deploy-Guide als Markdown

Um Missverständnisse zu verhindern und eskalierende Probleme rechtzeitig zu erkennen, soll besonders auf den direkten Kontakt und persönlichen Austausch im Team geachtet werden. Aus diesem Grund verabredet sich das Team nach dem Motto “Fridays for Falafel” wöchentlich, um sich gegenseitig abzusprechen. Überdies ist es das Ziel, die Möglichkeit des Austausches mit dem Kursleiter in den entsprechenden Veranstaltungen am Montag intensiv zu nutzen.

[6] (2016, Oktober 13). How to Use t-SNE Effectively - Distill.pub. Abgerufen am Mai 8, 2019, von https://distill.pub/2016/misread-tsne/

Meilensteinplanung

Es folgt eine grobe Übersicht über die Meilensteine des Projekts. Einzelne Blöcke werden dann im Laufe des Projekts in kleine Stories aufgeteilt und mittels GitHub Projects verwaltet und verfolgt.

Bezeichnung	Geschätzter Aufwand Arbeitstage	Geschätzte Aufwand Projektanteil	Priorität zu Beginn des Projekts	Risiko für Projekterfolg
Daten Beschaffen und Aufbereiten	-	-	hoch	hoch
Projekt abgrenzen und definieren	-	-	mittel	mittel
Einarbeitung Tensorflow	2 x 2	10%	mittel	gering
Word2Vec Modell erstellen	12	30%	hoch	hoch
tSNE Clustering entwerfen	8	20%	hoch	hoch
Fine Tuning	2	5%	gering	gering
Testing und Kontrolle	12	30%	hoch	mittel
Presentation entwerfen und üben	4	10%	mittel	mittel

Aufgrund der Berufstätigkeit beider Teammitglieder ist mit einem wöchentlichen Projekteinsatz von zwei bis drei Arbeitstagen je Mitglied zu rechnen. Daraus ergibt sich folgender, ungefährer Zeitplan:

- 10.05.2019: Abgabe Expose
- Mitte KW 20: Abschluss Einarbeitung Tensorflow
- Ende KW 21: Abschluss Word2Vec
- Mitte KW 23: Abschluss tSNE
- Mitte KW 24: Abschluss Fine-Tuning
- Bis einschließlich 24.06.2019: Testing und Kontrolle
- 24.06.2019: Projektabschluss
- 24.06. - 30.06.: Anfertigen der Präsentation
- 01.07.2019: Präsentation