Project1 SPOOKY Data Analysis

Yang He yh2825

January 29th, 2018

Section 0: Install required packages & dependencies. Load libraries & functions.

```r
packages.used <- c("dplyr", "tidyverse", "tidytext", "tidyr", "ggplot2", "reshape2", "ggridges", "scale

packages.needed <- setdiff(packages.used, intersect(installed.packages()[,1], packages.used))

if(length(packages.needed) > 0) {
  install.packages(packages.needed, dependencies = TRUE, repos = 'http://cran.us.r-project.org')
}

# data manipulation packages
library(dplyr)
library(tidyverse)
library(tidytext)
library(tidyr)
library(reshape2)

# visualization packages
library(ggplot2)
library(ggridges)
library(scales)

# NLP packages
library(wordcloud)
library(ngram)
library(qdap)
library(SnowballC)
library(stringr)
library(cleanNLP)
library(udpipe) # backend for cleanNLP package
library(topicmodels)


# required functions
# source("../lib/coreNLPfunctions.R")
```

Section 1: Read in the data, observe data pattern and check empty entries.

The dataset 'spooky.csv' is in '../data/' folder. Read in the data containing the headers. Then check the summary, the headers, and if there are empty entries in the dataset. The commented out section is loading in the spooky dataset as a cleanNLP readable tokenized format, in the 'output' folder the data has been pre-stored since the dataset is quite large.

```r
spooky <- read.csv('../data/spooky.csv', as.is=TRUE, header=TRUE)
head(spooky)
```

```
##        id
## 1 id26305
## 2 id17569
## 3 id11008
## 4 id27763
## 5 id12958
## 6 id22965
##
## 1
## 2
## 3
## 4
## 5
## 6 A youth passed in solitude, my best years spent under your gentle and feminine fosterage, has so r
##   author
## 1    EAP
## 2    HPL
## 3    EAP
## 4    MWS
## 5    HPL
## 6    MWS
```

```r
summary(spooky)
```

```
##       id                 text               author
##  Length:19579       Length:19579       Length:19579
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
```

```r
sum(is.na(spooky))
```

```
## [1] 0
```

```r
# cleanNLP output pre-processing
# spooky_tmp <- spooky
# spooky_tmp$author <- as.data.frame(spooky_tmp$author)
# cnlp_init_udpipe()
# cnlpobj <- cnlp_annotate(spooky_tmp)
# saveRDS(cnlpobj, file = "../output/cnlp.rds")
```

The dataset has three columns: 'id' (id for excerpts), 'text' (excerpt content), 'author' (acronym for author name). There is no empty entries in the dataset. Change 'author' to be a factor variable for easier data manipulation later.

```r
spooky$author <- as.factor(spooky$author)
```

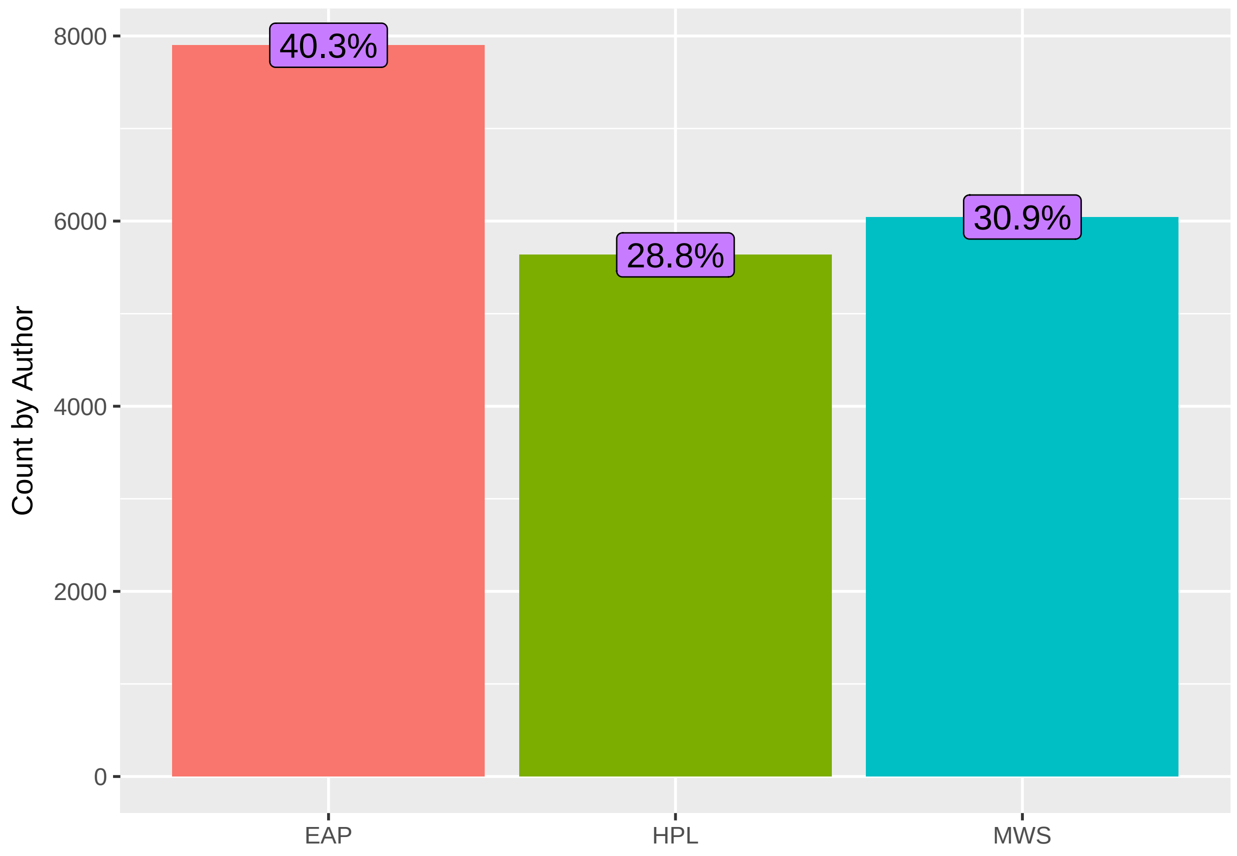
Section 2: Exploratory Analysis

Now that the cleaned up dataset is ready, an exploratory analysis of the data would hopefully provide some intuitions and insights for author identification purpose. The end goal here is to identify several plausible methods and metrics for author identification and feature engineering.

Count appearances for each author

First, we want to know how this dataset is distributed across different authors.

```r
total_count <- nrow(spooky)

author_count <- spooky %>%
  group_by(author) %>%
  summarize(count = n())


ggplot(author_count, aes(x = author, y = count, fill = author)) +
  geom_col() +
  geom_label(aes(label = percent(count/total_count),
              fill = "white",
              size=3.5),
              position = "identity") +
  xlab(NULL) +
  ylab("Count by Author") +
  theme(legend.position = 'none')
```

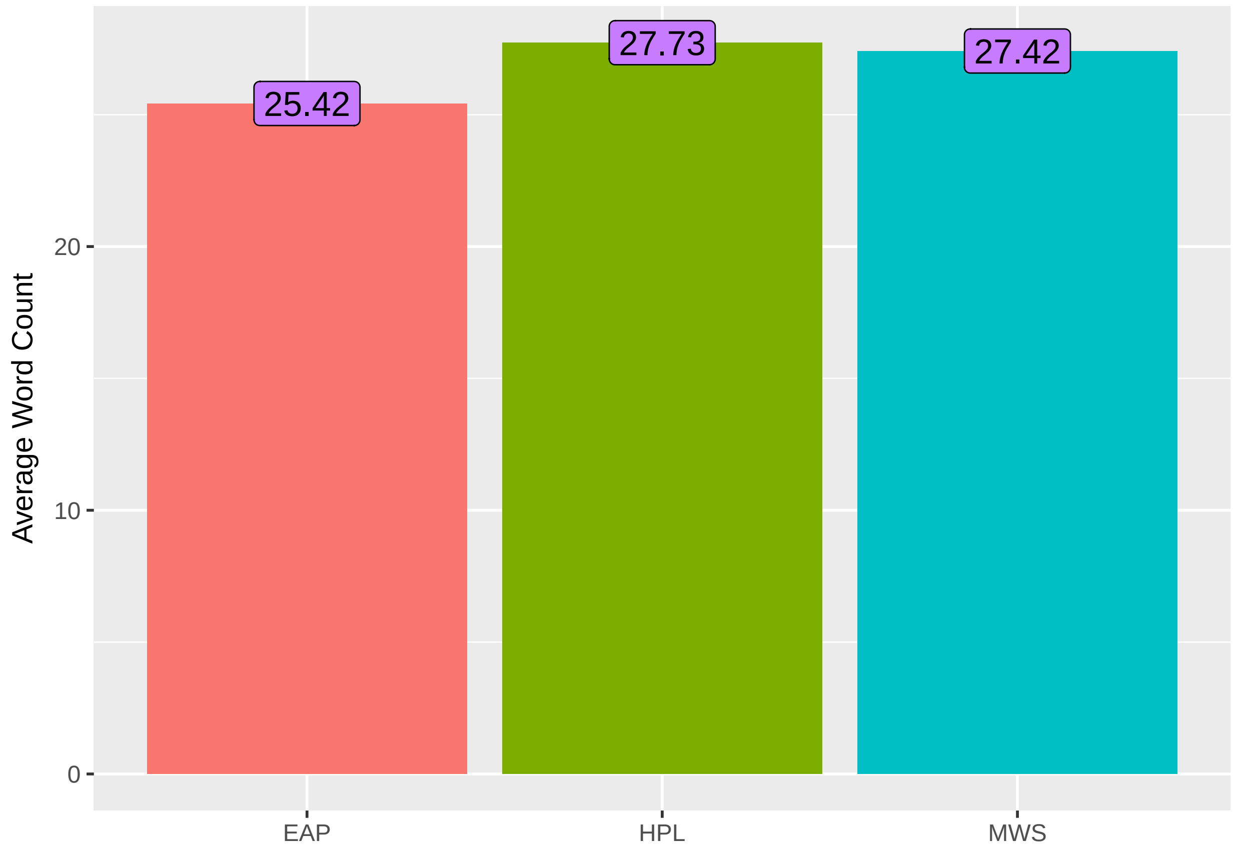


This shows that the dataset is not evenly distributed across different authors. ‘EAP’ has 40.3% of the excerpts, while ‘HPL’ and ‘MWS’ cover 28.8% and 30.9% respectively. Therefore, to show that an author identification method to be effective, these percentages are the “baseline” for prediction accuracies: any accuracy below than these for respective author are worse than a random guess.

Average excerpt length and length density

Second, check the excerpt length to see if there is a pattern. Plot both the average length histogram and density graph. Notice that the length are measured by word count (i.e. how many words are in the excerpt). The 'wc' function in 'qdap' package provides word count on sentences.

```
spooky_count <- spooky %>%
  mutate(length = wc(text))

word_count <- spooky_count %>%
  group_by(author) %>%
  summarize(avg_length = mean(as.numeric(length)))

ggplot(word_count, aes(x = author, y = avg_length, fill = author)) +
  geom_col() +
  geom_label(aes(label = format(avg_length, digits = 4),
                 fill = "white",
                 size=3.5),
                 position = "identity") +
  xlab(NULL) +
  ylab("Average Word Count") +
  theme(legend.position = 'none')
```

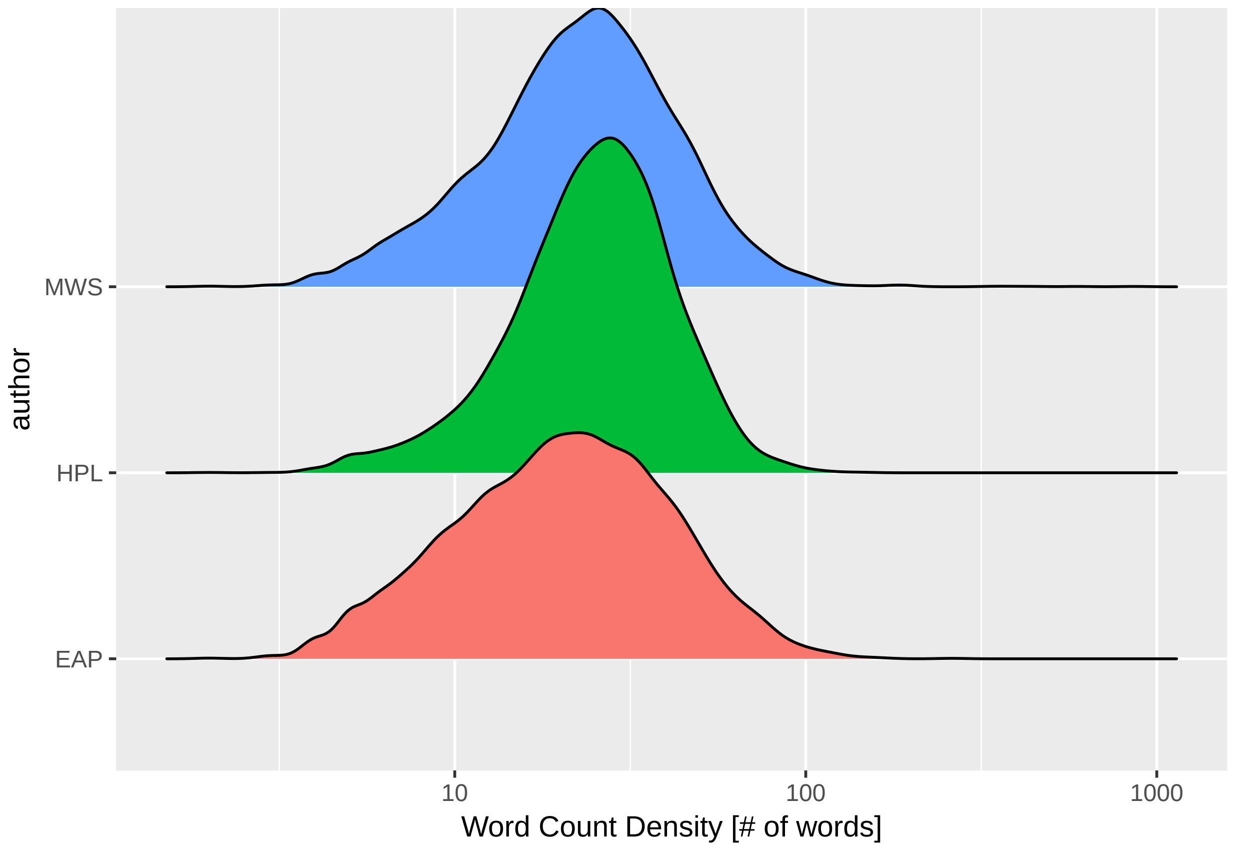


```
ggsave("../figs/avgwc.png")
```

```
## Saving 6.5 x 4.5 in image
```

```r
ggplot(spooky_count) +
  geom_density_ridges(aes(length, author, fill = author)) +
  scale_x_log10() +
  theme(legend.position = 'none') +
  labs(x = "Word Count Density [# of words]")
```

```
## Picking joint bandwidth of 0.0405
```



```r
ggsave("../figs/wcdens.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Picking joint bandwidth of 0.0405
```

On average, 'EAP''s excerpts are a bit shorter, but there is no significant differece on the average between the three. However, 'EAP' excerpts are distributed more evenly on length than the other two.

Section 3: Text Analysis

After analyzing how the dataset is structured, we want to dive deeper into the text and find out the differences.

Most frequent words and n-grams

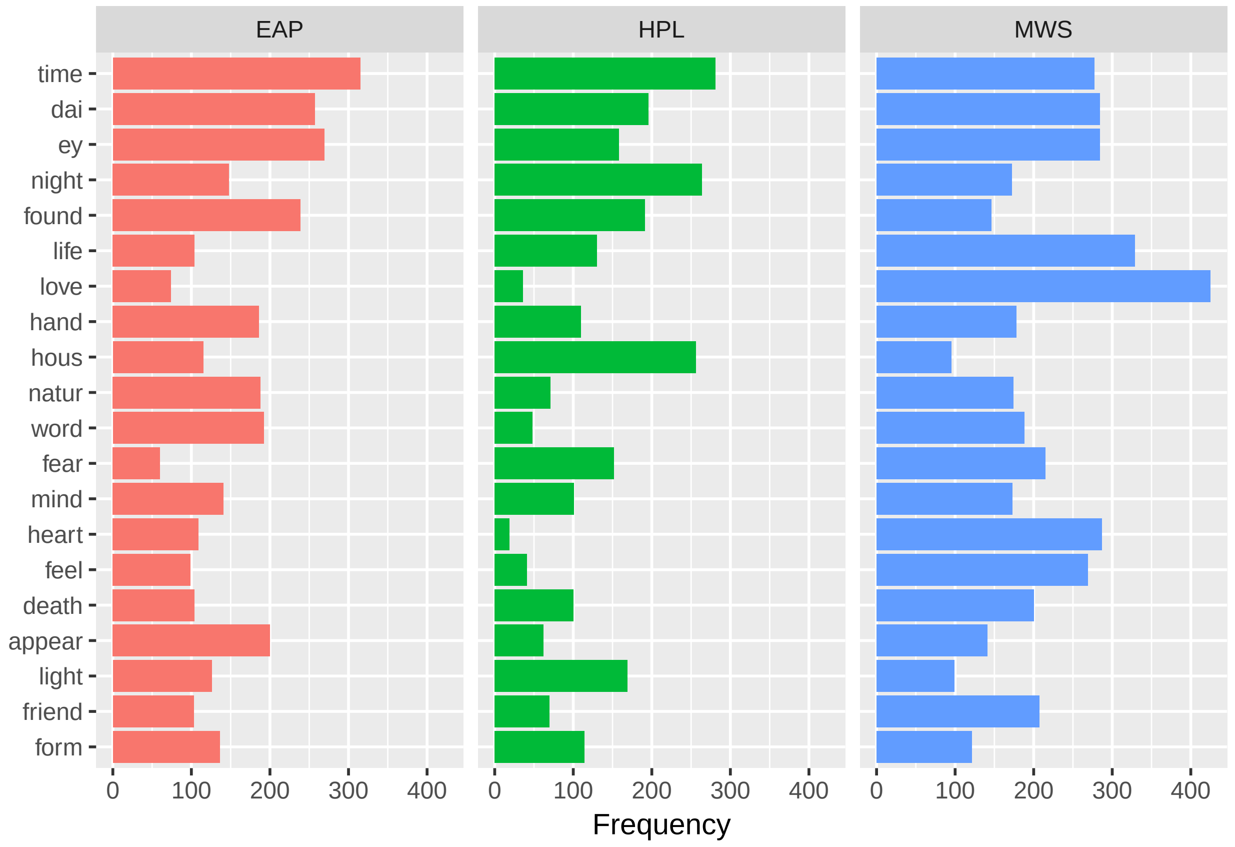
Here, we want to have a better understanding of the most frequent words or phrases that the three different authors would prefer - not only by choices of words, but also phrases. The intuition is that when filtering out stopwords one by one, sometimes the context is dropped; however, word context is also a demonstration of writing styles.

First, use 'unnest_tokens()' function in 'tidytext' package to tokenize all the words in the dataset. Then use 'stop_words' dictionary in 'tidytext' package to filter out words that does not necessarily impose emotions. Lastly, use 'wordStem' function in 'SnowballC' package to find stems for all the words.

```r
spooky_filtered <- spooky %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by = "word") %>%
  mutate(word = wordStem(word))
```

Then we find the most frequently used words. Here we want to find the most frequently used words across the overall dataset, and then compare between the authors - if we do it for each author, it will not facilitate cross comparison well. As one could imagine, there are definitely some common words that are shared among the authors, words like 'mind', 'time' or 'night' (since it's a SPOOKY dataset). The goal here is to find the differences between these common words.

```r
freq_word_author <- spooky_filtered %>%
  group_by(author, word) %>%
  summarize(count = n())

freq_word <- spooky_filtered %>%
  group_by(word) %>%
  summarize(tcount = n())

freq_word_author <- freq_word_author %>%
  left_join(freq_word, by = "word") %>%
  arrange(desc(tcount)) %>%
  head(60) %>% # top 20 for each author
  ungroup()

ggplot(freq_word_author, aes(reorder(word, tcount, FUN = min), y = count, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "Frequency") +
  theme(legend.position = "none") +
  facet_wrap(~ author) +
  coord_flip()
```



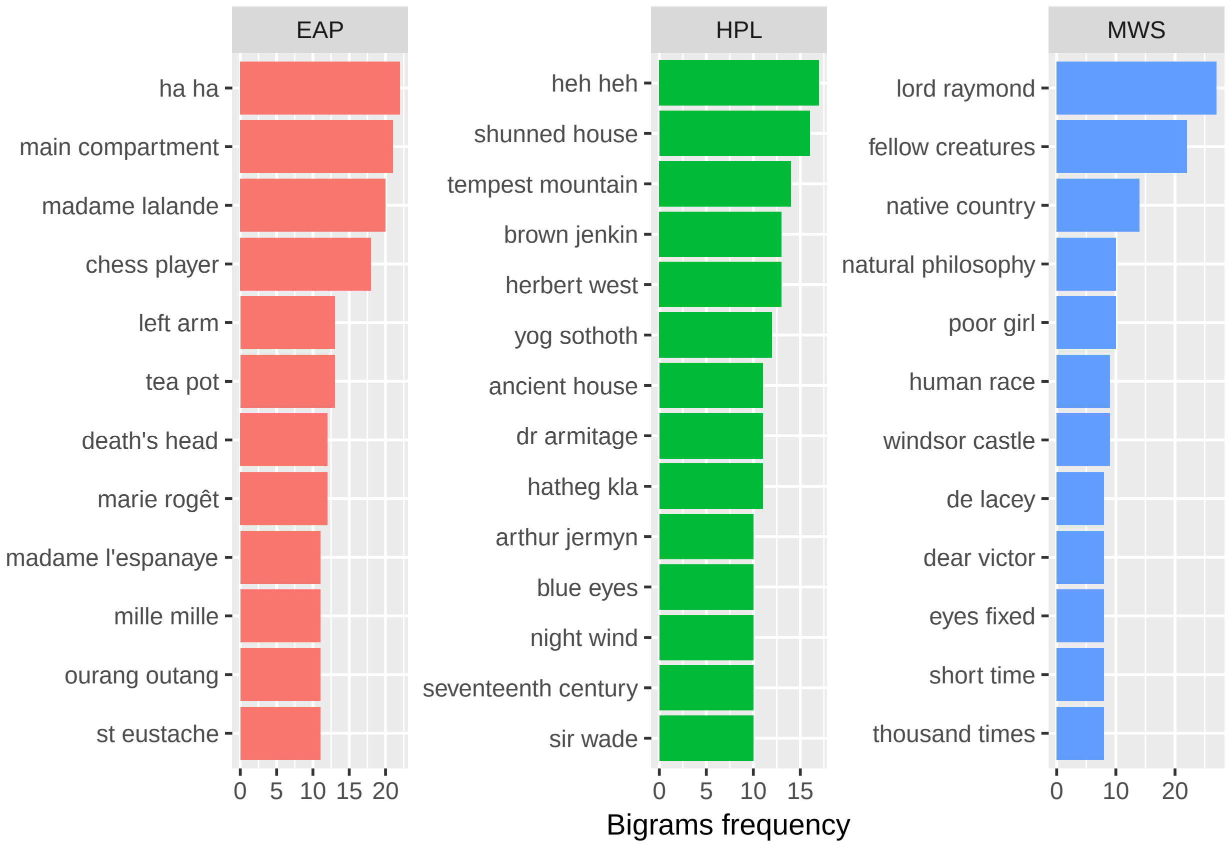
```
ggsave("../figs/wf.png")
```

```
## Saving 6.5 x 4.5 in image
```

It would be a reasonable guess to say that ‘MWS’‘s story is based on “love life”, mostly. Words like “friend” and “heart” also appears more in ’MWS’ than others, which possibly suggests that her story happens most likely in a social context. ‘HPL’‘s story seems to happen a lot in “house” and at “night”, while in ’EAP”s story stuff happens to be “found” and “appear” more often, maybe suggesting that there are more elements of surprise.

Next, let’s study bigrams…

```
spooky_bigram <- spooky %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2) %>%
  separate(bigram, c("word1", "word2"), sep = " ")

spooky_bigram_filtered <- spooky_bigram %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  unite(bigram, word1, word2, sep = " ")

spooky_bigram_count_by_author <- spooky_bigram_filtered %>%
  group_by(author, bigram) %>%
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
  ungroup() %>%
  mutate(bigram = factor(bigram, levels = rev(unique(bigram)))) %>%
  group_by(author) %>%
  top_n(10, count) %>%
  ungroup()

ggplot(spooky_bigram_count_by_author, aes(bigram, count, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "Bigrams frequency") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
  coord_flip()
```

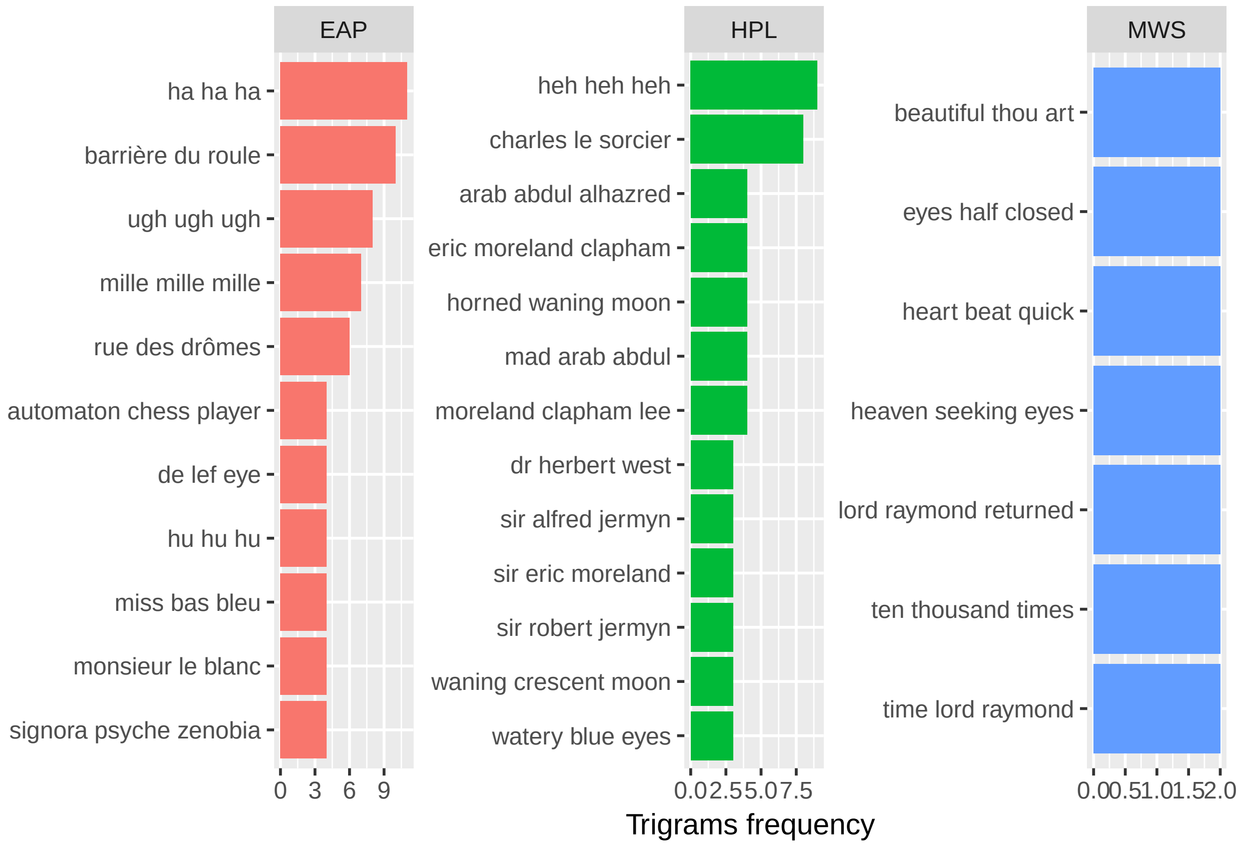


```
ggsave("../figs/bigramf.png")
```

```
## Saving 6.5 x 4.5 in image
```

And trigrams.

```
spooky_trigram <- spooky %>%
  unnest_tokens(trigram, text, token = "ngrams", n = 3) %>%
  separate(trigram, c("word1", "word2", "word3"), sep = " ")

spooky_trigram_filtered <- spooky_trigram %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  filter(!word3 %in% stop_words$word) %>%
  unite(trigram, word1, word2, word3, sep = " ")

spooky_trigram_count_by_author <- spooky_trigram_filtered %>%
  group_by(author, trigram) %>%
  summarize(count = n()) %>%
  arrange(desc(count)) %>%
  ungroup() %>%
  mutate(trigram = factor(trigram, levels = rev(unique(trigram)))) %>%
  group_by(author) %>%
  top_n(10, count) %>%
  filter(count > 1) %>% # MWS does not have a lot repeated trigrams...
  ungroup()

ggplot(spooky_trigram_count_by_author, aes(trigram, count, fill = author)) +
  geom_col() +
  labs(x = NULL, y = "Trigrams frequency") +
  theme(legend.position = "none") +
  facet_wrap(~ author, ncol = 3, scales = "free") +
  coord_flip()
```



```
ggsave("../figs/trigramf.png")
```

```
## Saving 6.5 x 4.5 in image
```

EAP: “ha ha”. HPL: “heh heh”. MWS: “. . . ”.

Besides EAP and HPL’s laughter and MWS’s slience, there are some characteristics differ from that of single words:

- A lot of the bigrams and trigrams are name specific, these will quickly help us to identify the author - and it should be one of the most features when doing author identification.
- HPL’s story happens when the “waning moon” appears, and this coincides with his relatively higher frequency of “night” in the single word analysis; same thing with the “shunned house” or “ancient

house" in the bigram analysis, as "house" appears quite often singly as well.
- MWS hardly ever repeats three words together, with only seven trigrams has frequency > 1 - although she does use "thousand times" repeatly. In both her bigrams and trigrams, she likes to describe a person through the "eyes".

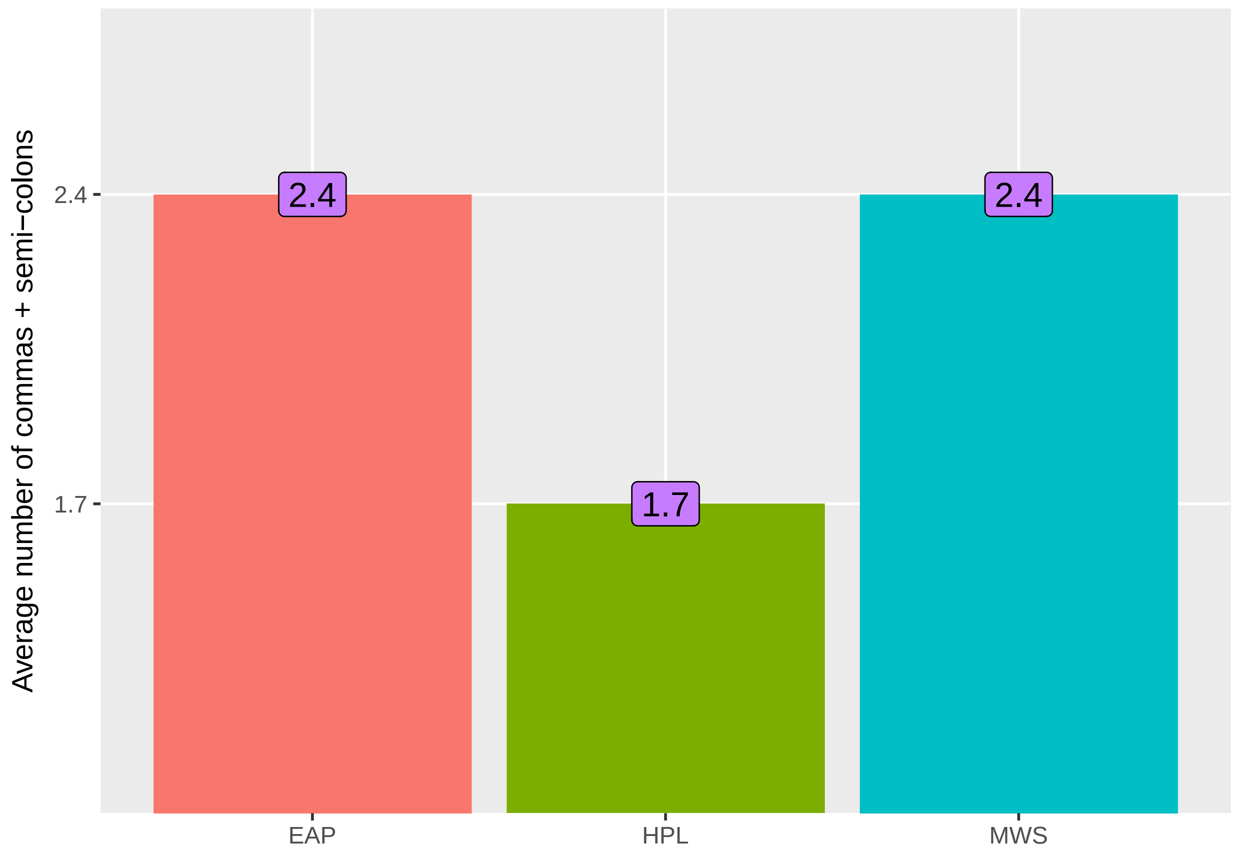
Sentence structure and Part of speech

HPL's work is almost 100 years later than the other two authors. Besides, although the excerpts share the same language, the authors came from different places. It is possible that the different language practices will have an effect on the usage of sentence structures, clauses etc.

Commas and semi-colons

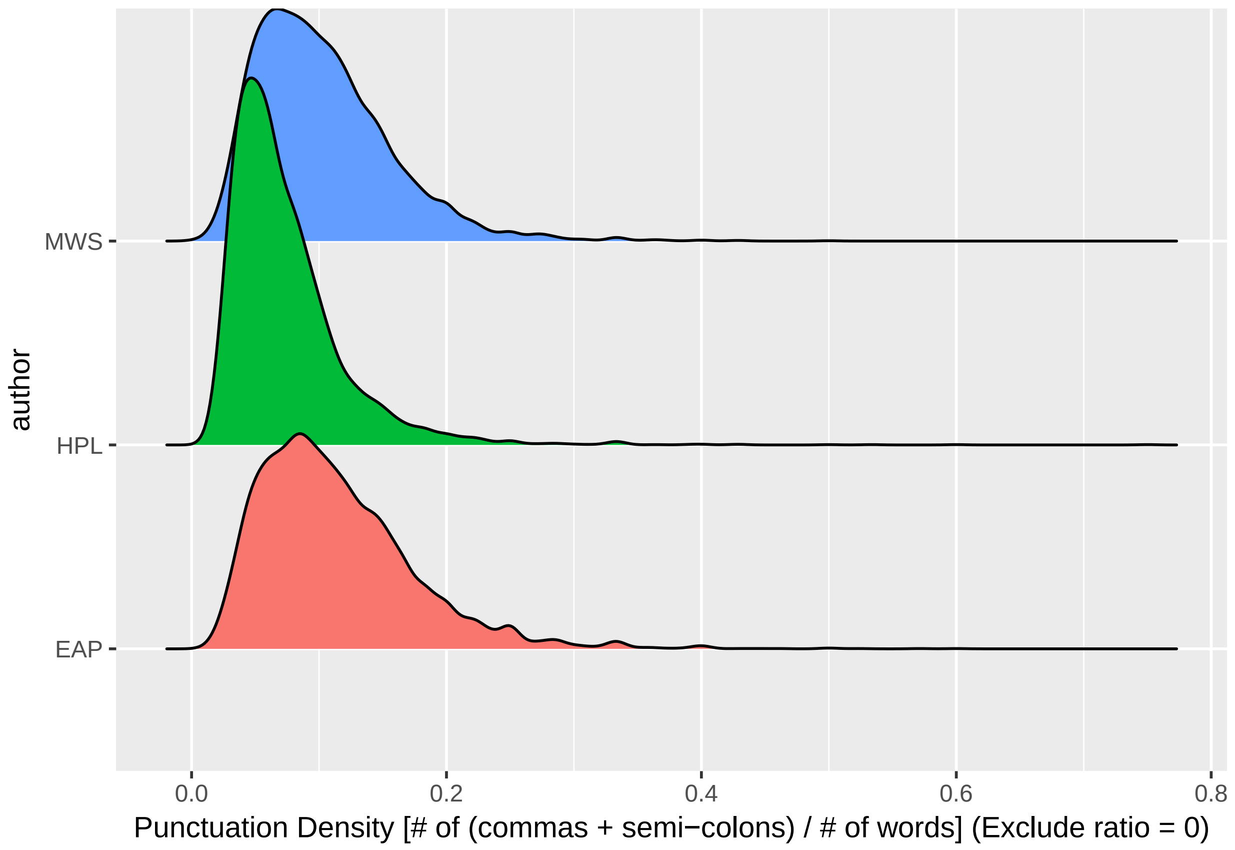
Let's look at how often does these authors uses commas and semi-colonns. Define the metric to be number of words in the excerpt per comma or semi-colomn used, and find the averages and frequency densities.

```r
spooky_punc_count <- spooky_count %>%
  mutate(comma_count = str_count(text, ',')) %>%
  mutate(sc_count = str_count(text, ';')) %>%
  mutate(quote_count = str_count(text, '\"')) %>%
  mutate(ratio = (comma_count+sc_count)/length)

spooky_punc_avg <- spooky_punc_count %>%
  group_by(author) %>%
  summarize(avg = format(mean(comma_count + sc_count), digits = 2))

ggplot(spooky_punc_avg, aes(x = author, y = avg, fill = author)) +
  geom_col() +
  geom_label(aes(label = avg,
              fill = "white",
              size=3.5),
              position = "identity") +
  xlab(NULL) +
  ylab("Average number of commas + semi-colons") +
  theme(legend.position = 'none')
```



```
ggplot(subset(spooky_punc_count, ratio > 0)) +
  geom_density_ridges(aes(ratio, author, fill = author)) +
  scale_x_continuous() +
  theme(legend.position = 'none') +
  labs(x = "Punctuation Density [# of (commas + semi-colons) / # of words] (Exclude ratio = 0)")
```

```
## Picking joint bandwidth of 0.00763
```



```r
ggsave("../figs/puncdens.png")
```

```
## Saving 6.5 x 4.5 in image
## Picking joint bandwidth of 0.00763
```

HPL seems to use commas and semi-colomns a lot less often than the other two authors, meaning typically his subsentences are longer and sentence structure are more complex. For MWS and EAP, the density distribution appears to be similar, althought MWS on average uses less punctuations per excerpt. This would match our expectation - modern english writing are generally more complex with frequent use of clauses etc. Remember that HPL's work is about 100 years later than the other two, and we see that the difference is significant, while the earlier works show some similarity.

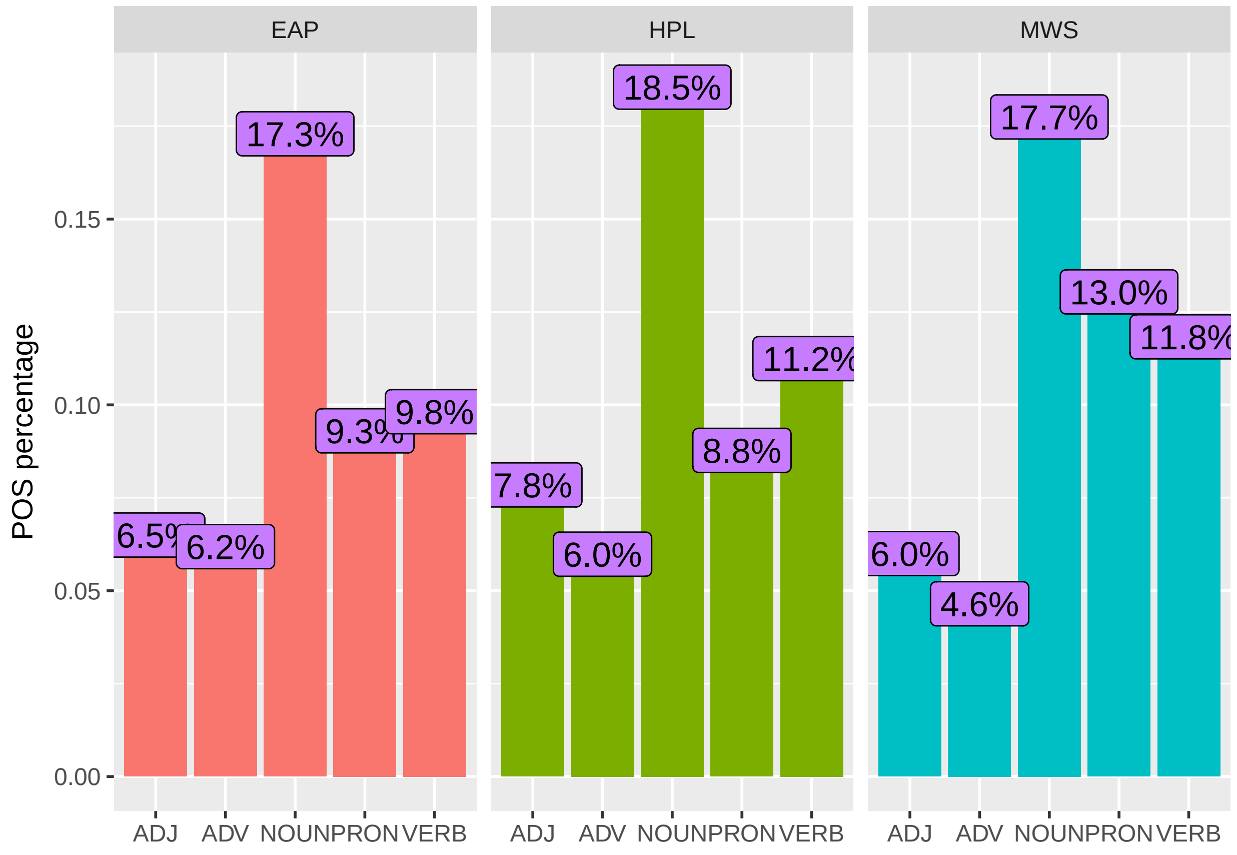
Part of speech

Here we want to make use of the cleanNLP package with strong POS analysis feature. First, let's load in the data.

```r
obj <- readRDS("../output/cnlp.rds")
spooky_id_author <- spooky %>%
  select(id, author)
spooky_tk <- cnlp_get_token(obj) %>%
  mutate(word = tolower(word)) %>%
  left_join(spooky_id_author, by = "id")
```

The dataset has several useful columns: 'upos', 'pos', 'gender', 'tense' etc. Let's look at these interesting columns one by one, and for this part let's look at POS first:

```r
spooky_POS <- spooky_tk %>%
  group_by(author, upos) %>%
  summarize(count = n()) %>%
  ungroup()

tmp <- spooky_POS %>%
  group_by(author) %>%
  summarize(total = sum(count))

spooky_POS_tmp <- spooky_POS %>%
  left_join(tmp, by = "author") %>%
  filter(upos %in% c("ADJ", "ADV", "NOUN", "VERB", "PRON"))

ggplot(spooky_POS_tmp, aes(x = upos, y = (count/total), fill = author)) +
  geom_col() +
  geom_label(aes(label = percent(count/total),
              fill = "white",
              size=3.5),
              position = "identity") +
  xlab(NULL) +
  ylab("POS percentage") +
  facet_wrap(~ author) +
  theme(legend.position = 'none')
```



```r
ggsave("../figs/POS.png")
```

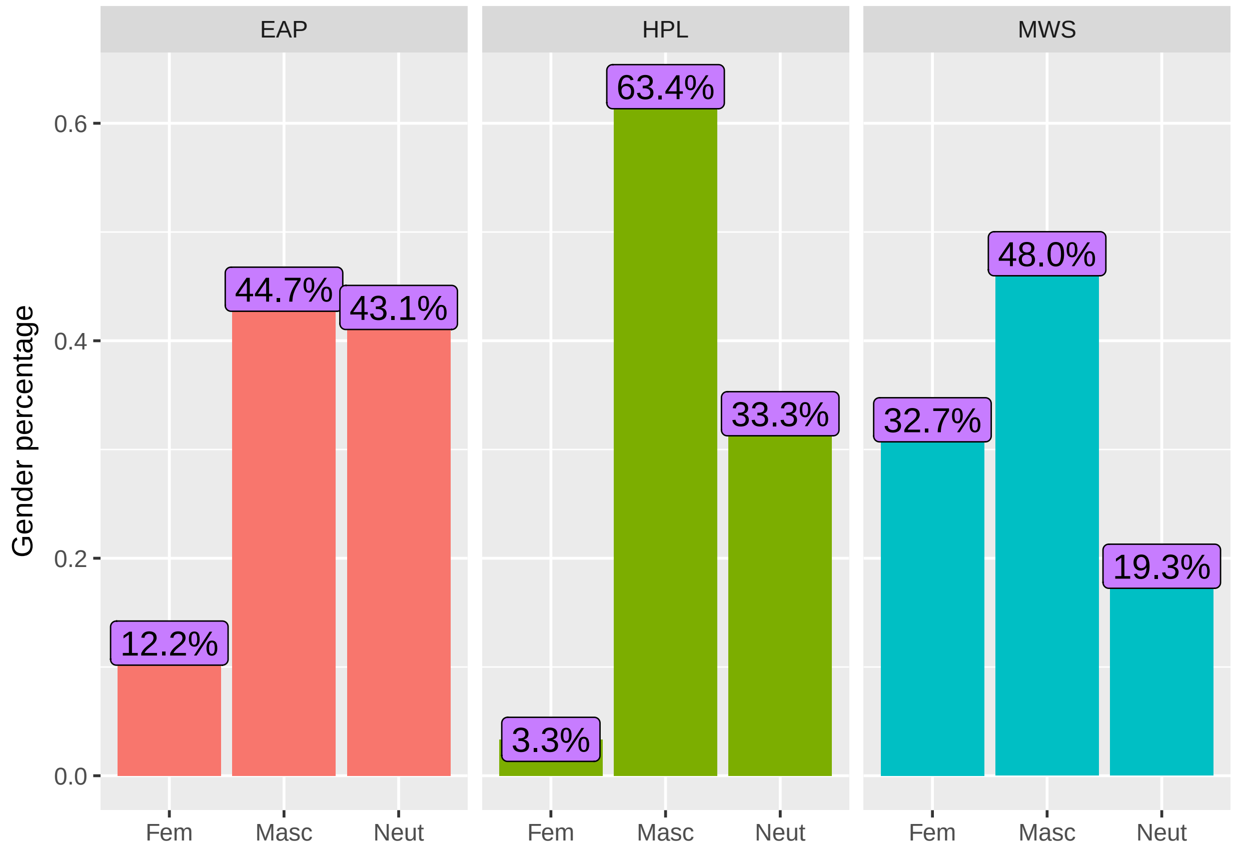
```
## Saving 6.5 x 4.5 in image
```

'EAP' and 'HPL' seem to have similar distributions of POS; however, MWS use pronouns a lot often, suggesting that there are more third-person narratives in MWS's work.

Masculine or Feminine?

Although we are looking into horror story excerpts, it might be still interesting to look at how the authors chose their wording according to gender. This gives us an easier way to see whether the main protagonist of the story is male or female, and may indicate the author's preference.

```
spooky_MF <- spooky_tk %>%
  filter(!is.na(gender)) %>%
  group_by(author, gender) %>%
  summarize(count = n()) %>%
  ungroup()

tmp <- spooky_MF %>%
  group_by(author) %>%
  summarize(total = sum(count))

spooky_MF_tmp <- spooky_MF %>%
  left_join(tmp, by = "author")

ggplot(spooky_MF_tmp, aes(x = gender, y = (count/total), fill = author)) +
  geom_col() +
  geom_label(aes(label = percent(count/total),
              fill = "white",
              size=3.5),
              position = "identity") +
  xlab(NULL) +
  ylab("Gender percentage") +
  facet_wrap(~ author) +
  theme(legend.position = 'none')
```



```r
ggsave("../figs/gender.png")
```

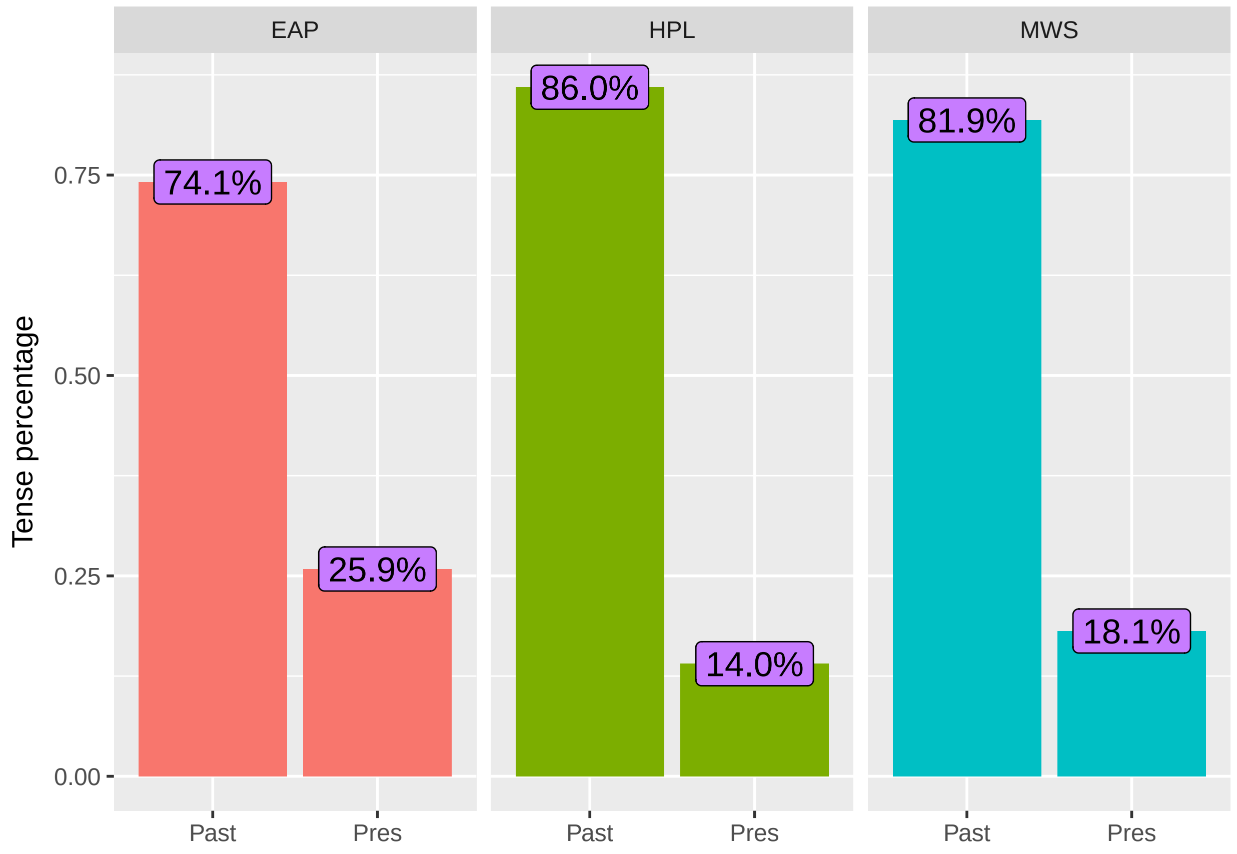
```
## Saving 6.5 x 4.5 in image
```

The three authors all show distinct pattern. MWS is the only female writer among the three and she reasonably uses more feminine words, suggesting that there may be a decent comparison of genders in MWS's story, possibly a story of "love" as indicated in the frequent word analysis. HPL's writing is a lot more "masculine" as only 3.3% is feminine word. EAP uses more neutural words than the other two.

Tense Use

The 'tense' column only has two values: past and present. Let's see the distribution between these two tenses.

```r
spooky_tense <- spooky_tk %>%
  filter(!is.na(tense)) %>%
  group_by(author, tense) %>%
  summarize(count = n()) %>%
  ungroup()

tmp <- spooky_tense %>%
  group_by(author) %>%
  summarize(total = sum(count))

spooky_tense_tmp <- spooky_tense %>%
  left_join(tmp, by = "author")

ggplot(spooky_tense_tmp, aes(x = tense, y = (count/total), fill = author)) +
  geom_col() +
  geom_label(aes(label = percent(count/total),
                 fill = "white",
                 size=3.5),
                 position = "identity") +
  xlab(NULL) +
  ylab("Tense percentage") +
  facet_wrap(~ author) +
  theme(legend.position = 'none')
```



```
ggsave("../figs/tense.png")
```

```
## Saving 6.5 x 4.5 in image
```

Past Tense Usage: HPL > MWS > EAP. A higher percentage of use of past tense might indicate more story-telling narrative in the excerpts, or possibly more

Section 4: Sentiment Analysis

For sentiment analysis of the excerpts, the bing sentiment lexicon is used, which provides a binary (positive/negative) result. The goal here is to compare the most frequently used negative and positive words used by different authors, and see the density of positive/negative words per sentence.

Wordclound for positive & negative words

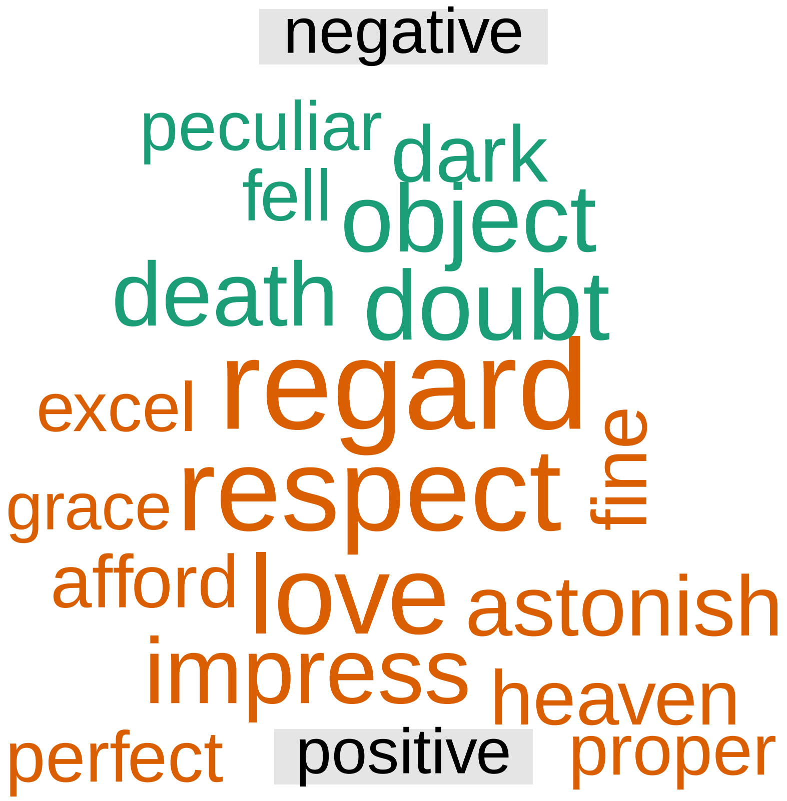
First, genenerate the sentiment table by inner joining with the bing lexicon.

```r
spooky_stmt <- inner_join(spooky_filtered, get_sentiments('bing'), by = "word")
```

Then, generate a word cloud for each of the authors, separating between negative and positive words.

For EAP:

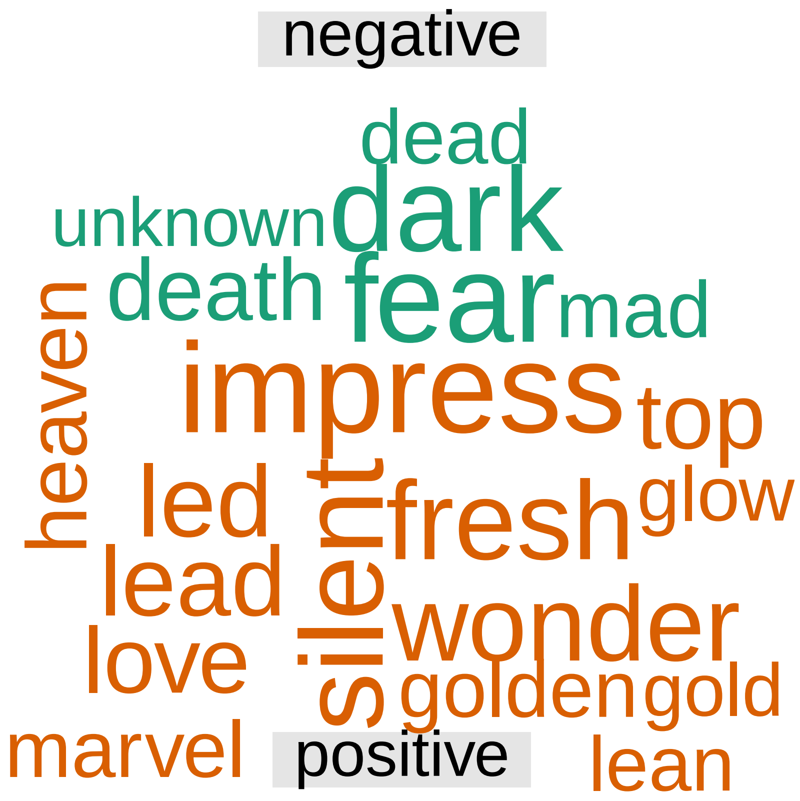
```r
# EAP
# png("../figs/stmtEAP.png")
spooky_stmt %>%
  filter(author == "EAP") %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(max.words = 20, title.size = 2)
```



```r
# dev.off()
```

For HPL:

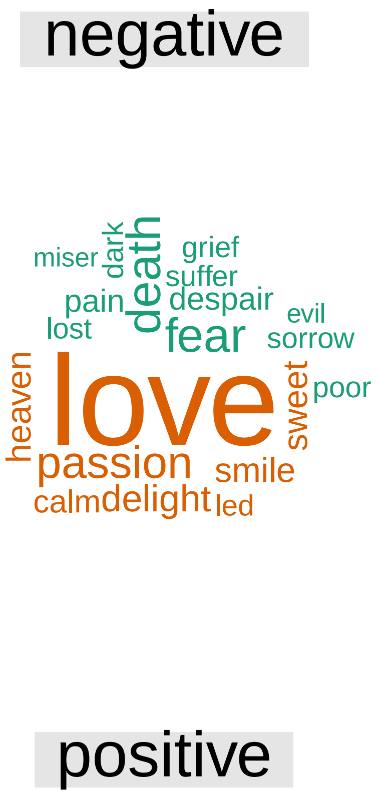
```r
# HPL
# png("../figs/stmtHPL.png")
spooky_stmt_HPL <- spooky_stmt %>%
  filter(author == "HPL") %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(max.words = 20, title.size = 2)
```



```
# dev.off()
```

For MWS:

```
# MWS
# png("../figs/stmtMWS.png")
spooky_stmt_MWS <- spooky_stmt %>%
  filter(author == "MWS") %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(max.words = 20, title.size = 2)
```



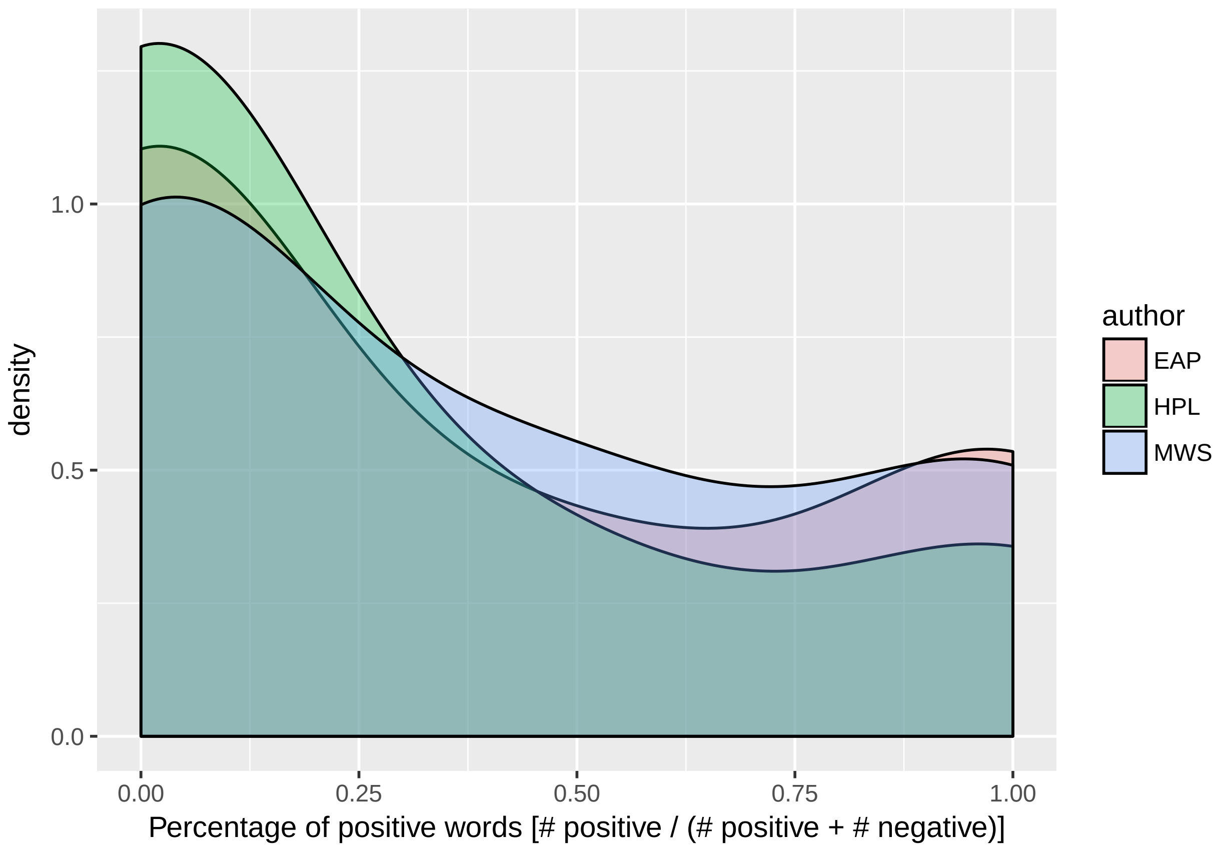
```r
# dev.off()
```

The word scales is weird for MWS because "love" was such a big part of it that other words become much smaller comparing to it - still very consistent with our previous analysis.

Density for positive word frequency

For each sentence, let us find the percentage ratio of positive words to the total number of words with either positive or negative sentiment. Then we show the density graphs for the three authors.

```r
spooky_stmt_density <- spooky_stmt %>%
  group_by(author, id, sentiment) %>%
  summarize(count = n()) %>%
  spread(sentiment, count, fill = 0) %>%
  group_by(author, id) %>%
  summarize(count_negative = sum(negative), count_positive = sum(positive)) %>%
  mutate(perc = (count_positive/(count_negative+count_positive)))

ggplot(spooky_stmt_density, aes(x = perc, fill = author)) +
  geom_density(bw = 0.2, alpha = 0.3) +
  xlab("Percentage of positive words [# positive / (# positive + # negative)]")
```



```
ggsave("../figs/stmtdensity.png")
```

```
## Saving 6.5 x 4.5 in image
```

HPL's excerpts are more negative than others', since the density where the ratio equals to zero is pretty high. The density graph is also skewed towards left more for HPL with a roughly monotounous decreasing trend, while for the other two there are slight variations. However, overall speaking, the spooky text dataset is contains more negative sentences than positive.

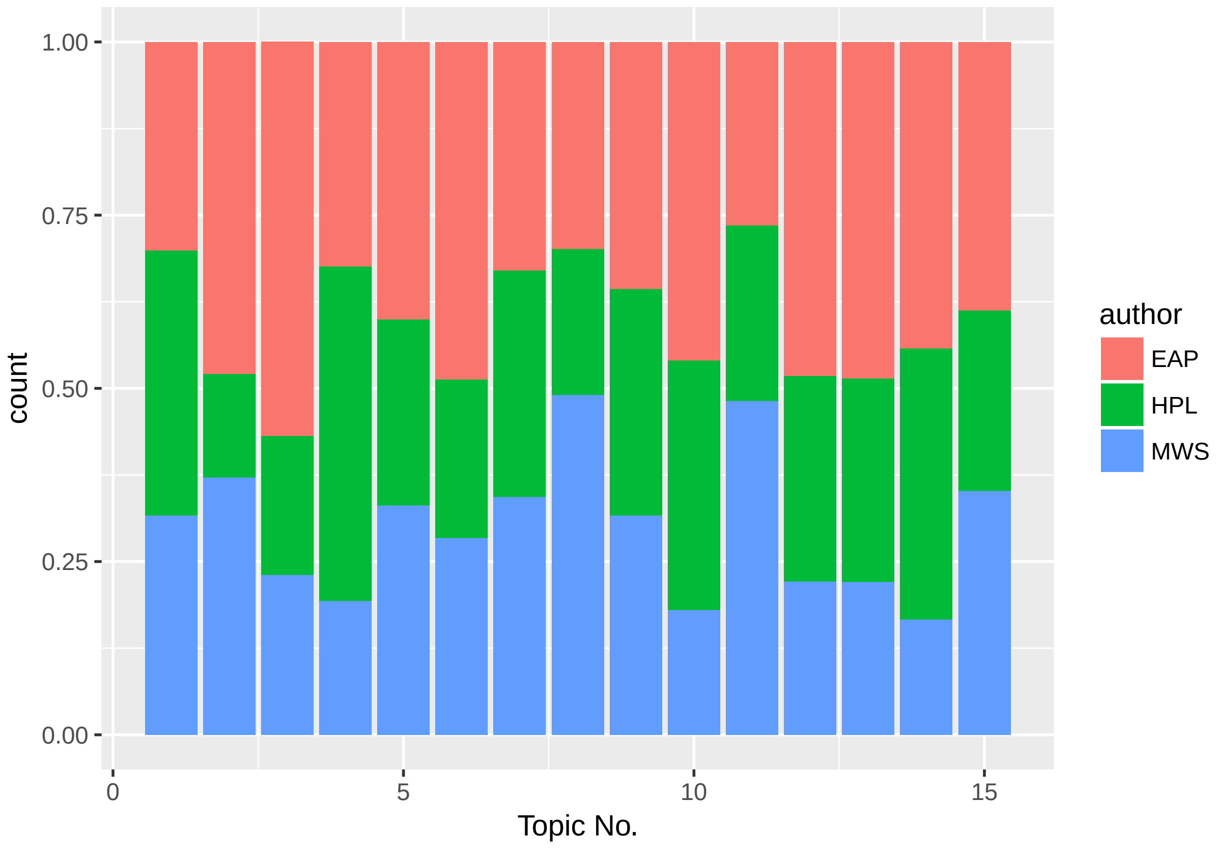
Section 5: Topic Modelling

For the topic modelling, the goal here is to see if there is a pattern of which authors using certain topics more often than others at sentence level.

```
spooky_tm <- spooky_tk %>%
  cnlp_get_tfidf(min_df = 0.05, max_df = 0.95, type = "tf", tf_weight = "raw") %>%
  LDA(k = 15, control = list(verbose = 1))
```

```
## **** em iteration 1 ****
## document 19560
## new alpha = 4.18321
## **** em iteration 2 ****
## document 19560
## new alpha = 5.07978
## **** em iteration 3 ****
## document 19560
## new alpha = 5.98895
## **** em iteration 4 ****
## document 19560
## new alpha = 6.90730
## **** em iteration 5 ****
## document 19560
## new alpha = 7.83266
## **** em iteration 6 ****
## document 19560
## new alpha = 8.76352
## **** em iteration 7 ****
## document 19560
## new alpha = 9.69880
## **** em iteration 8 ****
## document 19560
## new alpha = 10.63766
## **** em iteration 9 ****
## document 19560
## new alpha = 11.57948
## **** em iteration 10 ****
## document 19560
## new alpha = 12.52378
## **** em iteration 11 ****
## document 19560
## new alpha = 13.47016
## **** em iteration 12 ****
## document 19560
## new alpha = 14.41830
## **** em iteration 13 ****
## document 19560
## new alpha = 15.36796
## **** em iteration 14 ****
## document 19560
## new alpha = 16.31891
## **** em iteration 15 ****
## document 19560
## new alpha = 17.27097
## **** em iteration 16 ****
## document 19560
## new alpha = 18.22398
## **** em iteration 17 ****
## document 19560
## new alpha = 19.17784
## **** em iteration 18 ****
## document 19560
## new alpha = 20.13240
## **** em iteration 19 ****
## document 19560
## new alpha = 21.08760
## **** em iteration 20 ****
## document 19560
## new alpha = 22.04333
## **** em iteration 21 ****
## document 19560
## new alpha = 22.99952
## **** em iteration 22 ****
## document 19560
## new alpha = 23.95612
## **** em iteration 23 ****
## document 19560
## new alpha = 24.91305
## **** em iteration 24 ****
## document 19560
## new alpha = 25.87029
## **** em iteration 25 ****
## document 19560
## new alpha = 26.82777
## **** em iteration 26 ****
## document 19560
## new alpha = 27.78545
## **** em iteration 27 ****
## document 19560
## new alpha = 28.74331
## **** em iteration 28 ****
## document 19560
## new alpha = 29.70130
## **** em iteration 29 ****
## document 19560
## new alpha = 30.65940
## **** em iteration 30 ****
## document 19560
## new alpha = 31.61758
## **** em iteration 31 ****
## document 19560
## new alpha = 32.57581
## **** em iteration 32 ****
## document 19560
## new alpha = 33.53408
## **** em iteration 33 ****
## document 19560
## new alpha = 34.49237
## **** em iteration 34 ****
## document 19560
## new alpha = 35.45064
## **** em iteration 35 ****
## document 19560
## new alpha = 36.40890
## **** em iteration 36 ****
## document 19560
## new alpha = 37.36712
## final e step document 19560
```

```r
spooky_tm_topics <- tidy(spooky_tm, matrix = "gamma")

spooky_author_topic <- spooky_tm_topics %>%
  left_join(spooky, by = c("document" = "id")) %>%
  select(-text) %>%
  group_by(document) %>%
  top_n(1, gamma) %>%
  ungroup()

ggplot(spooky_author_topic, aes(topic, fill = author)) +
  geom_bar(position = "fill") +
  xlab("Topic No.")
```

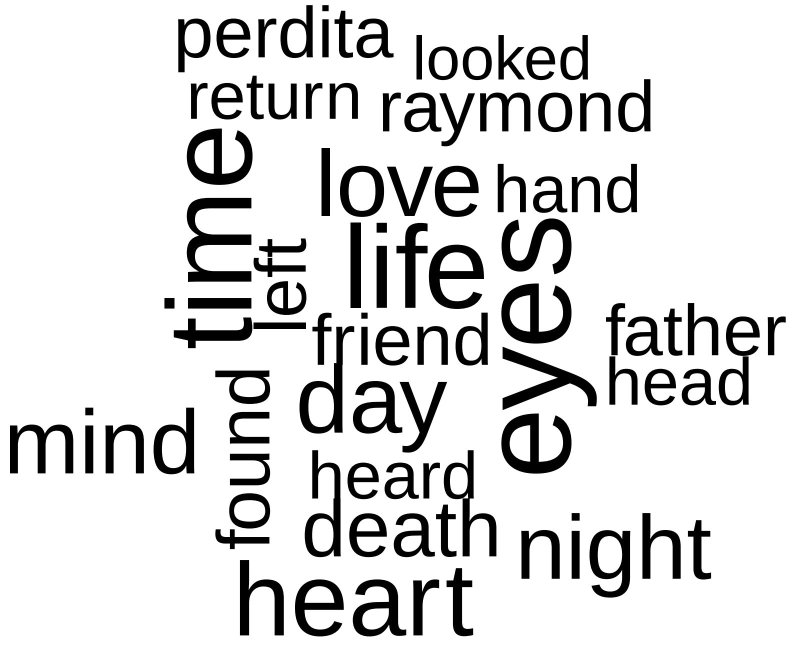


```r
ggsave("../figs/tm.png")
```

```
## Saving 6.5 x 4.5 in image
```

Looks like EAP leans toward topic 1 and 15; HPL leans toward topic 4; MWS leans toward topic 2 and 13. Take MWS as an example and look at all sentences in those two topics. (The order of topic no. is different each time running the above code; therefore, manually choosing the two topics with highest MWS percentage)

```r
# pick top 2 topic that most relates to MWS
MWS <- spooky_author_topic %>%
  group_by(author, topic) %>%
  summarize(tcount = n()) %>%
  spread(author, tcount, fill = 0) %>%
  top_n(2, MWS/(MWS+EAP+HPL)) %>%
  ungroup()

spooky_MWS <- spooky_author_topic %>%
  filter(topic %in% MWS$topic) %>%
  inner_join(spooky, by = c("document" = "id")) %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words, by = "word") %>%
  group_by(word) %>%
  summarize(count = n()) %>%
  ungroup()

wordcloud(spooky_MWS$word, spooky_MWS$count, max.words = 20)
```



Looking at the wordcloud, the most frequent words do match our previous results! In fact, the dominant words matches the words that “mark” MWS’s excerpts: “love”, “life”, “eyes” and “life”; while the rest may not appear in the most frequent words for MWS, those should be probably topic specific words.