Analysis of The Lord of the Rings Book Scripts

Maharan Murshed
Mathematics '21
mmurshed@princeton.edu

Phillip Yoon
Computer Science '20
pyoon@princeton.edu

Hamza Mahmood
Computer Science '20
hmahmood@princeton.edu

Diamond Acharya
Computer Science '21
dacharya@princeton.edu

Abstract

In this paper, we intend to break into the popular series of novels *The Lord of the Rings* and perform interesting supervised and unsupervised learning. We divide the analysis into three broad tasks. The first involves predicting the identity of the character given a random dialogue. The second involves analyzing sentiment arcs using a tool called VADER, and performing LDA to find latent structure. Similarly, the third involves performing network analysis to see the connections between characters. We use different models for each and find that a decision tree classifier works the best for character prediction based on accuracy, precision, recall, and F1-score, LDA generates relevant topics, sentiment analysis tracks the emotional trajectory of the series, and Louvain's method for community detection with a resolution of 0.6 works the best for network analysis.

1 Introduction

LOTR is an epic high fantasy novel series written by the English author and scholar J. R. R. Tolkien [7]. In this paper, we try to analyze the LOTR books using both supervised and unsupervised learning. The motivation for our work is to achieve a better understanding of characters distinctive traits and their mutual relationships in the LOTR series. In addition, the visualization of story arcs often leads to a better understanding of a plot, especially for more complex storylines, such as LOTR. Finally, we are able to use our methods to compare different methods for text analysis.

First, we try to predict the identity of the characters given the dialogue. For this we process the data accordingly and compare the performances of different classifiers. Second, we perform sentiment analysis on the book texts and LDA on both the texts and dialogue. Third, we do social network analysis. This involves finding out which character in the book is connected to which character and which particular pairs have more connections. We use the book texts for the second and the third task, whereas for character prediction from dialogue, we use movie scripts to make data processing easier. We acquire data by parsing text files of the book scripts and the movie screenplays.

2 Related Work

There have been many interesting machine learning analyses on popular novels and book series. For example, in the github repository: *https://github.com/mathbeveridge/gameofthrones*, the author has done network analysis on the characters from the popular fantasy series The Game of Thrones. Analysis have also been done to find out emotional arcs in literary works [2]. For this, the progression of the movies is plotted on the horizontal axis and the emotional level is plotted on the vertical axis. The 30th Nov, 2016 article in the World Economic Forum titled *Combining machine learning*

and Romeo and Juliet to explain what makes a good story notes that the majority of the popular literary works follow the same six emotional arcs in their stories [2]. There have also been projects on developing different character-based natural language models [5]. We will extend this work in order to determine if we can predict the character based on their languages, as well as determine if there are any features that are strongly indicative of a specific character.

3 Character Prediction from Dialogue

In this section, given a sentence of dialogue, we predict the identity of the character who spoke that dialogue. We do this using various methods of data engineering, feature extraction, and feature selection. We also test a variety of classifiers and then analyze the results. For this section, we utilize the movie scripts from the corresponding LOTR movies so that the dialogue is easily parsable.

3.1 Methods

After parsing the movie script to classify each sentence of dialogue with its speaker, we only consider the characters that have at least 100 sentences of spoken dialogue for the purpose of classification. This was done to prevent an imbalanced dataset, where some characters have significantly more dialogue than others. This reduced the number of characters to 7: Gandalf, Frodo, Sam, Aragorn, Bilbo, Pippin, and Gollum. We then used the Python NLTK library to tokenize, convert to lower case, remove stop words, and stem each word using the Porter stemming method. The resulting vocabulary contained 1268 words.

To extract features from the dataset, we first remove all stopwords from our reviews, as this has been shown to improve performance [14]. We then consider all 1-grams, 2-grams, and 3-grams of each training example in the dataset. Considering all various n-grams, we end up with a vocabulary that contains 5811 total words. Therefore, for each sentence, we have a feature vector of size 5811. We converted these features to the bag-of-words representation of the examples where each entry contains the number of occurrences of a specific n-gram in that example. We used the identity of the speaker that spoke each sentence of dialogue as its label. Therefore, our problem is as follows: Given X, an $n \times 5811$ matrix representing the bag of words of dialogue, predict Y a vector of size n that represents the speaker of each of row of X, where n is the number of sentences of dialogue.

In this work, we also consider the effect of feature selection on the performance of our classifiers. In feature selection, we select a subset of extracted features to train a model, reducing the dimension of the input space. Some advantages are that feature selection reduces computation time, may avoid overfitting, and reduces the overall model complexity. We assign a value to each feature according to some scoring function, and then select the K features with the highest values.

We examined the effectiveness of six types of classifiers from the scikit-learn Python libraries [11] and TensorFlow [13]:

1. Naive Bayes, with Multinomial implementation
2. Logistic Regression, with l_2 penalty
3. Support Vector Machine, with Linear kernel
4. Decision Tree Classifier
5. K-Nearest Neighbors Classifier, with K = 46
6. Neural Network, with Dense(512), Dense(256), Dropout(0.2), Dense(128), Dropout(0.2),Dense(64), Dense(7)

The hyperparameters of these models were tuned individually, but we have not shown these results because of space restrictions. For example, we experimented on the architecture of the neural network.

3.2 Results

We consider the Chi-squared statistic as a method for feature selection. This statistic tests the independence of two categorical variables: a feature variable and the class variable. This statistic detects

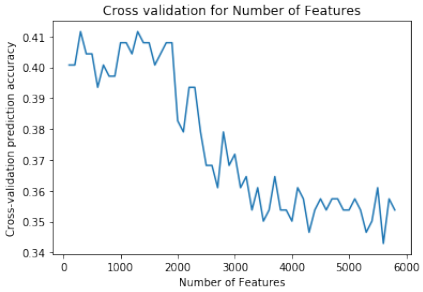

Figure 1: Cross-validation for number of features

Classifier	Accuracy	Precision	Recall	F1-Score
NB	0.3971	0.4978	0.3971	0.3179
LR	0.3827	0.4515	0.3827	0.3174
SVM	0.3791	0.4204	0.3791	0.3250
DT	**0.4116**	**0.5196**	**0.4116**	**0.3676**
K-NN	0.3755	0.4392	0.3755	0.30782
NN	**0.4116**	0.4549	**0.4116**	0.3598

Figure 2: Results with Chi-Squared Feature Selection (K=300)

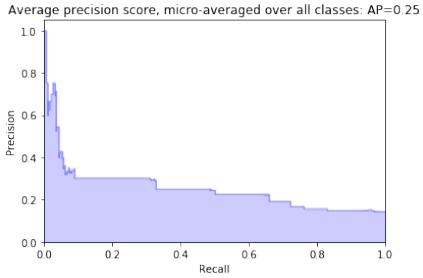

Figure 3: Average precision-recall curve

Classifier	Accuracy	Precision	Recall	F1-Score
NB	0.3827	0.4857	0.3827	0.3056
LR	0.3646	0.4130	0.3646	0.3017
SVM	0.3863	0.4162	0.3863	0.3341
DT	**0.4043**	**0.5228**	**0.4043**	**0.3588**
K-NN	0.3538	0.3748	0.3538	0.2887
NN	0.3755	0.4198	0.3755	0.3369

Figure 4: Results with ANOVA F-Value Feature Selection (K=300)

features that are likely to be independent of the class, and therefore, irrelevant for the purposes of classification. We use cross-validation to determine the optimal value of for K, the results of which are displayed in 3.2.

Using this method of feature selection, the results are displayed in Figure 2. The top 10 features selected were [‘good year’, ‘going sit’, ‘like lad extra’, ‘look mr frodo’, ‘going told’, ‘eleven years old’, ‘delaying old ag’, ‘thousand enough break’, ‘beacons minas tirith’, ‘accounted lost’]

We also consider the ANOVA F-value as a method for feature selection. This value tests whether or not a group of variables are jointly significant. We use cross-validation to determine the optimal value of for K. Using Using this method of feature selection, the results are displayed in Figure 4. The top 10 features selected were [‘stop’, ‘dungeon’, ‘good year’, ‘going told’, ‘look mr frodo’, ‘beacons minas tirith’, ‘demon ancient world’, ‘eleven years old’, ‘delaying old ag’, ‘accounted lost’].

Figure 3.2 displays the average Precision-Recall curves for the performance of our predictions with Chi-Squared Feature Selection (K=300) and using the Decision Tree classifier. Figure 3.2 displays the Precision-Recall curves for each individual class, i.e. each character. Precision-Recall curves were used as opposed to an ROC curve because the classes are inbalanced.

3.3 Discussion

We found that performing feature selection and choosing the top 300 features, which is 5.16% of the original feature set size, gave improved performance in each classifier, upon comparing to performance with no feature selection. Because the performance rapidly decreases in accuracy after the 2000 feature mark, this shows that the remaining features are generally not as important because including them does not improve our accuracy. In fact, including more features than necessary results in overfitting and reduces performance by considering too many features. Therefore, this suggests that only 2000 features are actually relevant for the purpose of classification, and 300 features is sufficient. The feature selection was also robust, as both Chi-Squared and ANOVA F-

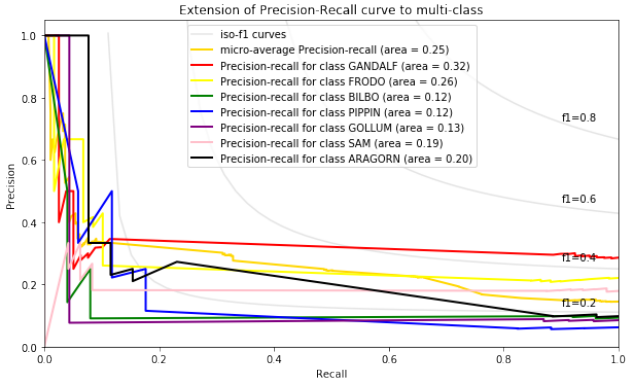

Figure 5: Precision-recall curve for each individual character/class

Value feature vector both gave very similar top ten features. The top features were also reasonable; the 'look mr frodo' feature is a very strong indicator for Sam, who says 'Look, Mr Frodo! We're almost there!' and 'eleven years old' is a very strong indicator for Gandalf, who says 'One hundred and eleven years old, who would believe it!' However, this strong correlation sometimes results in misclassification. When other characters spoke 'Mr. Frodo,' they were automatically classified as Sam because Sam says it so often.

The Decision Tree classifier consistently had the best performance for all metrics, accuracy, precision, recall, and F1-score after feature selection, which performed better than our baseline of Naive Bayes. One of the advantages of decision trees are that they focuses on the relationship among various events [10]. This would be helpful for our purposes because it would capture the relationships between words, which is especially pertinent when determining the speaker of a dialogue. Therefore, we see that Decision Tree is the best model when using feature selection. Our baseline of Naive Bayes might not have worked as well because it relies on the assumption that all features are independent. We violated this assumption when we took multiple k-grams that combined phrases together. The existence of some words would likely be dependent on the presence of other words. Meanwhile, K-Nearest Neighbors consistently had the lowest accuracy. This is probably because K-Nearest Neighbors is mostly used for clustering as opposed to classification, and most likely cannot model the intricacies needed for classifying dialogue by its speaker.

We wanted to determine the success of the model in classifying various speakers. From our results in Figure 3.2, we see that Gandalf, Frodo, and Sam had the highest performance. We see this because the area under the Precision-Recall curves for each of these characters was higher than the others. This was probably because they had the highest representation in the dataset, and as a result, our model had more training data to classify their data. If we had more data for the other speakers, we most likely could have had greater performance for them. We also see that recall is equal to accuracy for all of our results. In multiclass and multilabel classification task, the notions of precision, recall, and F-measures can be applied to each label independently [11]. This is expected behavior according to the documentation, in which the weighted recall is equal to accuracy.

4 Sentiment Analysis

In the following two sections (Sentiment Analysis, LDA), we hope to address the question of what we can learn about the Lord of the Ring books without having read them.

First, using a tool called VADER (Valence Aware Dictionary and Sentiment Reasoner), we analyze the intensity of the Lord of the Rings book texts [6]. The tool is defined as a lexicon and rule-based sentiment analysis tool. Specifically, the authors of the tool "first constructed a list of lexical features correlated with sentiment and then combined the list with some rules that describe how the grammatical structure of a phrase will intensify or diminish the sentiment"[12]. It was primarily

developed for the use of analyzing social media; but, it has been applied to other domains such as analyzing large novels like the Harry Potter series [12].

4.1 Methods

The text for all 3 books consisted of about 470,000 words, and 30,000 sentences. VADER needs as input raw sentences since it takes into account the grammatical construction of sentences. Then, the analyzer gives each sentence a negative, positive, and neutral scores.

So, in order to analyze the Lord of the Rings, we first broke the text down into individual sentences. Then, passing in this list of sentences into the analyzer, we obtained the average compound sentiment for each sentence; the compound sentiment is the average of the positive, negative, and neutral sentiments. Finally, we plotted the average compound sentiments using a moving average of 1000 sentences. 1000 sentences was chosen as we assumed each sentence is about 15 words, and so, 1000 sentences is 15000 words which is about the average length of a chapter [1].

4.2 Results

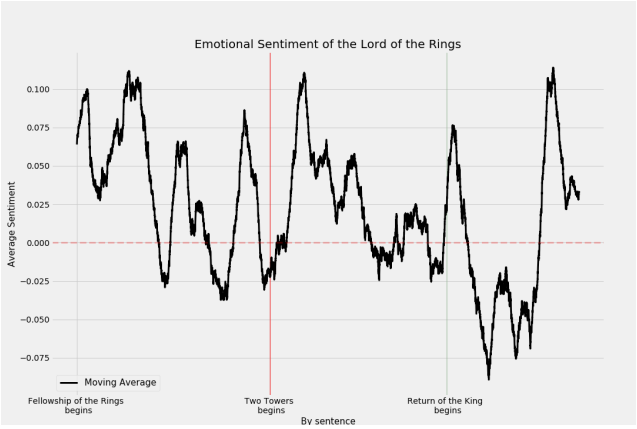

Figure 6: Intensity of book text

We indicate where each book begins; it ends when the next one begins. From this, we can gain insight into the sentiments associated with each book's beginning and ending.

As depicted, the story involves many "spikes" indicating regular periods of intense sentiment. This was also typical of the related work of Harry Potter. Perhaps these spikes are what keep the audience engaged with such a best-selling novel and movie series.

5 LDA

We performed Latent Dirichlet Allocation in order to discover the hidden structure within both the book text and movie script dialogue. LDA consists of generating topics from documents where documents were the individual sentences of dialogue / a chapter length of words.

5.1 Methods

5.1.1 Data preprocessing

For the movie scripts, we made a list of the sentences. From this, we made a dictionary consisting of an index for each word, and then a bag of words representation to model the distribution of words for each dialogue sentence.

Likewise, for the book script, which included both dialogue and narrative, we made a list in which each sublist had 15,000 word chunks. 15,000 words was about how long an average chapter from the trilogy is [1]. We additionally removed stopwords.

5.2 Results

We performed LDA using the gensim libary's implementation of LDA [4]. We ran LDA with the number of components (topics) set to 5.

Latent Topic	Words
0	must, sam, take, us, ring, frodo, back, oh, time, men
1	go, gandalf, frodo, let, never, mordor, precious, come, ring, well
2	frodo, ring, come, mr, one, bilbo, get, death, gandalg, power
3	know, would, us, look, smeagol, cannot, going, shire, frodo
4	one, ring, see, mean, aragorn, kill, things, time, could, tell

Table 1: LDA results on the movie scripts

5.3 Discussion

We found that commonly topics discuss "ring", and "frodo" or other characters. This could indicate that the series revolves around the characters' relationship with the ring, which was expected. However, at the same time, having topics with the same or quite similar words proves uninformative to our understanding of other themes in the book.

Moreover, since LDA on the book text and movie scripts produced similar topics, this is a good indication that the movie follows the books' trajectory, as it should.

Ultimately, further analysis of coupling LDA with sentiment analysis could reveal how the book's plot involving the ring and the characters' experience results in the emotional sentiment graph in the previous section. Specifically, we could indicate what happened at each peak and trough on the plot.

6 Network Analysis

In this section, the paper will analyze the network dynamics between the characters of the Lord of the Rings using the books as the source material. It then go on to perform measurements of centrality to see which characters are the most connected, bridge the most people, and are most influential. Finally, community detection will be performed to see what 'communities' can be found using the appropriate methods.

6.1 Methods

6.1.1 Pre-processing

First, the full text of the three (six, depending how you want to look at it) LOTR books was used as the source text. 30 characters, chosen from the LOTR wiki, were selected as our characters to work with. A note should be made on Treebeard/Fangorn which are two names to refer to the same person. We decided to count them separately because it bore out some interesting results as to who uses which name. The algorithm for getting the data worked as such:

1. A moving 25-word window (that moves one word at a time) is created
2. Only the last word is of concern, if it's a character, then
3. For all the 24 others words in the window, if any of them are a character alphabetically lower than the character in question, then add an edge to their relationship
4. If any two characters who've already been given an edge, have another one, the weight of the edge is increased by one
5. This continues until the end of the text file is reached

6.1.2 Methods Proper

For the network graph, an undirected graph was created using packages Networkx[8] and Plotly[9]. The nodes are to be placed using the Kamada-Kawai algorithm. The nodes then have two prop-

erties: their connections and colour. The colour of the nodes will be determined how many other nodes they're connected to. The lines between any node and another one will have a thickness commensurate (linearly) with the weight of the edge between those two nodes.
For centrality measures, three will be used: Centrality, Betweeness, and PageRank. We will perform this measure on the network of characters to see which characters rank highest on each metric of centrality, both with and without edge weights. Finally, community detection will be performed using the Louvain method (with Community[3]) at varying degrees of resolution.

6.2 Results

6.2.1 Network Graph

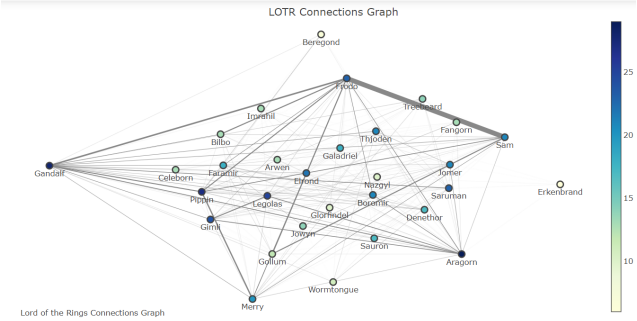

Figure 7: Connections Map of LOTR Characters

6.2.2 Centrality Measures

Ranking	Degree	Degree w	Betweeness	Betweeness w	PageRank	PageRank w
1	Aragorn	Frodo	Aragorn	Arwen	Aragorn	Frodo
2	Gandalf	Gandalf	Gandalf	Thjoden	Gandalf	Gandalf
3	Pippin	Sam	Pippin	Galadriel	Pippin	Sam
4	Legolas	Aragorn	Legolas	Jowyn	Legolas	Aragorn
5	Gimli	Pippin	Frodo	Celeborn	Gimli	Pippin

Table 2: Top 10 Characters with Regards to Various Measures of Centrality. w = weighted

6.2.3 Community Detection

Lone Warrior	Aragorn				
Elves	Arwen	Celeborn	Elrond	Galadriel	Glorfindel
Men	Beregond	Denethor	Faramir		
Bilbo	Bilbo				
Boromir	Boromir				
Rohan	Erkenbrand	Jomer	Jowyn	Imrahil	Thjoden
Fanhorn/Treebeard	Fangorn	Treebeard			
Frodo	Frodo				
Gandalf	Gandalf				
Love-Hate duo	Gimli	Legolas			
Hate Each other	Gollum	Sam			
Duo	Merry	Pippin			
Evil	Nazgyl	Sauron			
Evil's+	Saruman	Wormtongue			

Table 3: Louvain method, at resolution = 0.3, group-names on left-most column

Warriors	Aragorn	Boromir	Fangorn	Gimli	Legolas
Elves	Arwen	Celeborn	Elrond	Galadriel	Glorfindel
Men	Beregond	Denethor	Faramir		
Frodo and company	Bilbo	Frodo	Gollum	Sam	
Rohan	Erkenbrand	Jomer	Jowyn	Imrahil	Thjoden
Baddies and the guy they hate	Gandalf	Nazgyl	Saruman	Sauron	Wormtongue
Duo with Treebeard	Merry	Pippin	Treebeard		

Table 4: Louvain method, at resolution = 0.6, group-names on left-most column

6.3 Discussion

On the Network Analysis, it's interesting to note that a few characters seem to be mentioned a lot. In particular, Aragorn, Gandalf, and Frodo. Some Characters have a diverse range of connections, like Gandalf and Aragorn, others seem to have a lot of connections but only with certain people, like Sam's connection with Frodo.

On centrality, as perhaps seen in the previous network graph, Aragorn, Frodo, and Gandalf rank pretty highly in most measures, though Frodo seems to do worse than expected for a main character until the weights come into place. Aragorn and Gandalf both do well in most metrics, likely due to their extensive connections with most characters. Of interest is the column of weighted betweeness centrality, which differs from unweighted betweeness centrality principally in that characters' betweeness centrality also takes into account their total interactions, so characters who are more bridge-like per amount of interactions would rank higher in this. In this column, the elves seem to do particularly well, perhaps because their relatively fewer interactions take place with more people, and that Arwin and Galadriel have discussions with most of the main cast separately.

Finally, with regards to community detection, the communities spat out with resolution = 0.6 make a lot of sense. Of particular interest is the change in potential communities from when resolution is 0.3 to when it becomes 0.6. The group 'Elves', 'Men', and 'Rohan' remain virtually unchanged, and all the single person groups are removed. Evil, Evil+ and Gandalf are made into a new group, probably because all the bad guys mention Gandalf a lot and he always worries about them. The group Treebeard/Fangorn is broken up, with Treebeard going to the group with Merry and Pippin who mention him by his nickname, and Fangorn going to the group of warriors (which also consists of the Love/Hate Duo, Aragorn, and Boromir) who mention him by his more formal name. Finally, a group with Frodo and his various companions is made.

One possible suggestion for future research would be to combine the network graph with sentiment analysis. Specifically, it would be envisioned that the lines (or more formally, edges) between two nodes would be given a colour representing the sentiment/mood between the two characters. So, for example, though Gandalf has a decent connection with both Sauron and Saruman as well as with the other 'good guys', the former relations would be marked in a negative colour whereas the latter ones would be marked in a positive one.

7 Future Work

There are many ways we can further extend our project. We can apply the methods we used to analyze other literary texts as well. Prediction of events like character deaths and other plot events in ongoing series is also another avenue we could look at. Network connection combined with sentiment analysis can also be done to find out if characters have positive or negative connections between them.

References

[1] Chapter lengths. http://lotrproject.com/statistics/books/chapters.

[2] Combining machine learning and romeo and juliet to explain what makes a good story. https://www.weforum.org/agenda/2016/11/combining-machine-learning-and-romeo-and-juliet-to-explain-what-makes-a-good-story/.

[3] Community/python-louvain. https://python-louvain.readthedocs.io/en/latest/.

[4] gensim. https://radimrehurek.com/gensim/models/ldamodel.html.

[5] How to develop a character-based neural language model. https://blog.usejournal.com/how-to-develop-a-character-based-neural-language-model-99c18de1d4d2.

[6] Hutto, c.j. gilbert, e.e.. vader: A parsimonious rule-based model for sentiment analysis of social media text. eighth international conference on weblogs and social media (icwsm-14). ann arbor, mi, june 2014. https://github.com/cjhutto/vaderSentiment.

[7] The lord of the rings. https://en.wikipedia.org/wiki/The_Lord_of_the_Rings.

[8] Networkx. https://networkx.github.io/.

[9] Plotly. https://plot.ly/python/.

[10] A review of decision tree analysis advantages. https://www.brighthubpm.com/project-planning/106000-advantages-of-decision-tree-analysis/.

[11] scikit-learn: Machine learning in python. https://scikit-learn.org/stable/.

[12] Sentiment analysis on the texts of harry potter. https://towardsdatascience.com/basic-nlp-on-the-texts-of-harry-potter-sentiment-analysis-1b474b13651d.

[13] Tensorflow. https://www.tensorflow.org/.

[14] Tim O'Keefe and Irena Koprinska. Feature selection and weighting methods in sentiment analysis. 01 2009.