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

Literacy situation models knowledge base creation

Nace Gorenc, Jernej Vrhunc in Žan Pečovnik

Abstract

Although analyzing literary works and their content is a skill commonly taught and practiced by individuals, it poses significant challenges for machines. Machines struggle with the absence of human knowledge, common sense, and contextual awareness, all of which are crucial in the analysis of literary works. Researchers have made varying degrees of progress in addressing these challenges, some of which we mention in our related work. In our study, we narrow our focus to a specific aspect of literary analysis: fictional character analysis. Our investigation revolves around tackling the issues of character extraction, using named entity recognition and sentiment analysis of character relationships. To facilitate our research, we have created a new corpus of ground truths, which we annotated by hand and used for our model.

Keywords

natural language processing, named entity recognition, coreference resolution, sentiment analysis, character relationship analysis

Advisors: Slavko Žitnik, Aleš Žagar

Introduction

Natural language processing mainly focuses on different approaches of processing, analyzing and understanding large amounts of text. Extracting information from text can have great added value in many aspects of life such as science or economy. In recent years natural language processing became one of the hottest research topic. One of the driving forces was public availability of ChatGPT, which caused quite a stir since people started using it for various task which were formerly done by hand and with a little bit of help from search engines like Google. This report focuses on the analysis of personal relationships between fictional characters, using coreference resolution, named entity recognition (NER) and entity co-occurence graph (ECG).

Related work

A relation extraction boggled minds of the scientists long before NLP's renaissance driven by the invention of transformers.

In 2011, authors of [1] reviewed multiple supervised and semi-supervised classification approaches to the relation extraction task along with critical analysis. Among the supervised approaches, dependency path kernels stood out as the best both in term of computational complexity and performance. On the other hand, semi-supervised approaches seemed to be well suited for open domain relation extraction systems since they could easily scale with the database size and could extend to new relations easily.

Groza and Corde [2] focused on extracting information about literary characters from folktales. One of the tasks was to identify the main characters and the parts of the story where these characters are described or act. Their system relies on the folktale ontology which is based on Propp's model for folktales morphology and it defines three knowledge sources. The first one is *folktale morphology*, the second one is *folktale main entities* and the last one is *family relationships in folktale*. The authors used three sequential algorithms for retrieving information about characters. The main goals of these algortihms were to extract characters from the folktales, use decoreference resolution and finding the perspective of the characters.

More recently the authors of [3] proposed method of a hybrid approach which combines the features of unsupervised and supervised learning methods, which also uses some rules to extract relationships. The method identifies the main characters and collects the sentences related to them. Then these sentences are analyzed and classified to extract relationships. For testing purposes, 100 short stories were used with around 300 character pair relations. This model identified parent-child relationships, friendships or if there was no relation.

In a recent survey [4], Vincent Labatut and Xavier Bost

represented a character network, which can be used for extraction of relations between characters in works of fiction (e.g. novels, movies, plays, TV series). The main goal of this survey was to explain necessary steps, such as character identification, graph extraction and interaction detection, to construct such network.

Dataset analysis

We were not successful in finding any short stories that would already have annotated which characters appear in the stories and the relationships between them. Thus we decided to create our own dataset of stories and annotations for them. In order to create a dataset of sufficient size, we decided to find stories which would have approximately 1000 words or even less in some cases, to not spend too much time on just annotating the data. We found the stories on two different websites, the first one is American Literature [5] which offers a bunch of different stories of different length and from various authors for public reading. The second website is Mom Loves Best [6], which contains parenting tips from real moms and within these tips we also found some moral stories for children.

For the creation of the annotated corpus we created two new files for each selected story. The first file contained the raw text of the story and the second one contained the annotation of this story in JSON format. For each story we annotated:

- which characters appear in the story,
- what are the relationships between these characters.

To get this information we read and analyzed every story, the annotation can be seen below in listing 1. We can see that for each character we evaluated the sentiment relationship between him and everyone else and classified the relationship in 3 different classes:

- **-1**: negative sentiment,
- **0**: neutral sentiment,
- **1**: positive sentiment.

Note that each character has a neutral sentiment with itself. We had to remove a few stories that we annotated by hand, because they were not compatible with our model and we also expanded the dataset with additional stories gathered from last year's projects, which ends up for us having 65 short stories and their corresponding annotations. All of the data is available in our GitHub repository [7].

Listing 1. Annotation example

```
{
    "characters": [
        "john wick",
        "dog",
        "killer"
    ],
    "sentiments": {
        "john wick": {
            "john wick": 0,
            "dog": 1,
            "killer": -1
        },
        "dog": {
            "john wick": 1,
            "dog": 0,
            "killer": -1
        },
        "killer": {
            "john wick": -1,
            "dog": -1,
            "killer": 0
        }
    }
}
```

Methods

0.1 Character extraction

0.1.1 NER (Named entity recognition)

To identify characters, we employed Named Entity Recognition (NER), which is a text processing technique that aims to identify and classify named entities into predetermined categories such as PERSON, ORGANIZATION, LOCATION, etc. We utilized various NER models and libraries in our project, including Flair, Spacy and Stanza. Our primary focus was on recognizing people's names to create a character list.

Flair

Flair [8] is a robust NLP library that enables the utilization of cutting-edge natural language processing models on text, including Named Entity Recognition (NER), part-of-speech (PoS) tagging, and specialized support for biomedical data, sense disambiguation, and classification, with a constantly growing range of languages. Additionally, Flair functions as a text embedding library, which means it is used for representing text with its own embeddings, as well as BERT embeddings and ELMo embeddings. Furthermore, it is directly built on PyTorch, which facilitates the training of custom models. Flair representations are based on a bi-LSTM character-based monolingual model that has been pretrained on Wikipedia.

Spacy

Spacy [9] is a Python-based open-source library for advanced natural language processing. While it is primarily designed for production use rather than research, it is still quite user-friendly. Spacy offers a wide variety of models and supports many foreign languages. Spacy's NER model works by using machine learning algorithms to analyze the linguistic features of text, such as part-of-speech tags, dependency parse trees, and context clues. The model is trained on large datasets of annotated text to recognize patterns and associations between words and named entities.

Stanza

Stanza [10] is a suite of efficient and precise linguistic analysis tools that includes advanced NER models for eight different

languages. These modules are developed on PyTorch framework, which makes it easy to train and evaluate models using annotated data.

0.1.2 CR (Coreference resolution)

Coreference resolution is an essential task in natural language processing that identifies all the referring expressions in a text that refer to the same real-world entity which can be seen in figure 1. The aim is to cluster these expressions, replacing them with the most appropriate mention to represent the entity, and thus reducing ambiguity in the text. This task can be performed using various libraries, such as spaCy [9] and Allennlp [11]. By performing coreference resolution before named-entity recognition, the ambiguity in the text is reduced, leading to improved performance in NER tasks.

Figure 1. Example of how coreference resolution works

For the actual task of coreference resolution we used the implementation from Allennlp, which consists of a pre-trained model and for the tokenization of the input text we used SpaCy's tokenizer. The model tries to get the embedded representation of each span of the input text using SpanBERT. The embeddings are scored with the goal of removing the spans which would unlikely occur in coreference clusters and the model then decides which of the remaining spans are coreferent with each other.

As it turns out the used library is not the best at resolving coreferences, so we made some small adjustments to cluster head selection. Cluster head is a mention with which every coreference in the text would be replaced with. The selected library always selects the first one detected, so we changed that by ignoring the clusters which don't include any noun phrases and then selected the head in a more profound way by selecting the first noun phrase. After the head is selected all of its coreferences are replaced with in the text.

Since the Stanza NER yielded the best results, we decided to use this throughout the experiment and for better compatibility also use the Stanza's Pipeline sentiment analysis instead of Afinn.

0.2 Sentiment analysis of character relationships

0.2.1 Stanza

After Coreference Resolution (CR) and Named Entity Recognition (NER) were ran on a specific story, we then ran the sentiment analysis on the given processed story with Stanza's sentiment analysis Pipeline. This was done on a sentence level, meaning that we knew exactly in which sentence a certain character appears and thus we could evaluate each sentence separately and sum up the scores of each sentence to get the final evaluation. We then got a score for a certain character which could either be less than 0, 0 or greater than 0. If the score was less than 0, then the conclusion would be that he had a negative sentiment. If the score was greater than 0, then the conclusion would be that he had a positive sentiment. And if the score was exactly 0, the character had a neutral sentiment. We can see the examples below in figure 2 and figure 3. Thus we also obtained the character occurrence matrix which contains the occurrences of each character in each sentence of the story.

	text	sentence_sentiment
1	Misha	-1
2	Nafanail	-1
3	Porfiry	-1
4	Stanislav	-1
0	Luise	0

Figure 2. Example of sentiment analysis with Stanza's Pipeline of story 003.txt

	text	sentence_sentiment
0	Easton	-11
1	Easton's	-1
2	Mamma	0

Figure 3. Example of sentiment analysis with Stanza's Pipeline of story 005.txt

0.2.2 Co-occurrence and sentiment matrix

For computing the character relation sentiment we used 2 different approaches after character extraction. The first approach used the Afinn sentiment analyzer and the second approach used Stanza's pipeline. From both of these approaches we could compute the sentiment of each sentence in the text, from which we could then compute the character occurrence matrix.

After obtaining the character occurrence matrix we could now compute the character co-occurrence matrix by multiplying the occurrence matrix with its transposition. The computed co-occurrence matrix is of size $n \times n$, where n is

the number of extracted characters. Each entry in the co-occurrence matrix represents the number of co-occurrences in the text between the i-th and j-th character where i represents the row and j represents the column of the matrix.

From the co-occurrence matrix we could now also compute the sentiment matrix by multiplying the occurrence matrix with the product of the transposed occurrence matrix with the sentence sentiment vector. The computed sentiment matrix is of size $n \times n$, where n is the number of extracted characters. Each entry in the sentiment matrix represents the sentiment between the pair of i-th and j-th character where i represents the row and j represents the column of the matrix. The sentiment matrix is then aligned by adding the product of the co-occurrence matrix and the alignment rate variable to it, to account for the author's writing style.

Both matrices are transformed to their lower triangle form and their diagonals are set to 0, since a character does not have any co-occurrences with itself nor can it have any sentiment with itself. Lastly the sentiment matrix is normalized in the range of $[-1, 1]$ and from it we generate the JSON files containing the data about all of the extracted characters and their sentiments. The files are then used for evaluation and visualization.

0.2.3 Visualization

From the generated JSON files we could now generate character relation graphs. If the color of the edge between two characters is colored with green color, this means that those two characters have a positive sentiment and if the edge is colored with red color, this means that those two characters have a negative sentiment. There is one more possibility, where the edge is colored with gray color, which would mean that the characters have a neutral sentiment. The width of the colored edge represents the strength of positive or negative sentiment against chosen character. If the edge is thicker the sentiment is stronger and vice versa. The edges are directed, since it is possible that one character has a positive sentiment towards the other, while the other way around this does not have to be the case. Examples of these graphs can be seen in figure 4 and figure 5.

Results

0.3 Character extraction

We performed character extraction using 3 different NER methods with or without coreference resolution. For the evaluation of these methods we computed precision, recall and F1 score for each of the used models comparing them to the ground truth annotations that we provided. The performance of these methods can be seen in table1.

NER	Flair		Spacy		Stanza	
CR	No	Yes	No	Yes	No	Yes
Precision	0.697	0.632	0.462	0.442	0.75	0.752
Recall	0.811	0.715	0.638	0.642	0.841	0.781
F1	0.706	0.627	0.507	0.504	0.757	0.709

Table 1. Results of evaluating NER models with and without CR

As can be seen from the results, Stanza's model is the best out of 3 since it out-performs both Flair's and Spacy's model in terms of F1 score and recall, while in terms of precision the Spacy's model is a bit better. With the addition of coreference resolution we can notice that it has a great impact on the performance of all models, because the characters' names get repeated more in the text causing the NER models to more likely detect the characters.

0.4 Sentiment analysis of character relationships

From the generated JSON files, which contain the information about extracted characters and their sentiments, we could then generate the character relation graphs. We also computed the precision, recall and F1 score for both approaches of sentiment analysis and the results can be seen in table 2. As we can see, the Afinn approach works better than stanza.

Sentiment model	Precision	Recall	F1
Afinn	0.557	0.563	0.523
Stanza	0.463	0.516	0.458

Table 2. Results of evaluating sentiment analysis approaches

The visualizations show us which character is in relation with whom and what their sentiment is. Green directed edge from character $c1$ to character $c2$ means that the character $c1$ has a positive sentiment towards character $c2$. Similar to this, red directed edge means a negative sentiment and the gray directed edge means that the sentiment is neutral. We can see that it is possible that $c1$ does not have the same sentiment towards $c2$ as $c2$ has towards $c1$.

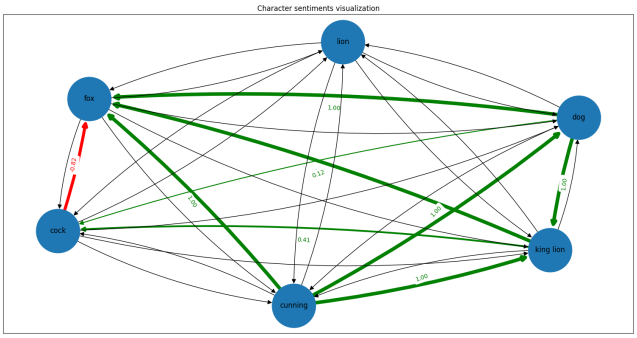

Figure 4. graph of character relationship sentiment

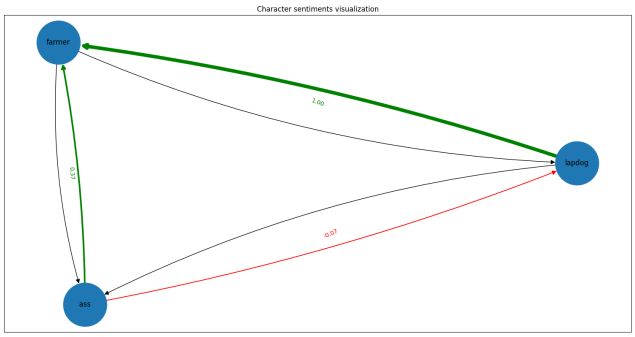

Figure 5. graph of character relationship sentiment

The visualizations are not very informative, since all of the stories have a relatively low number of characters, meaning that usually each character interacts with every other character in the story whereas if we take for comparison Harry Potter books, there are dozens of different characters and not all interact with each other, providing a much more interesting graph. Nevertheless we can still see different sentiments between characters in the graphs, proving that the model has successfully detected sentiment of their relationships. We can also see in the graphs that not all edges have the same width, meaning that the thinner the edge is, less co-occurrences the connected characters have.

Discussion

We are somewhat satisfied with the final results comparing the Afinn and the Stanza's pipeline sentiment analysis, even though we had some problems running NER and CR on our initial dataset due to the size of the stories, since they were so big that we ran out of RAM running the code. We tried splitting the initial stories into smaller chunks and running the algorithm separately on each chunk, but it turns out that the results were very bad. The CR did not know how to properly perform the algorithm, since it often occured that one person was mentioned by name in the previous chunk while the reference to its name was in the next chunk ending in the algorithm not recognizing this.

With switching up the dataset our algorithms started to work, comparing the Afinn analyzer with the Stanza's pipeline analyzer, the Afinn one was a bit better in all three scores, achieving better results for approximately 10-15%, but there is still lots of room for improvement since the F1 score is only 0.523 for the Afinn analyzer. We are very satisfied with the outcome of the visualization of the graphs, since they clearly show how much sentiment there is between two characters and what kind of sentiment this is. An idea for improvement would be to also include the number of occurrences of a specific character in the final graph, meaning that the bigger the node would be in the graph, the more occurrences this character would have in the story indicating that it probably plays a lead role in it. From this we could also then compute who is the protagonist or the antagonist of the story.

References

[1] Nguyen Bach and Sameer Badaskar. A review of relation extraction. 05 2011.

[2] Adrian Groza and Lidia Corde. Information retrieval in falktales using natural language processing. In *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 59–66. IEEE, 2015.

[3] V. Devisree and P.C. Reghu Raj. A hybrid approach to relationship extraction from stories. *Procedia Technology*, 24:1499–1506, 2016. International Conference on Emerging Trends in Engineering, Science and Technology (ICETEST - 2015).

[4] Vincent Labatut and Xavier Bost. Extraction and analysis of fictional character networks. *ACM Computing Surveys*, 52:1–40, 2020.

[5] 100 great short stories. https://americanliterature.com/100-great-short-stories/. Accessed: 2023-04-05.

[6] 20 good short moral stories for kids. https://momlovesbest.com/short-moral-stories-kids. Accessed: 2023-04-05.

[7] Github repository - skupina123. https://github.com/UL-FRI-NLP-Course-2022-23/nlp-course-skupina-123. Accessed: 2023-04-05.

[8] Flair - a very simple framework for state-of-the-art nlp. https://github.com/flairNLP/flair. Accessed: 19.3.2023.

[9] Spacy - industrial-strength natural language processing. https://spacy.io/. Accessed: 19.3.2023.

[10] Stanza - nlp package for many human languages. https://stanfordnlp.github.io/stanza/ner.html. Accessed: 19.3.2023.

[11] Allen nlp. https://allenai.org/allennlp. Accessed: 19.3.2023.