

Projektreport

1. Thema und Motivation

Das letzte Jahr hat für viele Menschen eine Ausnahmesituation dargestellt. Eine weltweite Pandemie und der daraus resultierende, nicht enden wollende Lockdown hat uns alle stark in unserem Sozialleben eingeschränkt. Durch soziale Medien wie Instagram oder Messenger-Dienste wie WhatsApp konnten die meisten Menschen trotzdem noch mit Freunden und Familie kommunizieren und so wenigstens einen Teil ihres Soziallebens wahren. Doch was ist mit Menschen, die in diesen Zeiten niemanden haben, mit dem sie ein lockeres Gespräch führen können? Allerdings ist der Bedarf an solch einfachen Gesprächen auch in Zeiten ohne Lockdown vorhanden, man denke nur an die vielen alleinlebenden Rentner, welche die meiste Zeit des Tages vergeblich nach einem Gesprächspartner suchen. Die Nachfrage für ein Tool, welches dieses Problem lösen kann, ist vorhanden und damit ist dieses Problem auch vom wirtschaftlichen Punkt gesehen relevant.

Die oben dargestellte Problematik wollten wir im Rahmen unseres Data Exploration Projects angehen, indem wir den Plan gefasst haben, einen Chatbot zu entwickeln, mit welchem man zwanglosen Smalltalk führen kann. Konkreter besteht das Ziel unseres Projekts darin, auf Basis eines konversationellen Datensatzes ein Modell zu trainieren, welches Input in Form von Texteingaben aufnimmt und algorithmusgesteuert passende Antworten als Output zurückgibt, um so ein Gespräch entstehen zu lassen.

2. Verwendete Technologien und Bibliotheken

Für die Umsetzung unseres Vorhabens haben wir uns für Jupyter Notebook als Entwicklungsumgebung entschieden, da hier zum einen eine gute und einfache Kollaboration möglich ist, zum anderen der Code gut in einzelne Blöcke aufteilbar ist und dadurch auch nur Teilstücke ausgeführt werden können.

Zur Datenverarbeitung haben wir uns für das Pandas-Paket entschieden.

Als Framework zum Lernen des Modells haben wir uns für Scikit-learn entschieden. Dieses Framework ist äußerst gut dokumentiert und speziell für die Anwendung beim Natural Language Processing (NLP) gut geeignet.

Für die Datenvorverarbeitung haben wir die spaCy-library verwendet. Konkret haben wir damit die Tokenisierung und Lemmatisierung unseres Datensatzes umgesetzt. Die Vorteile der

spaCy-Library lagen für uns vor allem bei der guten Dokumentation, der einfachen Nutzung und dass bereits trainierte, statistische Modelle verfügbar sind.

Zudem wurde auch die Python-Library Gensim eingesetzt. Konkret wurden hier die vorverarbeiteten Daten mittels Word2Vec in Wort-Vektoren umgewandelt, um die Erstellung einer für das Clustering notwendigen Feature-Matrix zu ermöglichen.

3. Entwicklungsprozess

Zu Beginn des Projekts haben wir auf Basis des in der Machine Learning Fundamentals Vorlesung vermittelten Wissens und unserer Recherchen einen Ablaufplan für die Entwicklung des Chatbots festgelegt. Der Ablaufplan besteht aus den folgenden 5 Schritten:

- Datentransformation
- Datenvorverarbeitung
- Clustering
- Lernen eines Modells zur Klassifikation
- Finale Implementierung des Chatbots

Im Schritt Datentransformation wurde der Datensatz im CSV-Format geladen und für die weitere Verarbeitung vorbereitet. Anschließend wurde der Text mit Hilfe der spaCy-Library zu Tokens umgewandelt, allerdings haben wir die Daten auch im String-Format beibehalten und für die folgenden Schritte auch nur das Array mit den Daten im String-Format verwendet.

Bei der Datenvorverarbeitung haben wir mit der Gensim-Library gearbeitet. Hier wurden die Textdaten mittels der Methode Word2Vec in Wort-Vektoren transformiert. Auf Basis derer wurde eine Feature-Matrix erstellt, in der alle Wort-Vektoren des Datensatzes enthalten sind.

Auf Basis der in der Datenvorverarbeitung erstellten Feature-Matrix haben wir das Clustering mittels der Ward-Hierarchical-Clustering Methode durchgeführt. Zudem haben wir auch die Cluster Methoden k-means, k-nearest-neighbors und Gaussian-Mixture ausprobiert. Jedoch waren hier die Ergebnisse unbefriedigend, da die resultierenden Cluster zum einen sehr unregelmäßig verteilt waren (Cluster mit mehr als 1000 Einträgen in Kontrast zu Clustern mit einem einzigen Eintrag), zum anderen da die Cluster-Einteilung größtenteils über die vorkommenden Satzzeichen und die Satzlängen bestimmt wurde.

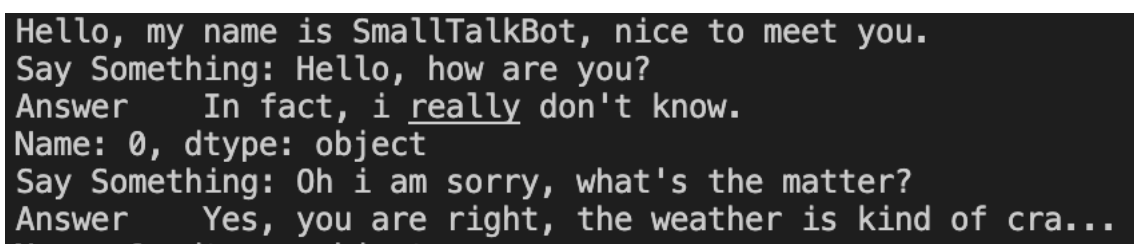
Unsere Entscheidung, final das Ward-Hierachical-Clustering zu verwenden ist darauf begründet, dass hier die Aufteilung nach großen und kleinen Clustern relativ ausgewogen ist und auch thematische Zusammenhänge innerhalb der Cluster erkennbar sind.

Als nächster Schritt erfolgte das Erlernen eines Modells zur Klassifikation neuer Texteingaben auf Basis des zuvor erstellten Cluster-Modells. Der aus dem Clustering resultierende annotierte Datensatz wurde im CSV-Format eingelesen. Die Schritte der Datentransformation- und vorverarbeitung erfolgten analog wie oben bereits beschrieben. Darauffolgend wurden die Daten in Trainings- und Testdaten aufgeteilt. Auf Basis dieser Daten haben wir zum Lernen eines Klassifikators folgende Klassifizierungsalgorithmen ausprobiert: AdaBoost, MLP Neural Network, k-nearest-neighbor und Random Forrest. Unter Berücksichtigung der Accuracy und der Laufzeiten haben wir uns final für k-nearest-neighbor entschieden.

Der letzte Schritt unseres Entwicklungsprozesses bestand darin, den Code unseres Klassifikators in eine ausführbare Python-File zu packen. Diese enthält drei Funktionen, welche beim Ausführen des Codes aufgerufen werden. Der Anwender wird hier nach einem Textinput gefragt, welcher dann durch den Klassifikator einem Cluster zugeordnet wird, um dann eine vorgefertigte Antwort passend zu diesem Cluster als Output zurückzugeben.

4. Erzielte Ergebnisse und Reflexion

Das Hauptziel unseres Projekts war die Entwicklung eines funktionalen Chatbots, welcher dazu in der Lage ist, Smalltalk zu führen. Dieses Ziel wurde erreicht. Wie in Abbildung 1 zu erkennen ist, ist unser Chatbot dazu in der Lage, Texteingaben zu verarbeiten und Antworten zurückzugeben.



```
Hello, my name is SmallTalkBot, nice to meet you.  
Say Something: Hello, how are you?  
Answer In fact, i really don't know.  
Name: 0, dtype: object  
Say Something: Oh i am sorry, what's the matter?  
Answer Yes, you are right, the weather is kind of cra...
```

Abbildung 1: Screenshot Chatbot-Gespräch

Allerdings muss man, kritisch betrachtet, das Ergebnis als unbefriedigend bewerten. Die Antworten des Chatbots sind oft kontextlos und nicht wirklich passend auf den jeweiligen Input. Dieser Sachverhalt ist auf Schwächen in unserem Entwicklungsprozess zurückzuführen, welcher im Folgenden kurz reflektiert wird.

Den verwendeten Trainingsdatensatz haben wir von kaggle.com geladen, unter der Annahme, dass dieser aufgrund der Quelle qualitativ ausreichend ist. Im fortgeschrittenen Entwicklungsprozess sind wir zur Einsicht gekommen, dass dem nicht so war. Dieser konversationelle Datensatz beruht auf einem Gespräch geführt von zwei Bots, was zu Folge

hat, dass die einzelnen Gesprächsbestandteile nur selten in Kontext zueinanderstehen, was die Durchführung eines Clusterings mit ausreichender Güte immens erschwert.

Wir haben uns im Vorlauf der Entwicklungsphase auf einen Cluster Ansatz verständigt, der nicht auf vorgegebenen Themen beruht. Damit wollten wir unserer Ambition, einen Smalltalk Chatbot zu kreieren gerecht werden. Dies führte allerdings dazu, dass unsere Cluster keinen ausreichenden thematischen Bezug hatten. Dies würden wir rückblickend anders angehen und gegebenenfalls Themen beim Clustering vorgeben.

Zudem hatten wir im Vorhinein kein Gütemaß für die Überprüfung der Cluster-Ergebnisse festgelegt, was die Interpretation derer um einiges erschwert hat, dies würden wir Rückblickend auch ändern.

Im Endeffekt sind wir davon überzeugt, dass wir mit unseren neu gewonnen Erkenntnissen bezüglich der Entwicklung eines Chatbots ein nächstes Projekt um einiges effizienter und erfolgreicher gestalten könnten.

5. Related Work

Wissenschaftliche Publikationen zur Entwicklung eines Chatbots gibt es einige. Beispielhaft wird hier kurz auf das Buch von Andreas Kohne mit dem Titel „Chatbots – Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten“¹ eingegangen. Hier wird die Bedeutung des Natural Language Processing (NLP) und konkret die Schritte der Lemmatisierung, Wortklassifikation und Wort-Vektoren behandelt, welche auch Bestandteile unseres Entwicklungsprozesses waren.

Für die Implementation unseres Chatbots wurden zudem die Scikit²-, spaCy³-, und Gensim-Dokumentation⁴ zu Rate gezogen.

¹ Kohne, Andreas (2020): „Chatbots – Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten“, Springer Vieweg, Wiesbaden

² Scikit-Learning, URL: <https://scikit-learn.org/stable/>

³ spaCy, URL: <https://spacy.io/api/lemmatizer>

⁴ Radim Řehůřek, URL: <https://radimrehurek.com/gensim/models/word2vec.html>

Abbildungsverzeichnis

Abbildung 1	Screenshot Chatbot-Gespräch	3
Abbildung 2	Screenshot Chatbot booting up	6

Literaturverzeichnis

Kohne, Andreas (2020): "Chatbots – Aufbau und Anwendungsmöglichkeiten von autonomen Sprachassistenten", Springer Vieweg, Wiesbaden

Radim Řehůřek, URL: <https://radimrehurek.com/gensim/models/word2vec.html>, Abruf am 10.06.2021

Scikit-Learning, URL: <https://scikit-learn.org/stable/>, Abruf am 23.05.2021

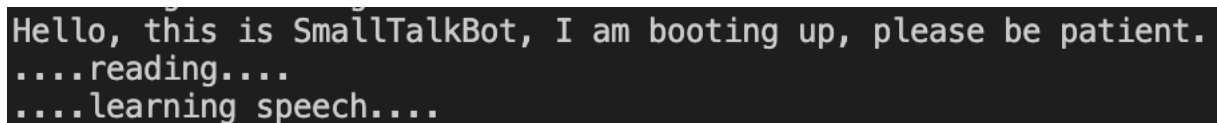
spaCy, URL: <https://spacy.io/api/lemmatizer>, Abruf am 27.05.2021

Anhang

Anmerkungen zum Quellcode

Die ausführbare Python-Datei mit der Bezeichnung „Chatbot_app.py“ findet sich im GitHub Repository im Ordner „Chatbot_Complete“. Diesen Ordner als ganzen downloaden und lokal speichern. Darauf die Python-Datei mit der IDE Ihres Vertrauens öffnen und darauf achten, dass sich im selben Ordner wie die Python-Datei auch die CSV-Dateien „Answers.csv“ und „cluster_model.csv“ befinden. Für die Ausführung des Codes sind die Python Libraries pandas, gensim, spaCy und das Framework Scikit erforderlich, falls nicht schon geschehen, diese downloaden.

Dann den Code ausführen, wenn die oben genannten Schritte durchgeführt wurden, sollte die Kommandozeile wie in Abbildung 2 aussehen. Der Chatbot braucht dann noch einen Moment zu laden und steht dann zu Ihrer Verfügung.

A screenshot of a terminal window with a dark background and light-colored text. The text displayed is: "Hello, this is SmallTalkBot, I am booting up, please be patient.", followed by "....reading...." and "....learning speech...." on separate lines.

```
Hello, this is SmallTalkBot, I am booting up, please be patient.
....reading....
....learning speech....
```

Abbildung 2: Screenshot Chatbot booting up