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

Character analysis in Slovene short stories

Gal Petkovšek, Marija Marolt, Jana Štremfelj

Abstract

The abstract goes here.

Keywords

Natural Language Processing, character analysis, knowledge base, character relations

Advisors: Slavko Žitnik

Introduction

Natural Language Processing (NLP) is an evolving field trying to automatically extract the knowledge from the abundant volume of natural language text. Machines can process significantly larger amounts of text, but there are some challenges such as ambiguity of the text, the sentiment analysis, commonsense reasoning, information extraction etc.

The main task of our project is the knowledge base creation, which is a sub-field of Natural Language Processing. This includes extraction of relations between the characters are created. Each character is also classified as positive, negative or neutral and the protagonist of the story is defined.

The pipeline and models are adjusted to Slovene language. We get a graph from the information extraction pipeline in which the nodes represent the characters and links the relations between them. First step in the pipeline is name entity recognition. In more general case this is refereed to recognition of all entities is made (e.g. person, location, organization...) however we only focus on characters. In the next step coreference resolution is applied where all expressions that refer to the same entity are found. Next additional information is obtained from the text about the characters or the relations between them. In the last step the knowledge graph is constructed.

The visualization is going to represent the relations of the characters and their profiles, which we obtain from the knowledge graph. It will provide a visual depiction of the story structure and character interactions, which help readers better understand and analyze the narrative.

Related work

Character analysis and relationship extraction is already a well researched topic that falls under the category of knowledge base extraction from text. Different authors report different approaches to tackling the problem of knowledge base creation, while most of them still roughly separate the pipeline steps to the ones meant to extract nodes ans the ones to extract edges. In an article by Kertkeidkachorn *et. al.* [1] they describe a pipeline for knowledge graph construction using with five steps: entity recognition (and mapping), coreference resolution, triple extraction, triple integration and predicate mapping. In another article [2] the authors describe the an algorithm called Grapher that uses an encoder-decoder language model (sequence to sequence algorithm) for node extraction and for edge detection they tested two different approaches (LSTM/GRU and classification).

When we consider the character analysis problem (which is a sub-problem of the above described knowledge base creation problem). Different research is also being done in that area. The article [3] by Nalisnick *et. al.* for example predicts the relationships from characters on Shakespeare's plays, however their problem is different from ours in the way that since we have plays the problem is mainly focused on the dialog between characters. Another example of character analysis is described in the article [4] where the author's main focus is to classify the character as positive or negative which is done in 3 different ways: character's speech, character's actions and predicatives.

In Slovene however, the topic is not as well covered. Markovič [5] *et. al.* in his article for example uses network analysis for analysing Slovene fables, however it does not strictly cover sentiment analysis of characters. In another article [6] the authors describe how they prepared a Slovene database for the problem of classifying texts as positive, negative or neutral, however they do not focus on the characters in those texts. Therefore to the best of our knowledge not many authors have attempted to solve the character analysis problem purely in slovene language.

Methods

Data preparation

The dataset that we have chosen, is a collection of Slovene short stories. Our training corpus contains 74 fairy tales. They are obtained from dLib.si, which is a digital library and document server for Slovenian language resources. Our objective was to identify stories with similar themes and characteristics that would be interesting for analysis. These stories were specifically chosen due to their length, which allows for efficient processing and label with the available resources.

The stories were scraped from the website using a Python web scraper, which utilized the BeautifulSoup and requests libraries. The obtained text was then processed by replacing multiple spaces with single spaces and removing new line breaks.

To train the model, the data was labeled with the assistance of ChatGPT, which aided in identifying the characters from the fairy tales. The results of the tool were then reviewed and manually corrected, as it typically only provided a list of the main characters. Characters who appeared only a few times needed to be added manually.

Name entity recognition

Name entity recognition is in general the task of extracting different entities from the text (locations, people, organization...) which is usually done with specialized NER language models. Our goal was a bit more domain specific as we wanted to extract the names of characters of fairy tales. The problem that arose here is that NER models are usually trained to recognize names of people as person entity, which in fairy tales is rarely the case as most characters are not introduced by names (eg. king, queen, fox *etc.*).

We first tackle the problem by capitalizing the first letter of each character name which made the model perform very well, but since this usually meant that a human needs to manly perform this task we abandon this idea as it defeats the purpose of automatised entity extraction.

In our second approach we tackle the problem in two ways and then combine the results. First we focus on the characters that are refered to with names. For recognizing those we use classla NER model. For recognizing all the other characters (which are the majority) we use chatGPT to generate a list of most common character names in Slovene fairy tales. This list enables us to recognize characters such as zmaj, oče, čarovnica in addition named characters such as Janko, Matka. We assumed that there is a somewhat small set of characters that appear in the stories so it is possible to capture and list them all.The list was then also manually corrected (some characters were removed and some were added). We then use this list to identify characters. However because of different shapes of words we do not just search over the text but rather over the lemmatized words of the text, which is obtained from the classla model. After some initial testing we also applied Levenshtine distance as to compare the words as the lemma model sometimes fails (however in the large majority of cases it works very well). We also use the Part-of-speech tagging from classla to confirm that the found word is indeed a noun.

Additionally filtering was added as the component predicted some characters incorrectly (false positives). We therefore also filter out the characters that do not have many occurrences as the stories usually do not have characters that would just appear once.

The result of this module is a dictionary of all occurrences of all characters in the text, an example of which can be seen below

```
{
    "zmaj": [[54, 59],
             [147, 151],
             [188, 193],
             [558, 562]],
    "pastir": [[339, 345], [107, 116]],
    "ovca": [[474, 478]],
    "zver": [[259, 263],
             [121, 125],
             [314, 318]]
}
```

Coreference resolution

For the coreference model we chose the SloCOREF model (Coreference Resolution for Slovene language). It returns pairs, with the trustworthy value of the prediction of the model. Each element of the pair includes the starting and ending position in the text. From the Clarin site [7] we downloaded the model and we used some of the code from the GitHub site [8].

We then use the entity dictionary from the name entity recognition and the coreferenced pairs (nouns and pronouns) from the coreference model. If one element from the coreferenced pair is recognized as the person entity, the other start and end position is added to the list of the entity occurrences. If none of them is a person entity, but one is recognized as entity, we add the other occurrences to the list. If both are entities we add the second start/end position to the first list of occurrences.

Because the model cannot take the whole story at once, we spilt the story to parts and search for the correference pair on these parts with certain offset. For example the length of subtexts is 1000 characters and the offset is 500, thet means for the first part we take first 100 characters and for the second part from 500 to 1500 etc.

The output of the function is the modified dictionary of the entities and its starting/ending positions, just as in the previous step.

Relationship extraction and character analysis

In our future work we will also implement relationship extraction and character analysis. We will use the output of the NER and coreference resolution components to extract the embedding of the characters from the text, which will then be

used as an input for models that will determine the relationship between characters and come characteristics of a specific character.

Visualization

The visualization was created using the p5.js library in JavaScript. It takes two inputs in the form of CSV files, one containing information on the relationships between characters and the other containing information on the sentiment and probability of each character being a protagonist. The output is a graph that visualizes the relationships between characters, with vertices representing individual characters and edges connecting them. The thickness of the edges is proportional to the frequency of interaction between characters. The position of the vertices is determined by the sentiment of the characters and their likelihood of being a protagonist, with characters who are more likely to be protagonists being positioned closer to the center. The size of the vertices corresponds to the frequency of appearance of the characters in the story.

Knowledge base from slovene fairytales

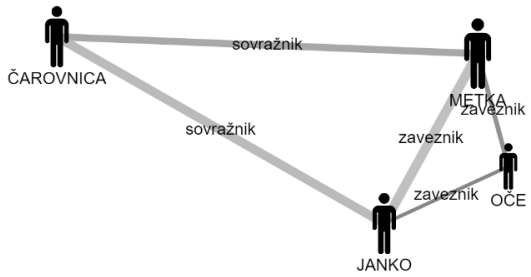

Figure 1. Visualization of characters and their relationships in the story.

On Figure 1 we see a graph that provides a clear and intuitive representation of the characters and their relationships in the story, which can aid in literary analysis and storytelling.

Results

We evalueated the outputs of the name entity recognition component and from coreference resolution. We focused mainly on which characters were identified (found at leas once in text) and which characters were left out. Therefore we define two metrics: $precision = \frac{TP}{TP+FP} = \frac{charactersthatwerecorrectlyidentified}{charactersthatwereidentified}$ and $recall = \frac{TP}{TP+FN} = \frac{charactersthatwerecorrectlyidentified}{charactersthatshouldbeidentified}$. Precision tells us how many characters that were outputted are actually correct characters from the stories (our visualization should not contain excessive characters) and recall tells us how many of the correct characters were identified (out visualization should not lack characters). We chose those matrices as it is easier to label the data in such way as oppose to finding every occurrence of every character. Those two metrices are also the most importaint as they clearly show how many FP and FN we have which is the most vital for our visualization. Results can be seen in Table 1. We report both the mean and the variance of precision and recall scores over all stories.

	average precision	precision standard dev.	average recall	recall standard dev.
only NER model	0.5451	0.1303	0.3595	0.0866
only list of characters	0.7083	0.0563	0.8038	0.0904
NER model with list of characters	0.8065	0.0453	0.9597	0.0232
NER model with list of characters filtered	0.8789	0.0437	0.6427	0.0874
NER model with updated list of characters	0.6767	0.031	**0.9939**	0.0013
NER model with updated list of characters filtered	0.8062	0.0454	0.6441	0.0932
crr precision	**0.8039**	0.0470	0.9067	0.0299

Table 1. Evaluation of NER and coreference components.

We can see that if we only used the classal NER model the recall would be very low, which is to be expected as most characters do not have names. The variance of precision and recall is also higher than with other approaches which means that the model is less stable when it comes to predicting different stories. What is surprising is that precision is also quite low for the NER model, which means that the model recognizes too many words as characters. By using only a list of characters generated by ChatGPT we obtain a quite good results both in terms of precision and recall. When the approaches are combined we obtain a quite good results with approximately 0.8 precision and 0.95 recall. As we examined the data we observed that many words that were wrongly detected as characters appeared only a few times (once or twice). Therefore we implemented filtering which excludes the characters that do not appear in more than 10% of appearances over all characters. This did bring the precision to 0.87 but also lowered the recall to 0.64 as some side characters actualy do appear very few times. We also tried to increase the recall by updating the list of allowed characters, however this also drasticaly decreased the precision as some characters (that are importaint in some tales eg. ovca, konj...) are not even side characters in other stories but they do appear as words.

We applied coreference resolution on the NER model with list of characters which slightly decreased the recall and precision. This is not problematic nor surprising as the main goal of coreference resolution is to find other indirect mentions of already detected entities. Since we can observe a change in the recall this must mean that the component joined some of the characters (since the model predicted the connection between them) that were actually not supposed to be joined. In the future we are going try to improve that behaviour.

From our manual analysis we concluded that the connections between the the correferenced elements are usually meaningless and do not reference the same entity at all. For example in "Vrnil sem se, dokler toliko ne zrasem, da bom goden za gosjega pastirja" it added "sem" to the entitiy of pastir. Some of them are also correct like in "je nekaj zatulilo" it found entity volk in the "je".

Discussion

Acknowledgments

References

[1] Natthawut Kertkeidkachorn and Ryutaro Ichise. T2kg: An end-to-end system for creating knowledge graph from unstructured text. In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[2] Igor Melnyk, Pierre Dognin, and Payel Das. Knowledge graph generation from text. *arXiv preprint arXiv:2211.10511*, 2022.

[3] Eric T Nalisnick and Henry S Baird. Character-to-character sentiment analysis in shakespeare's plays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 479–483, 2013.

[4] Lucie Flekova and Iryna Gurevych. Personality profiling of fictional characters using sense-level links between lexical resources. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1805–1816, 2015.

[5] Rene Markovič, Marko Gosak, Matjaž Perc, Marko Marhl, and Vladimir Grubelnik. Applying network theory to fables: complexity in Slovene belles-lettres for different age groups. *Journal of Complex Networks*, 7(1):114–127, 08 2018.

[6] Jože Bučar, Martin Žnidaršič, and Janez Povh. Annotated news corpora and a lexicon for sentiment analysis in slovene. *Language Resources and Evaluation*, 52:895–919, 2018.

[7]

[8]