Twitter Yukun

Yukun He

2017/12/12

===Introduction===

I was a player of both online games, League of Legends and Heroes of the Storm. I want to explore and compare online critics for these two games, since they are major competitors in the MOBA game category. Also I want to start my future career in the gaming industry, so that this project can prepare me for the future.

```
library(devtools)
library(twitteR)
library(streamR)
```

```
## Loading required package: RCurl

## Loading required package: bitops

## Loading required package: rjson
```

```
library(ROAuth)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:twitteR':
##
##     id, location

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(splitstackshape)
```

```
## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
library(stringr)
library(tm)
```

```
## Loading required package: NLP

##
## Attaching package: 'NLP'
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```r
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```r
library(grid)
library(tidytext)
library(tidyverse)
```

```
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr

## Conflicts with tidy packages ----------------------------------------------

## annotate():  ggplot2, NLP
## between():   dplyr, data.table
## complete():  tidyr, RCurl
## filter():    dplyr, plotly, stats
## first():     dplyr, data.table
## id():        dplyr, twitteR
## lag():       dplyr, stats
## last():      dplyr, data.table
## location():  dplyr, twitteR
## transpose(): purrr, data.table
```

```r
library(tidyr)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths

## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

```r
library(RColorBrewer)


api_key <-  "8jZnFpi5LCq9WiuGGNCRqePpI"
api_secret <- "NZP0lq9lT5cl2lbhAdTWMzLPGrVJSmAAptKRyNrbzSxH0GyiIi"
access_token <- "321852916-jhSeZgq948vcNUEWl90hHajh53La8cYhcn8yYqb3"
access_token_secret <- "pYlobWYfFbDgNu2acYbVBsKvI5xVlcbqzgl3z7DunkI2n"

setup_twitter_oauth(api_key, api_secret, access_token, access_token_secret)
```

```
## [1] "Using direct authentication"
```

```
## Warning in strptime(x, fmt, tz = "GMT"): unknown timezone 'default/America/
## New_York'
```

```r
###Data for League of Legends
league <- searchTwitter("'League of Legends' OR #leagueoflegends", n = 2000, lang="en", since="2016-12-

league_df <- twListToDF(league)
```

```r
###Data for Heroes of the Storm
heroes <- searchTwitter("'Heroes of the Storm' OR #heroesofthestorm", n = 2000, lang="en", since="2016-

heroes_df <- twListToDF(heroes)
```

```r
league_df$text<-iconv(league_df$text, from = "latin1", to = "ASCII", sub="")
league_df$text<-gsub('http\\S+\\s*', '', league_df$text)
league_df$tweetid <- c(1:nrow(league_df))
write.csv(league_df,"league_df.csv")

heroes_df$text<-iconv(heroes_df$text, from = "latin1", to = "ASCII", sub="")
heroes_df$text<-gsub('http\\S+\\s*', '', heroes_df$text)
heroes_df$tweetid <- c(1:nrow(heroes_df))
write.csv(heroes_df,"heroes_df.csv")
```

```r
league_text <- read.csv("league_df.csv")
league_text <- select(league_text, tweetid, text)
league_text$text <- str_replace_all(league_text$text, "@", "")

heroes_text <- read.csv("heroes_df.csv")
heroes_text <- select(heroes_text, tweetid, text)
heroes_text$text <- str_replace_all(heroes_text$text, "@", "")
```

===Sentiment===

From the sentiment analysis we can directly observe the most frequent words used in the Tweets we choose, and the positive and negative sentiments of those words.

Sentiment Analysis for League of Legends

```r
league_text <- data.frame(lapply(league_text, as.character), stringsAsFactors=FALSE)

league_text <- data_frame(line = league_text$tweetid, text = league_text$text)
league_text <- league_text %>%
  unnest_tokens(word, text)

data(stop_words)

league_text <- league_text %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```r
league_text %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 3,002 x 2
##         word     n
##        <chr> <int>
##  1    league  1665
##  2   legends  1643
##  3        rt   601
##  4   youtube   413
##  5     video   254
##  6      live   171
##  7      game   162
##  8 streaming   136
##  9   playing   133
## 10    twitch   133
## # ... with 2,992 more rows
```

```r
nrcjoy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

league_text_sentiment_stat <- league_text %>%
  inner_join(nrcjoy) %>%
  count(word, sort = TRUE)
```

```
## Joining, by = "word"
```

```r
league_text$line <- as.numeric(league_text$line)

bing_word_counts_league <- league_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

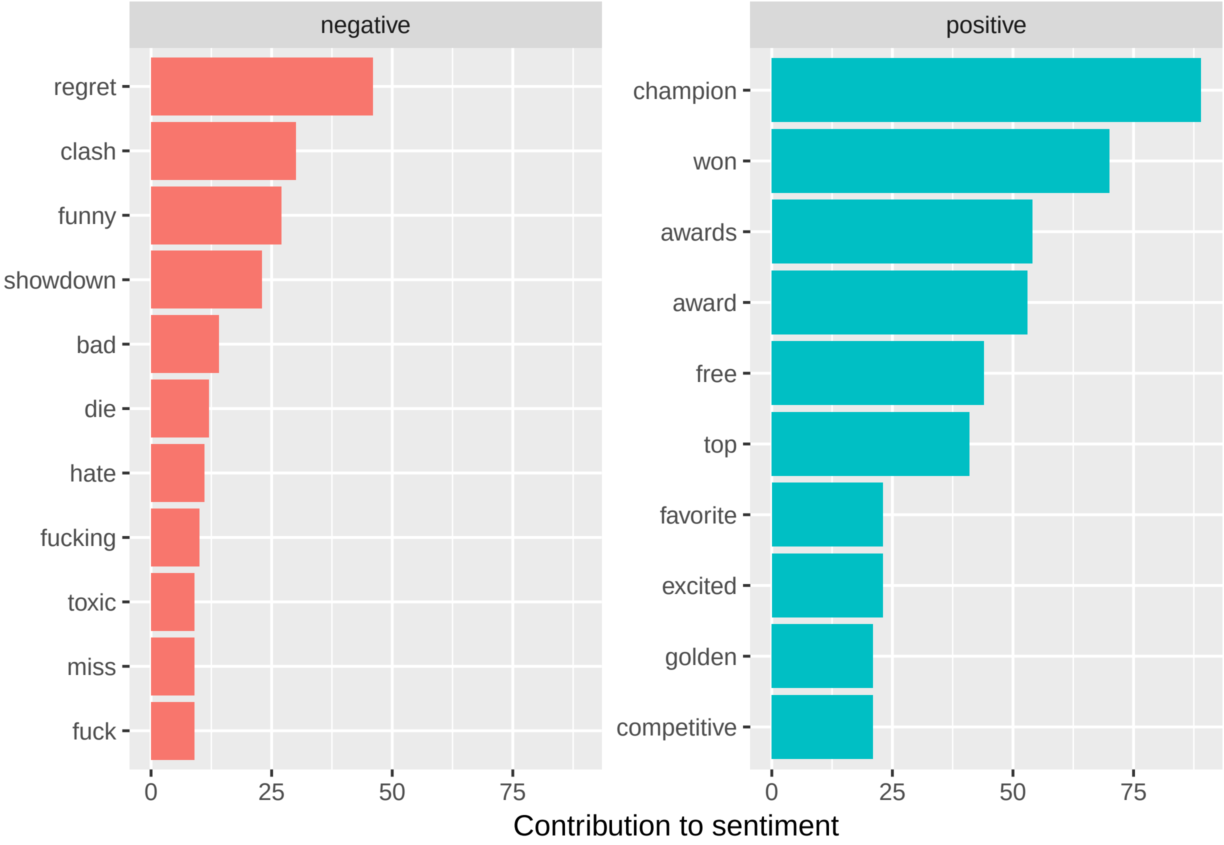
```
## Joining, by = "word"
```

```r
bing_word_counts_league
```

```
## # A tibble: 287 x 3
##        word sentiment     n
##       <chr>     <chr> <int>
##  1 champion  positive    89
##  2      won  positive    70
##  3   awards  positive    54
##  4    award  positive    53
##  5   regret  negative    46
##  6     free  positive    44
## 7      top  positive    41
## 8    clash  negative    30
## 9    funny  negative    27
## 10  excited  positive    23
## # ... with 277 more rows
```

```
bing_word_counts_league %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()
```

```
## Selecting by n
```



```
saveRDS(bing_word_counts_league,"bing_word_counts_league.rds")
```

Sentiment Analysis for Heroes of the Storm

```
heroes_text <- data.frame(lapply(heroes_text, as.character), stringsAsFactors=FALSE)
```

```r
heroes_text <- data_frame(line = heroes_text$tweetid, text = heroes_text$text)
heroes_text <- heroes_text %>%
  unnest_tokens(word, text)

data(stop_words)

heroes_text <- heroes_text %>%
  anti_join(stop_words)
```

```
## Joining, by = "word"
```

```r
heroes_text %>%
  count(word, sort = TRUE)
```

```
## # A tibble: 2,480 x 2
##       word     n
##      <chr> <int>
##  1  heroes  1497
##  2   storm  1417
##  3      rt   712
##  4     lil   576
##  5   patch   490
##  6 youtube   413
##  7   video   255
##  8   event   242
##  9   skins   241
## 10 junkrat   230
## # ... with 2,470 more rows
```

```r
nrcjoy <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

heroes_text_sentiment_stat <- heroes_text %>%
  inner_join(nrcjoy) %>%
  count(word, sort = TRUE)
```

```
## Joining, by = "word"
```

```r
heroes_text$line <- as.numeric(heroes_text$line)

bing_word_counts_heroes <- heroes_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

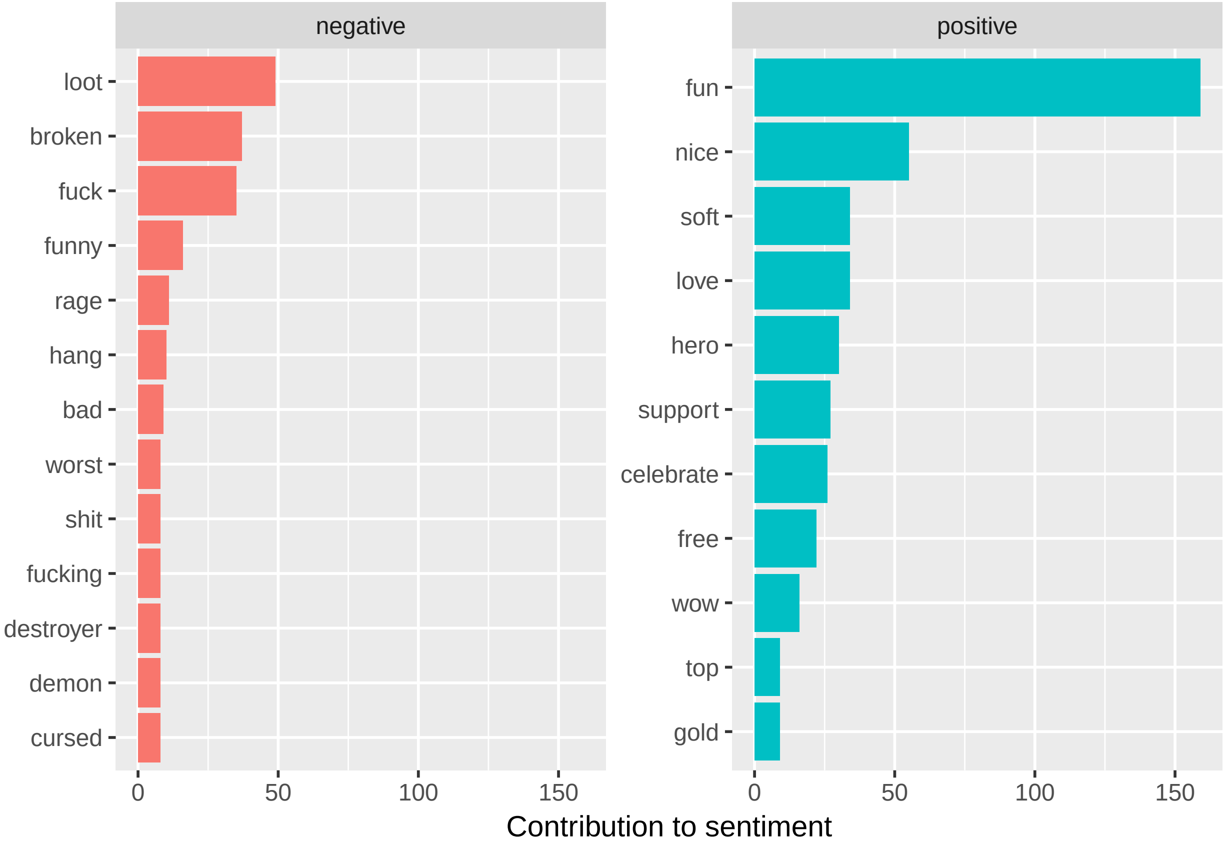
```
## Joining, by = "word"
```

```r
bing_word_counts_heroes
```

```
## # A tibble: 243 x 3
##         word sentiment     n
##        <chr>     <chr> <int>
##  1       fun  positive   159
##  2      nice  positive    55
##  3      loot  negative    49
##  4    broken  negative    37
##  5      fuck  negative    35
## 6      love  positive    34
## 7      soft  positive    34
## 8      hero  positive    30
## 9   support  positive    27
## 10 celebrate  positive    26
## # ... with 233 more rows
```

```
bing_word_counts_heroes %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y") +
  labs(y = "Contribution to sentiment",
       x = NULL) +
  coord_flip()
```

```
## Selecting by n
```



```
saveRDS(bing_word_counts_heroes,"bing_word_counts_heroes.rds")
```

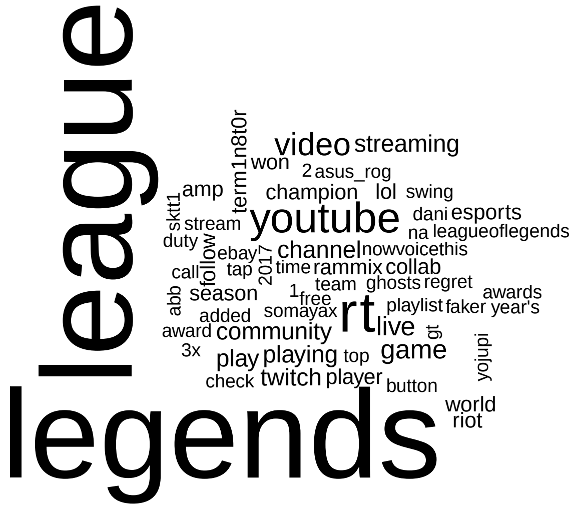
===Word Cloud===

From the word cloud, we can divide words into positive (Blue) and negative (Red) categories. And the larger the size, the bolder the font, the more frequently that word is observed.

Word Cloud League

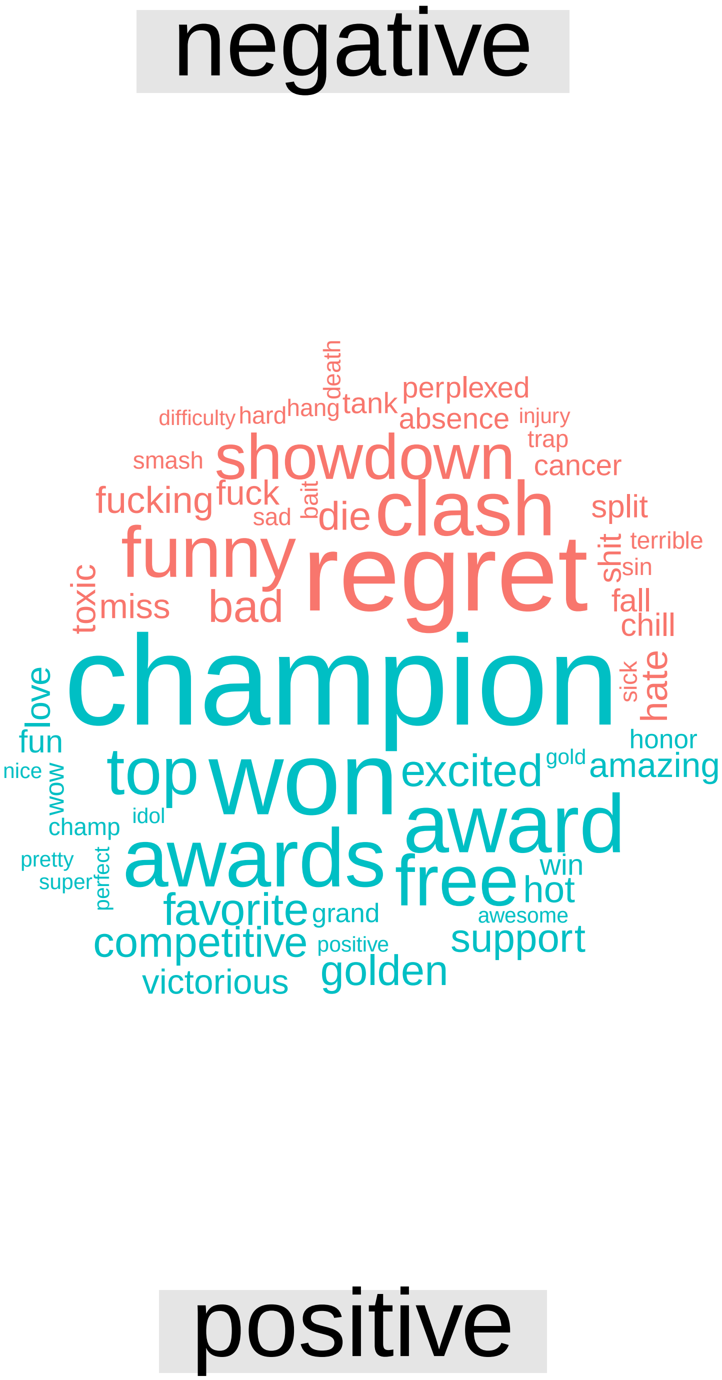
```
league_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 60))
```

```
## Joining, by = "word"
```



```
league_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 60,title.size=3)
```

```
## Joining, by = "word"
```

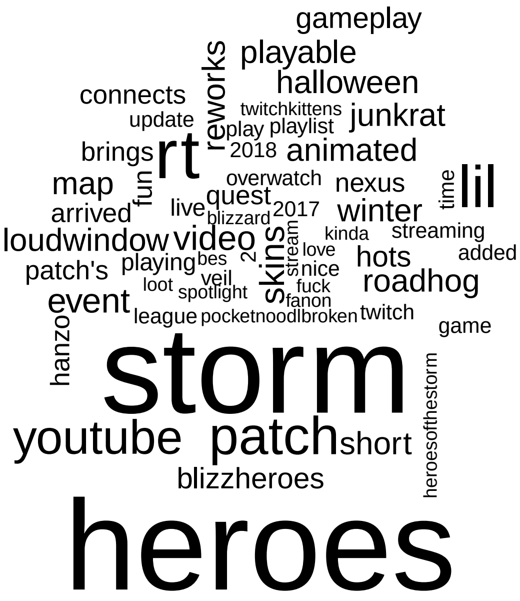


```r
saveRDS(league_text,"league_text.rds")
```

Word Cloud Heroes

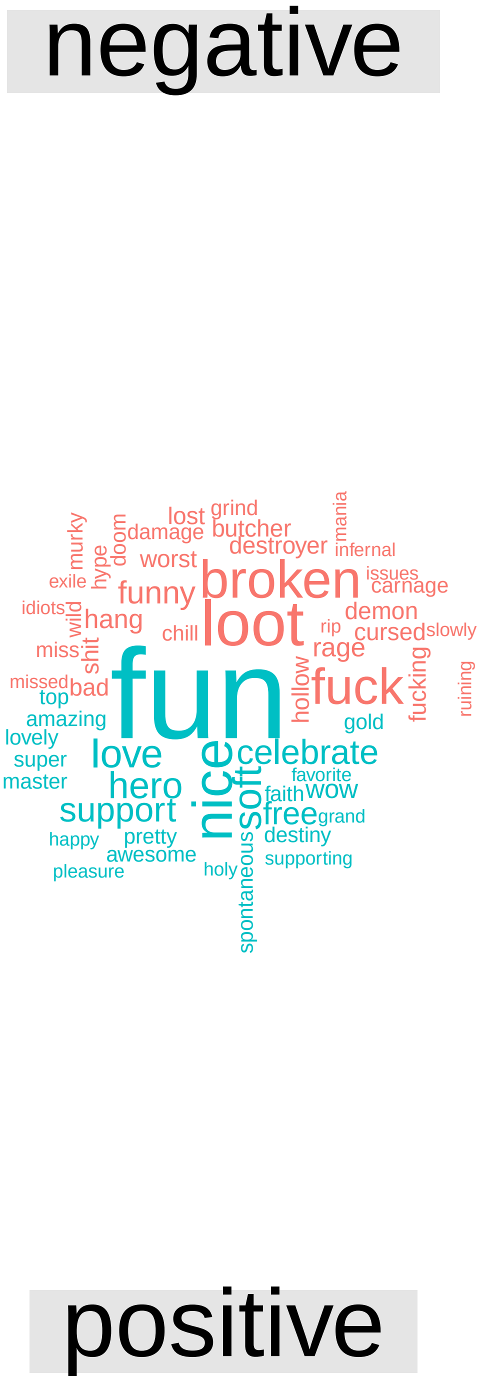
```r
heroes_text %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 60))
```

```
## Joining, by = "word"
```



```r
heroes_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 60,title.size=3)
```

```
## Joining, by = "word"
```



```
saveRDS(heroes_text,"heroes_text.rds")
```

===MAP===

```
##requestURL <- "https://api.twitter.com/oauth/request_token"
##accessURL <- "https://api.twitter.com/oauth/access_token"
##authURL <- "https://api.twitter.com/oauth/authorize"
##consumerKey <-  "LFNRqX5i1PkB69SjEEncXWloq"
##consumerSecret <- "4sDHqY6aLm7PRfJLxpq6GsWqphZxzX3dXLjssSLXYhO8wPwL3F"
```

```
##my_oauth <- OAuthFactory$new(consumerKey = consumerKey, consumerSecret = consumerSecret,
##                             requestURL = requestURL, accessURL = accessURL, authURL = authURL)
##my_oauth$handshake(cainfo = system.file("CurlSSL", "cacert.pem", package = "RCurl"))

##save(my_oauth, file = "my_oauth.Rdata")
```

Google allows only 2500 requests per day for geocode, these two maps are not comprehensive. However, these maps provide a general overview of the locations Leagues of Legends & Heroes of the Storm. From the map, we observe that Tweets are mainly located at West Coast, Great Boston and New York Area, Chicago, and Texas. These regions have a lot of colleges, which represents the fact that the players are mainly college students. Other than these regions, Tweets are observed all over the USA, which means that the two games are both popular over the country.

Map for League

```
load("my_oauth.Rdata")

##map

filterStream("league_map_setup.json",
             track=c("'League of Legends' OR #leagueoflegends"),
             locations = c(-125, 25, -66,50),
             timeout=200, oauth=my_oauth)
```

```
## Capturing tweets...
```

```
## Connection to Twitter stream was closed after 200 seconds with up to 6219 tweets downloaded.
```

```
league_map_setup<-parseTweets("league_map_setup.json", verbose = TRUE)
```

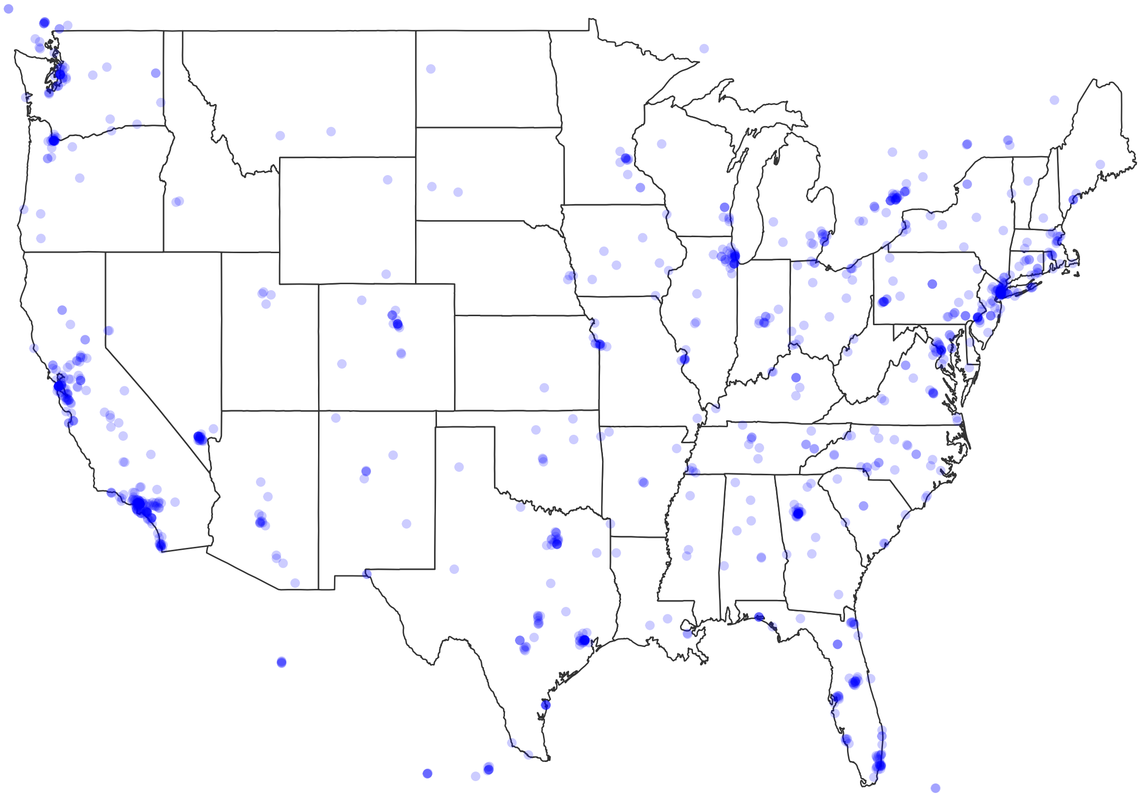
```
## 9257 tweets have been parsed.
```

```
ck1 <- sum(league_map_setup$lat>0, na.rm = TRUE)
ck2 <- sum(league_map_setup$place_lat>0, na.rm = TRUE)
ck3 <- sum(!is.na(league_map_setup$location))
map.data <- map_data("state")
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
league_map <- data.frame(x = as.numeric(league_map_setup$lon),
                       y = as.numeric(league_map_setup$lat))
league_map <- league_map[league_map$y > 25, ]
league_map<-filter(league_map,y>19&y<65,x>(-161.7)&x<(-68.01))
ggplot(map.data) +
  geom_map(aes(map_id = region),
           map = map.data,
           fill = "white",
           color = "grey20", size = 0.25) +
  expand_limits(x = map.data$long, y = map.data$lat) +
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        panel.background = element_blank(),
        panel.border = element_blank(),
        panel.grid.major = element_blank(),
        plot.background = element_blank(),
        plot.margin = unit(0 * c( -1.5, -1.5, -1.5, -1.5), "lines")) +
        geom_point(data = league_map,
        aes(x = x, y = y), size = 1,
        alpha = 1/5, color = "blue")
```



```
saveRDS(league_map, file="league_map.rds")
```

Map of Heroes

```
load("my_oauth.Rdata")

##map

filterStream("heroes_map_setup.json",
             track=c("'Heroes of the Storm' OR #heroesofthestorm"),
             locations = c(-125, 25, -66,50),
             timeout=200, oauth=my_oauth)
```

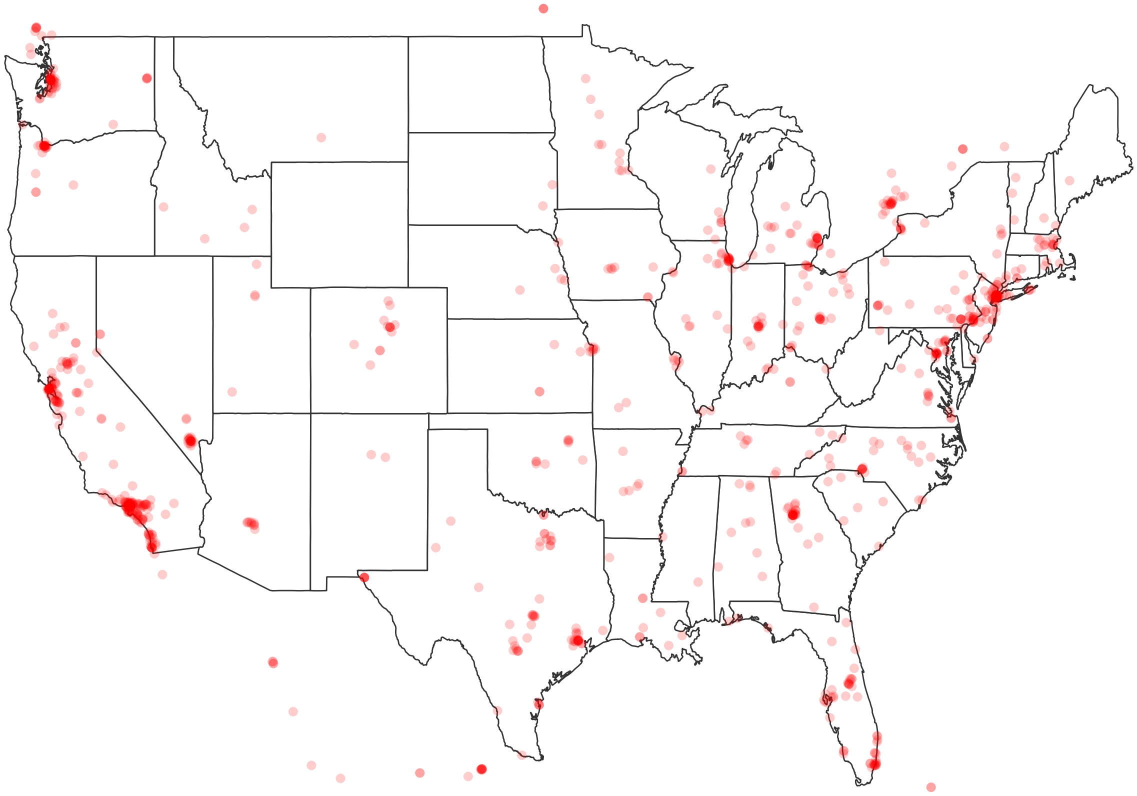
```
## Capturing tweets...

## Connection to Twitter stream was closed after 200 seconds with up to 6356 tweets downloaded.
```

```r
heroes_map_setup<-parseTweets("heroes_map_setup.json", verbose = TRUE)
```

```
## 8132 tweets have been parsed.
```

```r
ck1 <- sum(heroes_map_setup$lat>0, na.rm = TRUE)
ck2 <- sum(heroes_map_setup$place_lat>0, na.rm = TRUE)
ck3 <- sum(!is.na(heroes_map_setup$location))
map.data <- map_data("state")
heroes_map <- data.frame(x = as.numeric(heroes_map_setup$lon),
                         y = as.numeric(heroes_map_setup$lat))
heroes_map <- heroes_map[heroes_map$y > 25, ]
heroes_map<-filter(heroes_map,y>19&y<65,x>(-161.7)&x<(-68.01))
ggplot(map.data) +
  geom_map(aes(map_id = region),
           map = map.data,
           fill = "white",
           color = "grey20", size = 0.25) +
  expand_limits(x = map.data$long, y = map.data$lat) +
  theme(axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        panel.background = element_blank(),
        panel.border = element_blank(),
        panel.grid.major = element_blank(),
        plot.background = element_blank(),
        plot.margin = unit(0 * c( -1.5, -1.5, -1.5, -1.5), "lines")) +
  geom_point(data = heroes_map,
             aes(x = x, y = y), size = 1,
             alpha = 1/5, color = "red")
```



```r
saveRDS(heroes_map, file="heroes_map.rds")
```