Character Relationships in Fairy Tales

Burhan Can Akkuş
Department of Computer Engineering
Bogazici University
Istanbul Turkey
can.akkus@boun.edu.tr

July 1, 2021

Abstract

Our project is designed to process a given text and find out what each character in the story thinks of any other character they have a relation with. This is done by extracting the characters via certain Natural Language Processing tools and then analyzing the semantic relations between them.

1 Introduction

Our project is an Natural Language Processing(NLP) project we aim to analyze a given text (fairytales of Grimm Brothers in current scope) and extract the main characters of the text and to create a map of relationships between these characters. The relationships are labeled as either positive, indicating friendly and good intentions and opinions from one character to another, or negative, indicating hostile and bad intentions and opinions from one character to another. We aim to investigate these relationships and create a categorization of each characters opinion of the others.

2 Related Work

2.1 Extracting Social Network from Literature to Predict Antagonist and Protagonist

This research is done by a team of students from the Standford University. Their aim is to investigate the Social Networks in literature works and use this investigations results to find the Protagonist and the Antagonist of the works.

They followed similar steps to our work.

The first aim of this work is character extraction. For this they developed their coreference resolution system. Their results on Character Identification is quite promising with high recall rates and somewhat high precision rates.

For sentiment Analysis, they used a sentence based approach. Whenever a set of characters are present in a sentence together, that sentence is assumed to carry information about their relationship. The focus of the sentiment analysis is the verbs of these sentences.

Their final aim was to identify the antagonist and the protagonist of the story by examining the relationships they extracted in the previous step. They use triads of character relationships to determine certain groupings of characters.

Their findings are quite promising with the coreference F1 score of 0.815, sentiment analysis score of 0.71, protagonist score of 0.60, antagonist score of 0.25, and balanced prediction score of 0.60 [1].

2.2 Automatic Animacy Classification

In this paper, the researchers developed upon the binary classifier developed by Zaenen et al.(2004) and managed to train a classifier with 10 different classes of animacy. The work makes use of syntactical clues.

First of these clues is that the animacy of a Noun Phrase(NP) is highly correlated by the animacy of its head. "The Panama hat I gave to my uncle on Tuesday" contains several nominals of different animacy class but it is the "hat" that determines the animacy of the whole phrase.

Second clue they used makes use of the verb in the sentence. Verbs carry certain specifications about their subjects and objects. The paper's example is the verb "to see", which requires its subject to be one of human, machine,animal or organization.

Using these clues, the researchers managed to get an accuracy score of 86 percent [2].

3 Approach

3.1 Pre-Processing

The texts are taken from https://www.gutenberg.org website. They contain some unorthodox punctuation marks like " ‘ " " ’ ", and some of these are not recognized by python's file.read() function. For this reason, we had to pre-process the text files.

In this step, we read through the whole document and replaced any problematic characters with their acceptable equivalences. We also put spaces after each punctuation because the tools we used in later stages were having problems with tokenizing the sentences properly when the punctuation and the character after the punctuation were not seperated by a space.

```
f = open("./Stories/"+taleName,'r',encoding="utf8")

tale= f.read()
tale = tale.replace('\n', ' ')
tale = tale.replace('\r', ' ')
tale = tale.replace('‘', '\"')
tale = tale.replace('’', '\"')
tale = tale.replace('“', '\"')
tale = tale.replace('”', '\"')
```

Figure 3.1: Pre-Processing Pseudo Code.

3.2 Coreference Resolution

Our first step is to do a Coreference Resolution on the given text to replace pronouns with their proper nouns. For this task we used Stanford CoreNLP's Coreference Resolution module [3]. Each reference chain is resolved to its head noun.

```
Original Text:
Hansel and Gretel are trapped in the witch's house. They have to find a way out before she catches them!
Text after Coreference Resolution
Hansel and Gretel are trapped in the witch's house. Hansel and Gretel have to find a way out before the witch catches Hansel and Gretel!
```

Figure 3.2: Example use of Coreference Resolution.

The output of the Coreference Resolution module is an array of chains which contain references to the same entity. Each chain contains a list of mentions,

the location of the mentions in the text and a mention that is deemed to be the representative mention of the chain. For each chain the representative mention is extracted. For every mention in the chain, we located the mention in the text and replaced it with the representative mention of the chain. Thus we had replaced each pronoun with it's referring entity.

In this step we ran into a few problems. The output of CoreNLP's coreference chains were not as useful as we hoped. There were some chains with no proper noun as their head. They were consisting of only pronouns. This was partly due to CoreNLP being an insufficient tool and partly due to the convoluted writing of Grimm Brothers'. For the reasons above, we decided to manually resolve coreferences of a few tales(Briar Rose, Hansel and Gretel, Cinderella and The Cat and The Mouse) to get a good measurement of the success of the later steps.

3.3 Character Identification

3.3.1 Subject Extraction

For this task, we started with a simple intuition: The actions in a fairy tale are usually taken by the main characters of the story. By this thought in mind, we investigated the syntactic structure of every sentence in the text and extracted the subjects of each sentence. Sometimes the subject is not a single word but rather a phrase like "the third pig" or "the thirteenth fairy". These cases are ignored and only the head of the subject phrase is considered the subject of the sentence. After completing this for every sentence, we simply counted how many times each entity was the subject of a sentence. We then sorted this counts in descending order. After the sorting, the list is chopped down to its top N characters, where N is the number of characters for that story, labeled manually.

```
1
2  subjects=dict()
3
4  for sentence in doc['sentences']
5      for word in sentence['dependency']
6          if word['dep']=='subject'
7              if subjects[word]==nil
8                  subjects[word]=1
9              Else
10                 subjects[word]=subjects[word]+1
11             Endif
12         Endif
13     Endfor
14 Endfor
15
16 sortedSubjects=sort(subjects)
17
18 return sortedSubjects
19
```

Figure 3.3: Subject Extraction.

3.3.2 Animacy Detection

Any character in a tale should be an independent agent that can act on her will freely. Any entity who does not have free will or does not exercise it during the tale can not be a character of the tale. With this information in mind, we decided to add an animacy filter to the candidate characters list which was extracted from the tale in the Character Identification step.

There are a couple different possible approaches to this problem. The first and the easiest one we tried was to take each entity from the candidates list and compare it with an online English Word Animacy database. This approach was quickly disregarded after we saw the first results. The results were that only the human or humanoid characters were labeled as animate/free agent while animals and some other objects were labeled as inanimate/partially free agent. This was a problem because in the realm of fairy tales there are many characters who are animals and normally inanimate objects(The Big Bad Wolf from Little Red Riding Hood or The Talking Candlestick from Beauty and The Beast). We saw no possible improvement in this approach and decided to abandon any database driven ideas completely.

For our second approach we came up with a couple rules that helped us come to a conclusion on what kind of a solution we were seeking. The approach would

have to decide on animacy on a case by case solution, any approach who had a static answer would be insufficient for the task at hand. With this information, we decided to use a verb based analysis approach. The main idea of this approach is that, each verb has a connotation about the animacy any free agency of it's subject. Verbs like say, talk and think imply a human like conciousness for the subject. Any agent who can take these actions are considered to be a free agent who can act on her will. Some other verbs like walk, move, climb imply self decided movement capabilities for the subject. These words indicate animacy and partially free agency.

Our search for a verb based solution lead us to a few possible solutions. From these possibilities we decided to use the Animacy property of the Stanford CoreNLP's Coreference Resolution module. We were already using the Coreference Resolution module for the previous steps. This decision made it possible to keep our number of tools and outside dependencies to a minimum.

We investigated each coreference chain. From these chains we took out the ones labeled ANIMATE in the animacy property and made a list out of them. This list was then used as a filter for the Candidate Characters List from the previous step. Any entity who was not labeled as ANIMATE in any of the chains were considered as inanimate and discarded.

This approach was mostly successful. However it has a major flaw. Any main character who is not referred to with any pronouns or any character whose coreference chain is misidentified would not be considered as to be a major character of the story. This flaw caused problems for only 2 tales from our list. While it is not critically important this flaw still makes it a necessity to further improve our approach.

To select the major characters out, we used StanfordCoreNLP's Animacy property of coreference chains [2]. We exported coreference chains labeled as animate from the previous step and used them as a filter for our list from the Syntax analysis part. This significantly improved our model and we achieved a high precision rate for Character Identification.

```
1  subjects=read("./Characters"+taleName)
2  animates=read("./Animates"+taleName)
3  characters=[]
4
5  for subject in subjects:
6      if subject in animates:
7          characters.append(subject)
8      Endif
9  Endfor
10
11 return characters
```

Figure 3.4: Filtering Potential Characters based on Animacy.

3.4 Relationship Evaluation

For this step, we utilized SentimentIntensityAnalyzer of python's nltk library. This tool reads a sentence and returns its positive, negative and compound sentimental values. Positive score measures how positive elements the sentence contains and negative score measures how negative elements the sentence contains. The compound score is the measurement that decides the overall sentimental value of the sentence.

To find the relationships between characters we simply iterated through all sentences in a tale. For each sentence, a list of characters whose name is mentioned in the sentence is created. Then the sentence's compound score is added to the list of related sentences for each tuple in the list.

4 Results

For evaluation of our project,we could not find any labeling of the Grimm Brothers' Fairy tales. Because of this reason, we had to manually label the tales we used for evaluation. The data used in this step is all done manually by the project team. Results of our work are compared against this manual labeling. You can find the results of our manual work in our github repo.

4.1 Character Identification

For character Identification, the output list of candidate characters are chopped down to include only the top N number of candidates, where N is the number of characters for the story, identified manually. For ambigous cases like "pig" in "Three Little Pigs" or "son" in "The Golden Bird", the identified character is assumed to correspond the highest suitable character in the manually labeled data. "The third pig" in the case of Three Little Pigs and "The third son" in "The Golden Bird".

A candidate character guess is correct if the character is in the manually labeled list of characters and incorrect otherwise.

Tale	First Character Extracted is a Main Character	Second Character Extracted is a Main Character	Rest of The Characters
BRIAR ROSE	+	+	2/4
CAT AND MOUSE IN PARTNERSHIP	+	+	-
CINDERELLA	+	+	2/3
THE DOG AND THE SPARROW	+	+	1/2
THE FISHERMAN AND HIS WIFE	+	+	1/1
THE FROG-PRINCE	+	+	-
FUNDEVOGEL	+	-	2/3
THE GOLDEN BIRD	+	+	2/4
HANSEL AND GRETEL	+	+	1/3
HANS IN GOOD LUCK	+	-	3/6
JORINDA AND JORINDEL	+	+	1/1
LITTLE RED RIDING HOOD	+	+	2/3
OLD SULTAN	+	+	3/4
RAPUNZEL	+	+	1/3
THE GOOSE GIRL	+	+	1/3
THE STRAW, THE BEAN AND THE COAL	+	+	0/1
THE WILLOW WREN	+	-	2/5
THREE LITTLE PIGS	+	+	0/2
THE TRAVELING MUSICIANS	+	+	2/3
THE TWELVE DANCING PRINCESSES	+	+	1/3
Total	20/20	17/20	27/54
Total Success Rate	100%	85%	50%
Overall Success Rate		64/94=68.1%	

Figure 4.1: Character Identification Results.

We also did a test run with the tales which had their coreferences resolved manually to better see the success of our approach without the interference of the unsuccessful resolutions. These 4 tales had a 13/18(0.722) of their characters successfully extracted with automated coreference resolution. With manual coreference resolution, this number jumps up to 16/18(0.889). The two missing characters are "the thirteenth fairy" from "Briar Rose" and "the stepsisters" from Cinderella.

Tale	First Character Extracted is a Main Character	Second Character Extracted is a Main Character	Rest of The Characters
BRIAR ROSE	+	+	3/4
CINDERELLA	+	+	2/3
CAT AND MOUSE IN PARTNERSHIP	+	+	∅
HANSEL AND GRETEL	+	+	3/3
Success Rate	1	1	0.8
Overall Success Rate		0.889	

Figure 4.2: Character Identification from Tales which had their coreferences resolved manually.

4.2 Relationship Evaluation

We categorized each relationship between identified characters of a story as either positive or negative. These categorizations were then compared to our manually labeled relationships.

Tale	Positive Relations	Negative Relations	All Relations
BRIAR ROSE	5/9	0/3	5/12
CAT AND MOUSE IN PARTNERSHIP	ø	0/1	0/1
CINDERELLA	1/5	0/2	1/7
THE DOG AND THE SPARROW	0/2	1/3	1/5
THE FISHERMAN AND HIS WIFE	3/3	ø	3/3
THE FROG-PRINCE	1/1	ø	1/1
FUNDEVOGEL	0/3	2/3	2/6
THE GOLDEN BIRD	2/4	1/3	3/7
HANSEL AND GRETEL	0/4	0/4	0/8
HANS IN GOOD LUCK	0/2	0/4	0/6
JORINDA AND JORINDEL	0/1	0/2	0/3
LITTLE RED RIDING HOOD	1/5	2/4	3/9
OLD SULTAN	0/5	4/6	4/11
RAPUNZEL	0/3	3/5	3/8
THE GOOSE GIRL	2/3	0/3	2/6
THE STRAW, THE BEAN AND THE COAL	0/1	1/2	1/3
THE WILLOW WREN	0/8	1/8	1/16
THREE LITTLE PIGS	0/3	1/3	1/6
THE TRAVELING MUSICIANS	0/6	0/4	0/10
THE TWELVE DANCING PRINCESSES	2/4	1/5	3/9
Total	17/72	17/ 65	34/137
Success Rate	0.236	0.261	0.248

Figure 4.3: Results compared to all of the relationships found manually

To get a better understanding of how succesfully the code categorizes relationships, we also measured its success of categorization. Every relationship our code found was either labeled positive or negative. We compared this labels with our manually labeled relationships. The results were expectedly better than the previous ones, rising to 56 percent from 24 percent.

Tale	Number of Correctly Labeled Relations	Number of Total Labeled Relations	Success Rate
BRIAR ROSE	5	7	5/7
CAT AND MOUSE IN PARTNERSHIP	0	1	0/1
CINDERELLA	1	4	1/4
THE DOG AND THE SPARROW	1	2	1/2
THE FISHERMAN AND HIS WIFE	3	3	3/3
THE FROG-PRINCE	1	1	1/1
FUNDEVOGEL	2	3	2/3
THE GOLDEN BIRD	3	5	3/5
HANSEL AND GRETEL	0	1	0/1
HANS IN GOOD LUCK	*0*	*3*	*0/3*
JORINDA AND JORINDEL	0	3	0/3
LITTLE RED RIDING HOOD	3	4	3/4
OLD SULTAN	4	8	4/8
RAPUNZEL	3	3	3/3
THE GOOSE GIRL	2	2	2/2
THE STRAW, THE BEAN AND THE COAL	1	1	1/1
THE WILLOW WREN	1	2	1/2
THREE LITTLE PIGS	1	1	1/1
THE TRAVELING MUSICIANS	0	3	0/3
THE TWELVE DANCING PRINCESSES	3	3	3/3
TOTAL	34	60	34/60=56.7%

Figure 4.4: Labeling Success of the correctly extracted Relationships.

We also did a test run with the tales which had their coreferences resolved manually to better see the success of our approach without the interference of the unsuccessful resolutions. This resulted in our success of 6/28(0.214) from the automated coreference resolution to jump up to 18/28(0.643). This confirms our expectations that the coreference resolution is a crucial part for extracting sentimental relationships.

Tale	Positive Relations	Negative Relations	All Relations
BRIAR ROSE	8/9	0/3	8/12
CINDERELLA	2/4	2/3	4/7
CAT AND MOUSE IN PARTNERSHIP	∅	0/1	0/1
HANSEL AND GRETEL	3/4	3/4	6/8
Total	13/17	5/11	18/28
Success Rate	0,765	0,455	0,643

Figure 4.5: Results from the manually resolved tales compared to all of the relationships found manually.

5 Ongoing Development

5.1 Relationship Evaluation

5.1.1 Co-agency

Co-agency is when two or more characters are part of the same group taking or receiving an action. We developed a simple intuition that, characters who are acting together are expected to have a positive relationship with each other. To exploit this intuition, we divided the characters that appear in the same sentence into two categories. The co-actors group, which act as the subject of the sentence together, and co-objects, which are on the receiving end of the verb together. Any character tuple appearing together in one of these groups are given a positive relationship score. Whenever two characters do something together, it is taken as a positive indicator.

5.1.2 Opinion of the Actors Group towards the Object Group

The second intuition we are developing is that, the two groups of entities in a sentence in relationship to the verb(root) of the sentence have a relationship with each other. The whole sentence gives an idea about the opinion of the subject group towards the object group. After grouping the characters into these two categories, we take the sentimental value of the sentence and add it to the opinion of each member of the subject group towards each member of the object group.

6 Open Questions

6.1 Character Identification

6.1.1 Places of Occurrences

Characters in a story can be grouped into two main categories. First is the main characters and the second is the helping characters. A difference between these groups is that the main cast of characters are present throughout the whole story while the helping cast of characters are usually only present in a limited section of the story. By investigating the locations of the mentions that refer to a character, a better prediction can be made on which category the character belongs to.

6.1.2 Alias Resolution

A character referred to by a number of different strings is a pretty common occurrence in literature. The Little Red Riding Hood is a good example of this. Riding Hood is referred to by the strings "girl", "Hood" and "the little girl" depending on the context. Her grandmother is also referred as grandma and granny. These strings are obviously referring to the same characters and need to be handled as such.

6.1.3 Equivalent Relationships

The same relation between a set of characters can be put into text in differing ways. A grandmother can be referred to as the mother's mother. We can see several examples of this from our tales dataset. In "Hansel and Gretel" Hansel and Gretel's stepmother is first introduced as "the father's wife". Later, she is referred to as "the mother". These two obviously refer to the same character, a father's wife is the mother of his children. However, this can not be inferred from the content of the text. We need outside information on equivalency of different relationships. We found a few different ontologies which looked promising during our research but we had to abandon that side of our project due to time and workload constraints.

6.1.4 Dynamic Attributes

A change in the state of a character might have an effect on other characters and their names. For example, in "The Goose-Girl" tale, the young prince becomes king after his father dies. A change in the king's state of liveliness causes a change

in the naming and the state of the prince. To resolve this it is again required to employ an ontology to gain information about the different relationships and how certain actions affect them. This was also deemed out of this project's scope but it nonetheless remains to be an interesting question.

6.2 Relationship Evaluation

6.2.1 Deception

A common theme in fairy tales is when a character disguises herself and acts in the name of another. This gives an exciting twist to the story but it also causes the relationship detection to grossly mislabel the interaction. In "Little Red Riding Hood" the Wolf disguises himself as the Grandmother after he eats her. When Little Red Riding Hood arrives to Grandma's house, she thinks it is her grandmother in the bed. The Wolf then talks to her as if he was her Granny. The dialogue, no matter how interesting it makes the story, poses quite challenging task for the name based relationship extraction systems. Little girl says "Grandma, how big teeth you have!!" but what she really means is that the wolf has big teeth. To correctly identify the semantic meaning of the relationship between Little Red Riding Hood, the Wolf and the Grandmother, the case of deception has to be handled.

6.2.2 Teaming Up

"Enemy of my enemy is my friend." and "Enemy of my friend is my enemy.", these are the main clues on how to define a relationship between characters in a story which are never present in the same scene or know of each other. In "Hansel and Gretel", the witch and the mother are working together even though they don't know it. The mother wants the children to disappear and the witch wants to eat them. This common enemy creates an unrealized alliance between the witch and the mother. With further investigation into character relationship networks, better assesment of character alignments can be made.

7 Conclusion

We managed to get promising results in character Identification, with 100 percent confidence in the first candidate and 85 percent confidence in the second candidate of main character extraction. Our overall score is 68 percent. With further improvements in coreference resolutions and syntactic sentence analysis, this result can get even better.

Our results in relationship extraction were not as successful as the character extraction. We only managed to find 25 percent of all the relationships. This result has a number of factors involved. Most significant of these factors is the missing characters. 30 percent of the characters don't reach the relationship extraction step, causing a larger portion of the relationships going undetected completely. The relationships we managed to detect are labeled rather unsuccesfully. Only a 56 percent of the labeled relationships were labeled correctly. However, with improvements in the ongoing developments part and potential solutions of the problems in the open questions section, we believe that this result can get significantly better.

Bibliography

[1] Ben Ulmer Matt Fernandez, Michael Peterson. Extracting social network from literature to predict antagonist and protagonist, 2015.

[2] Harshit Chopra Samuel R. Bowman. Automatic animacy classification.

[3] Christopher Potts Marta Recasens, Marie-Catherine de Marneffe. The life and death of discourse entities: Identifying singleton mentions. in proceedings of the naacl, 2013.