Discussion

Hong Shi

2/2/2021

In this data gathering practice, I tried two methods collecting the data: (1) one through twitter API using `rtweet` package, and (2) the other through OCR using `tesseract` package:

(1) Using `rtweet` package

Get DOTA2 patch update tweets(A popular 5v5 MOBA game. OpenAI Five launched a machine learning project in this game and the OpenAI system showed the ability to defeat professional teams. Web URL of the project: https://openai.com/projects/five/) I went through following steps to gather DOTA2 patch update data:

- Since I would like to get tweets information from the official DOTA2 account, I used the `get_timeline()` function and apply to user "DOTA2" to collect all tweets published from the official account (raw data).
- When I wanted to save the raw data into csv file, an error showed that `hashtags` is a list column so I could not save the data to csv. I googled a function to convety any list columns to character type, and successfully save the csv file after applying the function.
- After my inspection of tweet texts, I notice that not all tweets are related to gameplay updates. But as a player of DOTA2, I understand the pattern of patch updates `"\\<[6,7]\\.[0-9]{2}.?\\>"` [1], so I could use Regex to select tweets related to updates.

Thoughts and possible followups: Functions in `rtweet` require some understandings of features in tweet (e.g.: like, follow, etc.) I could use the dataset to analyze the popularity of each patch.

(2) Using `tesseract` package

```
library(tesseract)
library(tidyverse)
library(here)
```

[1] Detailed explanation of this pattern in "inputs/scripts/01_data2_patch_tweet"

```r
library(bookdown)
# tesseract_download("jpn") if Japanese is not installed
```

Get the lyrics of a Japanese song「紅蓮華」(Gurenge) in txt format. I went through following steps to gather the lyrics in Japanese:

- I chose an image of with the "Gurenge" lyrics (Source from Twitter)
- I noticed that Japanese language was not installed in `tesseract` package, so I installed the Japanese language engine and extract the text from it. And it seems that `tesseract` recognizes Japanese characters relatively poor (falsely recognize the Kanji "震" as "選", "睨" as "明" and completely mess up the Hiragana sentence "どうしたって" as "2357 つう sGgl"). NOTE: I could display proper Japanese characters in R markdown, but not in html document.

```
## mlau<U+4EE4>                    0:54         $@<U+30A4>@( 69%(<U+8A08><U+30EA><U+30BF>
## <U+304F><U+30E1>><U+30E2>                                  @<U+3002> <U+4E2D>
##
## <U+7D05><U+84EE><U+83EF> <U+6B4C><U+8A5E>
##
## <U+5F37><U+304F><U+306A><U+308C><U+308B><U+7406><U+7531><U+3092><U+77E5><U+3063><U+305F>
##
## <U+50D5><U+3092><U+9023><U+308C><U+3066> <U+9032><U+3081>
##
## <U+6CE5><U+3060><U+3089><U+3051><U+306E><U+8D70><U+99AC><U+706F><U+306B><U+9154><U+3046>
##
## <U+5F37><U+3070><U+308B><U+5FC3> <U+9078><U+3048><U+308B><U+624B><U+306F>
##
## <U+63B4><U+307F><U+305F><U+3044><U+3082><U+306E><U+304C><U+3042><U+308B>
##
## <U+305D><U+308C><U+3060><U+3051><U+3055>
##
## <U+591C><U+306E><U+5302><U+3044><U+306B>(| spend all thirty nights)
##
## <U+7A7A><U+660E><U+3093><U+3067><U+3082>(Storing into the sky)
##
## <U+5909><U+308F><U+3063><U+3066><U+3044><U+3051><U+308B><U+306E><U+306F> <U+81EA><U+5206><U+81E
##
## <U+305D><U+308C><U+3060><U+3051><U+3055>
##
## <U+5F37><U+304F><U+306A><U+308C><U+308B><U+7406><U+7531><U+3092><U+77E5><U+3063><U+305F>
##
```

```
## <U+50D5><U+3092><U+9023><U+308C><U+3066> <U+9032><U+3081>
##
## 2357<U+3064><U+3046>sGgl
##
## <U+6D88><U+305B><U+306A><U+3044><U+5922><U+3082> <U+6B62><U+307E><U+308C><U+306A><U+3044><U+4EC
##
## <U+8AB0><U+304B><U+306E><U+305F><U+3081><U+306B><U+5F37><U+304F><U+306A><U+308C><U+308B><U+306A
##
## <U+3042><U+308A><U+304C><U+3068><U+3046> <U+60B2><U+3057><U+307F><U+3088>
##
## <U+4E16><U+754C><U+306B><U+6253><U+3061><U+306E><U+3081><U+3055><U+308C><U+3066>
##
## <U+89D2><U+3051><U+308B><U+610F><U+5473><U+3092><U+77E5><U+3063><U+305F>
##
##          <U+3007>          <U+3007>          <U+3081>          <U+30F2>
##
```

- (This made me suffer almost a whole night figuring out the solution) After getting the text of lyrics, I would like to save it to a txt file. However, the output text file is shown below, even though I wanted the text file to match what I show in R console:

```
## mlau<U+4EE4>                    0:54         $@<U+30A4>@( 69%(<U+8A08><U+30EA><U+30BF>
## <U+304F><U+30E1>><U+30E2>                                  @<U+3002> <U+4E2D>
##
## <U+7D05><U+84EE><U+83EF> <U+6B4C><U+8A5E>
##
## <U+5F37><U+304F><U+306A><U+308C><U+308B><U+7406><U+7531><U+3092><U+77E5><U+3063><U+305F>
##
## <U+50D5><U+3092><U+9023><U+308C><U+3066> <U+9032><U+3081>
##
## <U+6CE5><U+3060><U+3089><U+3051><U+306E><U+8D70><U+99AC><U+706F><U+306B><U+9154><U+3046>
##
## <U+5F37><U+3070><U+308B><U+5FC3> <U+9078><U+3048><U+308B><U+624B><U+306F>
##
## <U+63B4><U+307F><U+305F><U+3044><U+3082><U+306E><U+304C><U+3042><U+308B>
##
## <U+305D><U+308C><U+3060><U+3051><U+3055>
##
## <U+591C><U+306E><U+5302><U+3044><U+306B>(| spend all thirty nights)
##
## <U+7A7A><U+660E><U+3093><U+3067><U+3082>(Storing into the sky)
```

```
##
## <U+5909><U+308F><U+3063><U+3066><U+3044><U+3051><U+308B><U+306E><U+306F> <U+81EA><U+5206><U+81E
##
## <U+305D><U+308C><U+3060><U+3051><U+3055>
##
## <U+5F37><U+304F><U+306A><U+308C><U+308B><U+7406><U+7531><U+3092><U+77E5><U+3063><U+305F>
##
## <U+50D5><U+3092><U+9023><U+308C><U+3066> <U+9032><U+3081>
##
## 2357<U+3064><U+3046>sGgl
##
## <U+6D88><U+305B><U+306A><U+3044><U+5922><U+3082> <U+6B62><U+307E><U+308C><U+306A><U+3044><U+4EC
##
## <U+8AB0><U+304B><U+306E><U+305F><U+3081><U+306B><U+5F37><U+304F><U+306A><U+308C><U+308B><U+306A
##
## <U+3042><U+308A><U+304C><U+3068><U+3046> <U+60B2><U+3057><U+307F><U+3088>
##
## <U+4E16><U+754C><U+306B><U+6253><U+3061><U+306E><U+3081><U+3055><U+308C><U+3066>
##
## <U+89D2><U+3051><U+308B><U+610F><U+5473><U+3092><U+77E5><U+3063><U+305F>
##
##           <U+3007>          <U+3007>          <U+3081>          <U+30F2>
```

I understood that the "<U+XXXX>" is the Unicode representation of the Japanese character and the text I extracted using `tesseract` package is a character variable with Unicodes. I spent a great amount of time googling and figured out that the text of the character variable it self is already encoded in "UTF-8". So if I save the encoded "UTF-8" text, no matter which encoding method I chose to open the file, the text file will always display the Unicode text as shown above. So I need to reencode the text `tesseract` generated first, and then save the txt file to make sure I could display the exact Japanese characters in .txt file.

However, the output html document document still display the Japanese character in Unicode form and I could not knit the pdf document due to error: ! Package inputenc Error: Unicode character 「 (U+300C) (inputenc) not set up for use with LaTeX.

I would like to get some follow up analysis about DOTA2 patch updates hopefully in reading week :) And also have a closer look of Unicode output to solve my knitting problems.