Topic Modeling

Haoran Cui

Packages

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(tm)
```

```
Warning: package 'tm' was built under R version 4.4.2
```

```
Loading required package: NLP
```

```
Attaching package: 'NLP'
```

```
The following object is masked from 'package:ggplot2':

    annotate
```

```
library(topicmodels)
```

```
Warning: package 'topicmodels' was built under R version 4.4.2
```

```
library(ldatuning)
```

```
Warning: package 'ldatuning' was built under R version 4.4.2
```

```
library(tidytext)
```

```
Warning: package 'tidytext' was built under R version 4.4.2
```

```
library(Rtsne)
```

```
Warning: package 'Rtsne' was built under R version 4.4.2
```

```
library(ggplot2)
library(wordcloud)
```

```
Warning: package 'wordcloud' was built under R version 4.4.2
```

```
Loading required package: RColorBrewer
```

```
library(RColorBrewer)
```

Data Cleaning

```
movie_data = read.csv("movie_plots.csv")
view(movie_data)
```

```
plots_by_word <- movie_data %>% unnest_tokens(word, Plot)
plot_word_counts <- plots_by_word %>%
  anti_join(stop_words,by = join_by(word)) %>%
  count("Movie Name", word, sort = TRUE)
```

```r
corpus <- VCorpus(VectorSource(movie_data$Plot))

corpus <- corpus %>%
    tm_map(content_transformer(tolower)) %>%
    tm_map(removePunctuation) %>%
    tm_map(removeWords, stopwords("english")) %>%
    tm_map(stripWhitespace)

dtm <- DocumentTermMatrix(corpus)
```

- **Tokenizing and Counting Words**: Splits plot text into words, removes stop words, and counts word frequency for each movie.
- **Text Preprocessing**: Converts plot text to lowercase, removes punctuation, stop words, and extra whitespace.
- **Creating a Document-Term Matrix**: Constructs a matrix where each row is a movie plot and each column represents word frequencies, enabling further analys

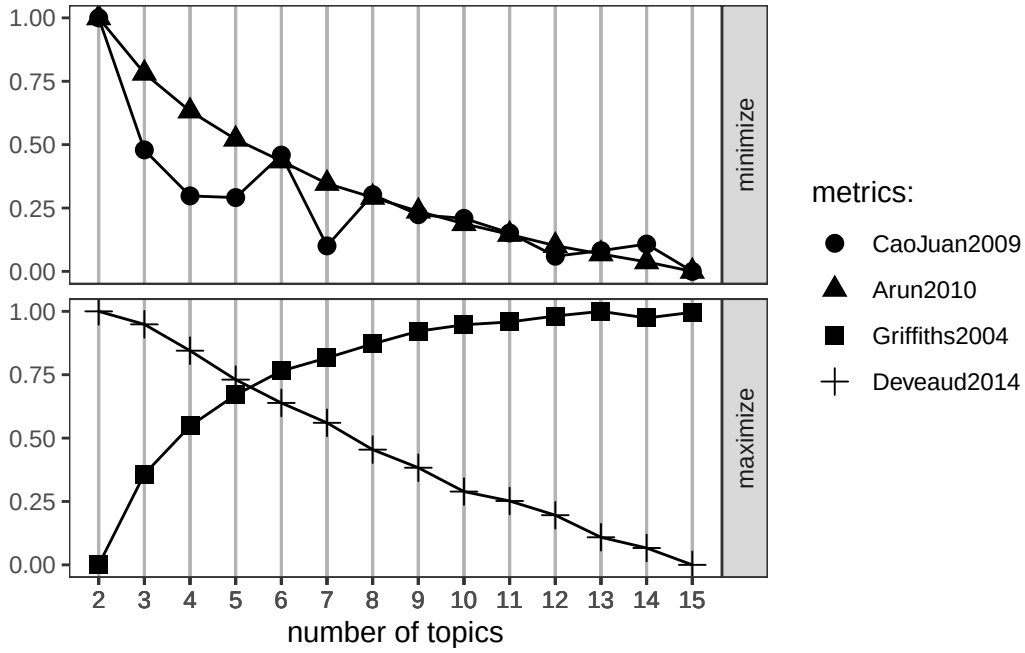
Visualizations

```r
result <- FindTopicsNumber(
    dtm,
    topics = seq(2, 15, by = 1),
    metrics = c("CaoJuan2009", "Arun2010", "Griffiths2004", "Deveaud2014"),
    method = "Gibbs",
    control = list(seed = 321),
    mc.cores = 1L,
    verbose = TRUE
)
```

```
fit models... done.
calculate metrics:
  CaoJuan2009... done.
  Arun2010... done.
  Griffiths2004... done.
  Deveaud2014... done.
```

```
FindTopicsNumber_plot(result)
```

```
Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
of ggplot2 3.3.4.
i The deprecated feature was likely used in the ldatuning package.
  Please report the issue at <https://github.com/nikita-moor/ldatuning/issues>.
```



- **Setting Up Topic Modeling Parameters**: Uses `FindTopicsNumber` to test different topic numbers (from 2 to 15) on the document-term matrix (DTM) with specified evaluation metrics.
- **Defining the Method and Controls**: Sets the topic modeling method to "Gibbs" sampling, uses a random seed for reproducibility, and specifies the number of processing cores.
- **Visualizing Optimal Topics**: Calls `FindTopicsNumber_plot(result)` to plot the results and help identify the optimal number of topics based on the evaluation metrics.

```
#set k to 8
k <- 8
lda_model <- LDA(dtm, k = k, control = list(seed = 321))
```

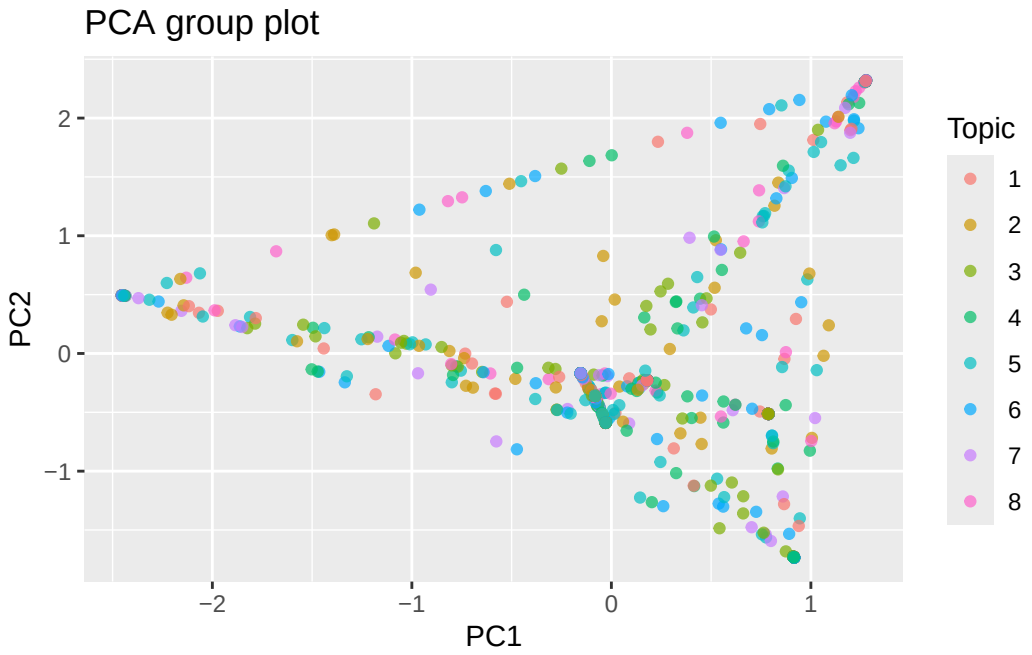
```r
gamma_matrix <- posterior(lda_model)$topics

pca_model <- prcomp(gamma_matrix, center = TRUE, scale. = TRUE)
pca_data <- as.data.frame(pca_model$x)

document_topics <- tidy(lda_model, matrix = "gamma")
doc_topic <- document_topics %>%
    group_by(document) %>%
    slice_max(gamma, n = 1) %>%
    ungroup()

pca_data$Topic <- factor(doc_topic$topic)

ggplot(pca_data, aes(x = PC1, y = PC2, color = Topic)) +
    geom_point(alpha = 0.7) +
    labs(title = "PCA group plot", x = "PC1", y = "PC2")
```

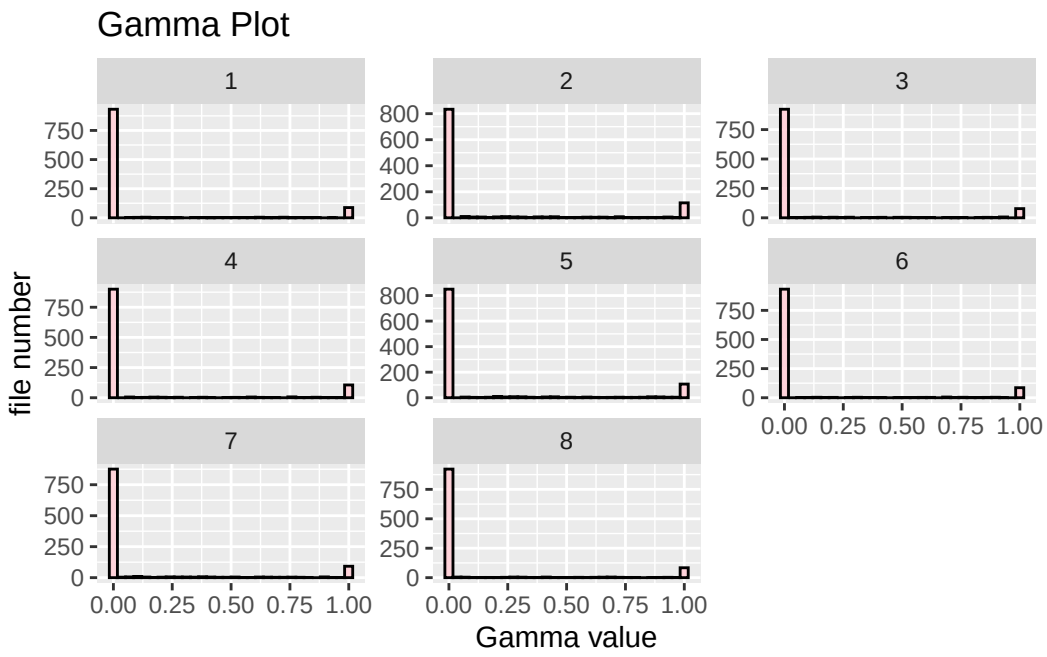


The code fits an LDA model with k=8 topics on a document-term matrix derived from movie plots, extracts the document-topic probabilities (gamma matrix), and uses PCA to reduce the dimensionality for visualization. Each document is assigned to its dominant topic based on the highest probability, and a scatter plot is generated to display the documents in a 2D space with colors representing different topics. The resulting PCA plot reveals how documents

cluster by topic, illustrating the distinctiveness and relationships between topics by showing how closely documents with similar themes group together.

```
document_topics <- tidy(lda_model, matrix = "gamma")

# Gamma plot
ggplot(document_topics, aes(x = gamma)) +
    geom_histogram(bins = 30, fill = "pink", color = "black", alpha = 0.7) +
    facet_wrap(~ topic, scales = "free_y") +
    labs(title = "Gamma Plot", x = "Gamma value", y = "file number")
```



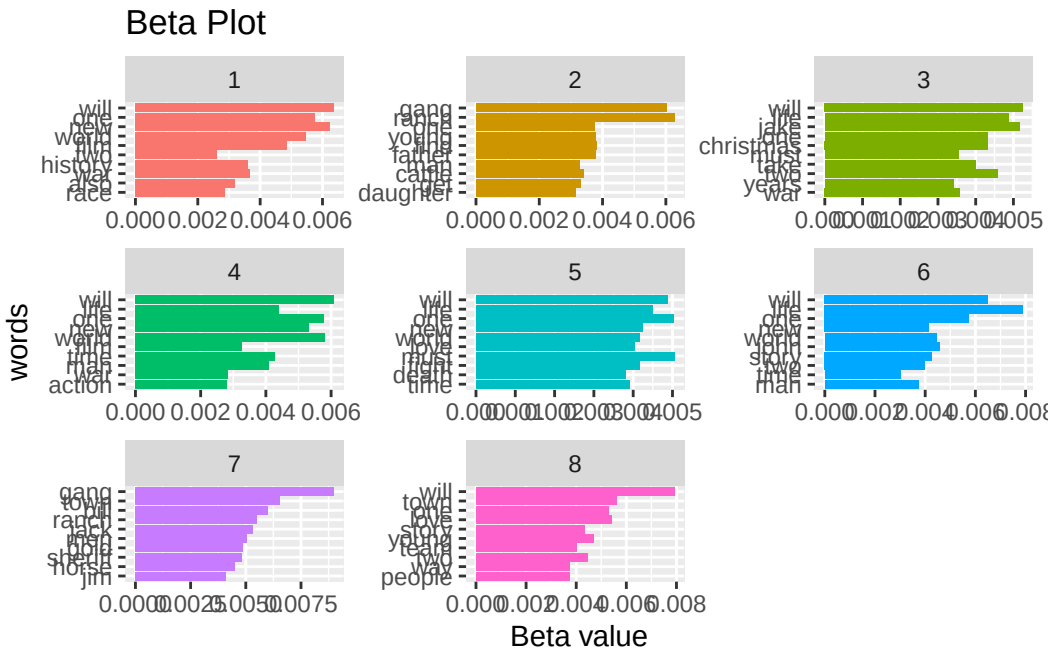
The Gamma Plot visualizes the distribution of topic probabilities (gamma values) for each document across the eight topics in the LDA model. Each subplot shows the gamma values for a specific topic, revealing that most documents have low gamma values for most topics but a high gamma value for one topic, indicating strong association with a single dominant topic. This pattern confirms that the model effectively assigns documents to distinct topics, as expected in well-separated topic modeling, where each document is primarily linked to one topic rather than spread across multiple topics.

```
topic_terms <- tidy(lda_model, matrix = "beta")

top_terms <- topic_terms %>%
    group_by(topic) %>%
```

```
    slice_max(beta, n = 10) %>%
    ungroup() %>%
    arrange(topic, -beta)

ggplot(top_terms, aes(x = reorder(term, beta), y = beta, fill = factor(topic))) +
    geom_bar(stat = "identity", show.legend = FALSE) +
    facet_wrap(~ topic, scales = "free") +
    coord_flip() +
    labs(title = "Beta Plot", x = "words", y = "Beta value")
```



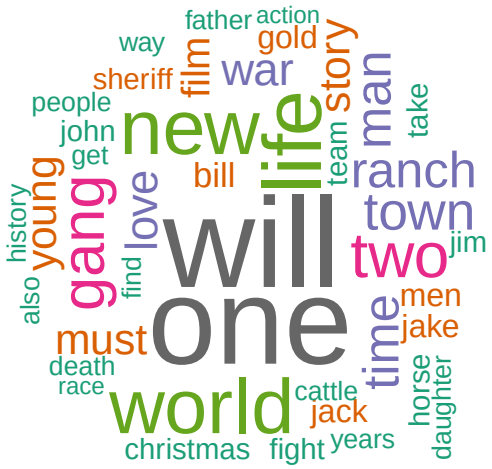
The Beta Plot displays the top 10 terms for each of the eight topics in the LDA model, with each term's importance measured by its beta value. Beta values represent the probability of a word being associated with a particular topic, so higher beta values indicate words that are more representative of the topic. Each subplot corresponds to one topic, showing the most significant words in descending order of their beta values. This plot provides insight into the defining terms of each topic, making it easier to interpret and label the topics based on the prominent words associated with them. This visualization helps understand the thematic structure of each topic in the model.

```
library(RColorBrewer)

all_terms <- top_terms %>%
```

```
    group_by(term) %>%
    summarize(total_beta = sum(beta)) %>%
    arrange(desc(total_beta))

palette <- brewer.pal(8, "Dark2")
wordcloud(words = all_terms$term,
          freq = all_terms$total_beta,
          min.freq = 0.001,
          colors = palette,
          random.order = FALSE,
          rot.per = 0.35,
          scale = c(4, 0.5),
          main = "wordcloud")
```



It generates a word cloud that visualizes the most important terms across all topics in the LDA model based on their aggregated beta values, which represent each term's overall significance in the topic structure. Terms are sized according to their total beta score, with larger words like "will," "one," "world," and "life" indicating higher importance across multiple topics. A color palette from library RColorBrewer is used to add visual distinction to the words, enhancing readability. This word cloud provides a quick, intuitive overview of the central themes and recurring terms in the movie plot dataset.