Post02 - A Comparison of Two of Dickens' Novels

1. Introduction

Charles Dickens was a British author in the 1800s. [1] You might recognize him from some of the novels he wrote, which many American schoolchildren are required to read (especially during high school). This blog post is about his books.

I should be more precise. This blog post is not about literary analysis, so it's okay if you forgot everything about him except his name. This blog post is about comparing the words in the Dickens' novels - in paritcular, *A Tale of Two Cities* (abbreviated TTC) and *Great Expectations* (abbreviated GE) - and looking for patterns and differences betweeen and among them.

The primary analysis will be in constructing data sets consisting of the text of the two novels and then comparing if they have roughly the same proportion of certain common words, a crude approximation of whether the two novels are written with the same style. Other miscellaneous comparisons may also be presented.

2. Background

A Tale of Two Cities takes place, as the name suggests, around two cities, London and Paris, around the time of the French Revolution. There is a love triangle, and someone dies in the end. However, this is just some context to the book. [1] It was first published in weekly installments.

Great Expectations is about Pip and his life over the span of several years, taking place in multiple places, though mostly in British-controlled territories (in the 1800s). [2] It was also published in serial form (a chapter every week).

In the course of this analysis, the R packages `readr`, `stringr`, and `ggplot2` (for plots) will be used, as we will be analyzing words from files downloaded from the internet, which are easily represented as characters in R. The `magrittr` library is used just for the pipe function.

```
library(readr)
```

```
## Warning: package 'readr' was built under R version 3.4.2
```

```
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 3.4.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
library(magrittr)
```

3. The Data

Here I procure the raw text of the files, from Project Gutenberg's online services.

```
# downloading the files
# A Tale of Two Cities ^[4]^
download.file('http://www.gutenberg.org/files/98/98-0.txt', 'a-tale-of-two-cities.txt')
# Great Expectations ^[5]^
download.file('http://www.gutenberg.org/files/1400/1400-0.txt', 'great-expectations.txt')
# use readr package to read the files into one character vector
temp1 <- read_file('a-tale-of-two-cities.txt')
temp2 <- read_file('great-expectations.txt')
```

Now we have the raw text, but we only want the words. Project Gutenberg attaches its own preamble and other licensing/etc. things, and the novels themselves contain a table of contents and chapter headings. We now remove the unwanted text.

3.a. Processing `ttc`

```
# remove chapter headings, e.g. "I. The Period"
ttc <- str_replace_all(temp1, '[\\n\\r][IVX]+\\. [^\\n\\r]+[\\n\\r]', " ")
# remove book headings, e.g. "Book the First--Recalled to Life"
ttc <- str_replace_all(ttc, "Book the (First|Second|Third)--[a-zA-Z ]+[\\n\\r]", " ")
# keep only the novel's text by searching for first and last phrases of novel
ttc <- str_sub(ttc,
               str_locate(ttc, 'It was the best of times')[1, 1],
               str_locate(ttc, 'End of the Project Gutenberg')[1, 1] - 1)
# replace all whitespaces and em dashes with just a single space
ttc <- str_replace_all(ttc, '(--)', " ")
ttc <- str_replace_all(ttc, '\\s+', " ")
# split up the words based on spaces and return the vector
ttc <- str_split(ttc, ' ')[[1]]
# get rid of the last element because it's empty
ttc <- ttc[1:length(ttc) - 1]
```

3.b. Processing `ge`

```
# remove chapter headings, e.g. Chapter I
ge <- str_replace_all(temp2, '[\\n\\r]Chapter [IVLX]+[\\n\\r]', " ")
# keep only the novel's text
ge <- str_sub(ge,
              str_locate(ge, 'My father')[1, 1],
              str_locate(ge, 'End of the Project Gutenberg')[1, 1] - 1)
# replace whitespaces and em dashes
ge <- str_replace_all(ge, '(--)', " ")
ge <- str_replace_all(ge, '\\s+', " ")
# split up words based on spaces
ge <- str_split(ge, ' ')[[1]]
# get rid of the last element
ge <- ge[1:length(ge) - 1]
```

Honestly, it took me a long time to figure out how to get these data vectors.

4. Exploratory Numbers (basically just easy stuff)

Let's take an introductory tour of these vectors.

4.a. Word Count

```
wc_ttc <- length(ttc)
wc_ttc
```

```
## [1] 136135
```

We expect the word count to be 135,420 [6], which is close enough.

```
wc_ge <- length(ge)
wc_ge
```

```
## [1] 185341
```

We expect the word count to be 183,349, which is close enough.

It seems that our actual word count is slightly higher than what my source states. This could be due to different methods of counting words, different editions used, or that my method extracted some more "words" than necessary.

4.b. Average words per sentence

Note that the vector elements still have punctuation. We first remove closing quotation marks (of dialogues), and then a non-exhaustive list of possible abbreviations. This means that we will probably end up with more sentences than there actually are, but this difference should not be too significant.

```
# I looked up the unicode, because the quotation marks were weird
ttc_sen <- str_replace_all(ttc, '\\.\u201D$', '\\.')
ge_sen <- str_replace_all(ge, '\\.\u201D$', '\\.')

# remove the abbreviations I could think of
ttc_sen <- ttc_sen[ttc_sen != 'M.|Mr.|Mrs.|Ms.|St.']
ge_sen <- ge_sen[ge_sen != 'M.|Mr.|Mrs.|Ms.|St.']
```

Now we can find the number of sentences quite easily.

```
# count the number of periods
ttc_sen <- ttc_sen %>% str_detect('\\.$') %>% sum()
ge_sen <- ge_sen %>% str_detect('\\.$') %>% sum()

# for ttc
wc_ttc / ttc_sen
```

```
## [1] 20.91489
```

```
# for ge
wc_ge / ge_sen
```

```
## [1] 21.90533
```

So it appears that the two novels *basically* have the same average number of words per sentence (though *Great Expectation* apparently has one more word per sentence on average than *A Tale of Two Cities*).

5. Some Actual Analysis

5.a. Vocabulary

We now remove all punctuation and store the words in two new vectors (that will have only words). Then we'll coerce them all to lower case.

```
# for ttc
ttc_words <- str_replace_all(ttc, '[^a-zA-Z-]+', '') %>% tolower()
# for ge
ge_words <- str_replace_all(ge, '[^a-zA-Z-]+', '') %>% tolower()
```

5.a.i. Unique words

```
# a function to count the number of unique words
unique_words <- function(x) {
  length(levels(factor(x)))
}
unique_words(ttc_words)
```

```
## [1] 10123
```

```
unique_words(ge_words)
```

```
## [1] 11326
```

Note that this includes proper nouns. As may be expected, the longer novel has more unique words.

We can actually make a scatterplot of the number of unique words vs. the number of total words, i.e. how many new words are introduced as the novel continues. I will normalize the axes, so that it becomes the number of unique words (of the total number of unique words) vs. the relative progress through the novel (e.g. 50% of the way through).

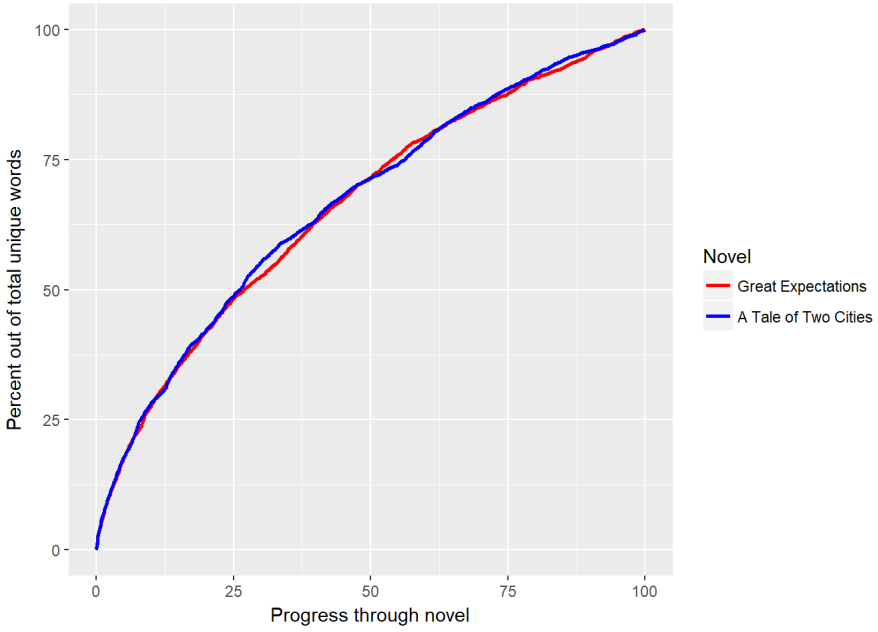
```
prog = seq(0, 100, length.out = 1000)
# the vectors of unique words vs. relative progress through the novel
ttc_rel <- rep(0, length(prog))
ge_rel <- rep(0, length(prog))

# WARNING: this loop takes a few minutes to run
for (i in 1:length(prog)) {
  ttc_rel[i] <-
    unique_words(ttc_words[1:round(wc_ttc * prog[i]/100, digits = 0)])
  ge_rel[i] <-
    unique_words(ge_words[1:round(wc_ge * prog[i]/100, digits = 0)])
}

ttc_rel_perc <- ttc_rel / unique_words(ttc_words) * 100
ge_rel_perc <- ge_rel / unique_words(ge_words) * 100

dat <-  data.frame(prog = prog,
                   perc = c(ttc_rel_perc, ge_rel_perc),
                   novel = c(rep('ttc', length(prog)), rep('ge', length(prog))))

ggplot(data = dat) +
  geom_line(aes(x = prog, y = perc, group = novel, color = novel), size = 1) +
  scale_color_manual(labels = c('Great Expectations',
                                'A Tale of Two Cities'),
                     values = c('red', 'blue'),
                     name = 'Novel') +
  labs(x = 'Progress through novel', y = 'Percent out of total unique words')
```



This graph shows that for both novels, new words appear even near the end of the novel (the lines do not reach 100 until the very end of the novel). We can, however, do better and compare whether new words show up at the same rate. This is not entirely apparent from the graph. We

can instead perform a chi-square test for homogeneity, using the above variables.

The null hypothesis is that the proportion of unique words (of total unique words) at a certain point in the novel is the same for both novels. The alternate hypothesis is that proportion is not the same.

```
chisq.test(ttc_rel, ge_rel)
```

```
## Warning in chisq.test(ttc_rel, ge_rel): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  ttc_rel and ge_rel
## X-squared = 984000, df = 983060, p-value = 0.2512
```

The p-value is rather large, so we cannot reject the null hypothesis. The proportion of unique words tend to show up at around the same rate for both novels.

Alright, now for some simpler stuff.

5.a.ii. Longest Word

```
# for ttc
max(str_length(ttc_words))
```

```
## [1] 21
```

```
ttc_words[str_length(ttc_words) == max(str_length(ttc_words))]
```

```
## [1] "garrison-and-dockyard" "pancras-in-the-fields"
```

```
# for ge
max(str_length(ge_words))
```

```
## [1] 25
```

```
ge_words[str_length(ge_words) == max(str_length(ge_words))]
```

```
## [1] "pocket-handkerchief-point"
```

These are both compound “words.” Let’s only include words that don’t have dashes in them.

```
# for ttc
ttc_single_words <- ttc_words[-grep('-', ttc_words)]
max(str_length(ttc_single_words))
```

```
## [1] 17
```

```
ttc_single_words[
  str_length(ttc_single_words) == max(str_length(ttc_single_words))
  ]
```

```
## [1] "undistinguishable"
```

```
# for ge
ge_single_words <- ge_words[-grep('-', ge_words)]
max(str_length(ge_single_words))
```

```
## [1] 19
```

```
ge_single_words[
  str_length(ge_single_words) == max(str_length(ge_single_words))
]
```

```
## [1] "architectooralooral"
```

I searched it up. Apparently someone in *Great Expectations* had difficulty with some pronunciation. Let’s display the first several longest words:

```
# for ttc
ttc_long_words <-
  ttc_single_words[order(str_length(ttc_single_words), decreasing = TRUE)] %>%
  head(n = 10)
data.frame(words = ttc_long_words,
           length = str_length(ttc_long_words))
```

```
##                words length
## 1  undistinguishable     17
## 2   incommodiousness     16
## 3   incomprehensible     16
## 4   enthusiastically     16
## 5   disproportionate     16
## 6    unconsciousness     15
## 7    acknowledgments     15
## 8    disinterestedly     15
## 9    accomplishments     15
## 10   characteristics     15
```

```
# for ge
ge_long_words <-
  ge_single_words[order(str_length(ge_single_words), decreasing = TRUE)] %>%
  head(n = 10)
data.frame(words = ge_long_words,
           length = str_length(ge_long_words))
```

```
##                  words length
## 1  architectooralooral     19
## 2   misrepresentations     18
## 3    unsympathetically     17
## 4    undistinguishable     17
## 5    disinterestedness     17
## 6    irreconcilability     17
## 7    disinterestedness     17
## 8     incomprehensible     16
## 9     constitutionally     16
## 10    constitutionally     16
```

Such length. It seems that GE's longest words are slightly longer than TTC's longest words.

5.a.iii. Most Used Word

```
# for ttc
sort(table(ttc_words), decreasing = TRUE)[1:10]
```

```
## ttc_words
##  the  and   of   to    a   in   it  his    i that
## 7995 4980 4001 3473 2930 2589 2031 2011 1937 1911
```

```
# for ge
sort(table(ge_words), decreasing = TRUE)[1:10]
```

```
## ge_words
##  the  and    i   to   of    a   in that  was   it
## 8143 7077 6482 5076 4432 4040 3022 2987 2836 2671
```

Extremely insightul /s.

5.a.iv. Miscellaneous

This book was written in the 1800s. The English language has evolved quite a bit since then.

```
str_detect(ttc_words, '^gay$') %>% sum()
```

```
## [1] 1
```

```
str_detect(ge_words, '^gay$') %>% sum()
```

```
## [1] 5
```

5.b. A Comparison of Word Usage

As promised, here's an analysis of whether the two novels have roughly the same proportion of common words. We will use the above computed table of most common words to (arbitrarily) select our most common words: "the", "and", "I", "to", "of", "a", "an", "that", "in", and "it."

```
common_words <- c('the', 'and', 'i', 'to', 'of', 'a', 'an', 'that', 'in', 'it')
# for ttc
common_ttc <- table(ttc_words)[common_words]
common_ttc
```

```
## ttc_words
##  the  and    i   to   of    a   an that   in   it
## 7995 4980 1937 3473 4001 2930  345 1911 2589 2031
```

```
# for ge
common_ge <- table(ge_words)[common_words]
common_ge
```

```
## ge_words
##  the  and    i   to   of    a   an that   in   it
## 8143 7077 6482 5076 4432 4040  446 2987 3022 2671
```

We can test whether these words show up in roughly the same proportion in their respective novels with (another) chi-square test of homogeneity. The null hypothesis is that each word appears at the same rate in both novels. The alternate hypothesis is that the rate is different for the two novels for each word.

```
chisq.test(common_ttc, common_ge)
```

```
## Warning in chisq.test(common_ttc, common_ge): Chi-squared approximation may
## be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  common_ttc and common_ge
## X-squared = 90, df = 81, p-value = 0.2313
```

Since the p-value is rather large, we find that we cannot reject the null hypothesis. One way of interpreting this result is that stylistically, these two novels are similar (at least by this crude measurement). So it is indeed likely that both novels were written by the same person (presumably Dickens himself).

5.c. Goodreads Ratings

For our final analysis, I shall bring in some new data - Goodreads user ratings. I'm literally going to just copy the numbers from their website (as of 2017 December 3), since otherwise, extraction of the data would be rather difficult.

I will use the entries with the most ratings. [7][8]

```
#` Function that creates the ratings vectors

#` @param a The number of 1-star ratings
#` @param b The number of 2-star ratings
#` @param c The number of 3-star ratings
#` @param d The number of 4-star ratings
#` @param e The number of 5-star ratings
rater <- function(a, b, c, d, e) {
  c(rep(1, a), rep(2, b), rep(3, c), rep(4, d), rep(5, e))
}

ttc_ratings <- rater(29282, 56326, 155551, 220519, 228670)
ge_ratings <- rater(23326, 45183, 123007, 174869, 153467)
```

Summary statistics:

```
summary(ttc_ratings)
```

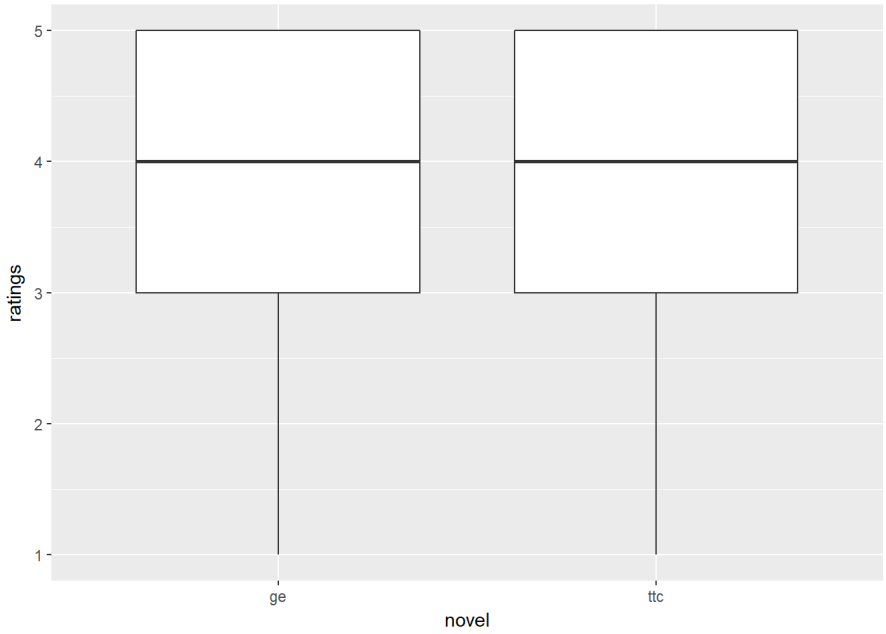
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   4.000   3.815   5.000   5.000
```

```
summary(ge_ratings)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    3.00    4.00    3.75    5.00    5.00
```

Here are some boxplots:

```
dat2 <- data.frame(ratings = c(ttc_ratings, ge_ratings),
                   novel = c(rep('ttc', length(ttc_ratings)),
                             rep('ge', length(ge_ratings))))
ggplot(data = dat2) +
  geom_boxplot(aes(y = ratings, x = novel))
```



Not going to lie, these boxplots aren't very insightful.

Let's test whether the mean of the ratings for the two books are equal. Our alternate hypothesis, after looking at our summary statistics, will be that the true average rating for *A Tale of Two Cities* is higher than the true average rating for *Great Expectations*. These ratings are probably not a random sample of the entire population of people who've read the book, but we can (hopefully) assume that this is roughly a simple random sample of the population of Goodreads users who review books they've read.

```
t.test(ttc_ratings, ge_ratings, alternative = 'greater')
```

```
##
##  Welch Two Sample t-test
##
## data:  ttc_ratings and ge_ratings
## t = 32.144, df = 1121800, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.06199063        Inf
## sample estimates:
## mean of x mean of y
##  3.815486  3.750152
```

Our p-value is actually really small (probably because the "sample" sizes are huge), so we can conclude that the true average rating for *A Tale of Two Cities* is greater than the true average rating for *Great Expectations*.

6. Conclusion

So after all those numbers, what can we conclude?

Honestly, we've mostly just played around with the data for fun and practiced manipulating strings. Let's recap what we've looked at in this post:

- word count
- average words per sentence
- unique words
- longest words
- most used words
- comparing proportions of most used words
- Goodreads ratings

In many of these comparisons, the two novels are very similar (statistically indistinguishable), but there are differences in some of them. GE has more words, more words per sentence on average, more unique words, longer words, and a lower Goodreads rating than TTC has. Though the evidence is not conclusive, I would wager that subjectively speaking, *Great Expectations* is a more difficult (more complex?) than *A Tale of Two Cities*. Stylistically they are similar, but based on word usage and customer satisfaction (assuming that customers dislike more the more complicated product), GE is slightly more challenging than TTC.

It may be interesting to note that *Great Expectations* was written after *A Tale of Two Cities*, so Dickens may have written more complex work as he wrote more.

References

[1] https://www.biography.com/people/charles-dickens-9274087
[2] http://www.sparknotes.com/lit/twocities/facts.html
[3] http://www.sparknotes.com/lit/greatex/facts.html
[4] http://www.gutenberg.org/ebooks/98
[5] http://www.gutenberg.org/ebooks/1400
[6] http://commonplacebook.com/art/books/word-count-for-famous-novels/
[7] https://www.goodreads.com/book/show/1953.A_Tale_of_Two_Cities

[8] https://www.goodreads.com/book/show/2623.Great_Expectations
[9] https://www.regular-expressions.info/