Analysis_file

Chenxi Li

2025-05-07

Conclusion

1. The language used in comments and the performance of emotions.

I found

What's the most used language in Cyberpunk 2077's steam review? English and Chinese.

What's the distribution of comments amount and ratings with the change of version?

What's the most used word in players' comments?

What's the attention differences between positive and negative comments?

Prepare necessary packages

In this section, we are going to load necessary packages for our further research.

```r
# remove objects
rm(list = ls())
# detach all libraries
detachAllPackages <- function() {
  basic.packages <- c("package:stats", "package:graphics", "package:grDevices", "package:utils", "packag
  package.list <- search()[ifelse(unlist(gregexpr("package:", search())) == 1, TRUE, FALSE)]
  package.list <- setdiff(package.list, basic.packages)
  if (length(package.list) > 0) for (package in package.list) detach(package, character.only = TRUE)
}
detachAllPackages()

# load libraries
pkgTest <- function(pkg) {
  new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
  if (length(new.pkg))
    install.packages(new.pkg, dependencies = TRUE)
  sapply(pkg, require, character.only = TRUE)
}

# here is where you load any necessary packages
# ex: stringr
# lapply(c("stringr"), pkgTest)

lapply(c("tidyverse",
         "cld3", # for language detect
         "ggplot2",
         "reshape2",
         "MetBrewer", # for visualisation color
         "stopwords", # for data tidy, to filter stopword
         "ggwordcloud", # for wordcloud
         "RColorBrewer",
         "tidytext",
         "dplyr", # for sampling
         "lubridate", # for date converse
         "scales",
         "quanteda", # for keyness
         "quanteda.textstats", # for keyness
         "lsa", # for text similarity
         "stm" # for STM analysis
         ), pkgTest)
```

```
## Loading required package: tidyverse
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.2     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 become error
## Loading required package: cld3
```

```
## Warning: package 'cld3' was built under R version 4.3.3
```

```
## Loading required package: reshape2
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
##
## Loading required package: MetBrewer
## Loading required package: stopwords
```

```
## Warning: package 'stopwords' was built under R version 4.3.3
```

```
## Loading required package: ggwordcloud
```

```
## Warning: package 'ggwordcloud' was built under R version 4.3.3
```

```
## Loading required package: RColorBrewer
## Loading required package: tidytext
## Loading required package: scales
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
```

```
##
##     col_factor
##
## Loading required package: quanteda

## Warning: package 'quanteda' was built under R version 4.3.3

## Package version: 4.3.0
## Unicode version: 14.0
## ICU version: 71.1
## Parallel computing: disabled
## See https://quanteda.io for tutorials and examples.
## Loading required package: quanteda.textstats

## Warning: package 'quanteda.textstats' was built under R version 4.3.3

## Loading required package: lsa

## Warning: package 'lsa' was built under R version 4.3.3

## Loading required package: SnowballC

## Warning: package 'SnowballC' was built under R version 4.3.3

## Loading required package: stm
## stm v1.3.7 successfully loaded. See ?stm for help.
##  Papers, resources, and other materials at structuraltopicmodel.com

## [[1]]
## tidyverse
##      TRUE
##
## [[2]]
## cld3
## TRUE
##
## [[3]]
## ggplot2
##    TRUE
##
## [[4]]
## reshape2
##     TRUE
##
## [[5]]
## MetBrewer
##      TRUE
##
## [[6]]
## stopwords
##      TRUE
##
## [[7]]
## ggwordcloud
##        TRUE
##
## [[8]]
## RColorBrewer
##         TRUE
##
## [[9]]
## tidytext
##     TRUE
##
## [[10]]
## dplyr
##  TRUE
##
## [[11]]
## lubridate
##      TRUE
##
## [[12]]
## scales
##   TRUE
##
## [[13]]
## quanteda
##     TRUE
##
## [[14]]
## quanteda.textstats
##               TRUE
##
## [[15]]
##  lsa
## TRUE
##
## [[16]]
##  stm
## TRUE
```

```r
options(scipen = 200)

set.seed(42)
```

```r
# Create a temporary path for data downloading
temp <- tempdir()
data <- file.path(temp, "Cyberpunk-2077-steam-reviews-as-of-aug-8-2024.zip")
unzip <- file.path(temp, "unzip")
dir.create(data, showWarnings = FALSE)

# Judge if there is already exist path. If yes, delete the old one.
if (dir.exists(data)) {
  unlink(data, recursive = TRUE)
}

# Data downloading
kaggle <- paste0(
  "kaggle datasets download -d filas1212/cyberpunk-2077-steam-reviews-as-of-aug-8-2024 ",
  "--path \"", temp, "\""
)
system(kaggle)
```

```r
# Data unzipping and checking the file name
unzip(zipfile = data, exdir = unzip)
list.files(unzip)
```

```
## [1] "Cyberpunk_2077_Steam_Reviews.csv"  "Cyberpunk_2077_Steam_Reviews.xlsx"
```

```r
raw <- read_csv(file.path(unzip, "Cyberpunk_2077_Steam_Reviews.csv"))
```

```
## Rows: 760330 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): SteamID, Rating, Review, Date Posted, Update
## dbl (1): ReviewID
## num (2): Hours Played, Voted Helpful
## lgl (1): Received for Free
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(df)
```

```
##
## 1 function (x, df1, df2, ncp, log = FALSE)
## 2 {
## 3     if (missing(ncp))
## 4         .Call(C_df, x, df1, df2, log)
## 5     else .Call(C_dnf, x, df1, df2, ncp, log)
## 6 }
```

Data Tidy

In this section, we are going to do the data tidy.

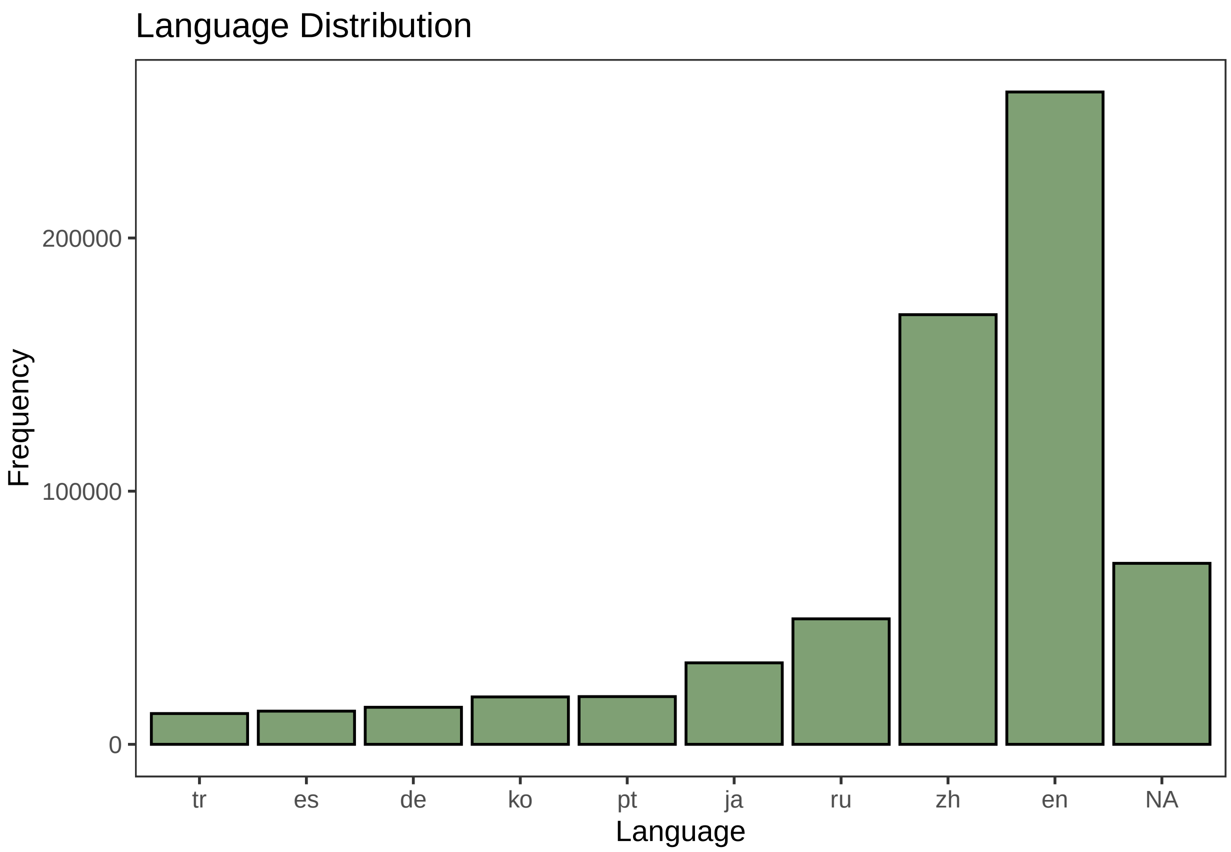
The first step is to clean several redunctant variables like ReviewID and SteamID due to privacy policy.

And also we notice in the previous section that there are several kinds of language in the data frame. So we are going to see the distribution of different language and then I only extract English comments as the main data source of this analysis for convenience.

```r
# Remove row ReviewID and SteamID, this is private information
df <- raw %>%
  select(-ReviewID, -SteamID)

# Detect language in Review
df <- df %>%
  mutate(Language = cld3::detect_language(Review))

# Count the frequency of language
lang_counts <- df %>%
  count(Language, sort = TRUE) %>%
  slice_max(n, n = 10)
```

English and Chinese are two most popular languages used in all comment players. Going further, we can speculate that the largest communities for Cyberpunk 2077 are English and Chinese communities.

```r
# Visulize the distribution of language
ggplot(lang_counts, aes(x = reorder(Language, n), y = n)) +
  geom_col(fill = met.brewer("Cassatt2")[7],
           color = "black") +
  labs(
    title = "Language Distribution",
    x = "Language",
    y = "Frequency",
  ) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
```



In this case, we only analysis for English comments, but probably one day I might do the Chinese part as well :)

```
df_en <- df %>%
  filter(Language == "en")
dim(df_en)
```

```
## [1] 257662      8
```

There are still 257662 observations in English comments, which will lead a very slow reaction during analysis, at least on my computer :(, so I decided to sample the data set to make it accessibility.

The strategy of sampling in this case will be stratified sampling by month, the reasons are:

(1) Based on empirical observations and the theories of game critics, there is a trend of reviews of the game Cyberpunk 2077 differentiating over time, from mixed reviews at the beginning of the game to positive reviews today, so time is an important dimension for understanding the shift in reviews.

(2) Compared with other sampling methods, such as simple random sampling, or non-random cluster

sampling, encounter sampling, etc., stratified sampling is more conducive to controlling the number of reviews at each time point and conducting subsequent statistical analysis.

```
# Prepared data for proportional sampling. Extract Date Posted into the format as Year-Month for sampli
df_en <- df_en %>%
  mutate(
    date_posted = parse_date_time(`Date Posted`,
                                  # "mdy" -> for 08-24-24
                                  # "d-b-y" -> for 18-Aug-24
                                  # "d-B-y" -> for 18-August-24
                                  orders = c("mdy", "d-b-y", "d-B-y")),
    year_month = format(date_posted, "%Y-%m")
  )
```

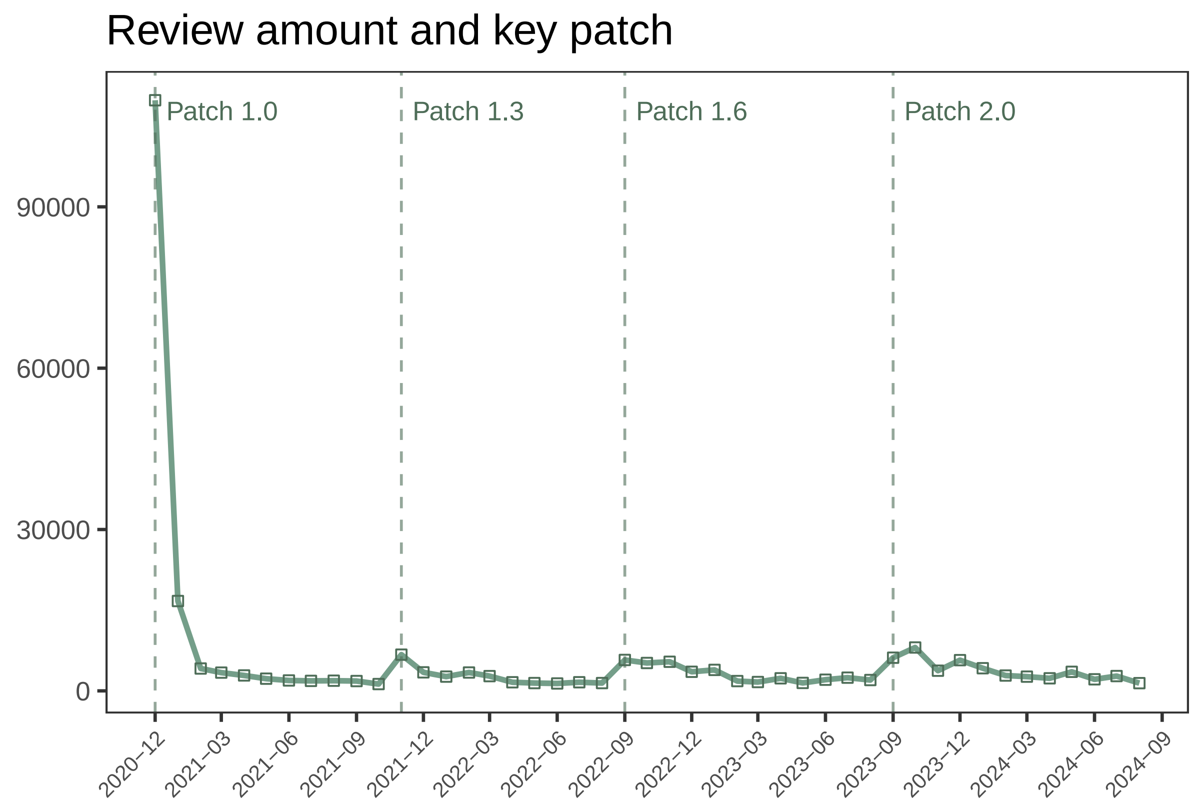
Basic Statistics Description

Before we sampling, it is necessary to see the tendency of comments amount with time. So here is a visualization of time and comments ammount:

We can find except 12/2020, which is the publication date of the cyberpunk 2077, there are also several peak review moment including 11/2021, 09/2022, 09/2023, which correspond to major update patch 1.3, 1.6, and 2.0 respectively. The detailed patch timeline can be found in this website: https://store.steampowered.com/news/app/1091500

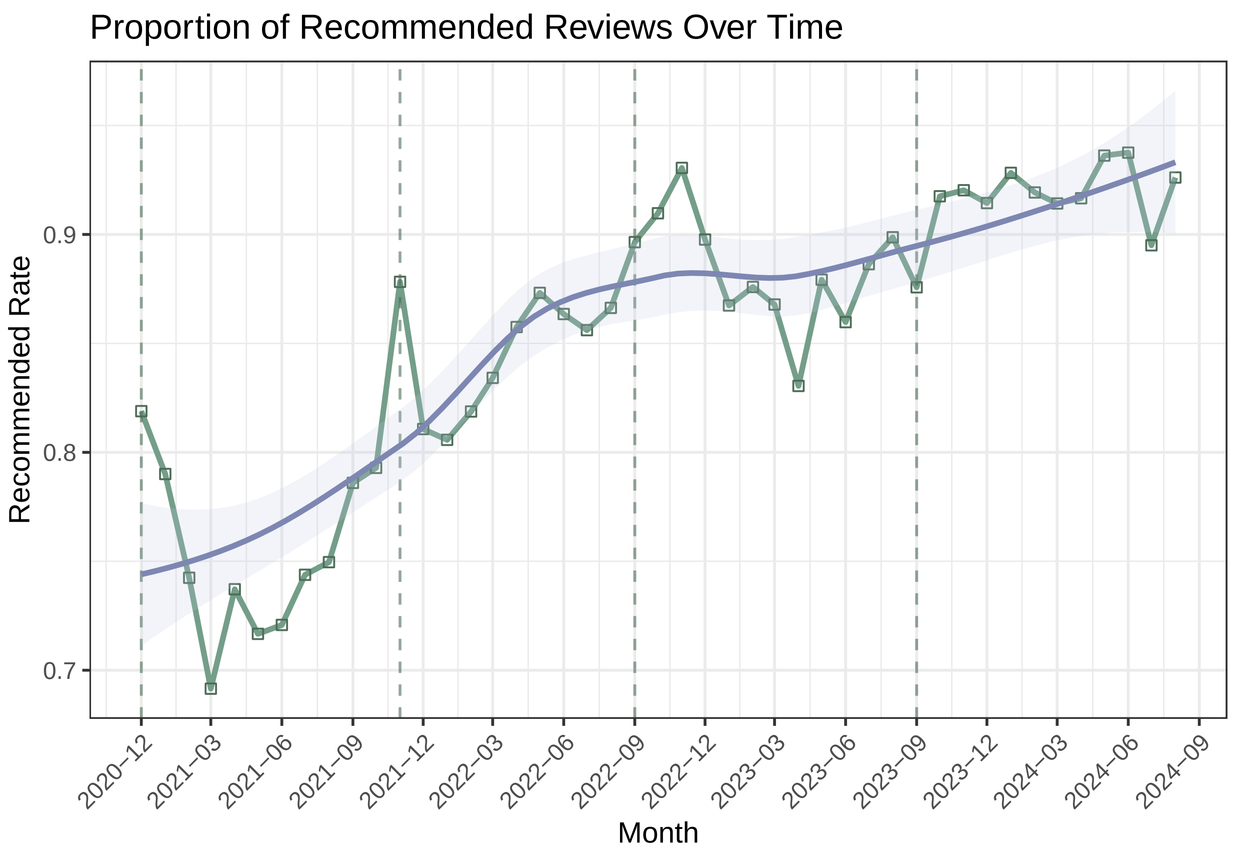
From the plot below we can analysis that every patch brought an increase in discussion, even before the patch relase, there were already discussion increase happened in community.

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
monthly_rating <- df_en %>%
  group_by(year_month, Rating) %>%
  summarise(count = n(), .groups = "drop") %>%
  group_by(year_month) %>%
  mutate(total = sum(count),
         proportion = count / total) %>%
  filter(Rating == "Recommended")

ggplot(monthly_rating, aes(x = as.Date(paste0(year_month, "-01")), y = proportion)) +
  geom_line(color = met.brewer("Monet")[2], size = 1) +
  geom_point(color = met.brewer("Monet")[1], size = 1.5, shape = 0) +
  geom_vline(data = key_time, aes(xintercept = x),
             linetype = "dashed", color = met.brewer("Monet")[1], alpha = 0.6) +
  scale_x_date(date_labels = "%Y-%m", date_breaks = "3 months") +
  geom_smooth(color = met.brewer("Monet")[8], fill = met.brewer("Monet")[9], alpha = 0.2) +
  labs(title = "Proportion of Recommended Reviews Over Time",
       x = "Month",
       y = "Recommended Rate") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



Next, we are going to sample the population to make the data set more clear.

```
# Proportional sampling
df_sam <- df_en %>%
  group_by(year_month) %>%
  mutate(group_n = n()) %>%
  sample_frac(0.05) %>% # sample rate, for every group keep 5%
  ungroup()

dim(df_sam)
```

```
## [1] 12884    11
```

Create a function for data tidy.

```
# Prepare the stopwords library
stopwords <- c(stopwords("en"), "game", "games", "can", "yes", "just")

# Function to text data tidy
review_tidy <- function(text) {
    text <- tolower(text)                    # Change to lower writing style
    text <- gsub("[[:punct:]]", "", text) # Delete character like !@
    text <- gsub("\\s+", " ", text)       # Delete extra space
    text <- trimws(text)                     # Delete extra lines
    text <- gsub("[0-9]+", "", text)      # Drop numbers
    words <- unlist(strsplit(text, "\\s+"))
    cleaned <- words[!tolower(words) %in% stopwords]
    return(paste(cleaned, collapse = " "))
}
```

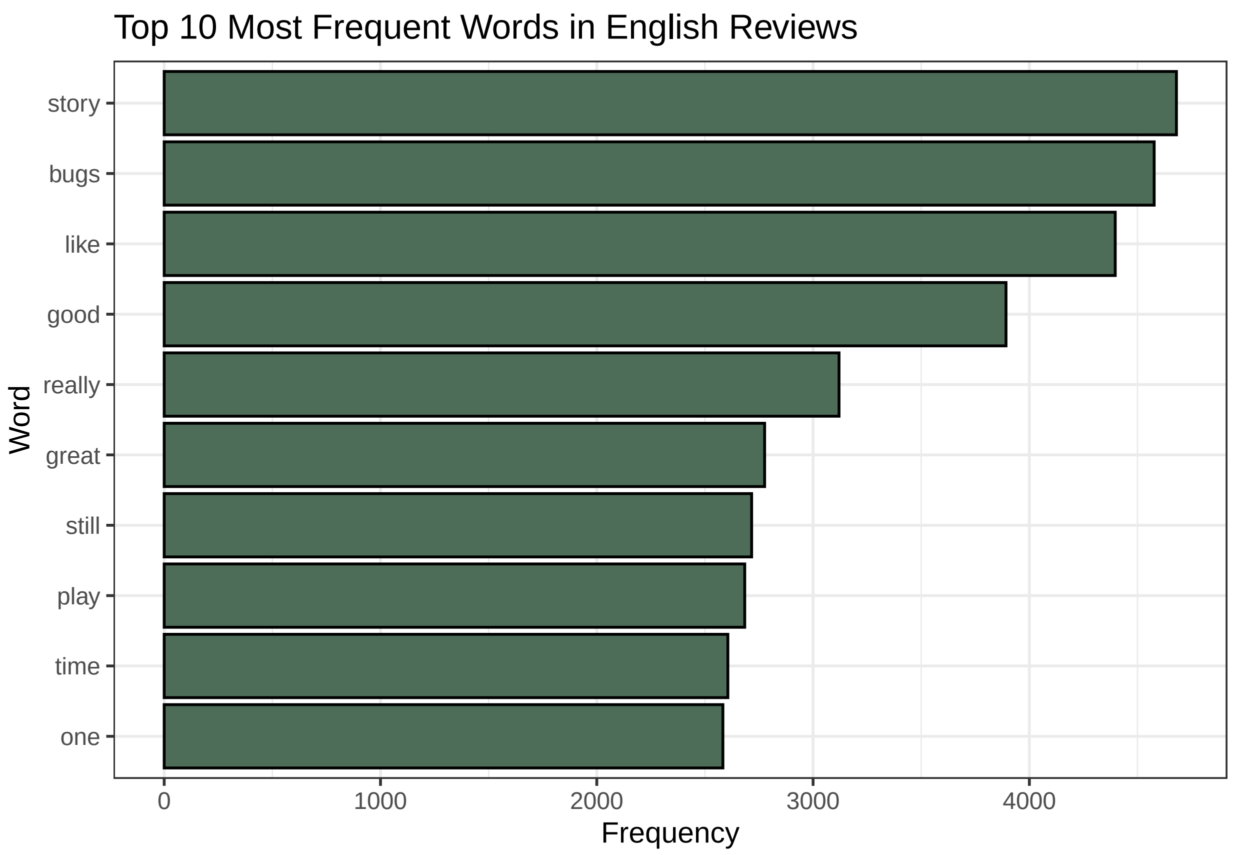
```r
# Data tidy
df_sam <- df_sam %>%
  mutate(Review = sapply(Review, review_tidy))
```

```r
word_freq <- df_sam %>%
  select(Review) %>%
  unnest_tokens(word, Review) %>%
  filter(!word %in% stopwords) %>%
  count(word, sort = TRUE)
head(word_freq)
```

```
## # A tibble: 6 x 2
##   word        n
##   <chr>   <int>
## 1 story    4680
## 2 bugs     4577
## 3 like     4397
## 4 good     3892
## 5 really   3120
## 6 great    2776
```

```r
top_words_bar <- word_freq %>%
  slice_max(n, n = 10)

top_words_cloud <- word_freq %>%
  slice_max(n, n = 200)
```

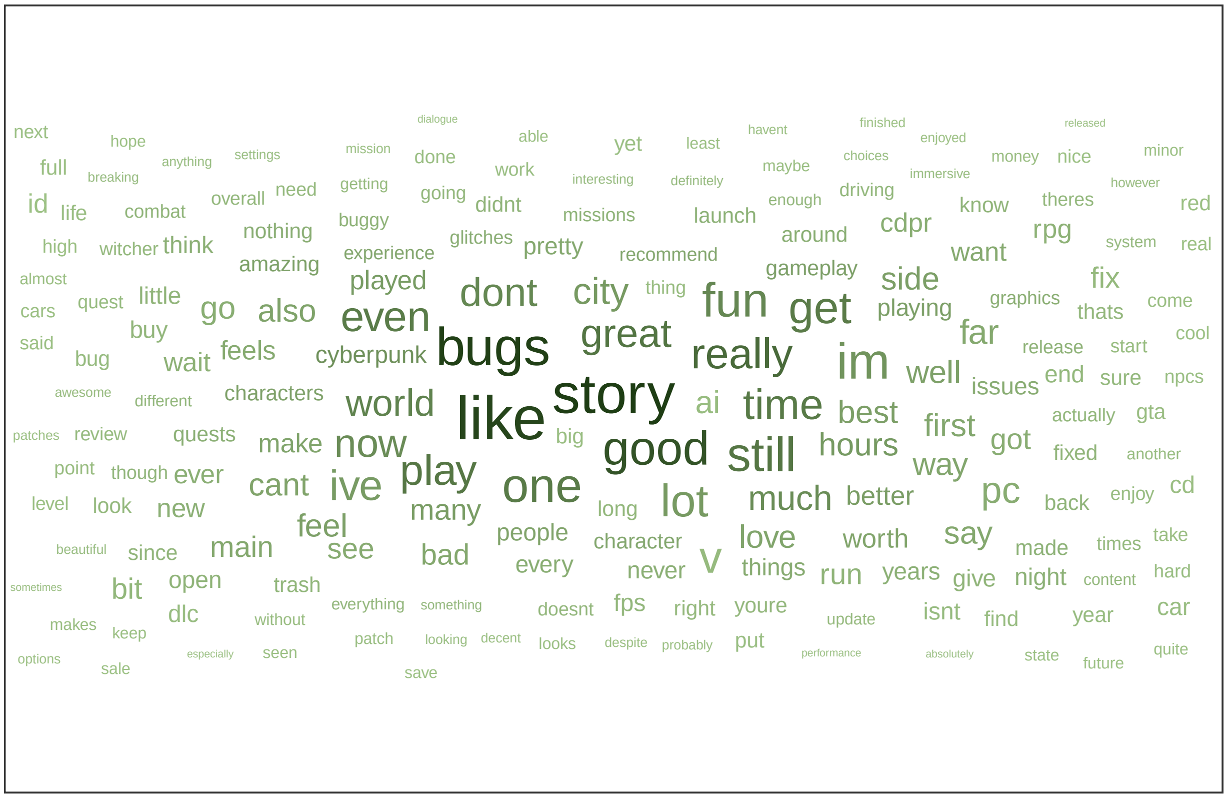
```r
ggplot(top_words_bar, aes(x = reorder(word, n), y = n)) +
  geom_col(fill = met.brewer("Monet")[1], color = "black") +
  coord_flip() +
  labs(title = "Top 10 Most Frequent Words in English Reviews",
       x = "Word",
       y = "Frequency") +
  theme_bw()
```



Word Cloud

```
ggplot(top_words_cloud, aes(label = word, size = n, color = n)) +
  geom_text_wordcloud(area_corr = TRUE, shape = "square") +
  scale_size_area(max_size = 14) +
  scale_color_gradient(low = met.brewer("VanGogh3")[3],
                       high = met.brewer("VanGogh3")[7]) +
  theme_bw() +
  labs(title = "Word Cloud of English Reviews")
```

Word Cloud of English Reviews



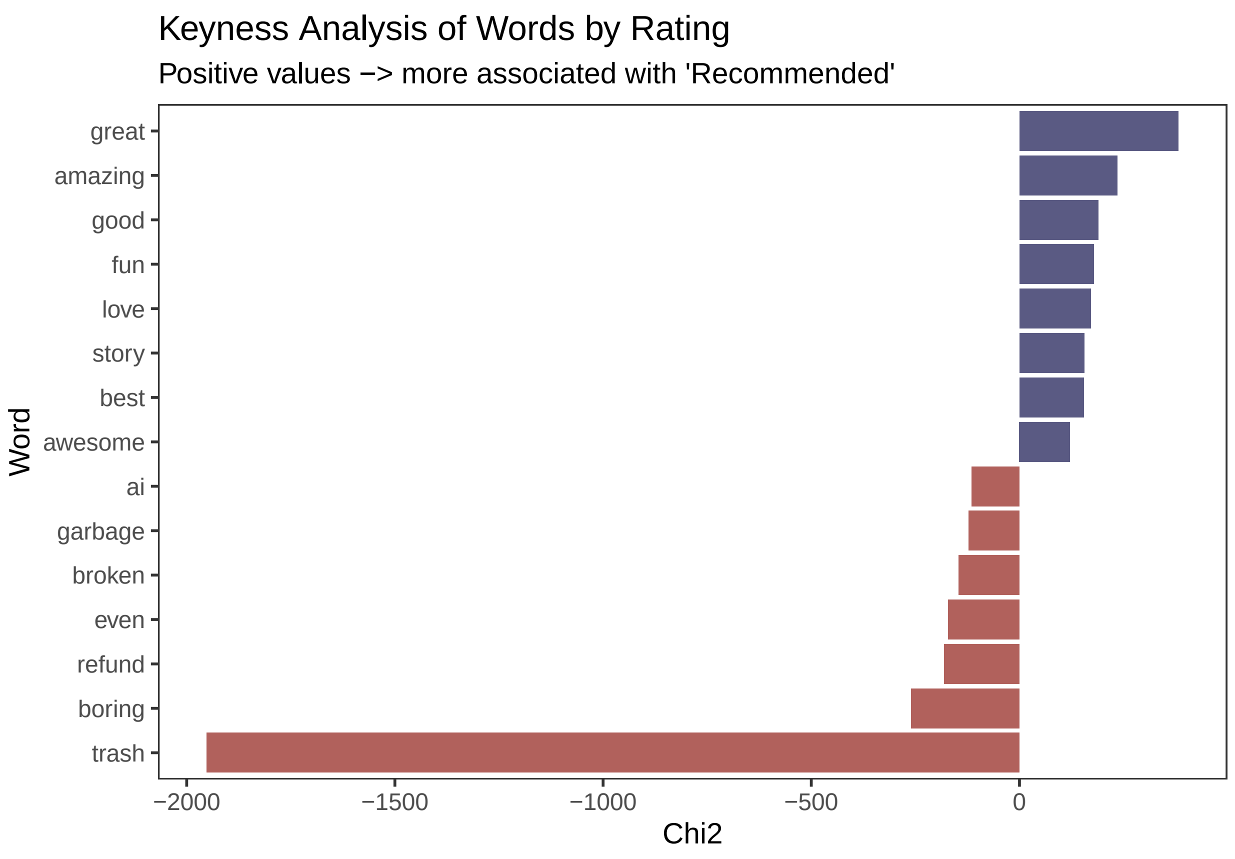
Corpus and Tokens

Using log-likelihood ratio (LLR), keyness method is a nice way to check the differences of word in two groups.

```r
# Build a corpus
corp <- corpus(df_sam, text_field = "Review")
docvars(corp, "Rating") <- df_sam$Rating

# Tokenization
toks <- tokens(corp, remove_punct = TRUE, remove_numbers = TRUE)
toks <- tokens_tolower(toks)
toks <- tokens_remove(toks, stopwords("en"))

# Build dfm and grouped by rating
dfm_grouped <- dfm(toks) %>%
  dfm_group(groups = docvars(corp, "Rating"))
```

Keyness Analysis

```r
# Keyness
keyness_result <- textstat_keyness(dfm_grouped, target = "Recommended")

top_key <- keyness_result %>%
  slice_max(abs(chi2), n = 15) %>%
  mutate(word = reorder(feature, chi2))

ggplot(top_key, aes(x = word, y = chi2, fill = chi2 > 0)) +
  geom_col(show.legend = FALSE) +
  coord_flip() +
  labs(title = "Keyness Analysis of Words by Rating",
       subtitle = "Positive values → more associated with 'Recommended'",
       x = "Word",
       y = "Chi2") +
  scale_fill_manual(values = c("TRUE" = met.brewer("Cassatt1")[8],
                               "FALSE" = met.brewer("Cassatt1")[1])) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
```



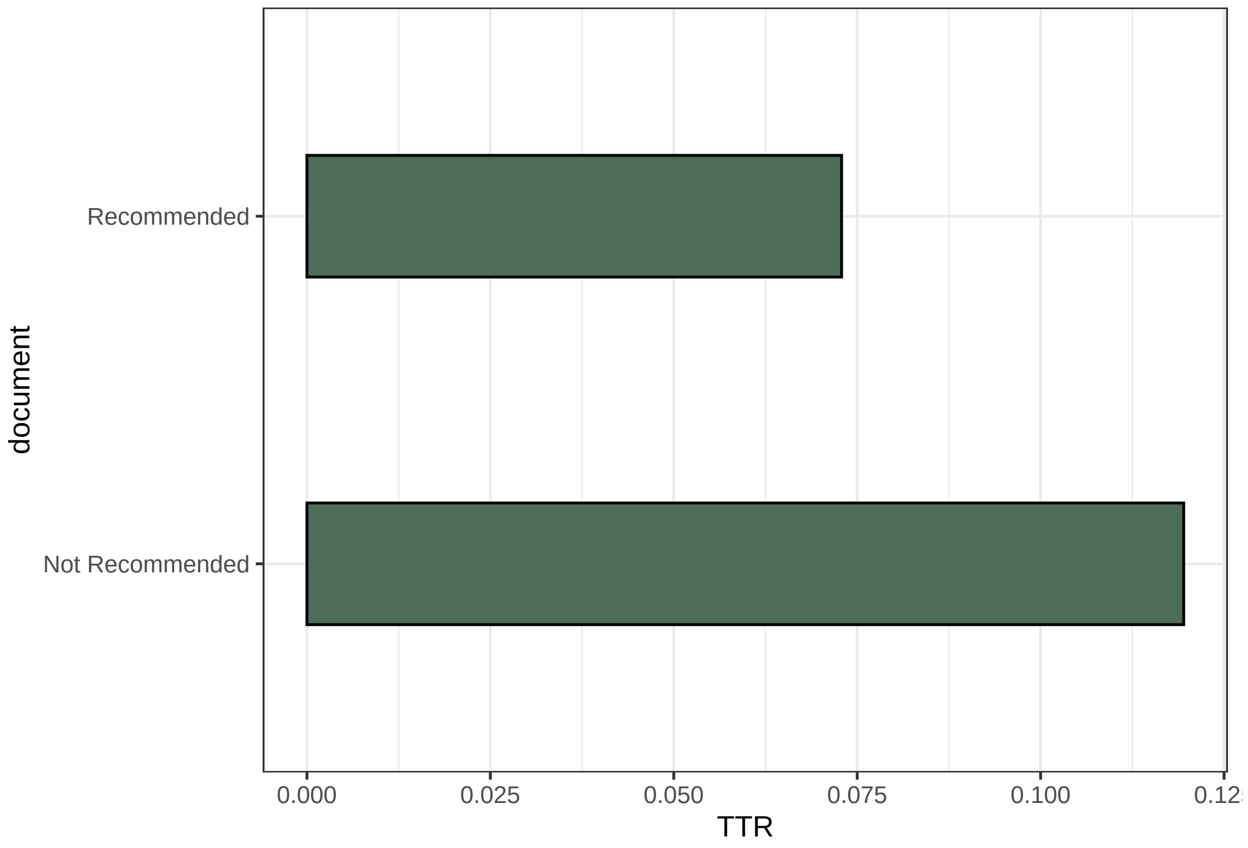
TTR - Readability

So we used the TTR to analysis lexical diversity of player reviews.

The results show that the Recommended reviews have a lower TTR (~0.074), while Not Recommended reviews have a higher TTR (~0.115). This suggests that recommended reviews tend to use simpler and more repetitive language, possibly reflecting common, formula praise. In contrast, not recommended reviews display greater lexical variety, which may indicate more detailed, emotionally charged, or nuanced expressions of dissatisfaction.

```
ttr <- textstat_lexdiv(dfm_grouped, measure = "TTR")

ggplot(ttr, aes(x = document, y = TTR)) +
  geom_bar(stat = "identity", width=0.35,
           color = "black", fill = met.brewer("Monet")[1]) +
  theme_bw() +
```

```r
  coord_flip()
```



Text similarity

In this section, we used a cosine similarity matrix to matures how similar two text documents are, which is based on the angle between word frequency vectors and range from:

- 1: perfectly similar
- 0: no similarity
- -1: completely opposite

For the results of negative and positive comments in different period of cyberpunk 2077, there are the key insights of this data:

1. More Variation in Negative Reviews:

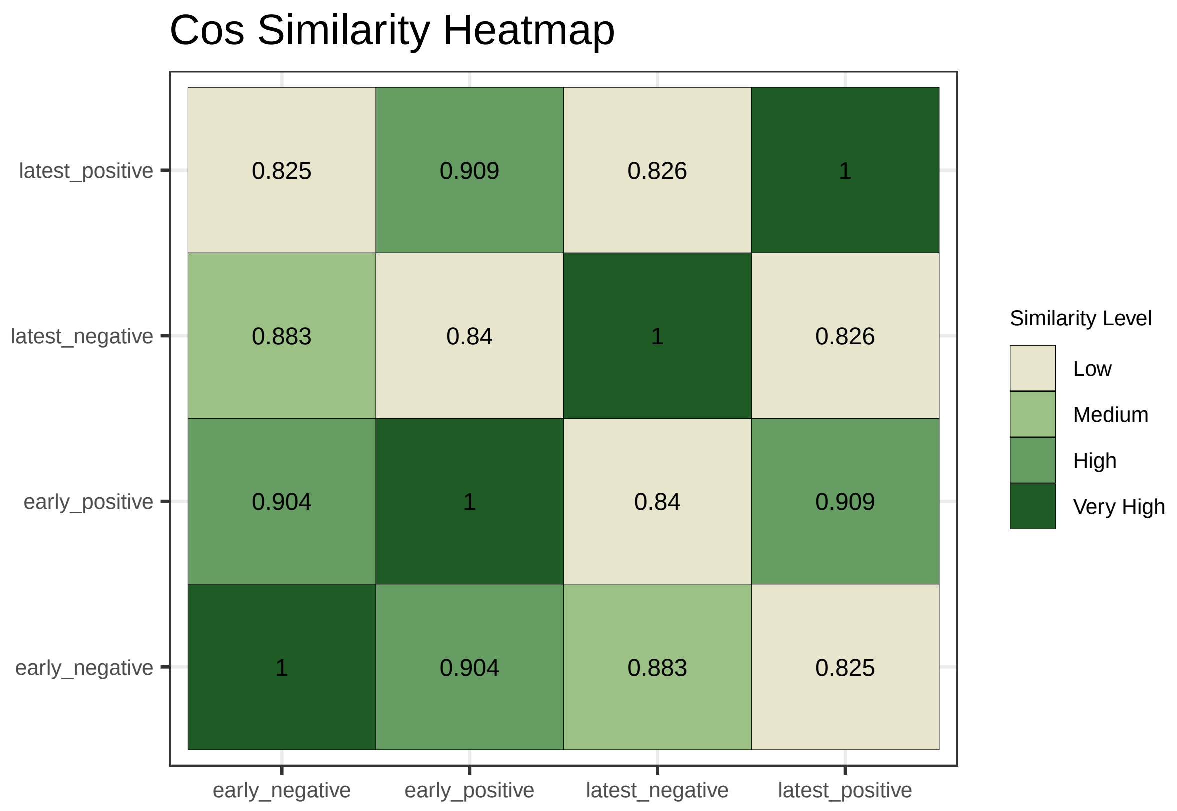
early positive vs. latest positive = 0.909 early negative vs. latest negative = 0.883

With time goes by, seems like there is a direction change in negative view, while for the positive comments are quite similar.

2. Negative and positive comments are still quite similar:

Even across sentiment lines (early positive vs. early negative = 0.904), similarity is high. This may suggest players often mention similar topics regardless ofrating, but frame them differently.

```r
# filter the time range
early_ver <- df_sam %>%
  filter(year_month >= "2020-12", year_month <= "2021-12")

latest_ver <- df_sam %>%
  filter(year_month >= "2023-09", year_month <= "2024-09")

# filter the rating
early_positive <- early_ver %>% filter(Rating == "Recommended")
early_negative <- early_ver %>% filter(Rating == "Not Recommended")
latest_positive <- latest_ver %>% filter(Rating == "Recommended")
latest_negative <- latest_ver %>% filter(Rating == "Not Recommended")

# Integrate to a list
group_texts <- c(
  early_positive = paste(early_positive$Review, collapse = " "),
  early_negative = paste(early_negative$Review, collapse = " "),
  latest_positive = paste(latest_positive$Review, collapse = " "),
  latest_negative = paste(latest_negative$Review, collapse = " ")
)

sim_crop <- corpus(group_texts)

group_dfm <- dfm(tokens(sim_crop, remove_punct = TRUE, remove_numbers = TRUE)) %>%
  dfm_tolower() %>%
  dfm_remove(stopwords("en")) %>%
  dfm_trim(min_termfreq = 5)

similarity_matrix <- textstat_simil(group_dfm, margin = "documents", method = "cosine")

sim_mat <- as.matrix(similarity_matrix)

sim_df <- as.data.frame(sim_mat) %>%
  rownames_to_column("doc1") %>%
  pivot_longer(-doc1, names_to = "doc2", values_to = "similarity")

# Due to the similarity is very close, so we use a categorical color platte
sim_df <- sim_df %>%
  mutate(sim_level = cut(similarity,
                         breaks = c(0.80, 0.85, 0.90, 0.95, 1.00),
                         labels = c("Low", "Medium", "High", "Very High")))

ggplot(sim_df, aes(x = doc1, y = doc2, fill = sim_level)) +
  geom_tile(color = "black") +
  scale_fill_manual(values = c("Low" = met.brewer("VanGogh3")[1],
                               "Medium" = met.brewer("VanGogh3")[3],
                               "High" = met.brewer("VanGogh3")[4],
                               "Very High" = met.brewer("VanGogh3")[6]),
                    name = "Similarity Level") +
  geom_text(aes(label = round(similarity, 3)), color = "black", size = 3) +
  labs(title = "Cos Similarity Heatmap",
       x = " ", y = " ") +
  theme_bw(base_size = 13) +
  theme(axis.text.x = element_text(size = 8),
        axis.text.y = element_text(size = 8),
        legend.text = element_text(size = 8),
        legend.title = element_text(size = 8))
```



STM Analysis

Using STM analysis, we can find there are 4 topics in the comments of player's discussions:

- Topics 1: story, bugs, good, really, great

The key word including positive aspect like good, really, but bugs are also found in this section. I interpret that in this topics player's may express good emotions in the game's story, but also suggests the bugs in the game. In a word, I will conclude this topics as **Stories are attracted but bugs are existed**.

- Topics 2: play, time, played, feel, say

This topics focused more on the user's experience and the game itself due to there are subjective feeling word like feel, say in this topics. I will conclude this topics as **The feeling when players are playing the game**.

- Topics 3: ive, best, want, year, launch

The expected word in this topics (ive probably refers to I've), as well as "best" and "want" indicate to the willings of players. So I will conclude this topics as **Exceptions on game's update or long-term performance**.

- Topics 4: like, world, fun, people, bad

There are opposite words in this topics (fun vs. bad), which indicate the conflict in the comments. The word "world" and "people" in this topics refers to the construct of the game's world and the npcs in it, which means players may hold two opposite sides of view in the game design. So I will conclude this topics as **The interactive of game's world**.

```
# File pre-processed
processed <- textProcessor(documents = df_sam$Review,
                           metadata = df_sam,
                           lowercase = TRUE,
```

```r
                           removestopwords = TRUE,
                           removenumbers = TRUE,
                           removepunctuation = TRUE,
                           stem = FALSE)
```

```
## Building corpus...
## Converting to Lower Case...
## Removing punctuation...
## Removing stopwords...
## Removing numbers...
## Creating Output...
```

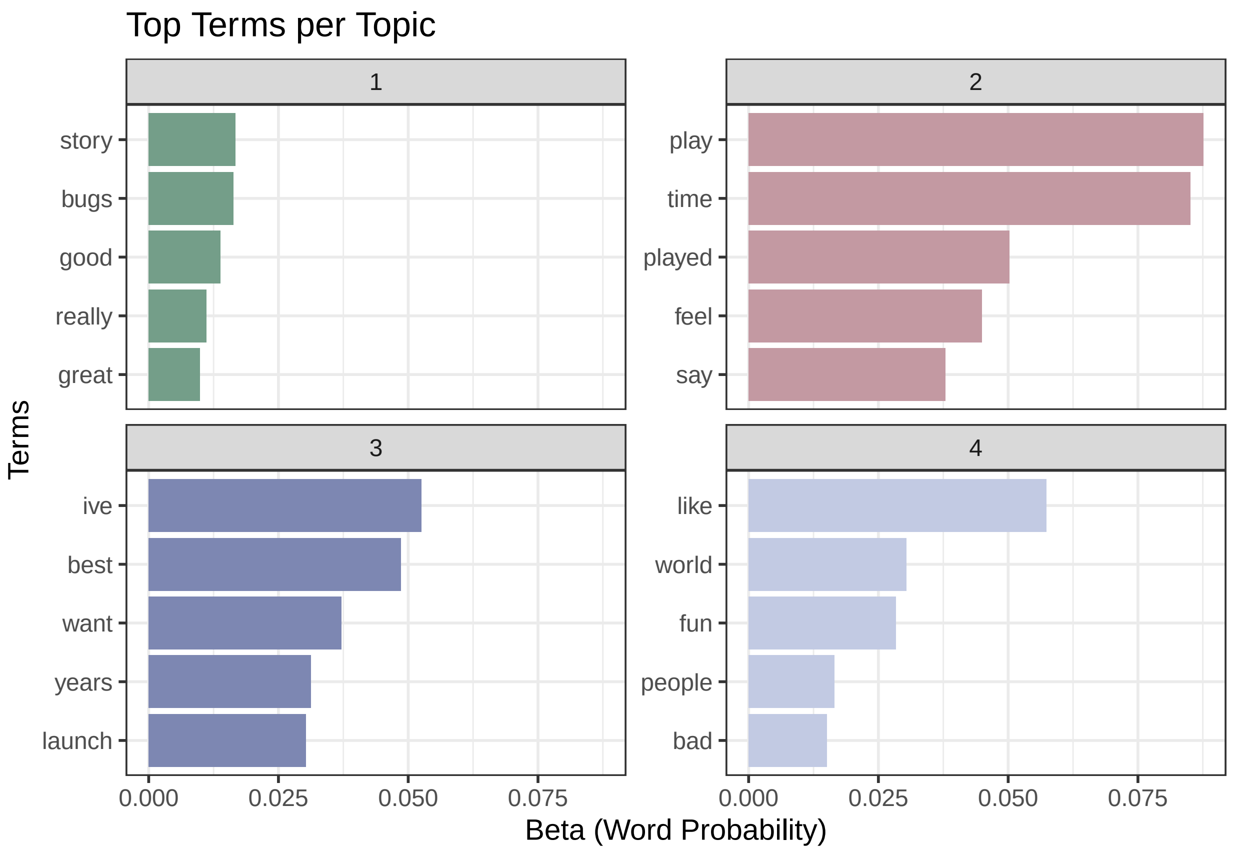
```r
# Create a file matrix
out <- prepDocuments(processed$documents, processed$vocab, processed$meta)
```

```
## Removing 17921 of 29016 terms (17921 of 373962 tokens) due to frequency
## Removing 36 Documents with No Words
## Your corpus now has 12621 documents, 11095 terms and 356041 tokens.
```

```r
docs <- out$documents
vocab <- out$vocab
meta <- out$meta

meta$year_month <- as.Date(paste0(meta$year_month, "-01"))
meta$month_num <- as.numeric(meta$year_month)
```

```r
stm_model <- suppressMessages(
  suppressWarnings(
    capture.output({
      model <- stm(documents = docs,
                   vocab = vocab,
                   K = 4,
                   prevalence = ~ Rating + s(month_num),
                   data = meta,
                   max.em.its = 20,
                   init.type = "Spectral")
    }, file = NULL)
  )
)
stm_model <- model
```

```r
td <- tidytext::tidy(stm_model)

# Every theme's top 5 words
top_terms <- td %>%
  group_by(topic) %>%
  slice_max(beta, n = 5) %>%
  ungroup() %>%
  mutate(term = reorder_within(term, beta, topic))

# ggplot
ggplot(top_terms, aes(x = term, y = beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free_y") +
  scale_x_reordered() +
  coord_flip() +
  labs(x = "Terms", y = "Beta (Word Probability)",
       title = "Top Terms per Topic") +
  scale_fill_manual(values = c(
    met.brewer("Monet")[2], met.brewer("Monet")[5],
    met.brewer("Monet")[8], met.brewer("Monet")[9])) +

  theme_bw()
```



The plot below shows the proportion change over time of the four topics.

Topics V1 (Story) shows a clear declining trend. It starts as the most dominant topics with a proportion around 0.67, but gradually decrease toward 2024, so a conclusion will be with time goes on, the discussion of story plot is decresing.

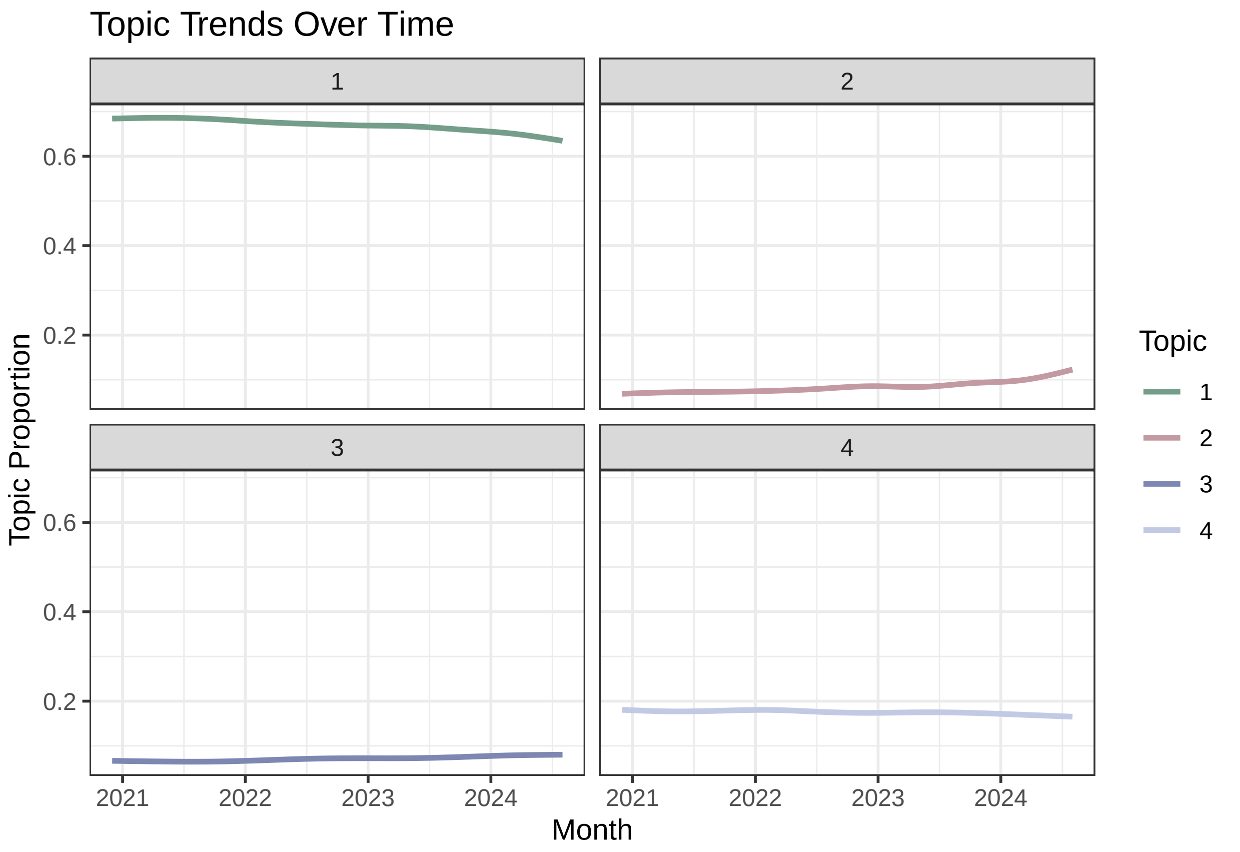
Topics V2 (Feeling) remains relatively stable at a low level in earlier years but increases slightly in 2024, which indicates people focus more on the ego feeling in recent months.

Topics V3 (Expectation) and V4 (Interactive) both stay flat and low, with only minor fluctuations.

```r
# Extract the distribution of themes
theta_df <- as.data.frame(stm_model$theta)
theta_df$year_month <- meta$year_month
theta_df$Rating <- meta$Rating

theta_long <- theta_df %>%
  pivot_longer(cols = starts_with("V"),
               names_to = "topic",
               values_to = "proportion",
               names_transform = list(topic = readr::parse_number))
```

```r
# Visualisation the tendency of 4 topics via time
ggplot(theta_long, aes(x = as.Date(paste0(year_month, "-01")),
                       y = proportion,
                       color = factor(topic))) +
  geom_smooth(method = "gam", se = FALSE) +
  facet_wrap(~ topic, ncol = 2) +
  scale_color_manual(values = c(
    met.brewer("Monet")[2], met.brewer("Monet")[5],
    met.brewer("Monet")[8], met.brewer("Monet")[9])) +
  labs(x = "Month", y = "Topic Proportion",
       title = "Topic Trends Over Time",
       color = "Topic") +
  theme_bw()
```

```
## `geom_smooth()` using formula = 'y ~ s(x, bs = "cs")'
```



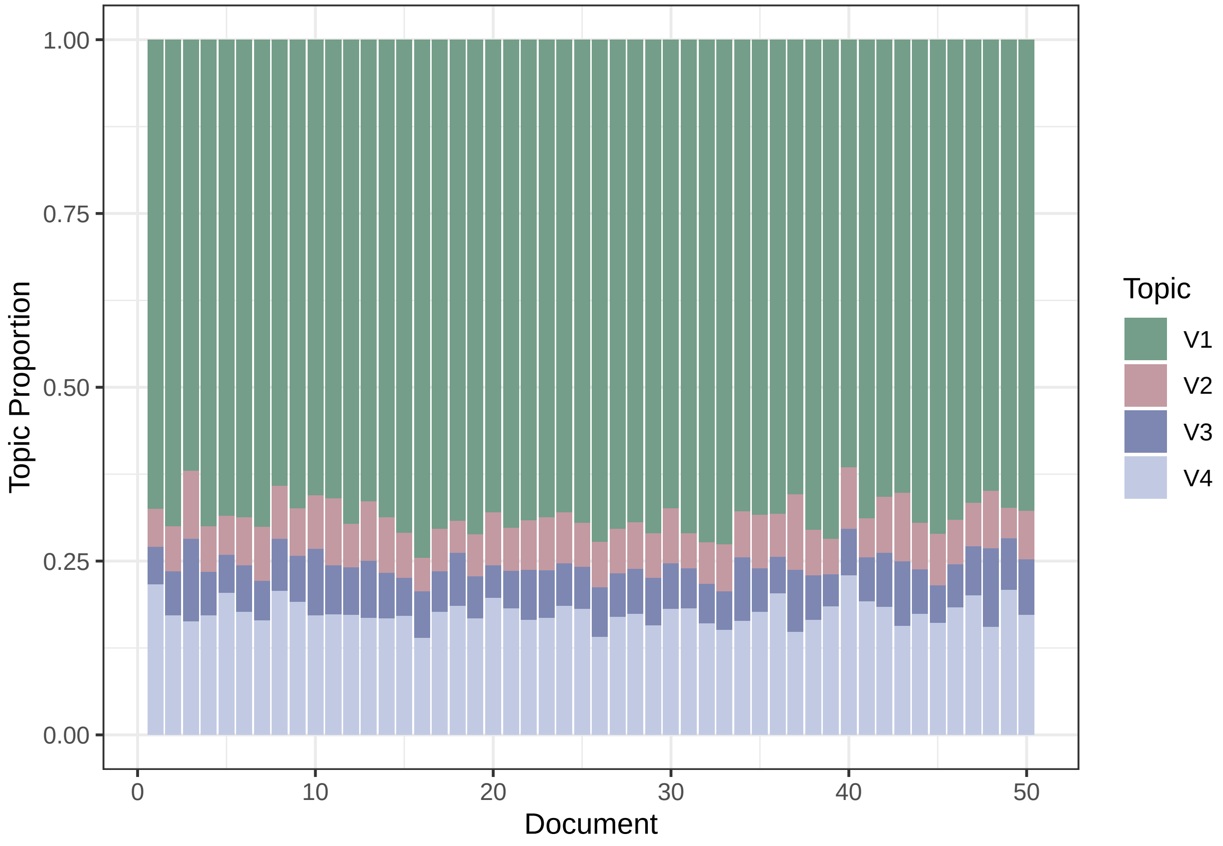
The bar chart below displays the topic proportion distribution of the first 50 documents in the database. Each bar represents a document, and the colored segments within each bar show the relative contribution of each topic to the document.

Key insights of the bar is firstly, V1 takes the overwhelmingly positions of the most documents. Usually in every document it accounts for the majority proportion, often over 60%.

Topics 2, 3, and 4 make up smaller proportions, with their contributions varying slightly across documents.

The mix of multiple topics suggests that each document can be about multiple themes to different extents while the consistent dominance of Topic 1 may indicate it represnts a common theme across the corpus, which is the **Story** aspects in player experience.

```r
# Find the distribution of document themes
theta_long <- stm_model$theta %>%
  as.data.frame() %>%
  mutate(doc_id = row_number()) %>%
  pivot_longer(-doc_id, names_to = "topic", values_to = "proportion")

# Visulisation top 50 documents
ggplot(theta_long %>% filter(doc_id <= 50),
       aes(x = doc_id, y = proportion, fill = topic)) +
  geom_bar(stat = "identity") +
  labs(x = "Document", y = "Topic Proportion", fill = "Topic") +
  scale_fill_manual(values = c(
    met.brewer("Monet")[2], met.brewer("Monet")[5],
    met.brewer("Monet")[8], met.brewer("Monet")[9])) +
  theme_bw()
```


Topic Proportion
Document
Topic
V1
V2
V3
V4