Final Project Report

STAT 231: BlueCookie B8

Cuong and Justin

Last updated 05/21/2022

Introduction

Have you ever read a book series that spanned almost a decade and wondered how the books developed over this time period? Is there a visible change in the series? For this project, we will use text analysis skills to explore the Harry Potter franchise. We will be using text analysis because text analysis is a relatively large field that we only briefly touched upon. There are many different applications of text analysis in today's rapidly growing media landscape due to the increase of text content that is being put into the world. In fact, text analysis is an active research topic which has many applications in a variety of industries. This project will demonstrate some basic as well as some more advanced utilizations of text analysis. We chose to use the Harry Potter franchise for this purpose because it is a series that many young adults are familiar with which makes the analysis more interesting for the audience. For any unfamiliar with the series, it is a series of seven fantasy books written by J.K. Rowling about a young wizard navigating a world of magic with his friends as they face danger and the perils of growing up. As discussed above, we will be analyzing all seven books in the Harry Potter franchise. Specifically, we want to look into a few different aspects of these books:

1. How has the word choice evolved over the series of the books?
2. How has the tone of the books changed?
3. Has the author's writing style also changed throughout the time writing this franchise?
4. Without knowing the books beforehand, can we classify every chapter to its corresponding book? (Classification)
5. What are the common themes and topics that are prevalent in each book?

Data

Our data are the Harry Potter novels. The data source that we will be using for this project is the R package `bradleyboehmke/harrypotter` which has been used in the community for various educational purposes, so we will also be using this package. This package contains the Harry Potter novels that we are able to import into R once we install the package.Bradley Boehmke was the person who collected all seven books in the Harry Potter series by scraping the books from an a website that displayed the books. The package was created in 2016 so we assume that he collected the data in 2016. We chose to use the `harrypotter` package because it was created specifically to analyze the Harry Potter books, which is our topic of interest. From the package, the data were separated by book and contained the whole book in text format.

In order to prepare the data for our visuals, we had to perform some initial wrangling. This proved to be rather challenging as we not only had to remove the stopwords, we also had to identify and remove some high frequency words that don't add any value to the analysis. For instance, since the novels have story lines, they contain many names over and over again. However, this doesn't add any insight to our analysis, so we removed many of the common names in the books. For example, in the wrangling below, we removed a number of names that appeared frequently and across many books. We did keep certain names in such as "Gilderoy Lockhart" because he only significantly appears in "The Chamber of Secrets."

Below is part of wrangling code that we think is new and other peers should see. There are two major parts: The first half takes all seven books, which are arrays in which each value in the array is a chapter, and tokenizes each chapter into words. Although we have seen this in class, we think the integration of a for-loop and the new use of an index make this code notable. The second half wrangles the data into a format that is compatible and friendly with the `cast_dtm()` function from the `topicmodel` package. We also want to show readers the common words that we decided to remove since it would give them a clear sense of why some words are not in the analysis.

```
# popular names that we removed for some of the analysis
pop_names <- c("harry", "potter", "hermione", "ron", "dumbledore", "hagrid",
               "snape", "malfoy", "voldemort", "vernon", "weasley", "dudley",
               "lupin", "sirius", "harry's", "umbridge", "mcgonagall",
               "dobby", "aunt", "professor", "madam", "uncle", "madame", "rita")


series <- tibble()
for(i in seq_along(titles)) {
  temp <- tibble(chapter = seq_along(books[[i]]),
                 text = books[[i]]) %>%
    unnest_tokens(token = "words", output = word, input = text) %>%
    # we tokenize each chapter into words
    mutate(book = titles[i]) %>%
    select(book, everything())
  series <- rbind(series, temp)
}
# set factor to keep books in order
series$book <- factor(series$book, levels = rev(titles))

# wrangling for LDA
`%notin%` <- Negate(`%in%`)
# Create word_counts data frame, aggregating word count for each chapter for
# all books
word_counts <- series %>%
  unite(document, book, chapter) %>%
  anti_join(stop_words) %>%
  count(document, word, sort = TRUE) %>%
```

```
  filter(word %notin% pop_words)
```

The rest of our wrangling is contained in a separate wrangling file "wrangling.R". If a reader wants to follow up on our work, we have attached a link to bradleyboehmke's git repository in our References section.

Methods

The first visualization that we did was a creating bigram word cloud that contained the top 10 words from each of the books and the overall series. We accomplished this by extracting bigrams from the novel instead of singular words and then cleaning the bigrams up by removing the stop words and the popular names. From this, we were able to generate word clouds.

Next, we created a graphic to show how the tone of the books has changed over the franchise. We used the BING lexicon to accomplish this. The BING lexicon characterizes words as either being positive or negative. We matched the words that are contained in the lexicon and then removed the words that were not contained. Thus, words without any sentiment value were discarded. Stop words were also removed. In order to create an index on which to scale the sentiment, we formed groups of 100 words and found the aggregate total of the group (number of positive words minus the number of negative words). Then we divided this number by 100 in order to scale the range of sentiment between negative one and one. Some of this code is shown below although the rest of it is contained within the wrangling file.

```
# remove stop words
sentiment <- series %>%
  anti_join(stop_words)

lexicon <- get_sentiments("bing")

# join the two datasets and create the score for each chunk
sentiment_analysis <- sentiment %>%
  inner_join(lexicon, by = "word") %>%
  mutate(id = 1:n(), index = id %/% 100) %>%
  count(book, index = index, sentiment) %>%
  pivot_wider(names_from = sentiment, values_from = n, values_fill = 0) %>%
  mutate(sentiment = (positive - negative)/100,
         book = factor(book, levels = titles)) %>%
  arrange(index)

# calculate the proportion of words that were included in this analysis
sentiment_lexicon <- sentiment %>%
  inner_join(lexicon, by = "word")
nrow(sentiment_lexicon)/nrow(sentiment)
```

The visual was created using stacked bar plots so that we can see the change over time using color. As we can see, the later books have a larger number of words, so there are more 100-word chunks. However, we use the same bar to represent the whole book to easier depict the changes in tone at the start and end of each book.

We also created a shiny app in which the user is able to able to search for any word and see how many times the word appears over the franchise and where. As shown below, each bar depicts a different chapter from the novels with the frequency being the number of times that the chose word appears in that chapter. A trend line was included to help visualize the patterns of specific words. The bars are arranged starting with the first chapter of the first book and ending with the last chapter of the last book. These initial visualizations will help us answer the first three of our questions of interest.

To address questions 4 and 5, we needed to go beyond: We implemented a model of unsupervised learning into our text analysis. Specifically, we used LDA (Latent Dirichlet Allocation), which is a very popular method of topic model. LDA helped us do two things: Classification and Genre-Per-Book.

Initially, we wanted to use this method of topic modeling to classify different topics/themes within a book. In this case, that means performing LDA on each Harry Potter book, and repeating that process seven times. However, during our self-researching period, we realized that LDA can also be used (and might be better

used) to classify underlying topics over a corpus. Therefore, we also ran LDA on all seven books at once to see if it can correctly classify those books.

Regarding how we went beyond: First of all, we taught ourselves the basic concept of LDA (the 3 links we studied from are listed in the References section). Then, we tried to find an R package that can be used to perform topic modeling. We found 2 functional and community-approved packages: `topicmodels` and `lda`, and we decided to use the former because it is more up to date and there is a vignette on how to use it. Lastly, as an attempt to solidify our understanding of LDA, we read through 3 scientific papers (also listed in the "References" section). However, it appeared that we aren't well-trained enough to comprehend everything written in those papers. Therefore, we deferred our goal of understanding how topic modeling works to a later time, and focused on knowing how to execute the functions and finish one cycle of analysis.

Below are the codes that we have decided to show since they include many of the new concepts.

```
chapters_dtm <- word_counts %>%
  cast_dtm(document, word, n)
```

This chunk takes a data frame and turn it into a document-term matrix (hence `cast_dtm`), the only format that is compatible to the function `lda()` from this package. Without going into too much detail, a document-term matrix has rows as unique documents and columns as unique words. So, if a word from document 1 doesn't appear in document 2, the corresponding entry will be empty.

```
chapters_lda <- LDA(chapters_dtm, k = 7, control = list(seed = 8))
```

Essentially, the function `LDA()` does all the heavy lifting, and is the most important line of all the LDA analysis. It takes a document-term matrix, the number of topics that a user pre-decides, and a seed to guarantee reproducibility. Then, we can call other functions to make sense of this output.

```
chapter_topics <- tidy(chapters_lda, matrix = "beta")
```

This line examines the per-topic-per-word probabilities. In other words, it calculates the probability of a term being generated from a specific topic.

```
chapters_gamma <- tidy(chapters_lda, matrix = "gamma")
```

This line examines the per-document classification. In other words, it estimates the proportion of words from a document that are generated from a specific topic

The 4 chunks above are the general framework of a LDA analysis. Of course, we used other visualizations to enhance our understanding, but we believe that our peers already know how to do the unmentioned wranglings and visualizations.

If the readers want to explore Latent Dirichlet Allocation further, they can find all the resources that we used in the References section.

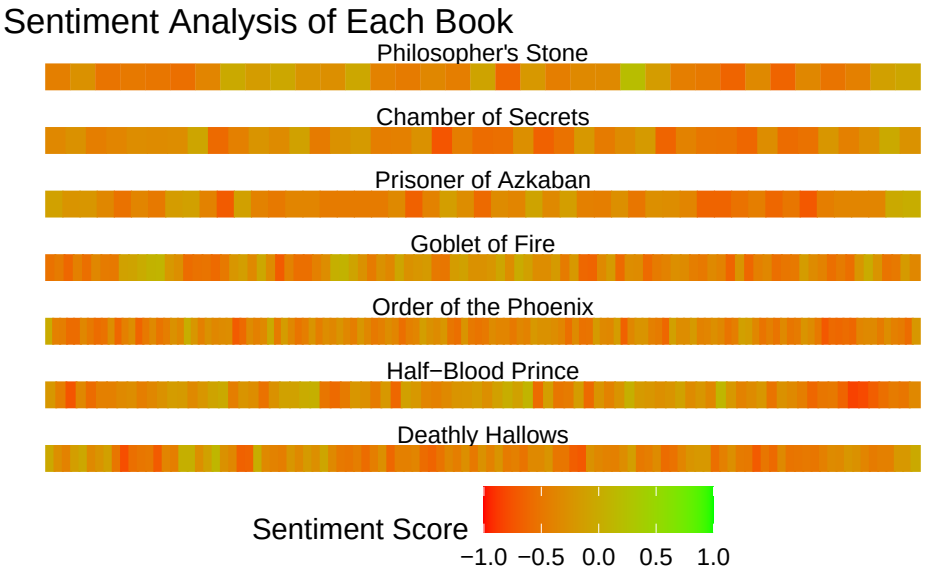
Results

Bigram Cloud

From the bigram clouds below, we are able to see the most popular bigrams for each of the books. We are able to see that each book has popular bigrams that are specific to the book as well as bigrams that are contained in other books. For example, we can see that the bigram, "Triwizard Tournament" is specific to "The Goblet of Fire", but the bigram, "Death Eaters" is contained in multiple of the word clouds (Goblet of Fire, Order of the Phoenix, Half-Blood Prince, and Deathly Hallows). By looking at the word cloud for bigrams that are popular over the whole series we are able to see many of the bigrams that occur in multiple words clouds. Along with "Death Eaters" are bigrams such as "Invisibility Cloak" or "Daily Prophet." It is interesting to see some of the bigrams that are important to each of the novels and how there are underlying elements that stay constant through the series.

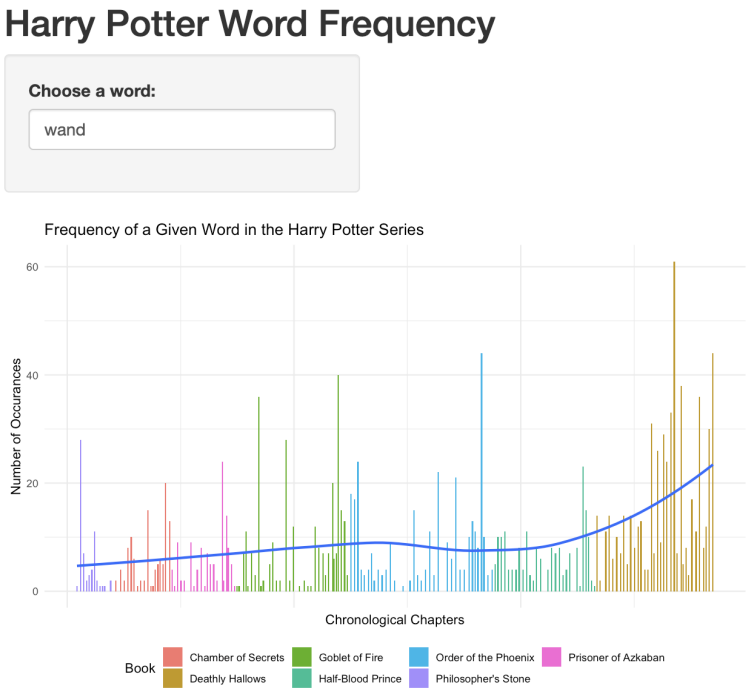
Sentiment Analysis

From our sentiment analysis below, we are able to see that overall, the general sentiment of the series ranges from neutral to fairly negative. As described above, the books tend to have a larger number of words as the series progresses so there are a greater number of "chunks" that make up each book. For most of the books, we can see that the beginning and the end of each book are relatively neutral while the middle of the books being made up of negative and neutral sections. For example, in the "Half-Blood Prince," there are patches of neutral sentiment and negative sentiment that are interwoven until there is group of chunks near the end that have a greatly negative sentiment score. From this visual, it appears that the tone of the books appears to stay fairly consistent through the series.



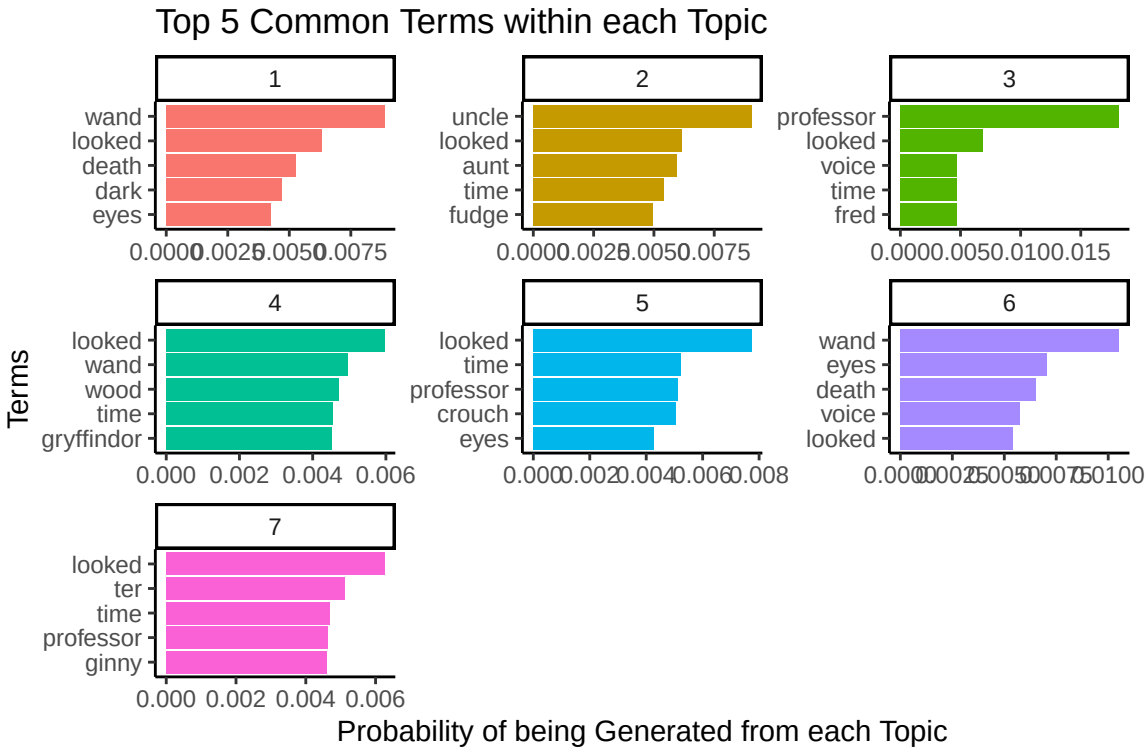
Shiny App

From the Shiny App (which can be found at the following link: https://r.amherst.edu/apps/jpapagelis24/final-project231/), we are able to see how the trend of a certain word appears in the series. Since this aspect of the project is user-input, the conclusions that can be taken from it may differ. However, when we use the word, "wand" (since the book *is* about magic), it appears that there is an upward trend in the use of the word per chapter. This can be done for any word that the user chooses.

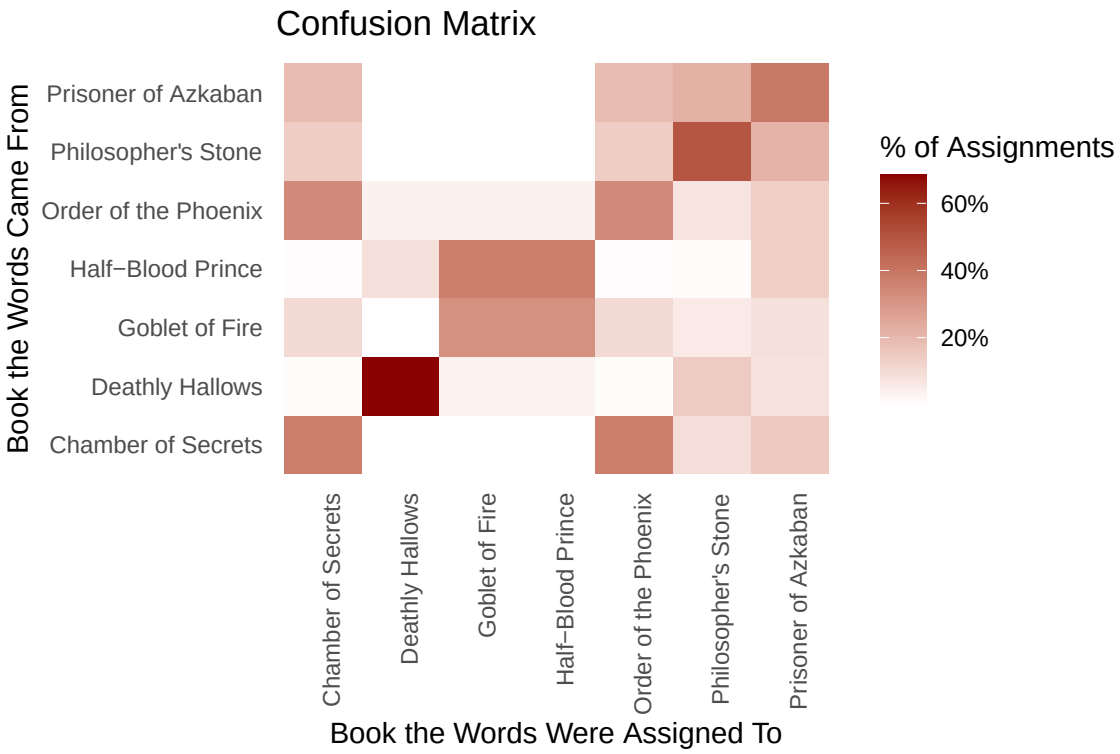


LDA

For the Latent Dirichlet Allocation classification, we found that the model, in both cases, had trouble finding very distinct topics. Specifically, for our first analysis, where we want to see if the model can successfully classify a chapter (document) to its corresponding book (topic), we saw that some words are dominantly common among all topics:

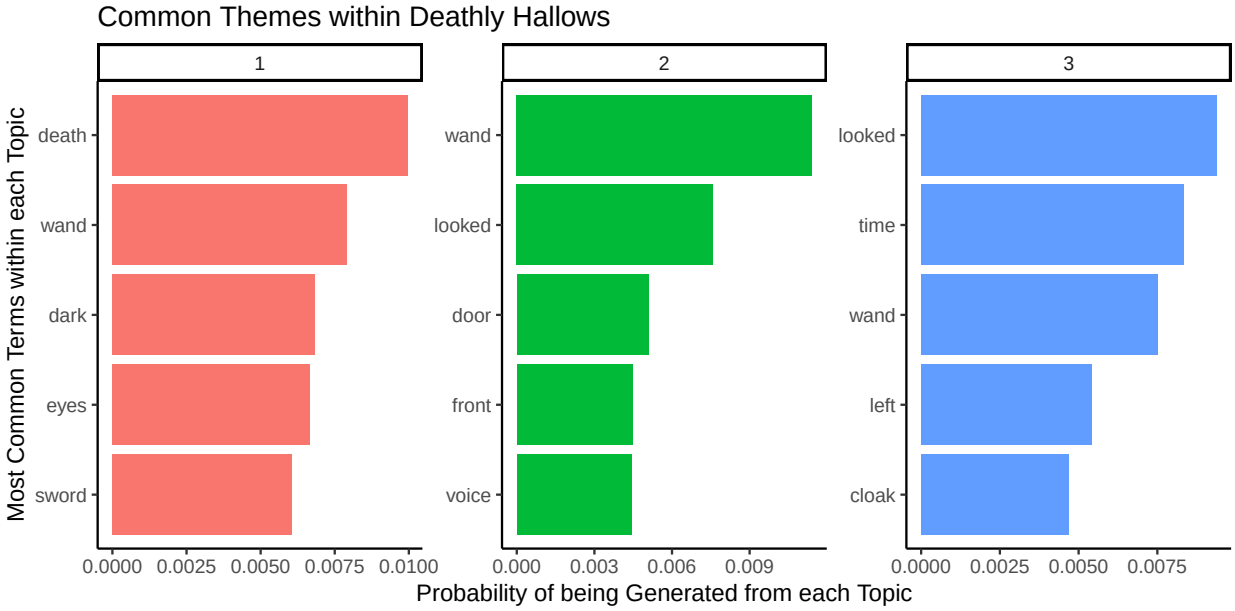


For instance, the word "looked" is consistently in the top 5 most common words among all seven topics. Furthermore, it is the most common word for topics 4, 5, and 7. This implies that the topics are not strongly disjointed. As a consequence, the model couldn't classify the books correctly.



Here, we can see that at best, the model assigned words coming from Deathly Hallows to the Deathly Hallows topic (implying that the identification is correct) 60% of the time, which is significantly better than random guessing. On the other extreme end of the spectrum, the model assigned words from Goblet of Fire to the Goblet of Fire topic only 20% of the time. One could argue that this is still better than randomly guessing (1/7 chance of getting the correct answer), but we think this means that the model is not very helpful in this setting.

For our second analysis, where we want to find the common themes/topics within each book, we saw a similar pattern. Take "Deathly Hallows" as an example:



Again, we found that there are certain words that appear on multiple topics, implying that the topics are not dissimilar to each other, at least to the level that we desired. As a consequence, the model couldn't find a clear, apparent topic for any book:

document	t1_proportion	t2_proportion	t3_proportion
Philosopher's Stone	0.3338852	0.3310015	0.3351133
Chamber of Secrets	0.3128063	0.3308932	0.3563005
Prisoner of Azkaban	0.3387294	0.3379194	0.3233512
Goblet of Fire	0.3278137	0.2973350	0.3748513
Order of the Phoenix	0.3182399	0.3663048	0.3154553
Half-Blood Prince	0.3330623	0.3284662	0.3384715
Deathly Hallows	0.3486131	0.2772557	0.3741312

Take Philosopher's Stone. The model says that it is 33% topic 1, 33% topic 2, and 33% topic 3. We observed that this trend holds for all 7 books. Without even identifying and naming the topic (something us users must do; the model couldn't do this), we conclude that there is not a dominant theme in each book.

Conclusion

Overall, we were able to roughly answer all of the questions that we had proposed although we do have some limitations to our analysis. In particular, our sentiment analysis is only as useful as the lexicon we use and although the BING lexicon worked fairly well, there were lots of words from the novel that were not included because they did not match any of the words in the lexicon. Overall, only about 13.05% of the words (excluding stop-words) had a sentiment value assigned to them so it is difficult to determine the effectiveness of this visual. With a larger and more specific lexicon, we could better analyze the sentiment as it varies through the novel with a greater variety and proportion of words being represented. We planned on doing a statistical test to see if there is a significant difference in the books, however we opted not to do this due to the limited lexicon not being very representative of the actual books. Additionally, the LDA analysis accomplishes this to a certain degree.

For the LDA, our biggest limitation is the nature of our data. Although topic modeling is a very popular method of multiple classification, our analysis serves to be an example of an edge case when the model may not produce absolute results: The Harry Potter franchise revolves around one main character, Harry Potter, so all 7 books are somewhat interconnected. Therefore, in a sense, some words are used repeatedly throughout the series. Therefore, if we look at a word that is used in all 7 books, we would not necessarily know which book this instance of the word belongs to. This is exactly what's happening with the LDA model. With that being said, our analysis doesn't undermine the prowess of Latent Dirichlet Allocation: Given 4 very different books (example in the References section), the model correctly classified words almost 100% of the time.

We used the Harry Potter books as a way to demonstrate text analysis on a series and this method could be used to analyze any series or group of books. We believe that our work can be generalized. That is, users can input an infinite number of books to form a corpus, and the model would still be able to do the analysis. Furthermore, our work can be extended with more computational power: As the number of topics increases, the running time also increases exponentially. A way to improve our study would be to pick unrelated data instead of books that are sequels of another.

References

Below are all the resources that we used to realize this ambitious project:

R package for data: `bradleyboehmke/harrypotter`

https://github.com/bradleyboehmke/harrypotter

R package for LDA: topicmodels

https://cran.r-project.org/web/packages/topicmodels/vignettes/topicmodels.pdf

Text Analysis Guide

https://www.tidytextmining.com/index.html

LDA concept guide:

https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2

http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/

https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation

LDA Example:

https://www.tidytextmining.com/topicmodeling.html

Machine Learning papers:

Blei D.M., Ng A.Y., Jordan M.I. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research, 3, 993–1022.

Phan X.H., Nguyen L.M., Horguchi S. (2008). Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections. In Proceedings of the 17th International World Wide Web Conference (WWW 2008), pages 91–100, Beijing, China.

Lu, B., Ott, M., Cardie, C., Tsou, B.K. (2011). Multi-aspect Sentiment Analysis with Topic Models. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, pages 81–88.