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

Literacy situation models knowledge base creation

Aljaž Sebastjan Ahtik, Jaša Kerec, Luka Kuzman

Abstract

In this assignment, we're faced with creation of a knowledge base creation for both English and Slovene stories and novels. Our pipeline consists of named entity extraction, co-reference resolution, character sentiment extraction and visualization of the obtained data. The pipeline is the same in both languages, enabling us to compare implantation challenges and results based on language. Our end product is protagonist and antagonist extraction for each of the stories in the corpora.

Keywords

corpus, knowledge base, named entity recognition, co-reference resolution, sentiment analysis, co-occurrence extraction

Advisors: Slavko Žitnik

Introduction

Text comprehension is a difficult task in the field of natural language processing, as it requires solving multiple sub tasks. In this project, we strive to build a knowledge base for a number of Slovenian and English novels which focuses on exploring the relationship sentiment between the characters in the stories, namely the antagonist/protagonist relationship. Using it, we can quickly grasp the contents and ideas of the books, even if we have not read it before.

For the purposes of our project, we constructed a corpus which consists of:

- **7 English novels** to test the correctness of our solutions,
- **33 Slovenian short stories** with simple plot lines using which we will determine the correctness of our solution on Slovenian texts and
- **21 Slovenian novels** which are more advanced texts and will be used to test the performance of our implementations on more difficult cases.

Due to a large amount of information we can extract from the corpus we limit ourselves to a small subset of them. Our initial plan consists of named entity recognition, character sentiment recognition, relationship extraction, antagonist and protagonist detection and visualization of the data.

Related work

When analysing stories the first task is often named entity recognition. One approach to solving this problem is by using large databases of language specific known entity names, however in most languages and domains there is very few training data available. Lample et al. [1] present two different LSTM-based models for named entity recognition that capture ortographic and distrubutional evidence without resorting to any language-specific resources.

The next important part of the story analysis is relation extraction. This problem is generally solved either through supervised or unsupervised learning algorithms. For using supervised algorithms we need a text corpus for which the entities and their relation types are known, which is not always the case. So the authors of [2] proposed a hybrid approach. This approach identifies the main characters and collects the sentences related to them. The collected sentences are then processed and classified to extract relationships between characters.

Sentiment analysis deals with understanding emotional tone behind a string of text. Sentiment analysis approaches can be split into three categories: machine learning based (such as Naive Bayes, Support Vector Machine and Maximum Entropy), lexicon/corpus based (which employs a sentiment dictionary, one such being NRC Emotion Lexicon [3], the drawback being that a human must be present) and hybrid methods. Article [4] present some existing solutions regarding sentiment analysis.

In order for the model to better understand the text, it needs to employ some common sense reasoning. Because of this many databases of commonsense knowledge were built, however, the data is spread over many sources with

different foci. Ilievski et al. [5] attempt to combine these different sources into one knowledge graph which comes with three key challenges: (1) the different knowledge modeling approaches, (2) imprecise descriptions of entities, and (3) the sparse overlap between the sources. They achieve this by constructing a commonsense knowledge graph linking seven key sources.

Another important concept in natural language processing is the concept of causality, which can informally be described as a relationship between two events such that one event causes the other. Shingo Nahatame [6] investigates the properties of global and local causality, semantic text relations and their effect on L2 readers' memory, finding that global structure of the text rather local has a stronger impact on how well the subject remembers the text. This is in contrast to semantic relations, where local relations have a larger impact. In light of this information, we investigate a work by Tirthankar et al. [7] which propose a linguistically informed deep neural network in order to extract casual relations from documents, finding that a bi-directional LSTM performs well on the task. Sendong Zhao et al.[8] employ an approach using Restricted Hidden Naive Bayes model to extract causality. The advantage of this approach is in its ability to cope with partial interaction amongst features, which helps avoid overfitting present with Hidden Naive Bayes model. Besides better text comprehension, causality is also useful when predicting medical results, future natural disasters and their aftereffects etc.

Caselli, T. and P. Vossen in [9] presented a new dataset for training and evaluating models for causal and temporal relation extraction. They also presented three baseline systems with their performance on the dataset which showed how complex the task is and gave directions for the development of more robust systems. The dataset (corpus) is meant to provide an intrinsic evaluation benchmark for the StoryLine Extraction task. The task is composed of three basic parts. (1) Event detection and classification - detect and classify events (which compose a topic) in each document. (2) Temporal anchoring of events - Anchor each event mention with the time in which it happened. (3) Explanatory Relation Identification and Classification - classify the storyline relation type based on the selection of event pairs that are temporally and logically connected.

The methods presented reach beyond the scopes of this project. For example, sentiment analysis finds its use in medicine. One such overview is presented by Kerstin Denecke and Yihan Deng [10].

Methods

We evaluate our methods on three different datasets. In this section we will describe our entire dataset and all the methods we use. We will also define and explain every step in our pipeline for protagonist and antagonist detection. The whole pipeline is shown in 1.

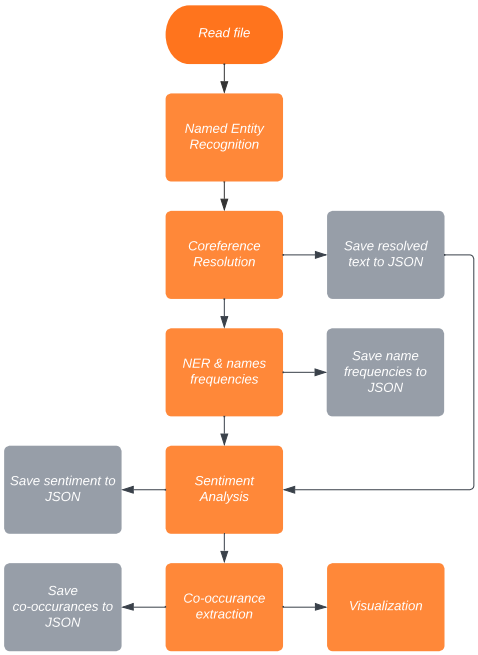

Figure 1. Pipeline for protagonist/antagonist extraction

Datasets

Our corpus consists of three datasets. First one is IMapBook project dataset in english. It consists 7 short stories:

- **The Ransom of Red Chief** (about 4.160 words).
- **Hills Like White Elephants** (about 1.470 words).
- **Leiningen Versus the Ants** (about 8.610 words).
- **The Lady, or The Tiger** (about 2.710 words).
- **The Most Dangerous Game** (about 7.950 words).
- **The Tell-Tale Heart** (about 2.200 words).
- **The Gift of the Magi** (about 1.860 words).

These stories are considered classics in English literature and have been widely read and studied by scholars, students and enthusiasts alike. The dataset is suitable for use in various natural language processing tasks. The stories were originally published in the late 19th and early 20th centuries, and offer a glimpse into the socio-cultural and historical context of that period. They showcase a variety of literary styles and themes, including suspense, irony, humor, and romance. Overall, this dataset provides a rich and diverse source of literary content for researchers and practitioners working in the field of natural language processing and related areas.

In the second dataset are 33 Slovenian short folktales. A folktale is a traditional story that has been passed down orally from generation to generation within a particular culture or community. These stories typically have anonymous authors and are often part of a larger body of oral tradition. Folktales may involve animals, supernatural creatures, or human characters and are usually used to teach moral lessons, explain natural phenomena, or simply entertain. On average, the stories from this dataset are a little bit shorter than the stories from IMapBook project. They also include fewer characters

than the previous mentioned stories, since there are rarely more than 4 in one story. In many stories there are no specific names of the characters or the main characters are animals.

The last dataset consists of a collection of Slovenian novels, each with a word count ranging from 1000 to 10000 words. The novels cover a wide range of genres, including romance, adventure, and mystery, and were written by a diverse group of authors. Overall, this dataset provides a rich and varied source of Slovenian literary content for analysis and exploration.

Named entity recognition

Named entity recognition (NER) is a process for determining which of the input tokens represent a named entity and is a fundamental step in identifying the relationship between characters in stories [2]. NER is a very challenging problem, especially in less common and complicated languages like Slovenian, where there is only a small amount of supervised training data available and there are few constraints on the word form of named entities. "As a result, carefully constructed orthographic features and language-specific knowledge resources, such as gazetteers, are widely used for solving this task" [1], however, these are costly to develop and adapt. More recently, semi-supervised machine learning methods are being used to learn the word grouping, orthographical, and distributional evidence of named entities. In this paper, we will use the built-in NER preprocessors from Stanza [11] for the English language and from Stanza-based Classla [12] for the Slovenian language.

Coreference resolution

The important step before the named entity extraction is coreference resolution. Because often characters are described with pronouns instead of full names. Coreference resolution (CR) is a task that involves identifying all expressions in a text that refer to the same entity. It is a crucial step in many NLP applications, including information extraction, machine translation, question answering, and sentiment analysis. The goal of CR is to enable NLP systems to correctly identify and resolve coreferent expressions, which can greatly improve the accuracy and performance of downstream tasks. The task of CR is challenging because coreferent expressions can have different forms and can appear in different parts of the text. For example, consider the sentence "John went to the store. He bought some milk." The pronoun "he" refers back to the noun phrase "John", and therefore, the two expressions are coreferent. In a given example we can see that in the second sentence "John" is replaced with "he" so the NER algorithm will not detect that as a "John" entity. But coreference resolution does exactly that, it replaces pronouns with actual entities. For the coreference resolution in Python we use the original implementation of Fastcoref method described in [13]. It is based on a neural network architecture that uses a combination of syntactic and semantic features to identify and resolve coreferent expressions in the text. One of the key features of Fastcoref [13] is its speed. It can process large amounts of text in a matter of seconds, making it suitable for use in real-time applications. This is achieved through the use of efficient data structures and optimized algorithms, as well as a parallel processing pipeline that takes advantage of multi-core CPUs and GPUs. For the Slovenian language, we used the BERT-based coreference model [14].

Character sentiment extraction

Our next task is to extract character sentiments from stories. The methods range from simpler to more complex. Due to the fact that we were using the same methods on both Slovene and English text (in order to compare performance), we limited ourselves to methods that were viable in both languages, those usually being the more popular ones (especially with a less popular language like Slovene). For each character that was extracted using NER, we compute its sentiment in the story. This is done by computing the sentiment of the sentences the word appears in, as we work under the assumption that protagonists on average appear in more positive sentences and protagonists more often appear in negative sentences.

Since we took the approach of saving each step of the pipeline into its own JSON files (decoupling the steps, allowing to run each of them on their own), we have three ways to extract the sentences which contain the characters. The most straightforward one is searching for the character string in a sentence that matches the name. This approach is the fastest, however, we have the largest chance to miss the sentences with characters included in them, since the names have to match exactly, however, it is also the fastest method. We also presume that it might work better being that sentences where the character is explicitly mentioned might end up being more important. The second one is first lemmatizing the sentence and name first and searching to determine in which sentences it appears. This is especially useful in Slovene where names can take on multiple forms (conjugations). This gives us a larger chance of finding the named entity, however, it requires additional text processing and is therefore more demanding. The third approach is the extraction of sentences using co-reference, which gives us the largest amount of sentences since we also include the sentences where the character is not mentioned directly (by his name), as discussed in the previous section. We also work with the co-reference data extracted in the previous section. In the end, we found that co-references worked best, so we use it in further experiments.

For examples of demonstration, we created a short story, presented below.

```
Janez je vesel. Tine je jezen. Janez je
videl Petra. Janez in Peter sta najbo-
ljša prijatelja. Janez ima sovražnika
Tineta.
```

For all of the books, we output the character sentiments in a JSON file. An example for the short story is presented on Listing 1. All of the sentiments are normalized to lie in the range -1 (negative sentiment) to 1 (positive sentiment).

If there was more than one character detected in the story, we also decided to bring the sentiment mean to a 0 and get a clearer comparison of sentiment between characters which is not as affected by the story's overall sentiment.

Listing 1. JSON object including character sentiments and frequencies.

```
sentiments = {
    'Janez': {
        'frequency': 5,
        'sentiment': 0.22203999999999996
    },
    'Peter': {
        'frequency': 2,
        'sentiment': 0.3602
    },
    'Tine': {
        'frequency': 2,
        'sentiment': -0.5264500000000001
    }
}
```

Here, Janez and Peter have a positive sentiment, while Tine has a negative sentiment.

AFINN

AFINN (Affective Norms for English Words) [15] [16] is a simple and very popular influential lexicon in the field of sentiment analysis. The lexicon consists of English words that are assigned a numerical score based on their polarity ranging from negative to positive. As such, this method works only on a word-per-word basis. Calculating the mean of the word scores in a given text gives us a quick and efficient way of determining its sentiment. Its simplicity also makes it easy to implement from scratch, requiring only a dictionary of words with their scores for the chosen language.

The simplicity of AFINN sentiment extraction however shows its weaknesses when it comes to more complicated texts or text where words with negative sentiment are used in conjunction with words with positive sentiment, producing unclear results. One such example is given below:

`I don't dislike rainy days, in fact I like them.`

The sentence is not necessarily negative (even positive), but the presence of negative words (such as `dislike` and `rainy`) give the sentence a negative score.

VADER

VADER (Valence Aware Dictionary and sEntiment Reasoner) [17] is a popular lexicon and rule-based sentiment analysis tool that came out in 2014. The tool is first and foremost fine-tuned for sentiments expressed in social media and as such can be easily generalized to other applications such as our own. Unlike AFINN which only focuses on the sentiment word by word, VADER is able to extract the sentiment more accurately. Some of the rules that VADER applies are considering capitalizations and punctuations in sentences, for example giving sentences containing capital letters a higher score. Another improvement over AFINN is it's ability to better account for intensifying (e.g. *extremely*) and negating (e.g. *not*) words. The phrase valence aware also suggests that the method is not only able to detect if the sentence is positive or negative, but it also determines the degree of its positiveness or negativeness.

The original idea was to preform sentiment analysis on Slovene text using a Slovene lexicon. However, upon further research, we stumbled upon article [18], we learned that even in German, a more popular language than Slovene and likely a better lexicon, preformed worse than its translation when it came to detecting sentiments. This is likely due to the fact that, even with proper lexicon for the model in the language of choice other than English, the method is first and foremost designed with English in mind and is most optimal for use as such. In light of this discovery, we preform a similar feat and instead of using a Slovene lexicon, we translate the text to English and preform sentiment analysis in English.

BERT and RoBERTa

BERT (Bidirectional Encoder Representations from Transformers) [19] is a more take on sentiment analysis, surfacing in 2018. BERT utilizes a transformer architecture that captures the intricate relationship between words by considering the context they appear in, meaning that the model is able to capture the sentiment expressed better. This is done by training the model on a large set of data, giving BERT a rich understanding of the language, allowing it to generate high-quality contextualized word embeddings. These can be tuned for specific tasks, enabling a more accurate sentiment extraction. A significant advantage of BERT is also the ability to capture long-range dependencies to text, i.e. capturing sentiment nuances spanning multiple words, enabled by its attention layer mechanism.

We decided to opt for RoBERTa (Robustly Optimized BERT approach) [20], introduced in 2019. It's design to improve upon BERT's pre-training methodology and fix some of its limitations. Some of the key features of RoBERTa include training on a larger corpus and longer training, however it also introduces dynamic masking and removes next sentence prediction (which is present in BERT), giving it a better sentence understanding, an improvement that benefits our task.

While RoBERTa performs well at sentiment analysis, it is also more computationally expensive than the aforementioned approaches. The model also requires a lot of labeled data to train, which might not always be available, especially in less popular languages (such as Slovene). Another limitation of BERT is that due to it's black-box nature, we cannot gain insight to its predictions, limiting its interpretability.

The models that we used for sentiment extractions are

- `cardiffnlp/twitter-roberta-base-sentiment-latest` (trained on Twitter posts) for English books and
- `cjvt/sloberta-sentinews-sentence` (trained on Slovene news articles) pretrained models for Slovene

books respectively.

Co-occurrence extraction

To extract antagonists and protagonists from our books, we have to make sure that said characters actually interact using co-occurrence extraction. Our approach to the co-occurrence extraction is simple: we make an assumption that characters that interact in the story also appear in the same sentences. The more often that these characters appear together, the more stronger the connection is, i.e. the more they interact and the more likely they are to represent protagonists and antagonists, if their sentiments are on the opposite ends. For all of the books, we output the character connections in a JSON file. An example from our example story is presented on Listing 2.

Listing 2. JSON object including character connections.

```
co_occurrences = {
    ('Janez', 'Peter'): 2,
    ('Janez', 'Tine'): 1
}
```

As such, we have extracted protagonists and antagonists from our stories. Since Janez and Tine have positive and negative sentiments respectively and are connected, they are the protagonist and antagonist of the story. Peter and Tine would also make good canditates for antagonist and protagonist, however due to the fact that they don't interact, we conclude that they are not.

The visualization of co-occurrences is preformed by constructing a circle graph with characters as nodes and edges as interactions between characters. The width of the line represents how connected the two characters are, while the color of the nodes represents the character sentiment and it's size the importance (i.e. frequency) of a character. The higher the difference in character's colours and the stronger the edge, the higher the likelihood is that the characters are in a antagonist-protagonist relationship. A visualization of the example is presented on Figure 2.

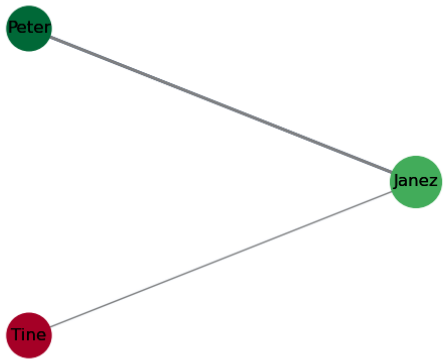

Figure 2. Co-occurrence visualization. Character sentiments (using VADER) and relations using the example above.

Results

To test the performance of our implementations we ran them on English and Slovene short stories corpora and compared the results to ground truths. At first, we determine if the main characters of the stories were extracted from the stories. This is done by looking at the frequencies the characters appeared in the stories: the more frequent the character, the more important it is, i.e. the more likely it is to be a main character. Our initial approach was to determine antagonist(s) and protagonist(s) in all of the stories by hand, however we quickly realised this was too large of a feat for the purposes of this project, as the stories are not that short still. Therefore, we hand-pick some examples and determine why our methods work and where they break.

Using NER and co-reference resolution, we extracted the characters. On the Table 1, we can see the comparison between how many main characters were in the story and how many of them actually appeared as one of the main characters (as in, it appeared as one of the top characters using NER, frequency wise). The Table 1 shows only the characters in English stories. The method mostly performed good. But it has big problems with stories where no names are used. Instead the characters are named ants, boy,... This happened in The Lady, or the Tiger and in The Tell-Tale Heart.

Table 1. Main characters.

Story	Real	Identified
The Ransom of Read Chief	4	4
Hills Like White Elephants	2	1
Leiningen Versus the Ants	2	1
The Lady, or the Tiger?	3	0
The Most Dangerous Game	3	3
The Tell-Tale Heart	2	0
The Gift of the Magi	2	2

Next, we compared how many character sentiments and their connections were correctly classified. This gives us information about protagonists and antagonists of the stories. Running sentiment analysis on simple examples (generated ourselves for testing purposes), the methods determined the character sentiments and their relations correctly. Furthermore, we analyzed the methods on English and Slovene short stories similarly as before, determining if antagonists and protagonists are really such. The sentiment and connection's actual values are present in the project's repository.

Determining the protagonist and antagonist in a story can sometimes be challenging because not all narratives fit neatly into the traditional hero-villain structure. We did analysis of protagonist and antagonist detection, but our ground-truth are based on our perception of the stories. The Table 2 shows the results for English short stories with VADER, which we find out the best on this dataset. The algorithm obviously fails in

Table 2. Protagonists and antagonists detection (Identified/Total).

Story	Protagonist	Antagonist
The Ransom of Read Chief	2/2	2/2
Hills Like White Elephants	1/1	0/1
Leiningen Versus the Ants	1/1	0/1
The Lady, or the Tiger?	0/1	0/1
The Most Dangerous Game	1/1	2/2
The Tell-Tale Heart	0/1	0/0
The Gift of the Magi	2/2	0/1

stories where the main characters were not identified (e.g. The Tell-Tale Heart). It also has problems where the antagonist is not a specific person, but an abstract thing (in The Gift of the Magi the antagonist could be money).

Additionally, the overall sentiment of the story might have an additional effect on whether the character is perceived as positive, as there would be more sentences with positive sentiment. We tried to mitigate that with moving the mean of all the character's sentiments to 0, however this could have also produced bad results if, for example, a story is neutral, yet its characters are mostly positive (reducing the mean means that some positive characters would end up being negative). This is the reason we try moving threshold by hand, observing if the results (as it would then be possible to employ a method to find that threshold).

An example of visualizations for one of the works using different sentiment analysis tools performed on English short stories is presented in Figures 3, 4 and 5. A similar example for Slovene short stories is presented in Figures 6, 7, and 8.

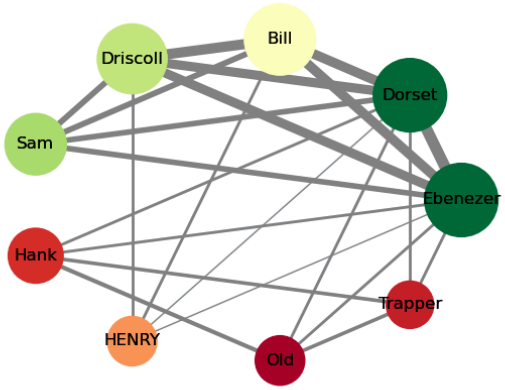

Figure 3. Protagonist(s) and antagonist(s) extraction from "The Ransom of Red Chief" using AFINN.

Since we didn't modify the English stories, we can see that NER also picks up the authors' names. This is not necessarily false, as the author could be referring to himself (e.g. with "I") in the story, however, this could also result in some false positives. The Slovene short stories don't have that problem,

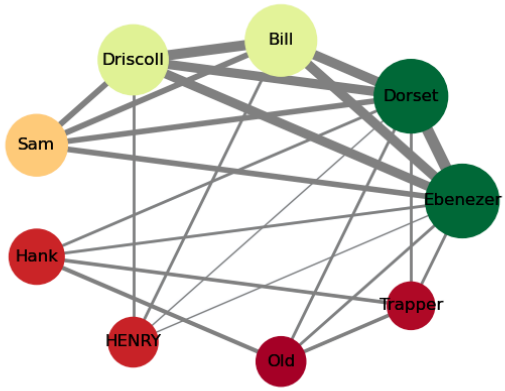

Figure 4. Protagonist(s) and antagonist(s) extraction from "The Ransom of Red Chief" using VADER.

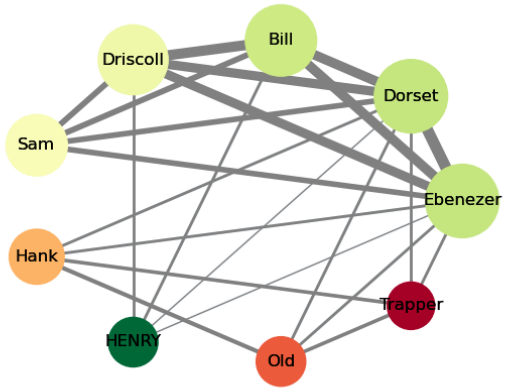

Figure 5. Protagonist(s) and antagonist(s) extraction from "The Ransom of Red Chief" using RoBERTa.

as they only contain the stories' texts.

As a proof of concept, we further ran our methods on Slovenian novels, the results of which are also available in the repository. However, due to a harder evaluation of our methods on them (as well as the fact that they are more intricate and contain more difficult phraseologies), we did not evaluate them[1]. An example of one of the Slovene novels is presented in Figure 9.

Discussion

In both named entity recognition and co-reference extraction we had to use separate models and methods for Slovenian and English. One of the issues with named entity recognition, which was especially prominent in Slovenian short stories, are appellatives like mother - *mati*, king - *kralj* etc. and animal characters like wolf - *volk*. We tackled this issue

[1]For the sake of speed, we also did not lemmatize the text, as it took longer since the works themselves are also longer.

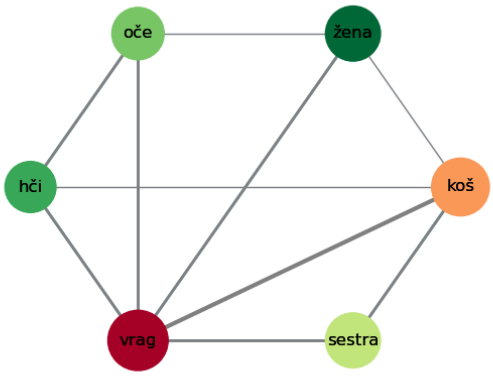

Figure 6. Protagonist(s) and antagonist(s) extraction from ”Vrag se ženi” using AFINN.

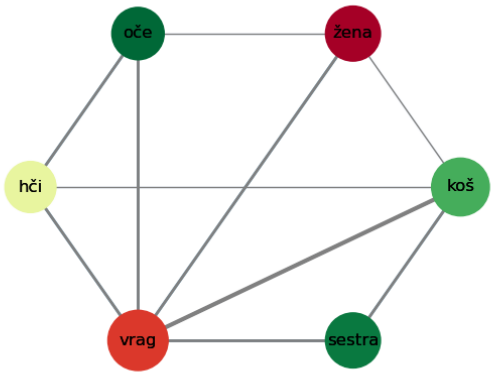

Figure 8. Protagonist(s) and antagonist(s) extraction from ”Vrag se ženi” using RoBERTa.

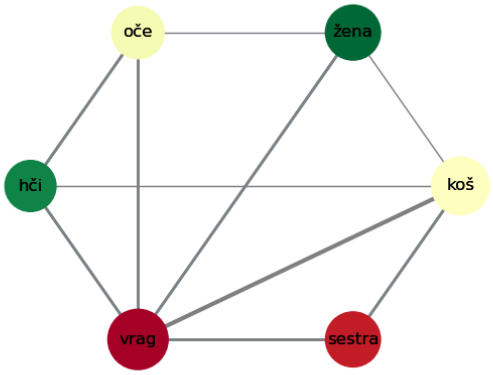

Figure 7. Protagonist(s) and antagonist(s) extraction from ”Vrag se ženi” using VADER.

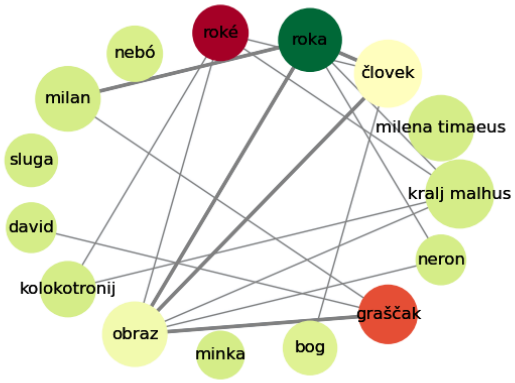

Figure 9. Protagonist(s) and antagonist(s) extraction from ”Kralj Malhus” using RoBERTa.

by including often occurring nouns and using a handmade character noun whitelist, while also filtering irrelevant nouns in the sentiment extraction process. Slovenian stories also presented a problem in co-reference extraction where we used a transformer based SloCOREF [14] model, which caused us three main issues. Firstly, its high computational complexity did not allow us to process entire stories on the available hardware, therefore we had to use a windowed approach with further resolving logic to process long texts. The complexity of the Slovenian language also caused the results of the model to be less than optimal with co-references being detected even between unconnected nouns. Lastly, the lack of available documentation made development using the model difficult.

Regarding sentiment extraction, we currently employed a simple procedure to extract character sentiments. However we can present two problems that come with that approach. First and foremost, a main character can be present in a lot of sentences in the story, taking over the story's overall sentiment, which is not always desired. Another problem is the possibility of a sentiment spanning multiple sentences. Additionally, a sentence's sentiment might not reflect character's actual sentiment, especially when both a protagonist and antagonist are present in it (one could try removing such sentences for the sake of sentiment analysis). A similar downside is therefore also present in co-occurrence extraction (not taking into account relations that span multiple sentences, as well as characters that are not actually related, but just appear in the same sentence).

Another flaw in our sentiment analysis approach is that with more frequent appearances of a subject, there was a higher likelihood that they would be included in sentences with different sentiments, therefore potentially moving the sentiment closer to 0, making him appear as less of a protagonist or antagonist than he actually was. We have already tried to take this into account by raising the sentiment depending on character frequencies, however we were not satisfied with

the results.

Conclusion

We've successfully developed methods that are able to extract characters from stories . Of course, since our approaches are meant to work both in Slovene and English, for comparison's sake, we opted to use simpler methods, in order to be able to compare the performance and test implementation difficulties. In the end, we acknowledge that there is a bunch of room for improvement, as already discussed in the previous chapter, for example, expanding the list of appellative and animal characters, and training custom models for named entity and sentiment extraction.

References

[1] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition, 2016.

[2] V. Devisree and Reghu P.C. A hybrid approach to relationship extraction from stories. *Procedia Technology*, 24:1499–1506, 12 2016.

[3] Saif M Mohammad and Peter D Turney. Nrc emotion lexicon. *National Research Council, Canada*, 2:234, 2013.

[4] Doaa Mohey El-Din Mohamed Hussein. A survey on sentiment analysis challenges. *Journal of King Saud University-Engineering Sciences*, 30(4):330–338, 2018.

[5] Filip Ilievski, Pedro A. Szekely, and Bin Zhang. CSKG: the commonsense knowledge graph. *CoRR*, abs/2012.11490, 2020.

[6] Shingo Nahatame. Revisiting second language readers' memory for narrative texts: the role of causal and semantic text relations. *Reading Psychology*, 41(8):753–777, 2020.

[7] Tirthankar Dasgupta, Rupsa Saha, Lipika Dey, and Abir Naskar. Automatic extraction of causal relations from text using linguistically informed deep neural networks. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 306–316, 2018.

[8] Sendong Zhao, Ting Liu, Sicheng Zhao, Yiheng Chen, and Jian-Yun Nie. Event causality extraction based on connectives analysis. *Neurocomputing*, 173:1943–1950, 2016.

[9] Tommaso Caselli and Piek Vossen. The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop*, pages 77–86, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[10] Kerstin Denecke and Yihan Deng. Sentiment analysis in medical settings: New opportunities and challenges. *Artificial intelligence in medicine*, 64(1):17–27, 2015.

[11] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020.

[12] Nikola Ljubešić and Kaja Dobrovoljc. What does neural bring? analysing improvements in morphosyntactic annotation and lemmatisation of Slovenian, Croatian and Serbian. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, pages 29–34, Florence, Italy, August 2019. Association for Computational Linguistics.

[13] Shon Otmazgin, Arie Cattan, and Yoav Goldberg. F-coref: Fast, accurate and easy to use coreference resolution. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 48–56, Taipei, Taiwan, November 2022. Association for Computational Linguistics.

[14] Matej Klemen and Slavko Žitnik. Neural coreference resolution for slovene language. *Computer Science and Information Systems*, 19(2):495–521, 2022.

[15] Finn Årup Nielsen. A new anew: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*, 2011.

[16] Jože Bučar. Slovene sentiment lexicon JOB 1.0, 2017. Slovenian language resource repository CLARIN.SI.

[17] 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.

[18] Karsten Tymann, Louis Steinkamp, Oxana Zhurakovskaya, and Carsten Gips. Native sentiment analysis tools vs. translation services-comparing gervader and vader. In *LWDA*, pages 100–104, 2020.

[19] 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.

[20] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.