Final Project Unstructured Data Analysis

Imperial College London

Department of Natural Science

Harry Potter and Friends(?)

Knowledge Graphs and Social Networks

Author: Marie-Luise Eppinger
CID: 02363795

Date: January 6, 2024

Contents

Statement of independent work
I confirm that this report and the attached code is the result of my own, independent work. Any ideas or quotations from the work of others, whether published or otherwise, are used in accordance with the standard referencing procedures. The code for the following report is fully given in github. It must be further noted at this stage that `Python 3.10.13` is used for this notebook. Further it must be noted that for comparing results with already published projects related to this topic, the results are compared by using the free available code published through this projects. If this is the case it is clearly stated inside the code. The usage of the code of this report is explained shortly in the Appendix and in the `README.md` file of the attached github repository: `https://github.com/ml99imperialep/mlds-uda-final-project`. In this public repository all code is freely available and can be downloaded as a `.zip` file.

1 Introduction

Books, especially book series, offer a wide range of possibilities for analysing texts simply because of their complexity. For example, different themes, relationships, characters/protagonists can be analysed and evaluated. In this thesis, the very well-known seven novels ***Harry Potter by J.K. Rowling*** are the subject of the analysis. Due to the large number of characters, they are suitable for analysing the relationships of different people and their representation in a **social network**, but also in the first instance for creating a **knowledge graph** to represent the role of the characters and a summary of their actions. The books are classified as children's and young adult literature, i.e. they are written in simple language and therefore, despite their length and complexity, can be analysed using pre-implemented text analysis techniques such as those found in the library `spaCy`.

This thesis is divided into three main chapters and a summary with a view on possible future work:

In the **first chapter** of the thesis, the dataset is preprocessed and cleaned and an initial **exploratory data analysis** is carried out together with **bigrams graph visualization**. In the **second part**, a **knowledge graph** for the essential main characters is created and analysed with the help of pre-implemented pipelines in `spaCy`.

In the third and final part of the thesis, firstly a character network based on co-occurrence is formed based on another general previously published work (see) and secondly a social network between the characters is created based on **sentiment analysis**. Both created graphs are then analysed and compared through **community detection** and **graph embedding** methods.

The images used in this report have been created using the notebooks from the Github repository (see `plt.savefig('filepath')` in the jupyter notebooks), however due to the volume of images, not all images produced using the code can be found in this report. Some screenshots of sections of networks and graphs have also been included. For this reason, there is also a folder in the repository with all the images used in this report called `report_media`.

1.1 Dataset: Harry Potter novels

The ***Harry Potter*** [1]series written by J.K. Rowling and published in the years 1997 - 2007 (see wikiharry [wik]) follows the life of a young wizard, Harry Potter, as he navigates through the wizarding world. Here's a summary of each book produced from my own knowledge about reading the books in the last years and (wikiharry [wik]) :

1. **HP 1: Harry Potter and the Sorcerer's Stone** (or **Philosopher's Stone** in some regions) : Harry discovers he's a wizard on his 11th birthday and attends Hogwarts School of Witchcraft and Wizardry. Further he learns about his past, defeats Voldemort for the first time, and makes friends like `Hermione Granger` and `Ron Weasley`.
2. **HP 2: Harry Potter and the Chamber of Secrets** : In his second year at Hogwarts, strange events occur, including students being petrified. Harry discovers a hidden chamber within the school and encounters a memory-bound Voldemort's younger self. He defeats a basilisk and saves Ginny Weasley.
3. **HP 3: Harry Potter and the Prisoner of Azkaban**: Harry learns more about his parents' past and the betrayal of his parents' friend, Sirius Black, one of the main characters of this book. He also encounters Dementors, mysterious creatures guarding the wizarding prison, which is called Azkaban. The truth about Sirius is revealed, exonerating him.
4. **HP 4: Harry Potter and the Goblet of Fire**: Harry is unexpectedly entered into the dangerous Triwizard Tournament, facing various challenges. Voldemort returns to power and, using Harry's participation, rises to his full form. Tragic events lead to the return of Voldemort.
5. **HP 5: Harry Potter and the Order of the Phoenix**: Harry returns to Hogwarts and faces disbelief and resistance from the wizarding community about Voldemort's return. He forms Dumbledore's Army to prepare students for defense against dark forces and faces tragedy when his godfather, Sirius Black, dies.
6. **HP 6: Harry Potter and the Half-Blood Prince**: Dumbledore reveals Voldemort's past and the Horcruxes - objects containing parts of Voldemort's soul. Harry learns more about Voldemort's history and discovers crucial information about Horcruxes. Tragedy strikes at the end when Dumbledore is killed by Professor Snape.
7. **HP 7: Harry Potter and the Deathly Hallows** : Harry, Ron, and Hermione leave Hogwarts to search and destroy Horcruxes, the keys to Voldemort's immortality. They face various challenges, losses, and revelations. Finally, Harry sacrifices himself to destroy the final Horcrux and defeats Voldemort, ending the dark wizard's reign.

Throughout the series, themes of friendship, bravery, love, and the struggle between good and evil are prevalent. Harry's journey from an orphaned boy to the vanquisher of Voldemort captivated readers worldwide.

This short summary of the seven Harry potter novels can interpreted as a story of a boy, who fights together with his friends against his enemies (Voldemort's friends). In children's and young adult literature, it is usually the case that

[1]It must be noted that through this report and the attached jupyter notebooks we only refer to HPi for $i \in \{1, \ldots, 7\}$ as a short form of the single books.

there is a clear distinction between good and evil. We will try to analyse in this project if one could derive with text and graph analytical methods **Harry's friends** and of course his enemies as well as general interaction between the characters and possible groups of character, who are strongly related to each other (called clusters).
The dataset containing the seven Harry Potter books (with `HPBooki.txt` for $i \in \{1, \ldots, 7\}$, which is used in this project can be downloaded from a public repository (see dataset1[data]). The chapters of each book are separated via @ and special symbols like "are given with a backslash. The other dataset called `characters.csv` is downloaded from a github repository as well (see dataset2[datb]). It has two columns and 99 rows. The first column called `Name` contains all characters of the Harry Potter books, the second column `Variation` the variation of their names in the book: e.g. *Ron* vs *Ronan*. This dataset will be used in section **4.** and **5.** of this report.

1.2 Literature Review

Since the dataset is a well-known and quite popular one, it is useful to start with a literature review at the beginning of this thesis to summarize existing and already published results, which are strongly related to the main analysis of this project.

- This is a work published in a github repository (see hzjken[hzj]) five years ago, which uses entity recognition and sentiment analysis to build a network graph as well. The author of this work produced two main graphs, one represents the number of interaction between characters in each book and in all books with name entity recognition and applies sentiment analysis on the characters related to the content of the book to build a social network.
- In this work published through a public github repository (see network[net]) a character network for HP1 is created on the basis of co-occurence of characters. We use the same method for all books to establish a comparison with the method presented in this report. Part of the freely available code is used and adapted for this purpose. It must be noted that the author was able to build a character network by counting the interactions between each characters in HP1, but the published project does not give an impression about a clear development of community detection or other underlying graph structure.
-

2 Initial Exploratory Data Analysis

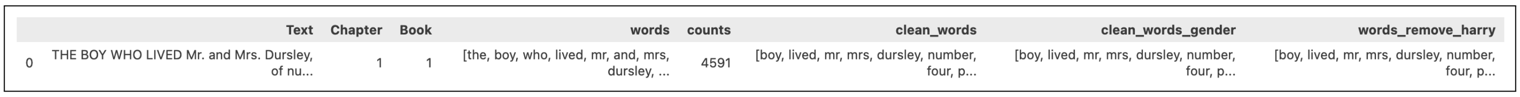
In the following, we will carry out an initial analysis of the text dataset, which relates to our research focus of creating a network of characters from the text. Of course, there are numerous other ways of analysing the text dataset, but these are not relevant to the later results of this project.

2.1 Initial Data Preprocessing

Cleaning the data set is essential for thorough text analyses [2]. The following steps were carried out for this purpose.

- Removal of **puncutation**, e.g. *„;:?!...*
- Removal of specific **stopwords** (general stopwords from `nltk` for generating wordclouds, gender specific for analysing gender words and removing most occurent words *harry, said*

The first row of the cleaned dataframe is shown in the Figure below:



	Text	Chapter	Book	words	counts	clean_words	clean_words_gender	words_remove_harry
0	THE BOY WHO LIVED Mr. and Mrs. Dursley, of nu...	1	1	[the, boy, who, lived, mr, and, mrs, dursley, ...	4591	[boy, lived, mr, mrs, dursley, number, four, p...	[boy, lived, mr, mrs, dursley, number, four, p...	[boy, lived, mr, mrs, dursley, number, four, p...

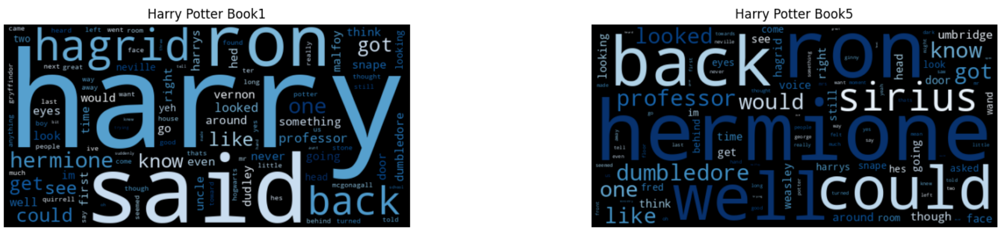
Figure 1: Cleaned dataframe of 200 rows and 8 columns

In column `Text` contains the uncleaned text per `Chapter` and `Book`. The column `words` the tokenized words with `counts` of words. In comparison to that `clean_words` lists the cleaned words after removing stopwords and punctuation, the column `clean_words_gender` represents the cleaned words expect gender specific words like *herself, himself, he, she,...*. The last column excludes additionally the most occurent unigrams *harry, said*.

2.2 Word Counts and Word Clouds

In order to get a first impression of the text content, it is advantageous to carry out wordclouds with regard to the individual books for better comparison and the most frequently occurring words (with the exception of stopwords in the English language of the `nltk` package) and to count the number of **unique** words in each book.

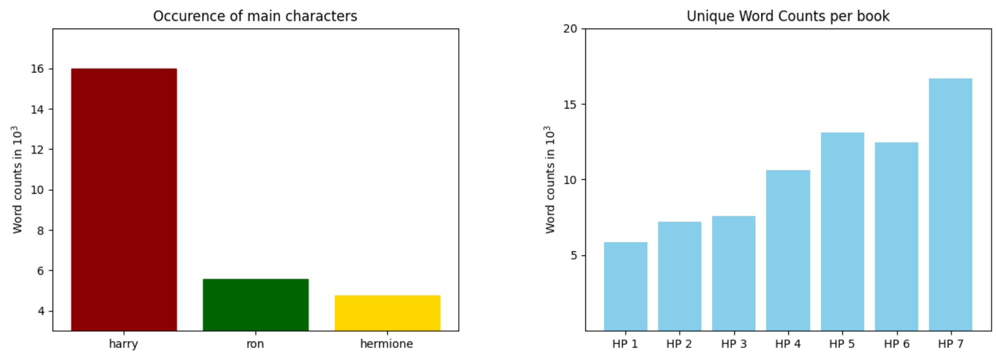
[2]for further text analysis we will extend this approach



a) left: Wordcloud of the HP1 (top 100 occurent words), ⇒ b) right: Wordcloud of the HP5 after removing *harry, said*

Figure 2: Exploratory Data Analysis with wordclouds

In Figure 2a) it is obvious that the words *harry, said* occurs most frequently in HP1. Figure 2 only shows the results of individually selected word clouds, as the similarity of the most frequently occurring words is significant when looking at all individual word clouds and therefore an explicit differentiation of the individual books is not necessary in this context. However, it can be seen that the words *harry, said* occur most frequently in all seven works, which is not surprising when you consider the titles of the works, which are also *Harry Potter xxx* and he is the main character and eponym at the same time. For better interpretation of the occurency of other different words we demonstrate in Figure 2b) the wordcloud of HP5 by using `words_remove_harry`. With this method one can get a initial impression about the main characters of the book series *harry, ron, hermione*, since after *harry* the words *ron, hermione* occurs second most frequently. In the figure below we plot this occurence in the histogram on the left.



a) left: Histogram of occurence of main characters *harry, hermione, ron*; b) right: Histogram of unique (cleaned) word counts per book

Figure 3: Initial Exploratory Data Analysis: Occurence of characters and word counts

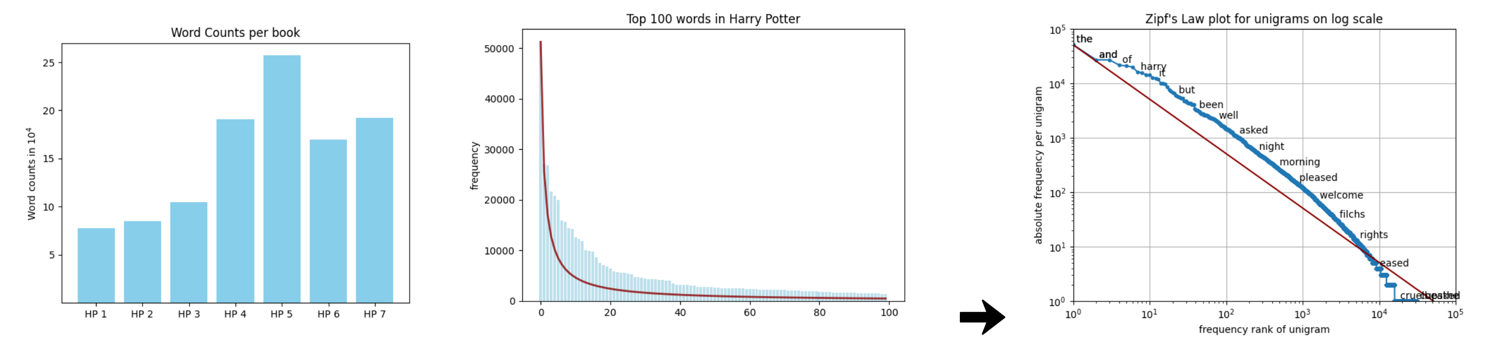
It is significant by considering Figure 3a) that in comparison to the other two main characters the word *harry* occurs more as three times; (e.g. *harry* occurs 15979 in comparison to *ron* 5579 and *hermione* 4759) Further it is obvious from the histogram on the right (Figure 3b) that the number of unique (cleaned) words in each book increases from HP1 to HP7.

2.3 Word frequency analysis: Zipf's law

Zipf's law states that the frequency that a word appears is inversely proportional to its rank. (see zipf[zip]) With this law we want to demonstrate that the language used in the HP series follows this typical distribution according to George Zipf. **Zipf's Law** describes a statistical distribution observed in various natural language phenomena, notably word frequencies. It states that in a large text corpus, the frequency of any word is inversely proportional to its rank in the frequency table (see zipfg[Zip49] and newman[New05]). The ith most frequent word with frequency $f(i)$ can be therefore scaled according to

$$f(i) \propto \frac{1}{i^s} \quad \text{with } s \approx 1$$

This means that the most frequent word will occur twice as often as the second most and so on. Therefore we first calculate the rank of each word in terms of its frequency. Afterwards one can calculate the term frequency and sort the values in a descending order. It must be noted that for an overall result at this stage we have to use all `words`, not only the `clean_words` with removed stop words. Therefore in Figure 4a) the number unique `words` for each book is shown in the histogram. From this result one can say that HP5 is the longest book. It has the highest word count with 8928 and the longest chapter which is chapter 38. Then HP4 is following and thirdly HP 7. It seems by comparison this observation with Figure 3b), the counts of `clean_words`, HP5 and HP4 seems to have more stopwords and punctuation in comparison to HP7.



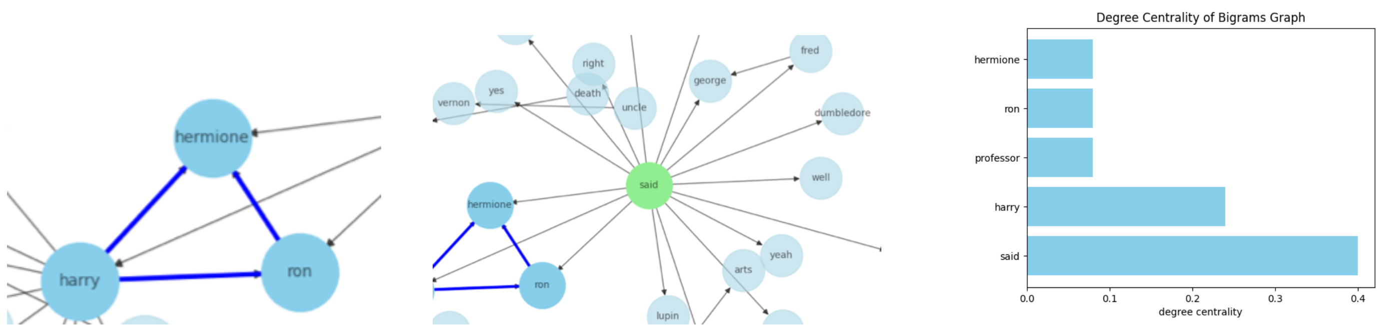
a) left: General word counts without cleaning, b) middle; Histogram of frequency of top 100 words in comparison to the Zipf's expectation, c) right: Linear plot on log scale for Zip"s law vs word frequencies in HP

Figure 4: (Un)cleaned words analysis for semantic analysis

Zipf's law is often visualized by plotting rank on the x-axis and term frequency on the y-axis, on logarithmic scales. Plotting this way, an inversely proportional relationship will have a constant, negative slope. A similar approach to this is given in ([tustla][tus] and adjusted for our dataset).
In Figure 4b) the x-axis is the rank of the frequency from highest rank from left up to 100th rank to the right and on the y-axis the frequency of the top 100 words counted through the text corpus. The red line plotted the expected relation by Zipf's law. It seems that this expectation is almost fulfilled, although the histogram has more area above the curve in higher ranked words from rank 10 to rank 20 f.e.. In Figure 4c) the Zipf's relation for unigrams is plotted on a log-log scale (see [tustla][tus]). If the blue curve of the words is near the red linear curve, then the law is satisfied. One can recognize that the given text corpus of the Harry Potter books does not strictly follow the red line, but overally it does. There is a deviation from the linear curve for higher ranked words and for lower rank words the blue curve does not strictly align on the red line. A reasonable explanation for this observation could be that in the novels sometimes magic words related to the general content of the books or certain exclamations such as *aaaa* are often used. However, the result confirms that the use of words in the text corpus is largely the usual one and the usual methods of text analysis can be applied without further ado.
In the following until the end of section **2** we will again only used `clean_words`

2.4 Bigrams - Graph Visualization

To derive a graph for text visualization there are different approaches. One approach, which is very common used in `R` is a Bigram graph visualization. Therefore we derive the bigrams of `clean_words` and use `FreqDist` to count there frequency. The most occurent bigram is as expected *'said', 'harry'* following, *'said','ron'*. From the 50 most frequent bigrams we create a directed graph with `word1` f.e. *said* as `source` and `word2`, *harry* as `target` of the graph. The edge attribute are the `counts`. The result is demonstrated below:



a) left: Interaction of main characters (shortest path in graph is blue), b) middle: Centrality of graph node *said* is green, c) right: Degree Centrality of Bigrams Graph

Figure 5: Bigrams Graph Visualization (important graph sections)

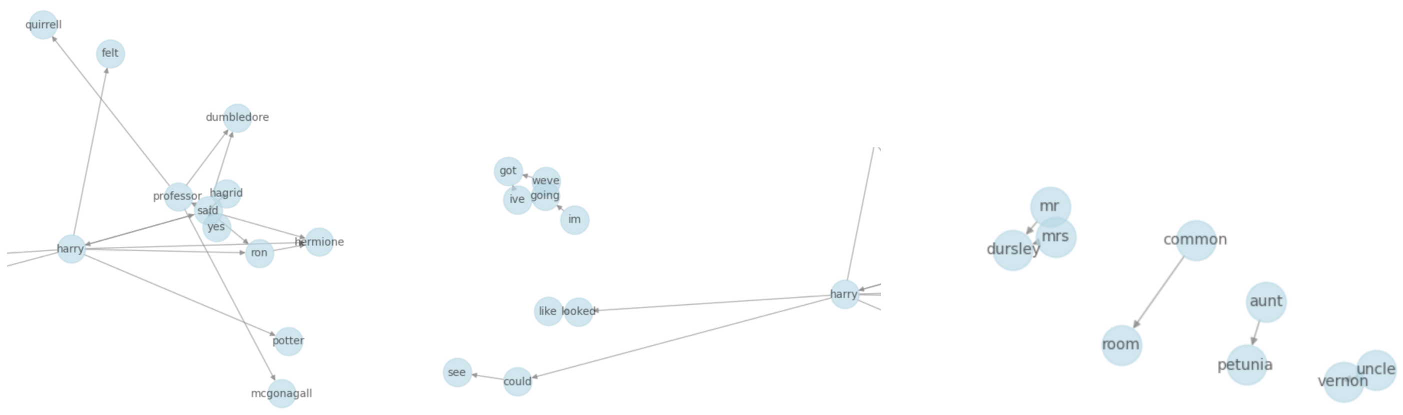
It is obvious from the Graph by zooming in to the center that there is a strong relation of *shortest paths* between the three main characters *harry, ron, hermione*, as well as around the node *said.* It seems like expected in children and youth literature that the direct speech plays a very important role. Further by analysing the **degree centrality** one gets reasonable results by considering that *said, harry, ron, hermione* have the highest degree centrality in the graph demonstrated in Figure c). With regard to the informative value of this graph visualisation, it should be noted at this point that it does not provide any real insight into the content of the books and does not deliver very different results from the word clouds. It should also be noted that it does not make sense to consider the entire text corpus as such for such a graph, but rather individual books. Additionally analysing trigrams is as well not appropriate for graph visualization, since they do not consider **relations** between characters, only counts the occurence of three following words. Overally on could say that bigrams graphs are useful to get a first impression but for shorter texts they could be more appropriate.

2.4.1 Bigrams - Graph Visualization with TF-IDF Vectorization

Therefore for better results in terms of Bigrams graph visualization we will consider now only one book. We decided to choose HP1 (`HPBook1.txt` and apply `CountVectorizer` and `TfidfVectorizer` one the `clean_words` to derive bigrams with `ngram_range` from $(2,2)$ and sort the bigrams in descending order of the `tfidf` value.
It must be noted at this step that the sklearn package smoothes the term frequency values. Let us consider the general formula [3] with

$$\text{tf-idf}(t,d) = \frac{\text{count } t \text{ in } d}{\text{total number of words in } d} \cdot \log\left(\frac{\text{total number of documents in text corpus}}{\text{number of documents containing } t}\right) \tag{1}$$

which can be implemented with the `sklearn` package (sklearntfidf[skl]). In HP1 for example one can derive as five most important bigrams again *said harry, said ron, uncle vernon, professor mcgonagall, said hagrid.* We apply the same methode as in the subsection before, but use as edge attribute the tfidf value times 10^3 for better visualization. The result is



a) left: Interaction of main characters, b) middle: Fragments of Harry's actions, c) right: Other single fragments of the Graph

Figure 6: Bigrams Graph Visualization with TF-IDF Vectorization (important graph sections)

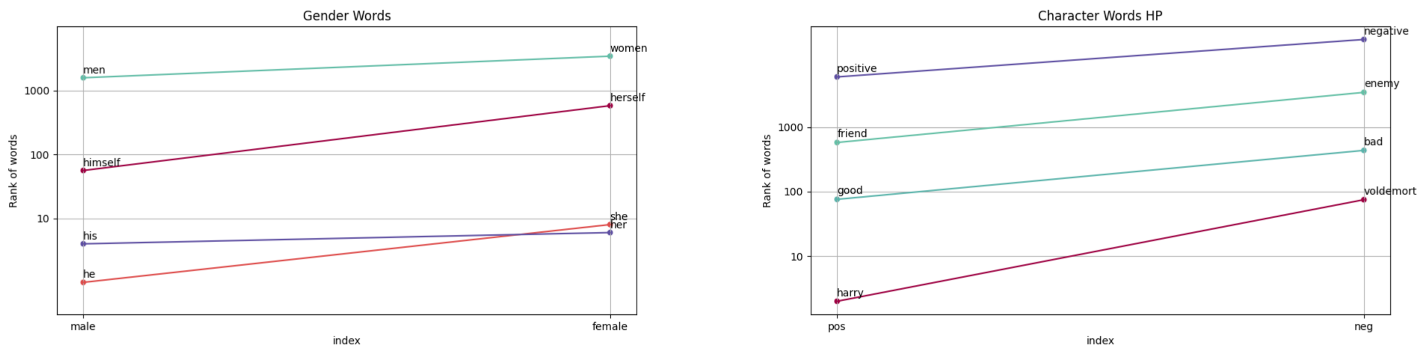
Figure 6 can be interpreted as follows: There are different characters, which occur in this book, like f.e. *harry, ron, hermione, hagrid, dumbledore*, which are important in the text and thus for Harry and there are f.e. other characters like *mr, mrs dursley, aunt petunia, uncle vernon*, which are not as related to Harry, since they do not occur with a high tfidf value together with *harry.* One could say that this aligns with our first impression of the Harry Potter novels, since the family Dursley do not have a positive relationship or an important relationship to Harry in comparison to Ron, Hermione, who are his friends and Dumbledore and Hagrid, his friends and advisors. This approach by considering only one book results in clearer results than the general Bigrams Visualization, but one has to say that is not a reasonable result for content summary or the building of social networks between the characters. We will come back to this in chapter 3 and 4.

2.5 Word rank slope charts (after Emil Hvitfeldt)

Since we recognized before from the Bigram Graph Visualization that there are as expected strong relationships between the main characters and other characters as well. At a first stage we compare gender specific words like *he, she, himself,*

[3](see lecture notes of this module not freely published, but available on Coursera as part of this module and thus not explicitly cited)

herself, men, women[4] through word rank slope charts. Therefore we use `clean_words_gender`, since the `stopwords` set of `nltk` would remove *he, she, himself, herself.* In a second step we will compare with the same approach the word rank of words which are important for sentiment analysis and character analysis, f.e. *positive, negative* and *harry, voldemort*; since we know that voldemort is harrys arch-enemy. The approach is fully taken from Emil Hvitfeldt (see [hvi]) and states that the word rank slope chart is generally designed to visualize the word rank difference of two different sets of words. So f.e. if the author of the book series often used masculine words in comparison to feminine words then then the word rank of f.e. *himself* is lower than for *herself.*



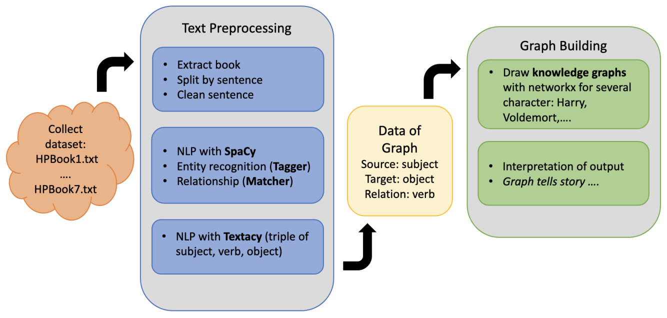
a) left: Word rank slope of gender specific words, b) right: Word rank slope chart of character specific words

Figure 7: Word rank analysis

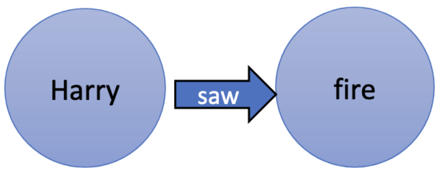
In Figure 6a) one can recognize that the importance of male words is much higher in comparison to female ones, since it has a lower rank. One could further say that words related to *himself* and *herself* have the same difference in comparison to *he* and *she*, since the two lines are parallel. The importance of *men, women* is in general not very high, since HP book series is based on children literature. In the word rank slope in Figure 6b) one can recognize that the author are more comfortable in using words with positive sentiments like *positve, friend, good.* Another significant characteristic of this chart is that the relation in terms of the rank difference between harry and voldemort can be compared with the occurence of positive and negative words. This once again confirms the assumption that Harry's friends can be categorised as positive and Voldemort's friends as negative.
Since we will analyse the relationship between the single characters it is useful to know if one can cluster them in terms of sentiment analysis, which will be done in section 4.

3 Knowledge Graph with `SpaCy`

A **Knowledge Graph**, also known as a semantic network, is a network of real entities. Entities are, for example, objects, events or situations. Such a graph can be used to visualise the relationships between such entities. The information from a text (in this case Harry Potter novels) is stored in a dataframe for graphs and the graph structure is then displayed (see [kg]). As is generally the case, the graph consists of three main components such as **entities 1 (source)**, **entities 2 (target)** and **relationships (relation)**. Each place or person can be such a node, i.e. source, while the edges visualise the relationships and interactions between them.



a) Pipeline of building a knowledge graph



Entity 1 related to Entity 2

b) Simple Knowledge graph with entity 1: *Harry*; relation: *saw*, entity 2: *fire* with subject, verb, object

Figure 8: Knowledge Graph: Process

[4]At this point it should be noted that the order in which the words are listed should not be interpreted in this sense.

In Figure 8a) the pipeline of building knowledge graph is given. The first part collecting the dataset is still done. The first part of the second part called text processing is also done in section 2.1. In this part we will now take advantage of `spaCy` to build a pipeline for entity recognition to find sources and target and using `Matcher` to get the relationship between them. A simple example is given in Figure 8a) with *Harry saw fire* a graph of two nodes and one edge. The pipeline which we used for this is given in Figure 9:

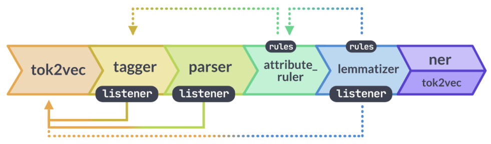


Figure 9: `en_core_web_sm` pipeline [spa]

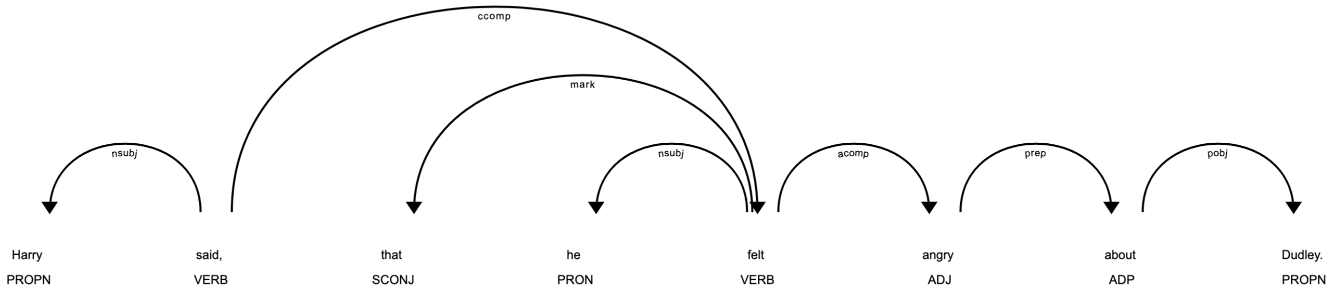
The `en_core_web_sm` pipeline in spaCy refers to a pre-trained statistical model specifically designed for English text processing. It consists of several components that work together sequentially to process text (see [spa]):

- **Tokenization (tok2vec):** The text is split into individual tokens, which represent basic units like words or punctuation marks.
- **Part-of-Speech (POS) Tagging:** Each token is assigned a grammatical category or tag indicating its part of speech (e.g., noun, verb, adjective).
- **Dependency Parsing (parser):** This component analyzes the syntactic structure of the sentence by assigning a dependency label to each token, determining how words relate to each other and their hierarchical relationships.
- **Attribute Ruler and Lemmatizer:** The attribute ruler lets you set token attributes for tokens identified by `Matcher` patterns and the lemmatizer is the component for assigning base forms to tokens using the rules of (POS).
- **Named Entity Recognition (NER):** Identifies and classifies named entities within the text, such as persons, organizations, locations, dates, etc.

The `en_core_web_sm` model is optimized for speed and resource efficiency while maintaining a reasonable level of accuracy across various NLP tasks. Its accuracy varies across different tasks and datasets, providing a good balance between performance and accuracy for general-purpose text processing. [5]

3.1 Knowledge Graph with POS

The POS tagger in spaCy is as mentioned above responsible for assigning specific grammatical categories or "tags" to each word in a given text. These tags represent the syntactic role of a word within a sentence. The POS tags assigned by spaCy include categories like noun, verb, adjective, adverb, pronoun, conjunction, etc., along with more specific tags within each category that detail tense, case, number, and other grammatical features. SpaCy's POS tagger uses deep learning models to predict these tags based on the context of each word in a sentence. The models are trained on a large amount of annotated text data to learn patterns and associations between words and their corresponding POS tags. Here's a basic example *Harry said, that he felt angry about Dudley* illustrating how you can use spaCy to perform POS tagging:



Simple Example of POS

POS is able to identify the categories of each word correctly in the above simple sentence. To get the entities and relations we choose a general approach (see [kg]), which can be summarize as follows The approach extracts subject and object entities from a cleaned sentence using `spaCy`. Thereby it identifies different parts of the sentence based on their

[5]Unfortunately the exact accuracy of this model ist not given.

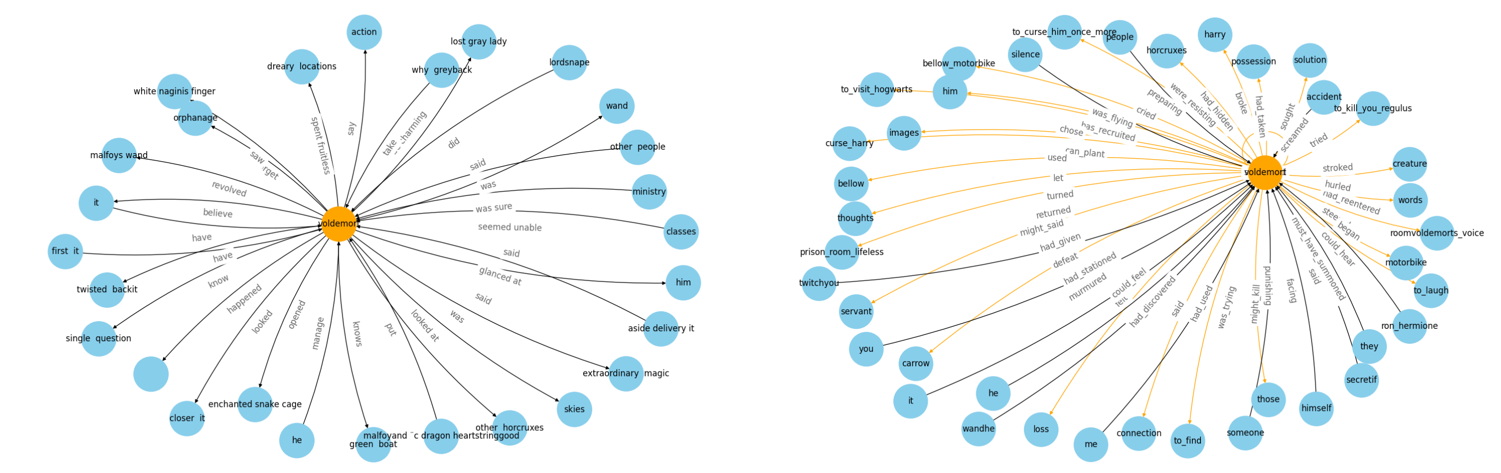
dependency relationships (subject, object, modifiers, etc.). A short summary of this approach is given below (function `get_entities`)

- It loops through the tokens in the sentence (with appropriate length >80) provided by `spaCy`.
- It checks the dependency label of each token to identify its role in the sentence (compound, modifier, subject, object).
- It constructs the subject (x) and object (y) entities based on these identified parts by concatenating the necessary tokens.

The process above is able to identify entities, in our simple example *he* and *Dudley*. In a second step the relationship `get_relation` identifies *path* as the relationship between the entities (source and target), but it neglects *Harry*. This function seems to be designed to work on sentences with a clear subject-object structure and might not cover more complex sentence structures. Remember that the accuracy of this function heavily relies on the accuracy of the `spaCy` parser and the specific sentence structures it encounters. For evaluation the performance of the result we choose HP7 and as source *harry* to analyse what happen with Harry in this book and what are his actions. We do not expect meaningful results, since the approach is not able to identify a clear structure. F.e. as a result one gets *harry said back*, which is a meaningless information extraction or *harry see ron*. Although there are several meaningless results, there are some good example like *harry straightened closer baggy bloodshot eyes*. However this approach is not appropriate to identify accurate relationships between words and reasonable information extractions.

3.2 Knowledge Graph with `Textacy`

For optimization of the results we take advantage of the library `Textacy`, which has the function `textacy.extract.subject_verb_object_triples` , which is specifically designed for extracting triples from text. These triples consist of Subject-Verb-Object (SVO) structures, which are commonly found in sentences and leads to better results, since it is designed to handle diverse structures even in complex sentences by using tokenization, part-of-speech tagging, and dependency parsing, to accurately identify the relationships between words and it useful for content and information extraction. Overall, `textacy.extract.subject_verb_object_triples` streamlines the process of extracting SVO triples, which can be valuable in various NLP tasks like information extraction, relationship analysis, and knowledge base construction. With this approach and a short sentiment analysis of the sentences with `Vader` to identify the sentiments of the sentences, one gets



a) left: Result of general approach in 3.1 for HP7 with Voldemort as source and target; b) right: Result of Textacy approach in 3.2. with all **negative** actions around Voldemort

Figure 10: Knowledge graphs around Voldemort

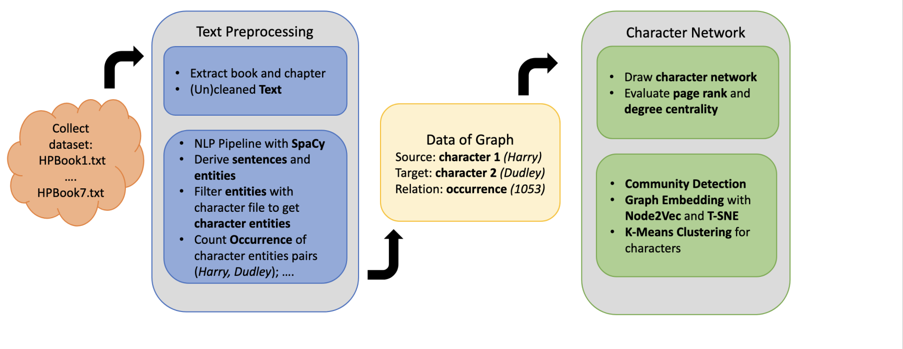
By considering Figure 10a) and Figure 10 b) one can see that we get from Figure 10b) a clearer impression about the content of HP7. F.e. the SVO triple *voldemort broke harry* or *voldemort had hidden horcruxes*, which is part of the content of HP7. In Figure 10a) some nodes are missing and expressions like *voldemort say action* is not a reasonable result.
We have seen in this approach that analysing SVO triples together with sentiment analysis yields into more reasonable results in the big textcorpus of the seven Harry Potter books.

4 Character and Social Networks with spaCy

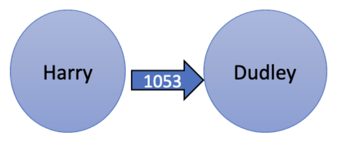
In this part we will compare two different ways of building a network graph with the HP corpus as basis. It is very common regarding networks of characters, that Character networks refer to a type of network representation used in literary studies and text analysis. We will use the following pipeline of building a character network with co-occurence.

4.1 Character Network (widget character network)

This approach is taken from [net] (so called widget character network)and calculate the number of interaction between the single characters. We use again the natural language pipeline `en_core_web_sm` of `spaCy`, but further use the dataset `characters.csv` from [datb], which has 2 columns and 99 rows. It contains all characters of the Harry Potter novels in the column `Name` and their variation like *Ron, Ronan* in the second column `Variation`. After applying the natural language processing model on the uncleaned dataset and identify the sentence entities automatically via the pipeline through tokenization of the sentence, we filter the resulting entities with the character names in the dataset `characters.csv` (with `filter_entities`). In the pipeline in Figure 11a) one can see the approach as well and how to derive the data for the graph with source (character 1) and target (character 2). The relation one can calculate through co-occurence of each character, which results in the dataframe `relationship_df` with source equal to character 1 and target to character 2 and the value is the number of interaction between them.


Collect dataset: HPBook1.txt HPBook7.txt
Text Preprocessing
Extract book and chapter
(Un)cleaned Text
NLP Pipeline with SpaCy
Derive sentences and entities
Filter entities with character file to get character entities
Count Occurrence of character entities pairs (Harry, Dudley); ....
Data of Graph
Source: character 1 (Harry)
Target: character 2 (Dudley)
Relation: occurrence (1053)
Character Network
Draw character network
Evaluate page rank and degree centrality
Community Detection
Graph Embedding with Node2Vec and T-SNE
K-Means Clustering for characters

a) Pipeline of building a character network (Co-occurence graph)

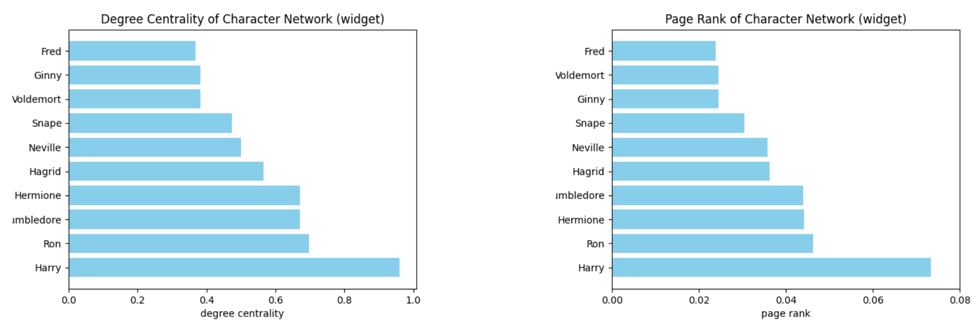


Entity 1 related to Entity 2 with occurence

b) Simple character network with entity 1: *Harry*; value: 1053, entity 2: *Dudley*

Figure 11: Character Network: Process

A simple example is given in Figure 11b) with Harry is related to Dudley 1051 times in the whole text corpus. To interpret the results we analyse the degree centrality and page rank of the builded graph (`G_widget`) . The results are given below:



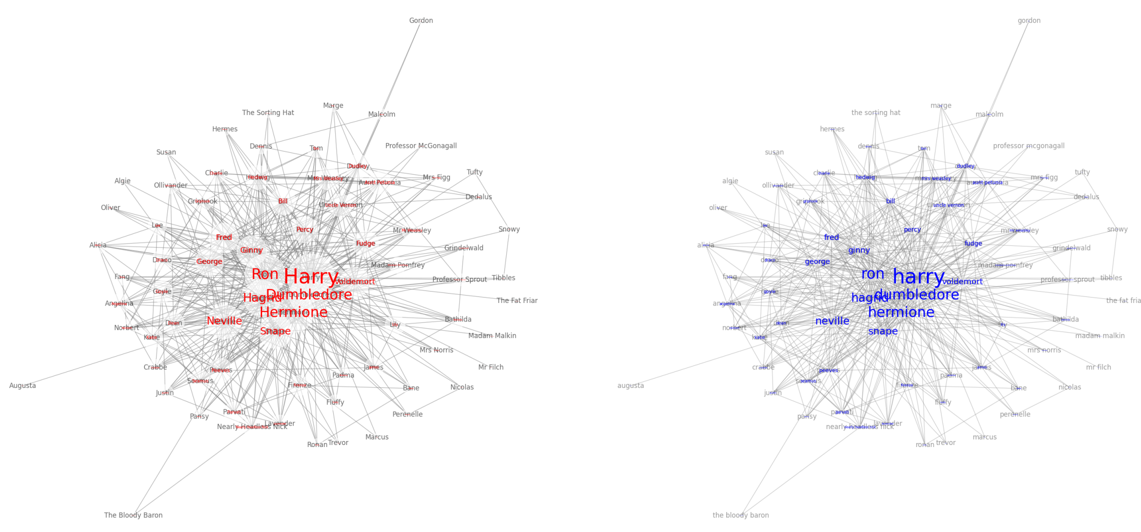
a) left: Degree centrality, b) right: Page rank of nodes in increasing order of character network graph

Figure 12: Measures of evaluation the performance of the character network

The results are overaly reasonable since in terms of page rank and degree centrality the expected main characters like *Harry, Ron, Hermione, Voldemort*, Dumbledore. The degree centrality of *Harry* is near 1 as it is expected, since degree centrality is a measure in network analysis that quantifies the importance of a node within a network based on the number of links it has. It counts how many connections (edges) a node has in a network. Nodes with a higher degree centrality are considered more central or important within the network as they have more connections, which one would expected in the seven books all named after Harry. Additionally the page rank assigns a score to each node in a directed graph, with the score representing the probability that a person randomly traversing the graph will visit that node. Again the node *Harry* has the highest value, followed by *Ron, Hermione, Dumbledore,...*.

4.2 Social network with sentiment analysis

In a second step we will compare this approach with a model based on sentiment analysis, which is often used for building social networks, which are representations of are representations of social relationships or interactions between individuals, groups, or entities. Nodes in a social network represent individuals or entities, and edges represent relationships or interactions between them. These relationships can be varied, including friendships, collaborations, interactions, or any other form of connection between social entities. Therefore we analyse the sentiments in each sentence and calculate the `compound` (with `VADER` and the `SentimentIntensityAnalyser`for each character by summing the compound value in each sentence or action the character appears. Thus each character has its own compound value (saved in dataframe `sentiment_df`. Afterwards we assign this values to the source and target of the `relationship_df` dataframe. Then we can calculate the sentiment of a **relation** by weighting the `sentiments_source` and `sentiments_target` values of the two characters, by dividing the compound value by the number of occurrences of the character in the text corpus for each source and target and then adding the respective weighted values of source and target sentiment values. It results the dataset `rel_sent_data`, in which we can identify whether two characters are `friend` or `enemy` or `object` to each other. If the compound value of the relation is > 0.05 they are friends, it it is < -0.05 they are enemies and else objects to each other. The resulting (undirected) graph `G_sent` has comparable values in terms of page rank and degree centrality as the character network graph of 4.1 `G_widget` (undirected) , with 77 nodes and 523 edges. The results of both graphs are given in the following figure, where the font size of the nodes are relation to their degree.



a) left: Character Network after network [net] b) right: Social Network with sentiment analysis and `kamada_lawai_layout`

Figure 13: Character Network and Social Network

It is obvious that the results are comparable and that both graphs measure the importance of the main characters correctly.

4.3 Community Detection

In the following we will analyse if one can detect communities inside both networks. **Community Detection** is a process within network analysis that aims to identify groups or communities of nodes within a network where nodes are more densely connected internally compared to the rest of the network. We use **modularity** as a measure to evaluate the strength of division of a network into modules or communities. It measures the density of links inside communities against links between communities. The modularity Q for a given partition of a network into communities is calculated using the following equation:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j) \quad Q \in (-0.5, 1) \tag{2}$$

where: Q is the modularity score, A_{ij} is the adjacency matrix element, k_i, k_j are the degrees of nodes i and j, m is the total number of edges, $\delta(c_i, c_j) = 1$ if nodes belong to same community and 0 otherwise The goal of community detection algorithms is to find a partition of nodes that maximizes the modularity score, indicating a strong division of the network into communities. If this is the case Q is near 1. By testing both graphs with the **Louvain** algorithm and the well-known **Girwan-Newman** algorithm for community detection with the `networkx` package, one gets very low modularity scores. For `G_widget` the optimized result by using `naive_greedy_modularity_communites` (see [mod]) a function to optimize modularity one gets $Q = 0.11$, for `G_sent` one gets $Q = 0.17$. Both scores are very low and thus the resulting graph with labelled communites are not given in this report. An explanation could be that on would need further information about a number of 77 characters to derive well-defined communities.

4.4 Graph Embedding and Clustering

For further analysis we use graph embedding, which involves transforming the structural information of a graph into low-dimensional, continuous vector representations. These embeddings capture the relationships and topology of the graph in a format suitable for machine learning algorithms. Thereby we use `Node2Vec` to define the word embeddings with 200 number of walks, $T = 30$ walk length and 10 workers in a dimension of 40. After training we use `tsne` to reduce the dimension and visualize the embedding in the 2D space. Afterwards we apply `KMeans` Clustering and `Agglomerative` Clustering on the embedding in 2D space. In general the k means clustering gives better results with a highest silhouette score around 0.44 for $k = 3$ of the social network with sentiment analysis. The result is given below

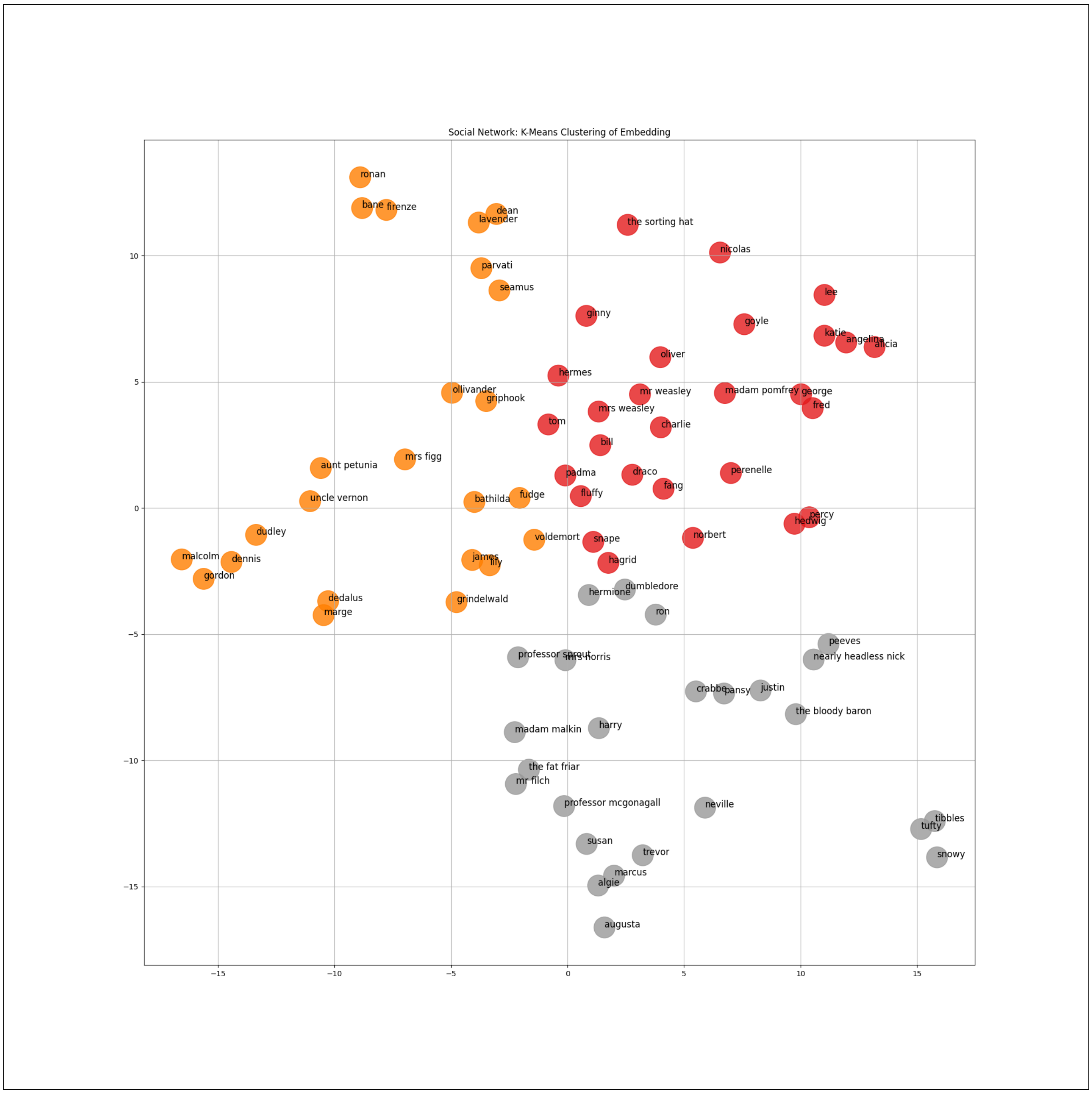


Figure 14: K-Means Clustering of Node2Vec Embeddings in 2-dimensional space

One can see that this approach was able to identify clear clusters. It defines *harry, ron, hermione, dumbledore* into one and f.e. *voldemort, uncle vernon, dudley*[6] into another. In another group f.e. the weasley family is located (family of Ron) with *ginny, mr weasly and mrs weasley,...* It seems that graph embedding and clustering gives the best results

5 Conclusion and Future work

To conclude one could say that

- Initial exploratory data analysis can give initial impression about a large text corpus
- Knowledge graph give reasonable results, if one specify the content they should know
- Character network and social networks give better results after graph embedding and clustering

For future work it would be interesting to use large scale language models to get more information about the book series.

References

[data] https://github.com/ErikaJacobs/Harry-Potter-Text-Mining/tree/master/Book%20Text

[datb] https://github.com/ohumu/HarryPotterNetwork/blob/main/characters.csv

[hvi] https://emilhvitfeldt.com/post/2020-03-17-word-rank-slope-charts/

[hzj] https://github.com/hzjken/character-network

[kg] https://www.kaggle.com/code/nageshsingh/build-knowledge-graph-using-python

[mod] https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.modularity_max.naive_greedy_modularity_communities.html

[net] https://github.com/ohumu/HarryPotterNetwork/blob/main/extracting_relationships.ipynb

[New05] Newman, M. E. J.: *Power laws, Pareto distributions and Zipf's law.* Contemporary Physics 46(5), 323-351, 2005

[skl] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html#sklearn.feature_extraction.text.TfidfTransformer

[spa] https://spacy.io/models

[tus] https://github.com/tthustla/twitter_sentiment_analysis_part3/blob/master/Capstone_part3-Copy2.ipynb

[wik] https://en.wikipedia.org/wiki/Harry_Potter

[zip] https://www.tidytextmining.com/tfidf

[Zip49] Zipf, Georg: *Human behavior and the principle of least effort.* Addison-Wesley Press, 1949

[6]besides voldemort, the relation between the dursley and harry is not positive