University *of Ljubljana*
Faculty *of Computer and Information Science*

Constructing a Co-Occurrence Graph from a Short Story Database: A Pipeline Approach

Nina Velikajne, Jer Pelhan

Abstract

Literature is a diverse field with unique characters and relationships that interact in complex ways. NLP may struggle to understand these elements due to the ambiguity and unclear references in natural language. In this assignment, we focus on short story NLP analysis. We build our corpus from Gutenberg short stories. On the collected corpus, we apply coreference resolution and then test several methods. First, we test AllanNLP and Stanza models for named entity recognition (NER). We implement, train and test out own BERT model for NER, outperforming both aforementioned methods. An important aspect of literature is also single-character and character-to-character sentiment. We test Stanza and Vader models. Based on the captured information we build and analyse the co-occurrence graph and report the accuracy of each tested method.

Keywords

named entity recognition, sentiment analysis, co-occurrence graph, short story

Advisors: Slavko Žitnik

Introduction

Literature is a rich and diverse field that contains a vast set of characters and relationships. From novels and short stories to plays and poems, literature offers an endless variety of human experiences, emotions, and perspectives. Each literary work presents a unique set of characters with their own personalities, backgrounds, and relationships with each other. These characters interact with each other in complex ways, creating intricate webs of relationships and social dynamics. Understanding these elements might be easy for a human, but it can present a difficult task for natural language processing (NLP) since natural language contains ambiguities and unclear references that might be difficult to understand for machines.

In this paper, we describe our approach for creating a knowledge base of literacy situation models using a combination of named entity recognition (NER), character sentiment analysis, and character graph co-occurrence. Our approach is based on the analysis of short stories that most often have very contrasting characters and more superficial sentence structures. By analysing character interactions in our knowledge base, we will gain insights into how their interactions shape the story, and how the characters are portrayed.

The first step of our approach is applying NER to identify and classify different characters in stories. The entities identified through NER techniques might be noisy, ambiguous, and lack clear semantics. Consequently, identifying connections between related entities in a given dataset, as well as with existing knowledge might enrich the data, and increase disambiguation and data consolidation. Dataset providers often seek to enhance a particular dataset by adding links to comprehensive reference datasets in order to achieve mentioned goals [1]. Herein, we plan to analyse the text to determine which characters interact with each other and how frequent their interactions are and construct a character graph co-occurrence graph. Then, we apply sentimental analysis algorithms to determine character morality and character-to-character sentiment. In the following, we will gain information about the morality of characters and from this in combination with character graph concurrence, which holds side roles, etc.

Our corpus consists of 128 short stories collected from Gutenberg project[1]. 55 stories and their annotations were taken from last year's group project, and 73 stories were additionally annotated by us in order to enlarge the training corpus.

The main goal of our work is to create a pipeline for co-occurrence graph construction. Based on the best-performing models for NER and sentiment analysis we plan to merge them all into a pipeline for constructing such a graph. The co-

[1] https://www.gutenberg.org/

occurrence graph and its details offer readers a comprehensive understanding of the plot and serve as a concise summary of the story and may be important in story understanding.

Related works

There are several approaches to NER in NLP. The simplest one is the rule-based system. One of the first research papers in the field was presented by Lisa F. Rau [2]. It relies on heuristics and handcrafted rules. The system was tested on over one million words and it successfully extracted company names with over 95% accuracy. Newer approaches use machine learning in order to classify NER. Mikheev *et al.* [3] proposed a supervised method based on maximum entropy. They report a recall of 76% for locations, 49% for organizations and 26% for persons with precision ranging from 70% to 90%. The techniques used in (NER) generally depend on lexical resources (such as WordNet), lexical patterns, and statistics derived from analyzing large unannotated corpora. Akbik *et al.* [4] presented contextual string embedding. They use LSTM and Bi-LSTM language models and achieve F1 score of 93.09%. A similar approach is also used in the Stanza Python library, which we use.

The analysis of character interactions is a critical aspect of NLP, and relation extraction plays a key role in this process. Lehmann *et al.* [1] developed RelFinder, a tool that utilizes a BFS algorithm to identify semantic connections among multiple entities in an RDF dataset. This algorithm is responsible for finding all related entities in the tripleset. Rather than focusing on finding connections between two given entities, paper [5] aims to identify the most connected entities in relation to a given relationship and entity. Although their approach offers an intriguing perspective on the problem, it differs from our problem. They search for connected entities based on known relationships, while we will try to uncover such connections between known entities. Leskovec *et al.* [6] presented a technique suggesting positive and negative relationships between people in a social network. This might also be useful in detecting the negative and positive characters in our stories.

Sentiment analysis is a widely researched topic used in commercial applications to predict user opinions on product reviews, detect harmful speech on social media, etc. Without using massive amounts of texts with the annotated sentiment, knowledge-based methods such as the method presented by Andreevskaia [7] rely on sentiment lexicons. Mentioned lexicons are lists of words that typically express sentiment, negative or positive, with the corresponding label (usually an integer), that represents positive to negative sentiment. But the performance of these methods heavily relies on each sentimental word being in the lexical base. Furthermore, some words in different contexts have different sentimental values, which cannot be distinguished using knowledge-based methods. Strapparava [8] a lexical database WordNet-Affect, that extends WordNet for sentiment analysis applications. WordNet-Affect provides a way to identify the emotional tone of the text by linking the words in the text to their corresponding affective information in the WordNet-Affect database. This way, the method can achieve more detailed information about the emotional meaning, "happy" has a positive valence, even though the word "mourning" is next to it in the method presented by Andreevskaia [7], but not with WordNet.

Corpus Analysis

We used two corpora. The first corpus consists of 55 short stories. The data was taken from a project of a group from last year[2]. We decided to enlarge the corpus so we added 73 additional stories, but annotated only the characters, not the sentiment as well. These stories were taken from the Gutenberg project. The dataset and the annotations are publicly available in our GitHub repository[3]. The ground truth is annotated in JSON format for each story separately. The annotations for the first corpus contain a list of all present characters, protagonist and antagonist labels, and sentiments. Sentiments include pairwise sentiments between all the characters. Sentiment 1 annotates a positive relationship between the two characters, -1 a negative one, and 0 a neutral relationship. The second ground truths contain only the annotated characters.

We used the first corpus only for sentiment analysis since it contains annotated ground truth about sentiment. First corpus was also used for testing our NER BERT model. The second corpus was used for training BERT. For all other analyses we used merged corpus – the first and the second corpus together. Our corpus is analysed in Table 1. We can see that the stories are short, and the sentences are simple and not complex.

Methods

Coreference Resolution

Coreference resolution is an essential step of text processing as it identifies pronouns or nouns that refer to the same entity, Furthermore, it also removes named entity deduplicates. We use coreference resolution from AlenNLP [9], which is a strong tool as it employs a network-based approach to link entities across a text. It is trained on large corpora of annotated texts, enabling it to identify co-referent phrases, pronouns and nouns based on their linguistic features such as gender, number and semantic similarity. We use a trained model to identify and link to-referent entities, which enables a more accurate and detailed analysis of relationships between the characters in a literary work. As is demonstrated in Figure 1, our preliminary study has revealed that NeuralCoref's performance was unsatisfactory. Consequently, we have made the decision to exclude it from the subsequent stages of our method's pipeline.

In the preliminary study, we also found that AllenNLP [9] always uses the first mention of the character as the cluster

[2] https://github.com/anzemur/literacy-knowledge-base/tree/main/data

[3] https://github.com/UL-FRI-NLP-Course-2022-23/nlp-course-mataviuc/tree/main/data

Table 1. Basic corpus analysis.

	First corpus	Second Corpus	Merged Corpus
number of stories	55	73	128
shortest (#words)	86	43	43
longest (#words)	454	476	476
avg. num. of words	174.25	161.07	166.73
avg. num. of sentences	6.40	6.77	6.61
avg. num. of words in sentence	27.23	23.80	25.23
vocabulary size	1745	2215	2999
avg. vocabulary size	98.44	89.10	93.11

name. Thus if the first mention is "the lucky beautiful girl called Red Hat", it replaces all occurrences with this long confusing "character name". Thus, we redesign the character cluster naming procedure. Within each cluster, we removed pronouns, definite and indefinite articles, and stopwords from the clustered duplicates of the character. The cluster head was then chosen as the most frequently occurring name for each character. In the preliminary study, this approach yielded better results than the predefined approach of using the first occurance of the named entity.

```
Input:
The wolf hated granny. She pointed gun at him.
He ran out of the window as she watched him.
The grandchildren were watching her with the gun in
her hands.

AllenNLP:

The wolf hated granny. granny pointed gun at The
wolf.
The wolf ran out of the window as granny watched The
wolf.
The grandchildren were watching granny with the gun.

NeuralCoref:

The wolf hated granny. The wolf pointed gun at him.
him ran out of the window as The wolf watched him.
The grandchildren were watching The wolf with the gun
in The wolf hands.
```

Figure 1. Comparison of AlenNLP [9] and NeuroCoref [10] coreference resolution performance on a challenging paragraph.

Named-Entity Recognition

NER is a subfield of NLP that deals with identifying and classifying named entities in text. Named entities are specific entities that are referenced by proper names, such as people, organizations, locations, dates, and other miscellaneous entities like products and events. It involves training machine learning models to identify and classify these named entities within a given text. Once the model is learned it can be used to automatically identify and extract named entities from new texts. We use two methods in order to detect NER in our dataset, namely the AllenNLP NER [4] and the Stanza NER [5].

[4]https://demo.allennlp.org/named-entity-recognition/named-entity-recognition

[5]https://stanfordnlp.github.io/stanza/ner.html

Last, we tried to learn our own BERT model for character detection in short stories.

AllenNLP NER uses a bidirectional LSTM-CRF model. Each word is represented as a vector pre-trained word embeddings, such as GloVe or FastText. These vectors are the input to a bi-directional LSTM layer, which captures the context and other dependencies of each word with respect to other neighboring words. This is then inputted into Conditional Random Field (CRF) layer. It applies a probability distribution over all possible NER tags for each word. The model is optimized using a maximum likelihood objective function, that tries to maximize the probability of the correct tag for each input [11].

Stanza NER uses a Bidirectional Encoder Representation from Transformers (BERT). BERT is a powerful language model that has been pre-trained on massive amounts of text data. Stanza NER first tokenizes text and then produces a probability distribution for all possible NER tags for each word. The model uses word embeddings, attention mechanisms, and convolutional neural networks to analyze the context surrounding each token and make predictions about the entity tag. After prediction, the model uses a post-processing step to combine adjacent tokens with the same entity label into a single entity. This improves the overall accuracy of the NER [12].

BERT or Bidirectional Encoder Representations from Transformers is a pre-trained language model that is based on transformer architecture and uses a large-scale unsupervised training method to learn contextual relationships between words in a sentence. The model was shown to outperform many other methods [13]. To enhance the performance of BERT in character detection we propose a two-step training procedure. In the first step, the pre-trained model is trained on a modified NER dataset for a specific NLP task of character detection. In this step, we modified the training data tags "B-per" and "I-per" into a "PER" tag and other tags e.g., time, and location, to O – denoting "other" since they are not relevant in character detection. We further removed the POS tags from the NER dataset in order to avoid the tedious annotation of POS tags in the next training step. In the second step, we utilized the second corpus to train the model based on the ground truth annotations of characters. We used the pre-trained BERT model from the previous step and finetuned

it for the specific task of character detection.

Sentiment analysis

For sentiment analysis of stories, we use the VADER [14] rule-based approach for sentiment analysis. It uses a lexicon-based approach to determine the polarity (positive, neutral, or negative) of the input text. For character-to-character sentiment analysis, we find all sentences and blocks of sentences where two characters interact with each other and analyse the polarity of the language towards each other. This way we determine their relationship and sentiment towards each other. For character morality analysis VADER and Stanza models are used to analyse the actions and words of a character in a story. By analysing the polarity of sentences or actions that are used/performed by a character, we portray the character as good or bad.

Co-occurrence graph

For constructing the co-occurrence graph we implemented our own method. First, we applied coreference resolution to uniform all character occurrences. Next, we extracted the named entities, namely all the characters in the story. For extracting the NER we used our trained BERT model. Once we extract the characters, we initiate an adjacency matrix, where columns and rows represent specific characters. We iterate through the sentences and count how many times two characters occur together. The obtained adjacency matrix is symmetric. Based on the adjacency matrix we construct an undirected graph. Each character is represented as a node. Two nodes are connected if two characters co-occur. Based on the counts in the adjacency matrix, we also determine the weights of links. In order to get a better story interpretation from the constructed graph, we also added different colors that notate the positive/negative/neutral relationships between characters. Sentiment was detected with the Vader model as described in the previous section.

Results

Named-Entity Recognition

We tested AllenNLP and Stanza NER on a short story database. We first applied coreference resolution to uniform all character occurrences. Next, we extracted characters in every story using the two mentioned NER models. Once the characters were extracted, we compared the obtained predicted characters with the ground truth characters. For each model, we calculated the precision, recall, and F1 score. One of the biggest issues was detecting characters without a name e.g., queen, wolf, farmer, possibly written in lowercase. Several models do not recognize this as a named entity. So tried to train our own BERT model on the labeled dataset in order to obtain more accurate results. We fine-tuned the BERT model for character detection using a learning rate of 5e-5, an epsilon value of 1-e8, a batch size of 32, and warmup steps of 5. We used warmup steps, to gradually increase the learning rate from a low value to an optimal value during the initial stages of the training, concluding in more stable training. This helps the model to converge faster and achieve better accuracy. These hyperparameters were selected based on previous research and our own experimentation. We tested our BERT model on the same corpus as the mentioned AllenNLP and Stanza models. Results are in Table 2.

NER model	Precision	Recall	F1 score
AllenNLP	0.74	0.77	0.72
Stanza	0.80	0.65	0.68
BERT (NER)	0.00	0.00	0.00
BERT (FINETUNED)	**0.71**	**0.92**	**0.78**

Table 2. Obtained precision, recall, and F1 score for AllenNLP and Stanza NER, BERT trained for named entity recognition and proposed version of BERT models on the fables dataset.

Sentiment analysis

We performed character sentiment analysis or character morality. We analysed the sentiment of sentences and actions that are used/performed by the character and took the most occurring polarity of the character as the morality of the character. Morality is classified as 1, 0, and -1, which stands for positive, neutral, and negative character, respectively. As visible in Table 3, Vader outperforms the Stanza method.

Sentiment model	Precision	Recall	F1 score
Stanza	0.77	0.63	0.68
Vader	**0.78**	**0.70**	**0.70**

Table 3. Obtained precision, recall, and F1 score for Vader and Stanza pipelines on fables dataset for character morality classification.

We also added a character-to-character sentiment to our knowledge base. Within this part of our method, we searched for subject-verb-object triplets using the Spacy [15] language processing tool. In each such sentence, we performed sentiment analysis. The resulting value was put in an affinity matrix between the subject to object, with both being the characters. The results are presented in Table 4. The Vader outperforms Stanza on the sentiment prediction task with respect to all used metrics.

Sentiment model	Precision	Recall	F1 score
Stanza	0.66	0.61	0.61
Vader	**0.71**	**0.60**	**0.63**

Table 4. Obtained precision, recall, and F1 score for Vader and Stanza pipelines on short story dataset for character-to-character sentiment analysis.

Co-occurence graph

For constructing the co-occurrence graph we built the adjacency matrix as presented in Methods. We constructed a graph for each story in the dataset. We tested the accuracy of the method by reading the story, manually constructing the co-occurrence graph, and comparing it to the automatically constructed graph. We decided to represent only the most interesting examples. The graph also contains sentiment values between characters. The sentiment is shown with colored edges between two characters. Green means positive sentiment, red negative, and orange neutral sentiment. Two examples of such constructed graphs can be seen in Figure 2 and Figure 3.

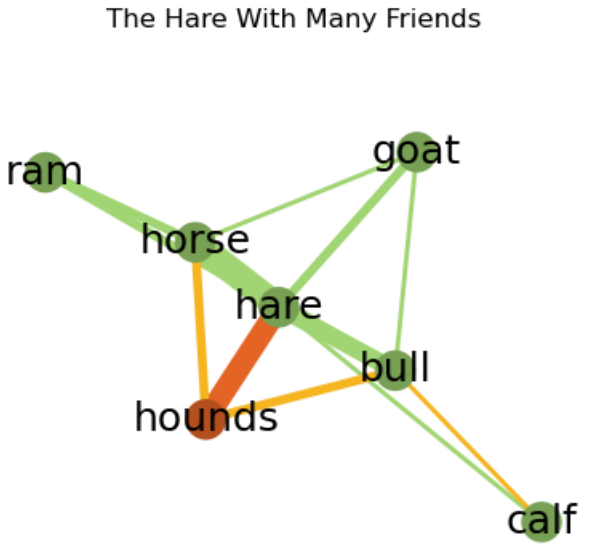

Figure 2. Co-occurence graph for The Hare With Many Friends. Thicker links between nodes symbolize higher weights, and more co-occurrences of the two characters. Edge colors represent character-to-character sentiment, and node colors represent single-character sentiment. Green represents positive sentiment, orange neutral and red negative.

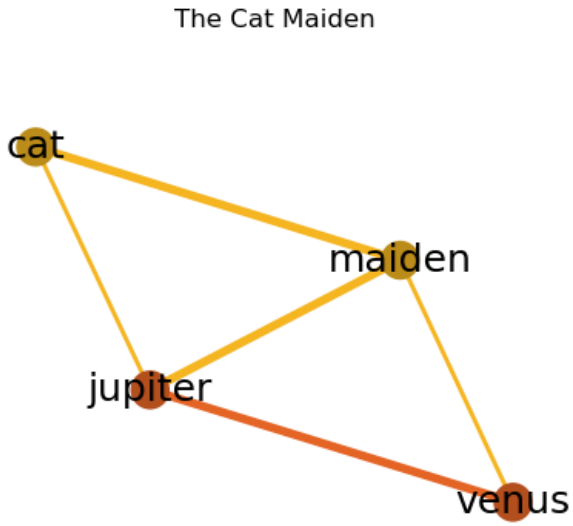

Figure 3. Co-occurence graph for The Cat Maiden. Thicker links between nodes symbolize higher weights, and more co-occurrences of the two characters. Edge colors represent character-to-character sentiment, and node colors represent single-character sentiment. Green represents positive sentiment, orange neutral and red negative.

1. Discussion

1.1 Named-Entity Recognition

We can see that Stanza has higher precision and lower recall (see Table 2), this means that the model is better at minimizing false positives (FP) compared to minimizing false negatives (FN). In other words, the model is more cautious in making positive predictions and tends to be more accurate when it does predict a positive instance. It detects fewer correct characters (more FN), but also fewer wrong characters (less FP).

AllenNLP on the other hand, has similar precision and recall. This means that the model is performing consistently in both minimizing false positives and minimizing false negatives. The rates of correctly identifying positive instances and correctly rejecting negative instances are balanced. The recall and F1 score for the Stanza model are much lower compared to the AllenNLP. AllanNLP compared to the Stanza model detects more characters, but not all are relevant. However, it misses less characters (less FN), herein higher recall.

By taking a look at the results we realized that a lot of characters are not detected. A significant challenge we encountered was identifying characters without explicitly mentioned names, such as ”queen,” ”wolf,” or ”farmer,” which may be written in lowercase. Many existing models failed to recognize these as named entities. As a solution, we embarked on training our own BERT model using a labeled dataset. Our aim was to improve the accuracy of character detection and achieve more precise results in such cases. We can see that our fine-tuned BERT demonstrated superior performance in character detection compared to both Stanza and AllenNLP, with 13% and 7% boost of F1, respectively. Furthermore, the recall of our trained version of the BERT is significantly higher, meaning that both other methods predict enormously less true characters. This is expected since our trained BERT model is adapted to characters occurring in short stories.

1.2 Sentiment analysis

We can see that the Vader model outperformed the Stanza model in both single-character and character-to-character sentiment analysis (see Table 3). The sentiment can be a challenging task for such a model. There are several challenges. For example, a character can first be positive but with the story it becomes negative. Or one can have positive sentiments with all other characters, except for one and it affects the sentiment as negative. Another major problem are ”bad” adjectives, if

an "evil queen" does something good, the sentiment might still be negative due to the word evil. Or in our case there is a character named "Ass", and due to the known word, the sentiment of such a character is always negative.

1.3 Co-occurence graph

The story The Hare With Many Friends contains the most characters – 7. The hare is the main character and the other animals are her friends. This can be nicely seen from the constructed graph (see Fig. 2). The hare is connected to all other animals, and the weights on the links are the heaviest, as one would expect. Hounds are an enemy of the hare – this means negative sentiment, and all other animals are friends – positive or neutral, sentiment. Another example was chosen randomly, namely The Cat Maiden (see Fig. 3). Jupiter changes a cat into a maiden and together with Venus they observe her behavior. The constructed graph is correct, and so are the links and their weights. The link between Jupiter and Venus is stronger, and so is the link between the cat and the maiden. Characters are in neutral relationships, except for Venus and Jupiter since they argue in the story.

From the constructed graph, we can assume a lot about the story. E.g., we can assume who is the main character – the node with the most links. Based on the edge thickness and weights we can assume which relationships are the most important in the story. And based on the color of the edges we can also assume who is the bad and who is the good character. Based on all this information a potential reader might get a good insight about the story and its plot. The graph can be also used as a short summary of the story.

2. Conclusion

In this assignment, our objective was to analyze short stories and devise a methodology for creating a co-occurrence graph. To accomplish this, we experimented with multiple models for NER and sentiment analysis. Our goal was to implement and evaluate these models to establish an effective method for our base creation process.

For NER we used the pre-trained AllenNLP and Stanza models. AllenNLP outperformed the Stanza model – achieving a lower trade-off between precision and recall (74% and 77%). Stanza obtained a recall of 65%. In order to obtain better NER results we also trained our own BERT model on a corpus collected by ourselves. We collected and annotated 73 short stories. Next, we tested the BERT model on the same corpus as AllenNLP and Stanza models. Our BERT model outperformed both models – achieving a precision of 71% and recall of 92%.

Next, we tested the Stanza and Vader models for sentiment analysis. We did two analyses, single-character, and character-to-character respectfully. Vader outperformed in both cases. For the single-character sentiment, Vader achieved a precision of 78% and recall of 70%. Stanza had problems with a lower recall score. And for character-to-character sentiment, Vader achieved a precision of 71% and recall of 60%.

Last, we attempted to create a graph using the information we extracted. Utilizing the BERT model for NER, we identified all character pairs and determined the frequency of their co-occurrences. This information was then used to construct an adjacency matrix for the graph. Additionally, we incorporated sentiment analysis to represent the relationship between characters by assigning different edge colors. By combining all the tested methods into a pipeline for graph construction, we provide valuable insights about the story. For instance, we can determine the main character by identifying the node with the highest number of links. By analyzing edge thickness and weights, we can infer the most significant relationships within the story. Moreover, the color of the edges allows us to distinguish between good and bad characters. All these details offer readers a comprehensive understanding of the plot and serve as a concise summary of the story.

All of the mentioned methods were tested on a short story corpus. Prior to implementing any of the previously mentioned methods, we applied an enhanced coreference resolution technique.

References

[1] Philipp Heim, Sebastian Hellmann, Jens Lehmann, Steffen Lohmann, and Timo Stegemann. Relfinder: Revealing relationships in rdf knowledge bases. *SAMT*, 5887:182–187, 2009.

[2] Lisa F Rau. Extracting company names from text. In *Proceedings the Seventh IEEE Conference on Artificial Intelligence Application*, pages 29–30. IEEE Computer Society, 1991.

[3] Andrei Mikheev, Marc Moens, and Claire Grover. Named entity recognition without gazetteers. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 1–8, 1999.

[4] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649, 2018.

[5] Yong-Jin Han, Seong-Bae Park, Sang-Jo Lee, Se Young Park, and Kweon Yang Kim. Ranking entities similar to an entity for a given relationship. In *PRICAI 2010: Trends in Artificial Intelligence: 11th Pacific Rim International Conference on Artificial Intelligence, Daegu, Korea, August 30–September 2, 2010. Proceedings 11*, pages 409–420. Springer, 2010.

[6] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 641–650, 2010.

[7] Alina Andreevskaia and Sabine Bergler. CLaC and CLaC-NB: Knowledge-based and corpus-based approaches to sentiment tagging. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-*

2007), pages 117–120, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[8] Carlo Strapparava and Alessandro Valitutti. Wordnet-affect: an affective extension of wordnet. *Vol 4.*, 4, 01 2004.

[9] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[10] Thomas Wolf. State-of-the-art neural coreference resolution for chatbots, Sep 2020.

[11] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[12] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[14] Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, volume 8, pages 216–225, 2014.

[15] Explosion AI. spacy: Industrial-strength natural language processing in Python. https://spacy.io, 2020.