Character Persona Identification and Summarization using NLP Techniques

Amrutha S
PES1201700829
Computer Science
PES University
Bangalore
amruthashetty299@gmail.com

Ria Kalia
PES1201700829
Computer Science
PES University
Bangalore
riakalia@gmail.com

Ashwini M Joshi
Professor
Computer Science
PES University
Bangalore
ashwinimjoshi@gmail.com

ABSTRACT

In this decade, the number of novels is increasing rapidly. Many studies have been proposed to assist users in novel understanding. We aim to map a layout of all the characters in a novel, while trying to identify their relationships, if any, with other characters in the novel. We focus on identifying all the characters using language processing tools and try to weed out all the conversations between the characters to recognize entities and also the adjectives used to describe them. Furthermore, we extract the features from the novel and create a NetworkX graph of all sentences of the novel and produce a summary of the novel. Character relationship graphs and several other evaluation measures outputs are produced as an output expanding upon the nature of relationships.

Keywords: Name Entity Recognition, Sentiment Analysis, Matrix multiplication, NetworkX, Character Relationship

1.INTRODUCTION

In today's world, while reading books - primarily novels, people often skip passages or chapters so that they can get to the interesting parts. However, one ends up missing important parts of the book while doing this, creating gaps in understanding of the plot and the interaction between characters. We wanted to help out such readers by providing them with the necessary information required to understand a text or novel.

A lot of research in the NLP field has been done in the textual summarization of novels aiming to describe the entire text in a brief and concise manner. These systems try to gauge the crux of the information being portrayed in the story to build up a knowledge base. The knowledge base is then converted back into a textual block summarizing the entire text in a few sentences. These techniques focus more on knowledge but less on sources or characters in the text that provide the knowledge. These summarization techniques, however, do not focus on the characters and their interaction with other characters to be able to generate a concrete entity-entity relationship survey.

Identifying the character to character relationships in a novel can be helpful not only to summarize the novels but also to find similar works of literature from the same author. E.g. The character of Elizabeth Bennet in the novel Pride and Prejudice resembles Elinor Dashwood in Sense and Sensibility, authored again, by Jane Austen; more than either character resembles Allen Quatermain in Allen Quatermain authored by H. Rider Haggard. Austenian protagonists should resemble each other more than protagonists from books are written by other authors (Bamman, 2014). Many literature studies prove that the character personalities and relationships tend to be similar when authored by the same person as compared to the works by other authors.

We have aimed to create a system that would take any novel as an input and return the top specific number of characters in the book as an output represented by a character relationship graph. We also show all the closest characters that a character has in the novel based on the interactions that took place in the novel through the thickness of the edges of the graph. Furthermore, the system recognizes the integrity

values of the character, trying to figure out the antagonists and the protagonists of the storyline.

We have taken data from the internet, for testing purposes, and any novel or text can be passed as an input to this program. As the common words tokens file is made available over the internet, we have used that for eliminating common words.

The method for implementing the plotting of character relationship can be broken down into 4 major parts, i.e. Character recognition in the text using, where the characters are extracted along with their frequency. Next, the character closeness relationships are identified based on the communication between the two characters. Thirdly, the integrity of characters is computed upon by referencing a common word file. Finally, the character relationship is being visualized in a NetworkX graph for better understanding.

Likewise for the summarization, we perform feature extraction then plot the nx graph for deriving the sentence relationship and we use the threshold filtering method to filter out the sentences.

For the evaluation measures, we have used several techniques such as a comparison of the character relationship NetworkX graph before and after summarization. We have also used Levenshtein distance, cosine distance, and document similarity in radians.

Many techniques from language processing and sentiment analysis have been used, individually or combined, to achieve the task that we aim to achieve. Although most of the work is flexible to work on any input dataset and are heavily annotated or transcribed to achieve optimum results, we aim to create a system that would sustain accepting any novel that has more than a few characters and provide a generalized output.

Some of the notable techniques used are extract characters and compute the co-occurrence matrix and sentiment matrix to learn entity-entity relationships and using a sentence to word NetworkXs to deduce the novel into a summary.

3. Related Work

Several techniques from language processing have already been used along with other inputs to achieve the summary and character relationship details. However most of the work is not efficient on any input dataset of any size. We aim to develop a system that can fill in the gaps and create a better and efficient output.

Character Relationship: This is a topic less addressed during the sentiment analysis summarization of novels. Characters form an important part of the analysis of the novel because in the process of sentence extraction there should not be a loss of important characters.

Abstractive Summarization: Abstractive methods use machine learning techniques which use most deep learning techniques like sequence, where recurrent networks read the text, encodes it and then generate target text These techniques use Abstract Meaning Representation (AMR) graphs using co-occurrence resolution, anaphora resolution and network generation by the creation of character nodes.

2. Data

As mentioned earlier, our goal was to come up with a generic model that performs well across all the genres of the literature. We handpicked several novels ranging from the genre of classic English literature to modern fantasy. Two major sources of our dataset are given below.

2.1 Immortals Of Meluha

The Immortals Of Meluha is an extremely popular series of novels by Amish Tripathi. The plot of this novel revolves around its central characters, Shiva (the protagonist), Sati, Nandi, Veerbhadra, and how they manage to save the Meluhans and expose them to various unknown truths of their kingdom and neighboring areas. The work is extremely rich in characters and plot. This novel was downloaded from the web in the form of text documents.

The Validation Dataset

In order to quantitatively analyze the results of our model, we built a validation dataset for two novels, namely IOM. This was prepared considering the processing time required for the complete series. The results were graphical and subjective. The subjective part included the co-occurrence and similarity matrix between the characters of the novel, using which the graphical output was being plotted. In the later stages when the summary has been generated, we have made use of the same graphical representation to validate the summary in terms of preserving the

integrity of the characters. The results also include a few numeric measures for the validation of the obtained summary.

3. Implementation

Step 1 *Name Entity Recognition*

Having no knowledge about the novel, the system needs to follow a series of steps to understand the characters and their relationships. From the related work, we have deduced to use the basic extraction method to recognize and extract character names. In this step, we use a pre-trained Spacy NER Classifier, since the initiation of the Spacy NLP class takes up the load of the memory. The process is carried forward by considering each sentence rather than the entire novel at once and identifying the name entities by removing stop words and common words. Therefore the sentence based processing approach is followed. One important processing done is to split the name into single words and count them as two different characters. According to the Association rules[1] we process the names obtained and discard the names which have occurrences lower than a defined threshold to get rid of unwanted characters.

Step 2 *Character Importance*

From the output of the previous step i.e the character name list, we perform the sentiment analysis based on the related work[3] on each of them by calculating a sentiment score for each of them which also describes the importance of the character in the novel[4]. This is done by using the Scikit-Learn text processing function CountVectorizer after which the top characters are selected based on their scores, here in the testing model we are making use of the top 10 characters.

```
The top 10 Character Names are :
['shiva', 'nandi', 'ayurvati', 'bhadra', 'pakratis', 'meluha', 'gunas', 'chitraangadh', 'srinagar', 'nurses']
Their freequencies are :
[133, 37, 36, 23, 18, 14, 14, 13, 7, 6]
```

Step 3 *Co-occurrence Matrix*

We make use of the concept of co-occurrence[5] and to define the character relationships. We have adopted a simple approach to calculating the co-occurrence score by observing the number of times two character names occur in the same sentence. For this purpose, we use the binary occurrence matrix that provides information on this, then the co-occurrence matrix is calculated by the dot product of the occurrence matrix and its transpose, according to the . We then triangularize the matrix and set them to 0, to eliminate the repeated numbers along the diagonal.

$$X_{cooccur} = X_{occur}{}^{T} \cdot X_{occur}$$

	s1	s2	s3	s4
n1	1	0	0	1
n2	0	0	1	1
n3	1	0	0	1

·

	n1	n2	n3
s1	1	0	1
s2	0	0	0
s3	0	1	0
s4	1	1	1

=

	n1	n2	n3
n1	2	1	2
n2	1	2	1
n3	2	1	2

➡

	n1	n2	n3
n1	0	0	0
n2	1	0	0
n3	2	1	0

The Co-occurrence Matrix generated for the input we used for testing is as follows :

```
Cooccurence Matrix
[[ 0  0  0  0  0  0  0  0  0  0]
 [23  0  0  0  0  0  0  0  0  0]
 [13  0  0  0  0  0  0  0  0  0]
 [ 4  4  0  0  0  0  0  0  0  0]
 [ 3  4  0  0  0  0  0  0  0  0]
 [ 2  1  0  0  0  0  0  0  0  0]
 [ 4  1  0  0  1  1  0  0  0  0]
 [ 1  0  1  0  0  0  0  0  0  0]
 [ 2  2  0  0  1  0  0  0  0  0]
 [ 3  0  0  0  0  0  1  0  0  0]]
```

Step 4 *Document-term matrix*

The features of the document are extracted using the Count Vectorizer function. The term frequency is normalized using TfidfTransformer which uses Term Frequency(TF) and Inverse Document Frequency(IDF) of the corpus. We use the TfidfTransformer library function for this purpose.

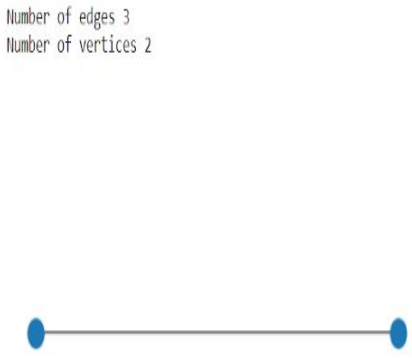
The Document Term Matrix generated for the input we used for testing is as follows :

```
matrix([[1.        , 0.04595628, 0.05875086, ..., 0.        , 0.07447009,
         0.04479651],
        [0.04595628, 1.        , 0.        , ..., 0.        , 0.01137111,
         0.02131267],
        [0.05875086, 0.        , 1.        , ..., 0.        , 0.        ,
         0.        ],
        ...,
        [0.        , 0.        , 0.        , ..., 1.        , 0.        ,
         0.        ],
        [0.07447009, 0.01137111, 0.        , ..., 0.        , 1.        ,
         0.02485713],
        [0.04479651, 0.02131267, 0.        , ..., 0.        , 0.02485713,
         1.        ]])
```

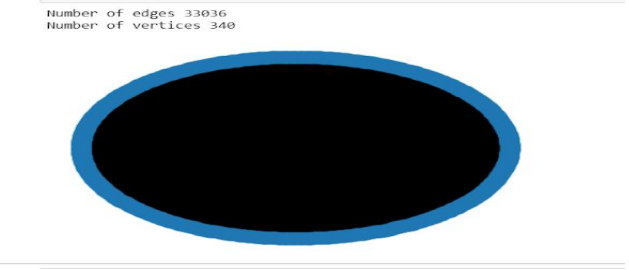
Step 5 *Page Rank*

The NetworkX graph is plotted using the output of the previous step with each node representing a sentence and an edge representing that they have words in common. The edge weight is the number of words that are common in both of the sentences.

The plot of a simple sentence is as follows :

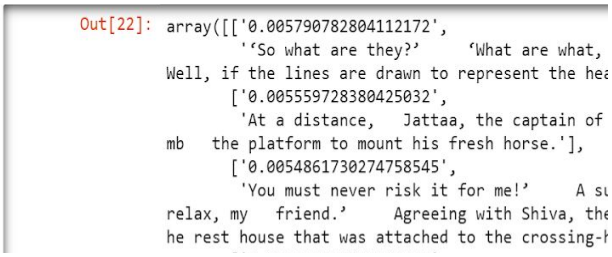


The NetworkX graph for the input we are providing is very dense as follows :



It is very dense because the number of sentences is very high, which is depicted by the number of edges and vertices displayed.

A snap of the ranks along with the sentences :



We use the PageRank function of the NetworkX library to rank the sentences forming a dictionary called ranks with key=node(sentences) and value=textrank (the rank of each of the sentences).

Step 6 *Summarization*

The logic used for summarization[7] is that if all sentences have equal ranks, meaning they are all the same, taking any sentence will give the summary of those lines, and if the sentence has different ranks, the rank is normalized.

Then the threshold is calculated by taking the mean value of all the normalized scores and any sentence with the normalized score 0.2 more than the mean value is considered to be :

threshold = (sum(temp_array) / len(temp_array)) + 0.2

The sentences that satisfy the criteria of having a score above the thresholds are segregated into a list to form the final output i.e the summary of the document.

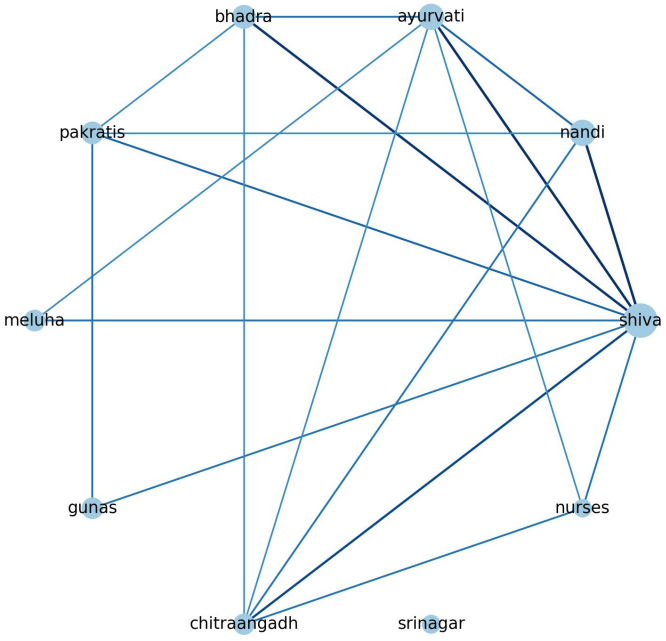
6. Quality Measures

1 *Relationship NetworkX graph*

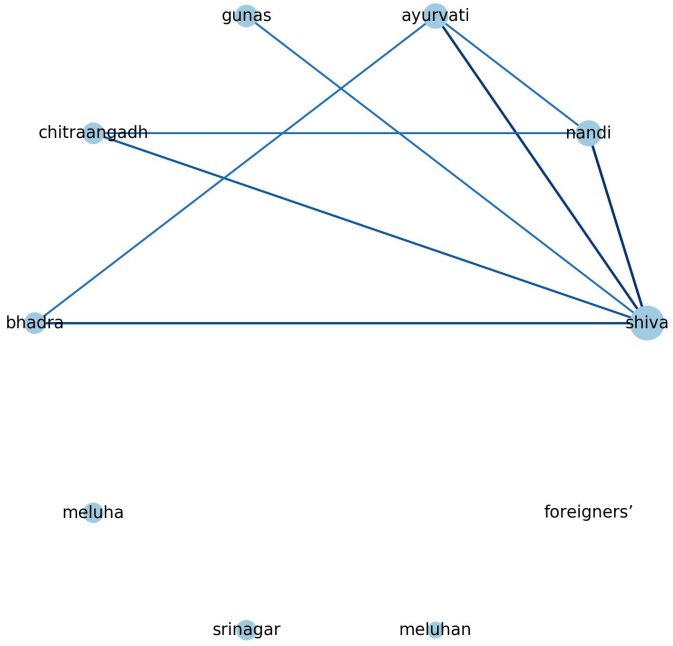
We have the two matrices computed in the previous steps, we can now transform them into the graph parameters and then plot the character relationship graph using the NetworkX library.[8]. In this process, the matrices are first normalized so as to make the magnitude consistent across different novels while keeping the diversity among characters in one novel. These graphs are just to depict the character relationship of the top characters of the novel.

They can be used as a quality measure for the summary by plotting the graph for the summary and checking if the character relationships are preserved.

Below is the Character Relationship graph of the Novel:



Below is the Character Relationship graph of the Summary:



We can see that the thick edges i.e Shiva and Nandi, Shiva and Ayurvathi, Shiva, and Bhadra reserved in the summary too, from this we can conclude that there is no loss of essential characters i.e character's integrity is preserved.

For our model, the percentage similarity of the characters between the novel and the summary is ***66.66666666666666***. We can also notice that the relationship between the important characters depicted by a thicker edge is preserved.

2 *Levenshtein Distance*

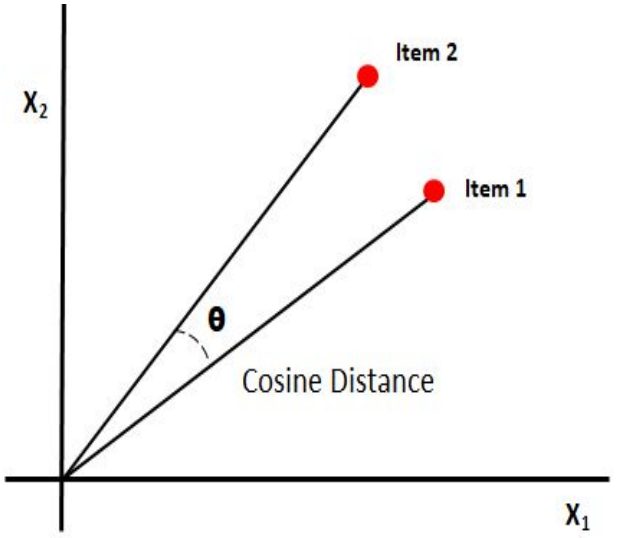
Levenshtein distance (LD) is a measure of the similarity between two strings, the greater the Levenshtein distance[9], the more different the strings are. The Levenshtein module can be imported to calculate the distance.

The Levenshtein distance we obtained is ***30651.***

3 *Cosine Similarity*

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space[10]. Cosine Similarity tends to determine how similar two words or sentence are,so here it determines the dot product between the vectors of two documents/sentences to find the angle and cosine of that angle to derive the similarity.

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$



The cosine similarity we obtained for our model is ***0.95.***

4 *Jaccard Similarity*

The Jaccard index, also known as Intersection over Union and the Jaccard similarity coefficient.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

(If A and B are both empty, define $J(A,B) = 1$.)

The Jaccard Similarity we obtained for our model is ***0.914.***

4 *Document Similarity*

Document similarity is calculated by calculating document distance. Document distance is a concept where words(documents) are treated as vectors and is calculated as the angle between two given document vectors. Document vectors are the frequency of occurrences of words in a given document.

The Document Similarity we obtained in radians for our model is ***0.351.***

```
File IOM.txt :
37656 lines,
6505 words,
1748 distinct words
File summary.txt :
139581 lines,
25078 words,
1709 distinct words
The distance between the documents is:  0.340215 (radians)
```

5 *Co-Selection Measures*

Precision, Recall and F-score

According to the research paper,

"EVALUATION MEASURES FOR TEXT SUMMARIZATION", [11]

Precision (P) is the number of sentences occurring in both systems and ideal summaries divided by the number of sentences in the system summary. Recall (R) is the number of sentences occurring in both systems and ideal summaries divided by the number of sentences in the ideal summary. F-score is a composite measure that combines precision and recall. The basic way how to compute the F-score is to count a harmonic average of precision and recall:

$$F = \frac{2 \cdot P \cdot R}{P + R}.$$

5. Future Work

The challenge we faced during name-entity recognition is that the same character was called with two different names (something like a nickname) and we failed to count them as the same but considered them to be two different characters which affected the quality broadly. We also faced issues in instances where many places where a character's name is replaced by "He" or "She" will not be captured, causing a loss of information.

During the same phase, we failed to not consider the place names as character names since our model could not differentiate between the place names and character names.

For the quality evaluation, we faced issues finding relevant evaluation measures such as accuracy, precision, and recall to portray the accuracy of the summary output.

The name entity recognition accuracy could be improved by covering glitches as mentioned in the

above, solving them could increase the accuracy as a whole.

Summarisation with an added aspect of character relationships would improve the summary quality.

In the model we have presented, for smooth and quick running there is a lot of loss of information encountered at every stage, which can be rectified by including a cloud approach for faster running.

Huge models might take a longer time to run on the model we presented, but the quality measures remain the same across novels.

6. Conclusion

We built a system that was flexible in accepting novels ranging from 10-line short stories to big novels as input. The system would return a graphical representation of the important character relationships aligned with their individual character integrity throughout the story. We made use of multiple NLP techniques and toolkits and also showed that our model works better than the stand-alone toolkits available online.

Our project has a lot of scope for improvement and has the potential to generate a higher accuracy than currently generated. The sentiment analysis could be even better by measuring the individual character's sentiment throughout the novel. Character relationships like father-son, brother-sister, etc. could also be identified along with the character closeness parameters that are calculated in the project.

7. Output

The two outputs included the visualization of the character relationship in a NetworkX graph format and the summary as a text file.

8. Acknowledgment

We would like to thank Dr. Shylaja S S, the Chairperson, Department of Computer Science and Engineering, and Computer Science Department of PES University for giving us the opportunity to work on this project.

10. References

[1]Shaalan, Khaled, and Hafsa Raza. "Person name entity recognition for Arabic." In Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources, pp. 17-24. Association for Computational Linguistics, 2007.

[2]Mansouri, Alireza, Lilly Suriani Affendey, and Ali Mamat. "Named entity recognition approaches." International Journal of Computer Science and Network Security 8, no. 2 (2008): 339-344.

[3]Nalisnick, Eric T., 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), pp. 479-483. 2013. https://cutt.ly/NetworkX

[4]Longstaff, D., Walker, R., Walker, R.F., and Jackway, P., 1995. Improving co-occurrence matrix feature discrimination. In Proc. of DICTA'95, 3 rd International Conference on Digital Image Computing: Techniques and Applications.

[5]https://en.wikipedia.org/wiki/Co-occurrence_matrix

[6]Pennycuff, C. and Weninger, T., 2016. Striations in PageRank-ordered matrices. Social Network Analysis and Mining, 6(1), p.30.

[7]Gupta, V. and Lehal, G.S., 2010. A survey of text summarization extractive techniques. Journal of emerging technologies in web intelligence, 2(3), pp.258-268.

[8]Bihari, A. and Pandia, M.K., 2015, February. Eigenvector centrality and its application in research professionals' relationship network. In 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE) (pp. 510-514). IEEE.

[9]Lavoie, T. and Merlo, E., 2012, June. An accurate estimation of the levenshtein distance using metric trees and manhattan distance. In 2012 6th international workshop on software clones (IWSC) (pp. 1-7). IEEE.

[10]Huang, A., 2008, April. Similarity measures for text document clustering. In Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand (Vol. 4, pp. 9-56).

[11]Steinberger, J. and Ježek, K., 2012. Evaluation measures for text summarization. Computing and Informatics, 28(2), pp.251-275.