FRI Natural language processing course 2023
Project report

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

Literacy situation models knowledge base creation

Jan Bajt, Anže Habjan, and Tadej Stanonik

Abstract

The abstract goes here. The abstract goes here.

Keywords
keyword extraction, natural language processing

Advisors: Slavko Žitnik

Introduction

Named entity recognition (NER) is the task of identifying and classifying named entities in text, such as people, organizations, locations, and dates. NER is an important component of many natural language processing applications, such as information extraction, machine translation, and question answering. However, NER is a challenging task for machines because named entities can appear in many different forms and contexts, and there can be ambiguity in the interpretation of entity mentions.

Relationship extraction is the task of identifying and extracting relationships between entities in text. These relationships can take many different forms, such as family relationships, social relationships, organizational relationships, and spatial relationships. Relationship extraction is important for many applications, such as social network analysis, event extraction, and knowledge graph construction. However, relationship extraction is a difficult task for machines because it requires understanding the complex semantics and context of natural language, as well as dealing with the variability and ambiguity in the expression of relationships.

Aforementioned methods are both important tasks in natural language processing because they enable machines to automatically extract structured information from unstructured text data. However, achieving high accuracy in these tasks requires sophisticated machine learning models and feature engineering techniques. Additionally, the variability and ambiguity of natural language make it difficult to achieve high accuracy, particularly for rare or ambiguous entities and relationships.

The literary work commonly recognized as a television show, *Game of Thrones*, originates from a collection of five books authored by George R. R. Martin. Each book features a copious number of characters who are all affiliated with a particular noble house (although this affiliation is not always precise). The overarching objective of each house is to acquire and sustain power and influence within the Seven Kingdoms. The narrative centers around the conflicts between the houses, each with its unique set of characters and interrelationships. Thus, this compilation of books offers an ideal resource for evaluating various techniques of named entity recognition and the extraction of family relationships from the identified entities.

Related work

In article [1] the authors tested and explained different NER libraries including Python's SpaCy, Apache OpenNLP, and TensorFlow. The comparison of these libraries is done based on training accuracy, F-score, prediction time, model size, and ease of training. They discovered that SpaCy gives better and accurate results as compared to Apache OpenNLP and TensorFlow when it comes to identifying entities in the input text. The prediction time of the Spacy model is also better, which indicates that Spacy is the best NER algorithm that was tested in this article.

In article [2] the authors compared the performances of different NER algorithms on old and modern novels. They discovered that although many studies on information extraction from literature typically focus on 19th and early 20th century source material there are no significant differences between old and modern novels but both are subject to a large amount of variance. They also found out that novels written in 3rd person perspective perform significantly better than those written in 1st person. From found entities they created social networks and performed network analysis by observing multiple network features.

For the characters that we find, we also want to extract relationships between them. In article [3] the authors presented their system CustRE, which is used for better identification and classification of family relations from English text. CustRE is a rule based system, that uses regular expressions for pattern matching to extract family relations explicitly mentioned in text, and uses co-reference and propagation rules to extract family relations implicitly implied in the text.

Methods

In this section we represent our pipeline and methods used for extracting the family relations between characters in Game of Thrones books series. The main steps of our pipeline are:

1. replacing aliases of characters with their name,

2. coreference resolution,

3. family relation extraction.

0.1 Coreference resolution

Coreference resolution is the task of finding all expressions that refer to the same entity in a text. It is an important step for a lot of higher level NLP tasks that involve natural language understanding such as document summarization, question answering, and information extraction [4]. This process can be challenging due to the complexity of language and the many ways that entities can be referred to in text. For example, an entity may be referred to using different names, pronouns, or descriptions throughout a text. Additionally, some expressions may be ambiguous and could refer to multiple entities.

Because characters can be represented with multiple different aliases, we decided to first replace these aliases with an actual name of the character. That way, each character is always referred to with only one name, which makes coreference resolution algorithms perform better. Removing aliases also helps NER algorithms find characters, because otherwise the algorithm would find a new entity for each alias of the same character.

The next step was to replace pronouns (such as 'he' or 'she') with the names of entities that these pronouns represent.

0.1.1 NeuralCoref

For Coreference Resolutin we used NeuralCoref [5], a pipeline extension for spaCy 2.1+ which annotates and resolves coreference clusters using a neural network.

0.2 Named Entity Recognition (NER)

Named entity resolution is a key task in natural language processing that involves identifying and classifying named entities in a text. Named entities are real-world objects that have specific names, such as people, organizations and locations. In our paper, we focused on finding characters in the book. Named entity resolution can be challenging due to the variety of named entities and the many ways that they can be referred to in text. For example, a person's name may be misspelled, abbreviated, or referred to using a nickname or title.

0.2.1 Flair

Flair [6] is a powerful NLP library, which allows you to apply our state-of-the-art natural language processing (NLP) models to your text, such as named entity recognition (NER), sentiment analysis, part-of-speech tagging (PoS), special support for biomedical data, sense disambiguation and classification, with support for a rapidly growing number of languages. The framework builds directly on PyTorch [7], making it easy to train your own models and experiment with new approaches using Flair embeddings and classes.

0.2.2 NLTK

The Natural Language Toolkit (NLTK) [8] is a Python library designed for natural language processing. It provides a wide range of tools and resources for tasks such as tokenization, stemming, and part-of-speech tagging. NLTK also offers pre-trained models and algorithms for sentiment analysis, topic modeling, and text classification. It supports multiple languages and integrates with other Python libraries such as NumPy and SciPy.

0.2.3 Spacy

Spacy [9] is an open-source Python library used for natural language processing. It offers tools for tokenization, part-of-speech tagging, named entity recognition, and dependency parsing. Spacy has pre-trained models for multiple languages and also allows for custom model training. It is fast and memory efficient, making it ideal for large-scale natural language processing tasks. Spacy is widely used in academia, industry, and government for various natural language processing applications.

0.2.4 Stanza

Stanza [10] is a Python natural language analysis package. It contains tools, which can be used in a pipeline, to convert a string containing human language text into lists of sentences and words, to generate base forms of those words, their parts of speech and morphological features, to give a syntactic structure dependency parse, and to recognize named entities. The toolkit is designed to be parallel among more than 70 languages. It is built with highly accurate neural network components that also enable efficient training and evaluation with your own annotated data. The modules are built on top of the PyTorch library.

0.3 Family relationship extraction

For the characters that we found with NER, we wanted to retrieve the family relationships between them. There are several NLP techniques used to extract family relationships from text, including rule-based methods and machine learning algorithms.

Rule-based methods, like [3], involve the creation of hand-crafted rules that specify patterns or keywords that indicate family relationships. These rules can be applied to text data to identify relationships between individuals. However, these methods can be limited by the complexity and variability of natural language, making it difficult to capture all possible variations of family relationships.

Machine learning algorithms, on the other hand, use statistical models to learn patterns in the data and automatically identify family relationships. These algorithms are trained on large datasets of annotated text, where each relationship is labeled with a specific category (e.g., parent-child, sibling, grandparent-grandchild). The model then uses these labeled examples to identify patterns in the data and predict relationships in unseen text.

0.3.1 Stanford CoreNLP

Stanford CoreNLP [11] enables users to derive linguistic annotations for text, including token and sentence boundaries, parts of speech, named entities, numeric and time values, dependency and constituency parses, coreference, sentiment, quote attributions, and relations.

Results

Family relationship extraction was evaluated in terms of standard evaluation metrics, *i. e.* precision, recall, and F1 score. It should be noted that in the current results, there are a large number of false negatives, resulting in low recall and consequently, a low F1 score. Many characters from the ground truth have multiple fathers or mothers, including biological, alleged, or rumored, and it is also possible to have multiple spouses. For the results presented in table 1, only the spouse and parental relationship evaluation results were taken into account when computing the evaluation. In other words, not all possible relations discovered by our model were evaluated because the ground truth is missing some of the relationships that would otherwise be predicted by the Stanford CoreNLP model. Visualization of extracted relationships for 20 characters with the highest number of detected relationships can be seen in Figure 1.

Model	Precision	Recall	F1 score
Stanford CoreNLP	0.14	0.004	0.008

Table 1. Evaluation results of family relationship extraction using different models.

Discussion

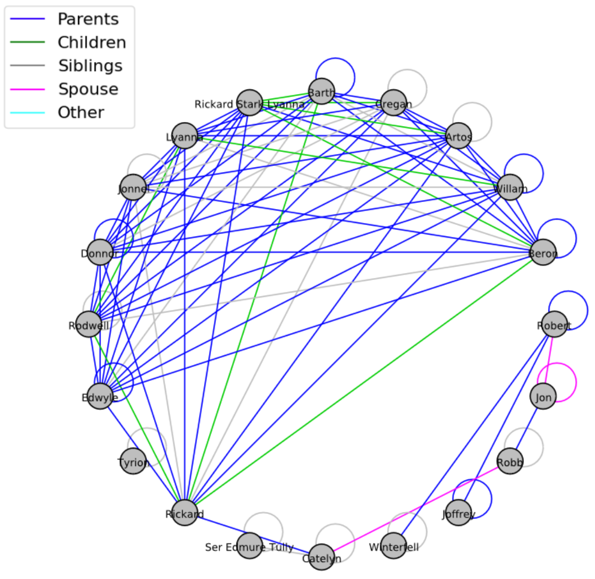

Figure 1. Visualization of relationships between characters with CoreNLP.

References

[1] Hemlata Shelar, Gagandeep Kaur, Neha Heda, and Poorva Agrawal. Named entity recognition approaches and their comparison for custom ner model. *Science & Technology Libraries*, 39(3):324–337, 2020.

[2] Niels Dekker, Tobias Kuhn, and Marieke van Erp. Evaluating named entity recognition tools for extracting social networks from novels. *PeerJ Computer Science*, 5:e189, April 2019.

[3] Raabia Mumtaz and Muhammad Abdul Qadir. CustRE: a rule based system for family relations extraction from english text. *Knowledge and Information Systems*, 64(7):1817–1844, June 2022.

[4] The stanford nlp group. https://nlp.stanford.edu/projects/coref.shtml.

[5] neuralcoref. https://github.com/huggingface/neuralcoref.

[6] Flair. https://github.com/flairNLP/flair.

[7] Pytorch. https://pytorch.org/.

[8] Nltk. https://www.nltk.org/.

[9] Spacy. https://spacy.io/.

[10] Stanza. https://stanfordnlp.github.io/stanza/.

[11] Stanford corenlp. https://stanfordnlp.github.io/CoreNLP/.