Transferaufgabe - Sentiment Analysis mit Word Embeddings

August 19, 2020

IDEEN / TODOS:

- allgemeines zu sentiment analysis finden (literatur)
- Medium Artikel zum Dataset: https://towardsdatascience.com/sentiment-analysis-and-product-recommendation-on-amazons-electronics-dataset-reviews-part-1-6b340de660c2

[1]:
```
import pandas as pd
```

Inhaltsverzeichnis

[27]:
```
ones = corpus[corpus.rating == 1.0]
sid = corpus[corpus.index == 991]
#ones.sample(5)
```

1 Einleitung

Die **Sentiment Analysis** ist ein wissenschaftliches Feld des Natural Language Processing, welches sich sich mit Texten befasst, die Meinungen, Stimmungen, Einschätzungen und Emotionen von Menschen beinhalten (Liu 2015, S. 1). Dazu gehören z.B. Filmkritiken, Produktreviews oder Twitterposts. In dieser Arbeit wird die Sentiment Analysis als eine besondere Form der Textklassifikation angesehen. Wichtig bei der Sentiment Analysis sind vor allem Schlüsselwörter oder -phrasen, die Auskunft über die Meinung, Stimmung oder Emotion des Textes gibt. In früheren Jahren wurden dafür zu Unterstützung der Textklassifikationstechniken "Stimmungslexika" verwendet, die den enstprechenden Wörter/Phrasen eine Stimmung (z.B. "gut", "schlecht", "neutral") zuordneten (Liu 2015, S. 10f.). Dadurch konnten jedoch Probleme wie die sich ändernde Semantik eines Wortes hinsichtlich des Kontextes nicht gelöst werden (Liu 2015, S. 10f.). In den letzten Jahren wurden daher immer häufiger die sich als sehr effektiv erweisenden **Word Embeddings** im Rahmen der Sentiment Analysis verwendet (Petrolito, Dell'Orletta 2019, S. 330), da sie beim Erstellen der Wortrepräsentation den Kontext eines Wortes berücksichtigen.

In dieser Arbeit wird eine Sentiment Analysis von Nutzerreviews des Onlineversandhändlers **Amazon** mithilfe eines Convolutional Neural Networks (CNN) durchgeführt. Es soll untersucht werden, welche Word Embeddings in Kombination mit welchen Parametern des Neuronalen Netzes die besten Ergebnisse liefern. Als Word Embeddings werden **GloVe**, **FastText** und **BERT** Embeddings miteinander verglichen. Die Ergebnisse der Sentiment Analysis in Kombination mit den Word Embeddings soll mit einem weiteren Experiment verglichen werden, beim dem das Korpus mithilfe des BERT-Modells *fine-tuned* wird.

Diese Ergebnisse sollen mit einem

Besonderheiten beim Nutzerreview-Korpus sind die Kürze der Texte und die fehlerhafte Orthographie.

TODO: warum cnns? - hier wird funktion für text erklärt: https://github.com/bentrevett/pytorch-sentiment-analysis/blob/master/4%20-%20Convolutional%20Sentiment%20Analysis.ipynb - best practices machine learning mastery: https://github.com/bentrevett/pytorch-sentiment-analysis/blob/master/4%20-%20Convolutional%20Sentiment%20Analysis.ipynb - gutes netzwerk: https://arxiv.org/pdf/2004.03705.pdf (S. 27) - spezielles CNN, **KimCNN**: https://towardsdatascience.com/identifying-hate-speech-with-bert-and-cnn-b7aa2cddd60d

TODO: fragstellung - Word Embeddings (GloVe, FastText) mit CNN vs. BERT/XLNet Transformer, die die Embeddings nutzen (SOTA) - Bert kann nicht so einfach mit CNN verwendet werden, da Modell mitgeliefert werden muss - interessant, wie CNN gegen BERT, welches eine Art Weiterentwicklung von RNN ist, funktioniert - verschiedene CNN Netze?

2 Das Korpus

Das verwendete Korpus ist ein Sammlung von englischsprachigen Nutzerreviews zu den Produkten des Onlineversandhändlers **Amazon** von Julian McAuley (Quelle). Der Zeitraum der Veröffentlichungsdaten der Reviews im originalen Korpus liegt zwischen dem Mai 1996 und dem Oktober 2018. Diese Zeitspanne umfasst ~ 233 Millionen Reviews aus 29 verschiedenen Produktkategorien. Zu jedem Produkt stehen die Bewertung in einer Skala von 1 bis 5 (sehr schlecht bis sehr gut) zur Verfügung, der Reviewtext, die Anzahl der "Nützlich"-Votierungen, eine Verifizierung von Amazon, die Produkt-Metadaten und weitere Links.

Für diese Arbeit wurde eine verkürzte Version des Korpus verwendet. Alle Produktreviews stammen aus der Kategorie "Elektronik" und lediglich aus dem Jahre 2018. Es wurden nur Reviews berücksichtigt, die zu jeder ausgewählten Metainformation ("Bewertung", "Nutzername", "Reviewtext", "Verifizierung", "Datum") Werte enthielten. Das resultiernde Korpus zeigte hinsichtlich der Klassenverteilung eine starke Unausgeglichenheit, weshalb mithilfe von zufälligem Downsampling zu jeder Klasse 15000 Nutzerreviews ausgewählt wurden, um ein ausgeglichenes Korpus zu erhalten (Größe: 75000 Reviews). Ein Einblick in das Korpus wird in der nächsten Zeile gegeben.

[2]:
```
corpus = pd.read_csv("../corpora/small_amazon_reviews_electronic.csv")
corpus.head(3)
```

[2]:

	rating	name	review	verified	vote	date
0	1.0	Mike L	Bought for Christmas present for my Grandson h…	True	0.0	01.02.2018
1	1.0	Gustavo Villalta Woltke	Broken in months	True	0.0	23.05.2018
2	1.0	David	The latest driver for this product on the Asus…	False	0.0	15.05.2018

3 Theoretische Grundlagen

3.1 Word Embeddings

Word Embeddings sind eine besondere Art der distributiven Repräsentation von Wörtern (PILEHEVAR 2020, S. 27). Word Embeddings bauen auf der Idee der **Distributionellen Hypothese** von John Rupert Firth auf, die besagt, dass die Bedeutung eines Wortes durch sein Umfeld geprägt ist. Wörter, die einen ähnlichen Kontext besitzen, haben eine ähnliche Bedeutung. Word Embeddings konstruieren diese Wortrepräsentationen mithilfe von Neuronalen Netzen und basieren meist auf Sprachmodellierungstechniken, mithilfe derer nachfolgende oder fehlende Wörter vorausgesagt werden.

In dieser Arbeit wurden die Word Embeddings **GloVe**, **FastText** und **BERT** verwendet. **GloVe** wurde 2014 von Pennigton et. al. veröffentlicht (Pennigton u.a. 2014). Anders als andere Word Embedding Verfahren verwendet GloVe für die Darstellung der Worthäufigkeiten keine Voraussagemodelle in Form von neuronalen Netzen, sondern eine Kookkurrenz-Matrix, die mithilfe einer Mischung aus maschinellem Lernen und statischen Verfahren aus den Texten gewonnen wird. GloVe hat den Nachteil, dass es nicht gut mit unbekannten Wörtern arbeiten kann (= *Out of vocabulary*-Fehler). Ein Verfahren, welches dieses Problem umgeht, ist das 2016 von Bojanowski et. al. veröffentlichte **FastText** (Bojanowski u.a. 2016). FastText löst das OOV-Problem, indem es während des Trainings anstatt ganzer Wörter Buchstaben N-Gramme lernt, aus denen unbekannte Wörter zusammengebaut werden können. Dies ist leider keine optimale Lösung, da Wörter zwar

aus ähnlichen Buchstaben N-Gramm-Bestandteilen bestehen, sich aber semantisch trotzdem stark voneinander unterscheiden können. Eine bessere Lösung des OOV-Problems bietet das 2018 von Devlin et. al. veröffentliche **BERT** (Devlin u.a. 2018). Wie FastText auch lernt BERT keine ganzen Wörter, sondern Teilwörter, aus welchen es unbekannte Wörter zusammenbauen kann. Anders als FastText oder GloVe zählt BERT jedoch zu den contextualised Word Embeddings, was bedeutet, dass es den Kontext eines Wortes bei der Bildung des Embeddings berücksichtigt. Dies erreicht BERT durch den sogenannten **Attention**-Mechanismus des **Transformers**-Modell, der es erlaubt, relevanten Worten in einer Sequenz mehr Bedeutung als anderen Worten zuzuschreiben. Dabei betrachtet BERT vorhergehende und nachfolgende Wörter (unidirektionaler Ansatz). Da sich durch diesen Ansatz Wörter jedoch "selber sehen" können, verwendet BERT zusätzlich noch die Konzepte **Next Sentence Prediction** (NSP) und **Masked Language Modeling** (MLM). Bei der Next Sentence Prediction überprüft BERT, ob der akutelle betrachtete Satz kontextuell zum nachfolgenden Satz passt. Beim Masked Language Modeling maskiert BERT nach einer gewissen Strategie Wörter, um diese mithilfe der umliegenden Wörter voraussagen zu können. Somit lernt BERT den Kontext von Wörtern, was es BERT erlaubt, zwischen mehrdeutigen Wörtern zu unterscheiden. Ein weiterer Unterschied von BERT zu GloVe und FastText ist, dass es keine **statische**, sondern eine **dynamische** Repräsentation der Wörter liefert. Worte, die die gleiche Schreibweise besitzen, können somit durch unterschiedliche Vektoren dargestellt werden, je nach Kontext und Reihenfolge. Dies bedeutet aber auch, dass auch nach dem Training des Modells dieses für die Benutzung der Embeddings obligatorisch ist. Bei den statischen Word Embeddings GloVe und FastText werden lediglich die Embeddings in Form von Wortvektoren benötigt.

3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) sind eine bestimmte Form von neuronalen Netzen, die vorwiegend für die Klassifizierung von Bildern verwendet werden. Es ist jedoch auch möglich, CNNs für andere Dateitypen wie Texte zu verwenden. Der wichtigste Bestandteil von CNNs sind die *Convolutional Layers*.

TODO

- KIM Paper: https://arxiv.org/pdf/1408.5882.pdf

4 Experimente

TODO:

4.1 Aufbau

TODO - Stoppwörter wurden beibehalten, da ansonsten zu wenig Text

5 Schlussbetrachtung

TODO

6 Literaturverzeichnis

BOJANOWSKI, Piotr, GRAVE, Edouard, JOULIN, Armand, MIKOLOV, Tomas, "Enriching Word Vectors with Subword Information", in: Transactions of the Association for Computational Linguistics, Bd. 5, Juli 2016, S. 135-146.

DEVLIN, Jacob, CHANG, Ming-Wei, LEE, Kenton, TOUTANOVA, Kristin, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", in: Proceedings of the NAACL-HLT Conference, S. 4171–4186.

LIU, Bing, Sentiment analysis. Mining opinions, sentiments, and emotions, Cambridge 2015.

PENNIGTON, Jeffrey, SOCHER, Richard, MANNING, Christopher D., "GloVe: Global Vectors for Word Representation", in: EMNLP (Januar 2014), S. 1532-1533.

PETROLITO, Ruggero, DELL'ORLETTA, Felice, Word Embeddings in Sentiment Analysis, in: Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it (Januar 2018), S. 330-334.

PILEHVAR, Mohammad Taher, CAMACHO-COLLADOS, Jose, "Embeddings in Natural Language Processing. Theory and Advances in Vector Representation of Meaning", 2020.

7 BibText

7.1 BOJANOWSKI 2016

@article{bojanowski2016, author = {Bojanowski, Piotr and Grave, Edouard and Joulin, Armand and Mikolov, Tomas}, year = {2016}, month = {07}, pages = {135-146}, title = {Enriching Word Vectors with Subword Information}, volume = {5}, journal = {Transactions of the Association for Computational Linguistics}, doi = {10.1162/tacl_a_00051} }

7.2 LIU 2015

@book{liu2015, author = {Liu, Bing}, title = {Sentiment analysis. Mining opinions, sentiments, and emotions}, year = {2015} }

7.3 DEVLIN 2018

@article{devlin2018, author = {{Devlin}, Jacob and {Chang}, Ming-Wei and {Lee}, Kenton and {Toutanova}, Kristin}, title = {BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding}, journal = {Proceedings of the NAACL-HLT Conference}, year = {2019}, pages = {4171-4186} }

7.4 PENNIGTION 2014

@inproceedings{pennington2014, author = {Pennington, Jeffrey and Socher, Richard and Manning, Christoper}, year = {2014}, month = {01}, pages = {1532-1543}, title = {Glove: Global Vectors for Word Representation}, volume = {14}, journal = {EMNLP}, doi = {10.3115/v1/D14-1162} }

7.5 PETROLITO 2018

@article{petrolito2018, author = {Petrolito, Ruggero, and Dell'Orletta, Felice}, title = {Word Embeddings in Sentiment Analysis}, journal = {Proceedings of the Fifth Italian Conference on Computational Linguistics CLiC-it}, month = {01}, year = {2018}, pages = {330-334} }

7.6 PILEHVAR 2020

@book{pilehvar2020, author = {Pilehvar, Mohammad Taher and Camacho-Collados, Jose}, title = {Embeddings in Natural Language Processing. Theory and Advances in Vector Representation of Meaning} year = {2020} }

[ ]: