Web Scraping in R

Sahil Singh

2023-02-24

Scrape https://www.azlyrics.com/b/beatles.html for all Beatles song lyrics to analyze and visualize the top occurring unigrams, bigram and trigrams

Code for scraping (do not run)

```
##########################
#  ---RUN ONLY ONCE---  #
##########################

# # Main page
# x <- scan("https://www.azlyrics.com/b/beatles.html", what = "", sep = "\n")
#
# ## Getting all urls from the main page for each song
# urls <- list()
# for (i in 1:length(x)) {
#   # extract URLs using regular expressions
#   urls[[i]] <- ifelse(grepl('href="/lyrics/beatles', x[i]),
#                       paste0("https://www.azlyrics.com", ## Completing link
#                              gsub('.*href="(/lyrics/beatles.*?)".*',
#                                   '\\1',
#                                   x[i])),
#                       NA)
# }
# urls <- urls[is.na(urls) == F] # Dropping NAs
#
# ## Getting lyrics from all the urls
#
# lyrics <- list()
#
# # Define the time interval in seconds for loop delay
# time_interval <- 330 # 5.5 minutes
#
# # Running loop at specified time intervals as IP Address updates in background
# for (j in 1:48) {
#   # Scraping each link to extract the lyrics
#   for (i in (length(lyrics)+1):(length(lyrics)+9)) {
#     y <- scan(urls[[i]], what = "", sep = "\n")
#     a <- grep("<!-- Usage of azlyrics.com content by any third-party lyrics provider is prohibited by
#     b <- grep("</div>", y[a+1:length(y)])[1] + a
#     slyrics <- y[(a+1):(b-1)]
#     lyrics[[i]] <- lapply(slyrics, function(x) gsub("<br>", "", x))
#   }
```

```
#   # Pause the execution of the loop for 5.5 minutes to update IP Address
#   Sys.sleep(time_interval)
# }
#
# # saving file for future use
# saveRDS(lyrics, file="lyrics")

#########################
# ^^^ DONT RUN AGAIN ^^^ #
#########################
```

Load scraped RDS file (to avoid scraping again and again)

```
lyrics <- readRDS("lyrics")
```

Finding top ten words (unigrams) excluding stopwords

```
mylrc <- lyrics

mylrc <- lapply(mylrc, function(x) {
  # Convert to lowercase
  x <- tolower(x)
  # Remove special characters
  x <- gsub("[^[:alpha:][:space:]]", "", x)
  # Split into words
  x <- unlist(strsplit(x, "\\s+"))
  # Return modified string
  return(x)
})

# Unlist the output
mylrc <- unlist(mylrc)

# Remove empty strings
mylrc <- mylrc[mylrc != ""]

# Scan stop words from "https://gist.githubusercontent.com/sebleier/554280/raw/7e0e4a1ce04c2bb7bd41089c
stopwords <- scan("https://gist.githubusercontent.com/sebleier/554280/raw/7e0e4a1ce04c2bb7bd41089c9821d
          what = "",
          sep = "\n")

# Remove stop words
mylrc <- mylrc[!(mylrc %in% stopwords)]

# Count the number of occurrences of each word
word_counts <- table(mylrc)

# Sort the word counts in descending order
word_counts <- sort(word_counts, decreasing = TRUE)

# Displaying top ten words
word_counts[1:10]
```

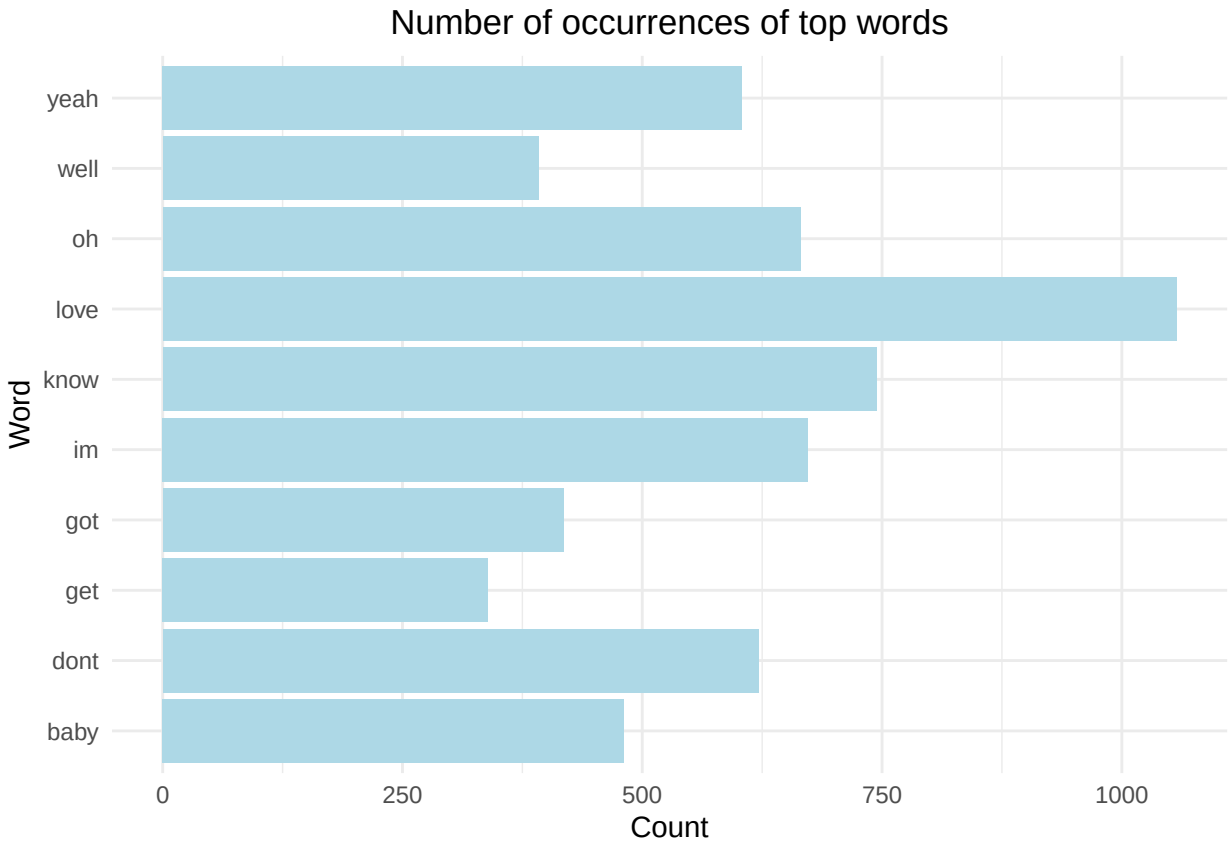
```
## mylrc
```

```
## love know   im   oh dont yeah baby  got well  get
## 1057  744  673  665  621  604  481  418  392  339
```

Visualizing the outcome

```
# Converting to dataframe for ease in visualizations
topwords <- data.frame("words" = as.character(names(word_counts)),
                       "counts" = as.numeric(word_counts))
```

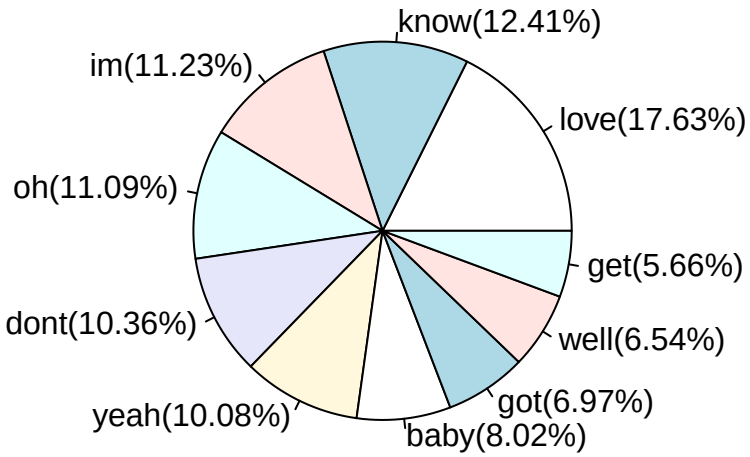
```
# Horizontal bar plot
ggplot(topwords[1:10,], aes(x = counts, y = words)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(x = "Count", y = "Word", title = "Number of occurrences of top words") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



Bar plot

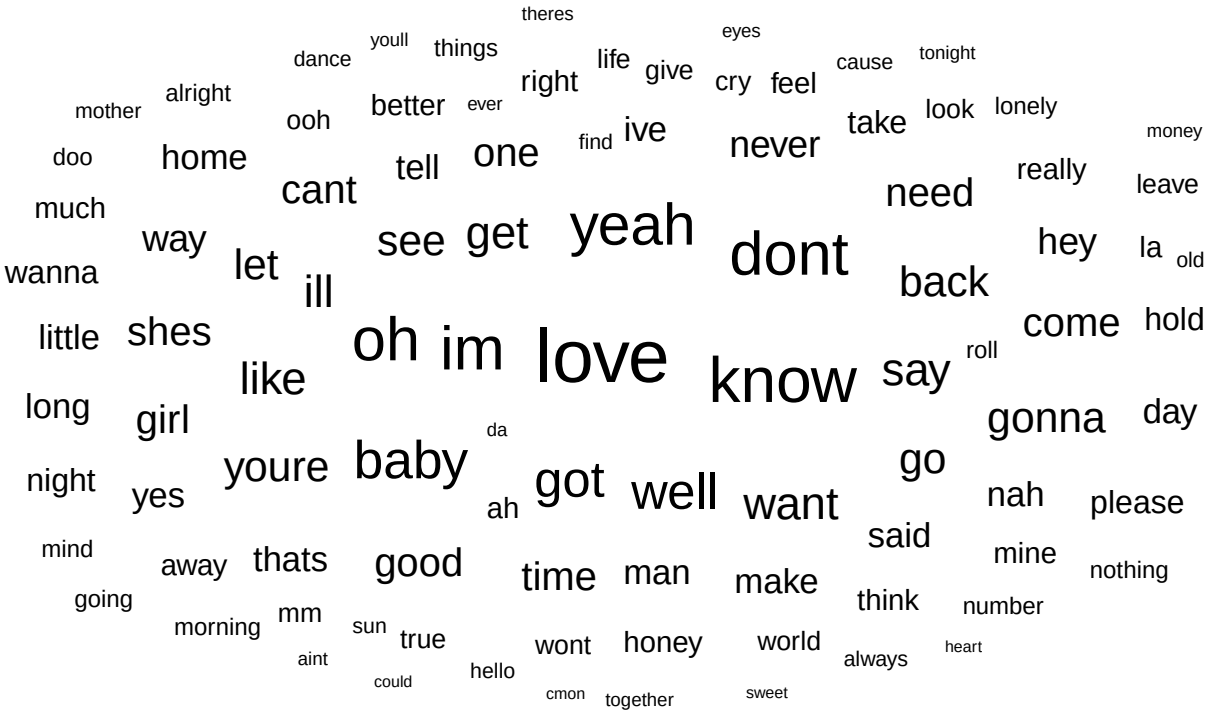
```
percentages <- round(topwords$counts[1:10]/sum(topwords$counts[1:10])*100, 2)
labels <- paste(topwords$words[1:10], "(", percentages, "%)", sep = "")

pie(topwords$counts[1:10], labels = labels, main = "Word Occurrences")
```



Pie Chart

```
# Create word cloud using ggplot2
ggplot(topwords[1:100,], aes(label = words, size = counts)) +
  geom_text_wordcloud() +
  scale_size(range = c(2, 10)) +
  theme_minimal()
```



Word cloud

Finding top bigrams

```
mylrc2 <- lyrics
```

```r
mylrc2 <- lapply(mylrc2, function(x) {
  # Convert to lowercase
  x <- tolower(x)
  # Remove special characters
  x <- gsub("[^[:alpha:][:space:]]", "", x)
  # Split into words
  x <- unlist(strsplit(x, "\\s+"))
  # Return modified string
  return(x)
})

# Unlist the output
mylrc2 <- unlist(mylrc2)

# Remove empty strings
mylrc2 <- mylrc2[mylrc2 != ""]
```

Preprocessing data

```r

# Creating bigrams
mylrc2 <- paste(mylrc2[-length(mylrc2)], mylrc2[-1], sep = " ")

# Count the number of occurrences of each bigram
bigram_counts <- table(mylrc2)

# Sort the bigram counts in descending order
bigram_counts <- sort(bigram_counts, decreasing = TRUE)

# Displaying top 20 bigrams
bigram_counts[1:20]
```

Displaying top bigrams

```
## mylrc2
##  you know    in the  love you     and i     got a   nah nah    i want     to be
##       393       297       206       203       178       168       163       159
##     to me    if you yeah yeah    when i     oh oh    i dont   love is     it be
##       159       157       156       148       126       122       122       121
##    let it    i love   oh yeah    on the
##       121       119       118       118
```

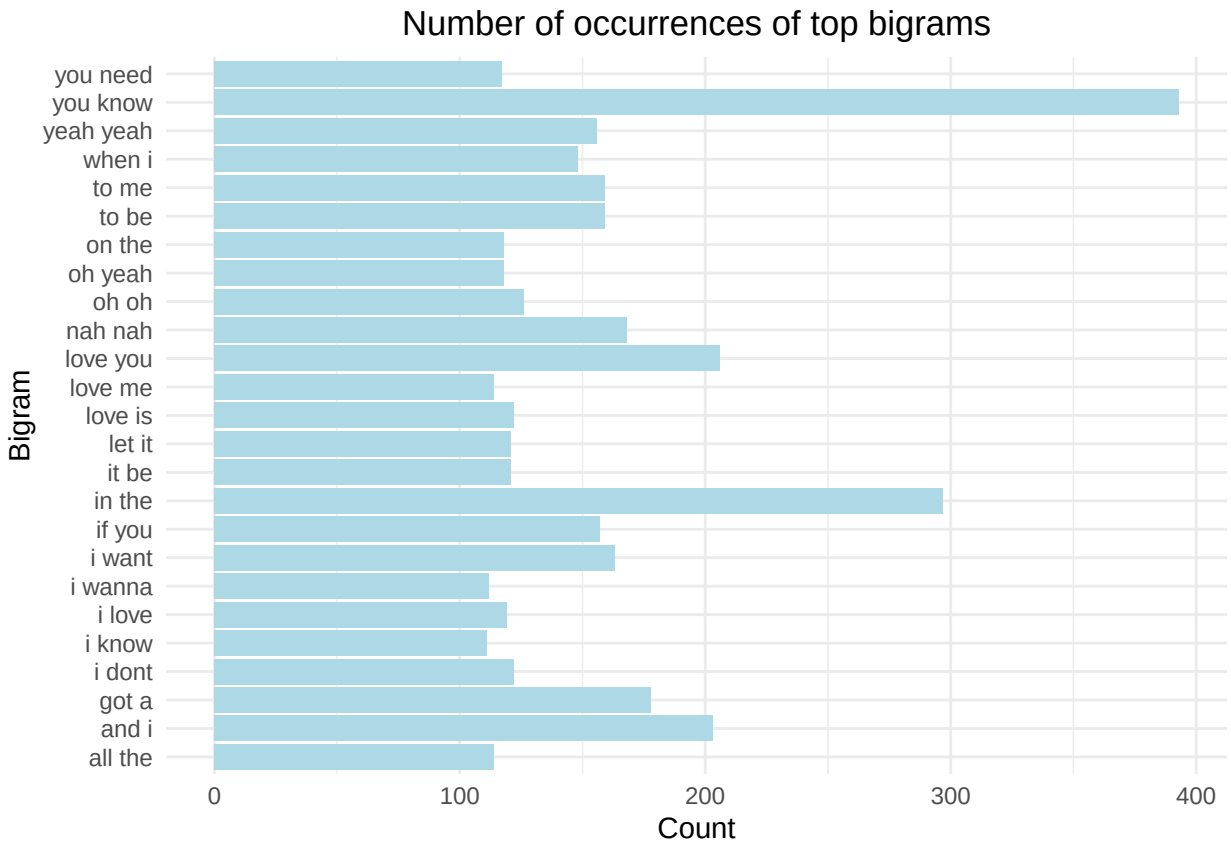
Visualizing the outcome

```r
# Converting to dataframe for ease in visualizations
topbigrams <- data.frame("bigrams" = as.character(names(bigram_counts)),
                         "counts" = as.numeric(bigram_counts))
```

```r

# Horizontal bar plot
ggplot(topbigrams[1:25,], aes(x = counts, y = bigrams)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(x = "Count", y = "Bigram", title = "Number of occurrences of top bigrams") +
```

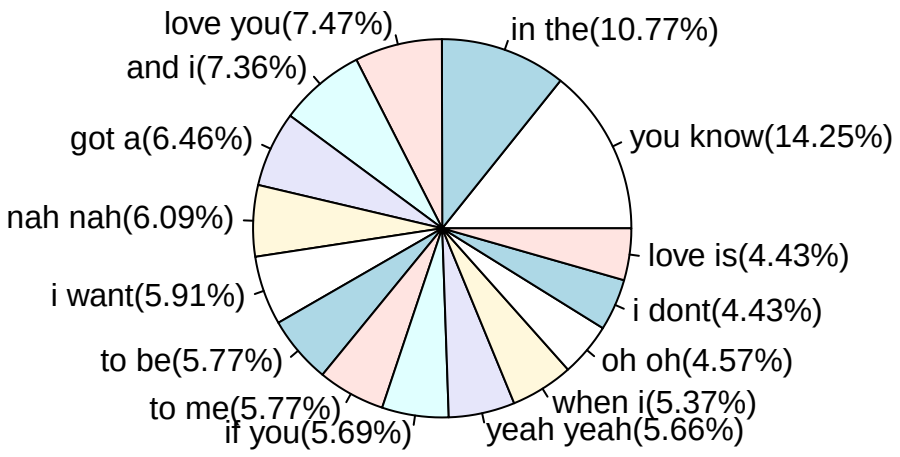
```
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



Bar plot

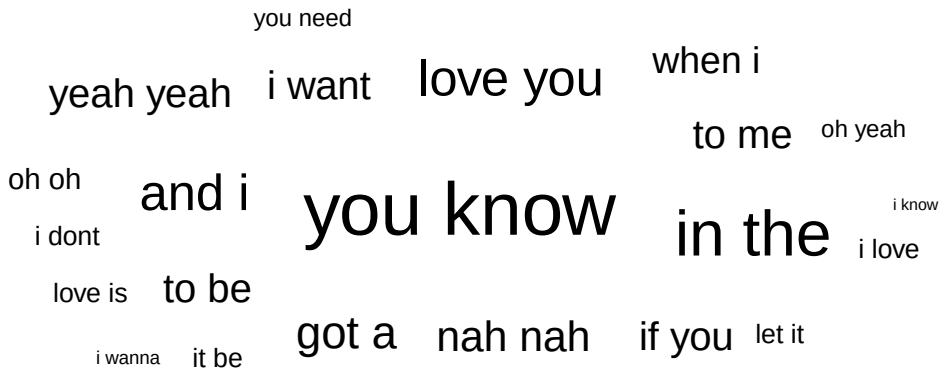
```
percentages <- round(topbigrams$counts[1:15]/sum(topbigrams$counts[1:15])*100, 2)
labels <- paste(topbigrams$bigrams[1:15], "(", percentages, "%)", sep = "")

pie(topbigrams$counts[1:15], labels = labels, main = "Bigram Occurrences")
```



Pie Chart

```
# Create word cloud of bigrams using ggplot2
ggplot(topbigrams[1:25,], aes(label = bigrams, size = counts)) +
  geom_text_wordcloud() +
  scale_size(range = c(2, 10)) +
  theme_minimal()
```



Word cloud

Finding top trigrams

```
mylrc3 <- lyrics

mylrc3 <- lapply(mylrc3, function(x) {
  # Convert to lowercase
  x <- tolower(x)
  # Remove special characters
  x <- gsub("[^[:alpha:][:space:]]", "", x)
  # Split into words
  x <- unlist(strsplit(x, "\\s+"))
  # Return modified string
  return(x)
})

# Unlist the output
mylrc3 <- unlist(mylrc3)

# Remove empty strings
mylrc3 <- mylrc3[mylrc3 != ""]
```

Preprocessing data

```
# Creating trigrams
mylrc3 <- paste(mylrc2, c(mylrc3[-(1:2)], ""), sep = " ")

# Count the number of occurrences of each trigrams
trigrams_counts <- table(mylrc3)

# Sort the trigrams counts in descending order
trigrams_counts <- sort(trigrams_counts, decreasing = TRUE)
```

```
# Displaying top 20 bigrams
trigrams_counts[1:20]
```

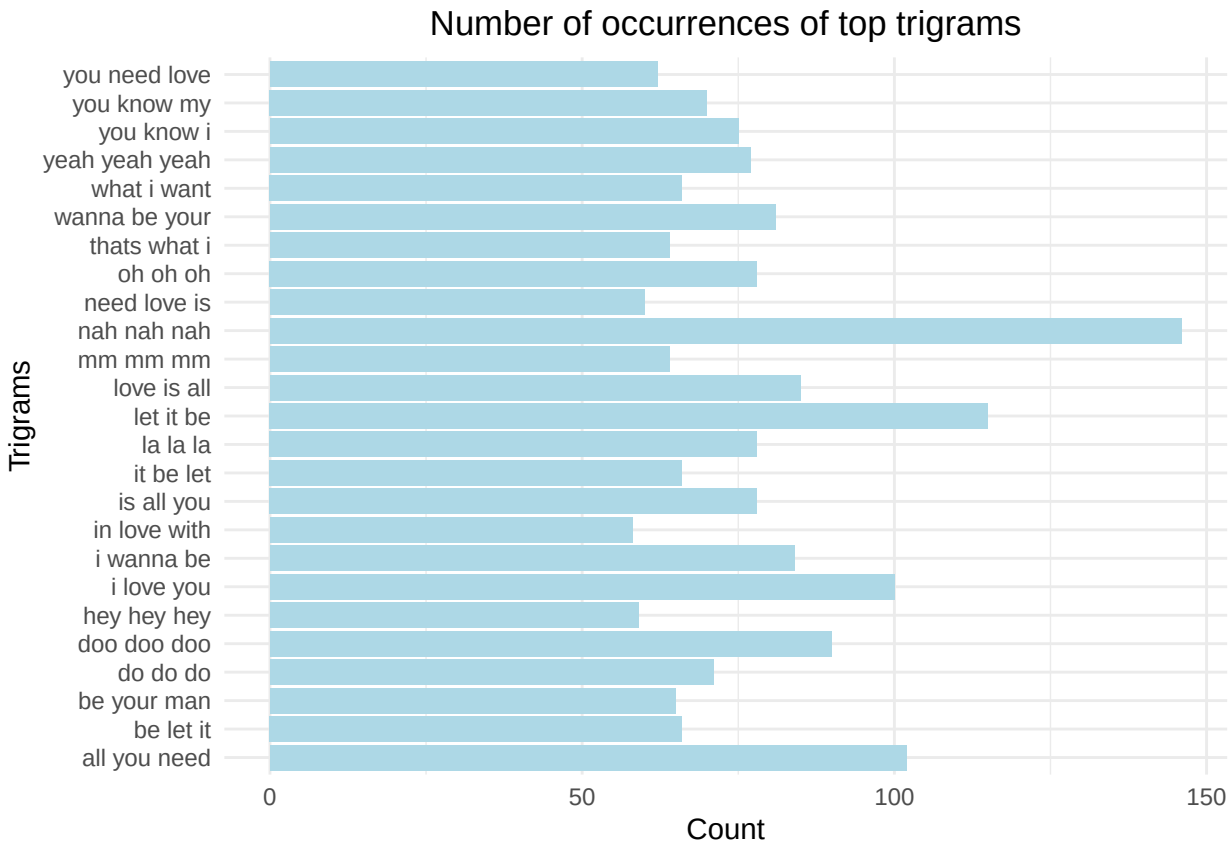
Displaying top trigrams

```
## mylrc3
##    nah nah nah      let it be   all you need     i love you    doo doo doo
##            146            115            102            100             90
##    love is all     i wanna be  wanna be your     is all you       la la la
##             85             84             81             78             78
##       oh oh oh yeah yeah yeah     you know i       do do do    you know my
##             78             77             75             71             70
##      be let it      it be let    what i want    be your man       mm mm mm
##             66             66             66             65             64
```

Visualizing the outcome

```
# Converting to dataframe for ease in visualizations
toptrigrams <- data.frame("trigrams" = as.character(names(trigrams_counts)),
                          "counts" = as.numeric(trigrams_counts))
```

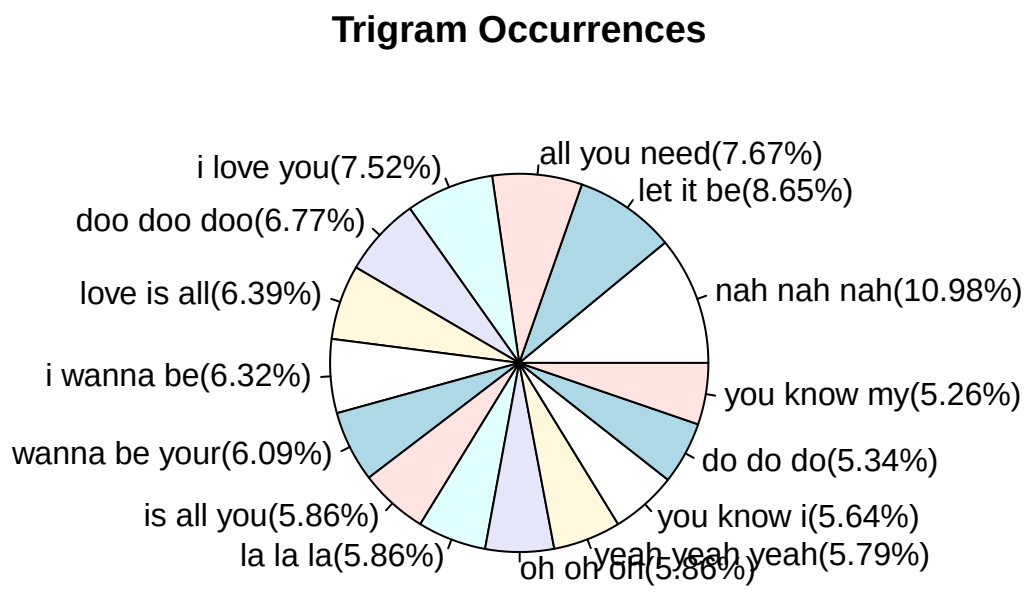
```
# Horizontal bar plot
ggplot(toptrigrams[1:25,], aes(x = counts, y = trigrams)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(x = "Count", y = "Trigrams", title = "Number of occurrences of top trigrams") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```



Bar plot

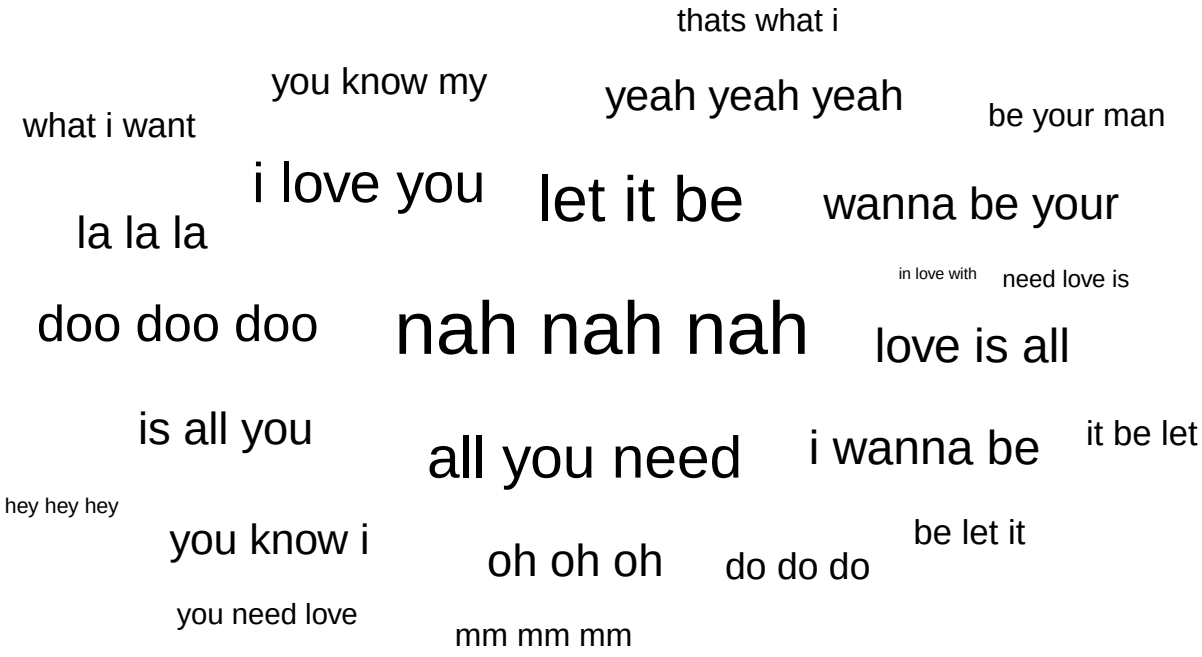
```
percentages <- round(toptrigrams$counts[1:15]/sum(toptrigrams$counts[1:15])*100, 2)
labels <- paste(toptrigrams$trigrams[1:15], "(", percentages, "%)", sep = "")

pie(toptrigrams$counts[1:15], labels = labels, main = "Trigram Occurrences")
```



Pie Chart

```
# Create word cloud of trigrams using ggplot2
ggplot(toptrigrams[1:25,], aes(label = trigrams, size = counts)) +
  geom_text_wordcloud() +
  scale_size(range = c(2, 10)) +
  theme_minimal()
```



Word cloud