Main

Smitesh Patil

2023-03-10

```
knitr::opts_chunk$set(echo = TRUE)
```

```
library(igraph)
```

```
## Warning: package ’igraph’ was built under R version 4.1.3
```

```
##
## Attaching package: ’igraph’
```

```
## The following objects are masked from ’package:stats’:
##
##     decompose, spectrum
```

```
## The following object is masked from ’package:base’:
##
##     union
```

```
library(ggraph)
```

```
## Warning: package ’ggraph’ was built under R version 4.1.3
```

```
## Loading required package: ggplot2
```

```
library(ggrepel)
```

```
## Warning: package ’ggrepel’ was built under R version 4.1.3
```

```
library(kableExtra)
```

```
## Warning: package ’kableExtra’ was built under R version 4.1.3
```

```
library(gt)
```

```
## Warning: package ’gt’ was built under R version 4.1.3
```

```
library(tidyr)
```

```
## Warning: package ’tidyr’ was built under R version 4.1.3
```

```
##
## Attaching package: ’tidyr’
```

```
## The following object is masked from ’package:igraph’:
##
##     crossing
```

```
library(dplyr)
```

```
## Warning: package ’dplyr’ was built under R version 4.1.3
```

```
##
## Attaching package: ’dplyr’
```

```
## The following object is masked from ’package:kableExtra’:
##
##     group_rows
```

```
## The following objects are masked from ’package:igraph’:
##
##     as_data_frame, groups, union
```

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

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

```
# reading the graph
g<- read_graph(file="./WordPairs.txt",format="pajek")
g<- as.undirected(g)
g<- simplify(g)

cues <- read.table("./cue.txt", header = F, sep="\t", skip=4)

V(g)$cue<-cues[[1]]

#checking the diameter value
print(diameter(g, weights = NA))
```

```
## [1] 7
```

```r
check_cue_words <- function(target_node_name1, target_node_name2){
# test if the selected words are cue words

  if(V(g)[target_node_name1]$cue & V(g)[target_node_name2]$cue){
    cat("Both target words are cue words \n")
  }else{
    cat("Both target words are NOT cue words \n")
    cat(target_node_name1, "cue = ", as.logical(V(g)[target_node_name1]$cue ),"\n")
    cat(target_node_name2, "cue = ", as.logical(V(g)[target_node_name2]$cue ),"\n")
  }
}
```

```r
random_walk_topic_network <- function(g,target_node_names, steps, walks, mode, topn){
  vertices_1 <- c()
  for (i in 1:walks){
    vertices_1 <- c(vertices_1, list(random_walk(g, target_node_names[1], steps, mode = mode)))
  }
  frequency_target1 <- head(sort(table(names(unlist(vertices_1))), decreasing = TRUE), 100)
  unique_words1 <- names(frequency_target1)

  vertices_2 <- c()
  for (i in 1:walks){
    vertices_2 <- c(vertices_2, list(random_walk(g, target_node_names[2], steps, mode = mode)))
  }
  frequency_target2 <- head(sort(table(names(unlist(vertices_2))), decreasing = TRUE), 100)
  unique_words2 <- names(frequency_target2)


  output = append(unique_words1, unique_words2)
  return(unique(output))

}

centralities = function(word_association_network){
  page_rank <- page_rank(word_association_network)$vector
  page_rank <- na.omit(page_rank[!names(page_rank) %in% c(target_word1, target_word2)])
  page_rank <- sort(page_rank, decreasing = TRUE)[1:5]

  betweenness <- betweenness(word_association_network)
  betweenness <- betweenness[!names(betweenness) %in% c(target_word1, target_word2)]
  betweenness <- sort(betweenness, decreasing = TRUE)[1:5]

  eigen_centrality <- eigen_centrality(word_association_network)$vector
  eigen_centrality <- eigen_centrality[!names(eigen_centrality) %in% c(target_word1, target_word2)]
  eigen_centrality <- sort(eigen_centrality, decreasing = TRUE)[1:5]

  return(list(page_rank, betweenness, eigen_centrality))
}
```

```r
target_word1 <- "BOOK"
target_word2 <- "DICTIONARY"

check_cue_words(target_word1, target_word2)
```

```
## Both target words are cue words
```

```r
out <- random_walk_topic_network(g, c(target_word1, target_word2), 3, 100, "all", 200)

Vertices_in_word_association <- V(g)[name %in% out]

word_association_network <- induced.subgraph(g, Vertices_in_word_association)

centrality = centralities(word_association_network)

df <- tibble(names(centrality[[1]]) , names(centrality[[2]]), names(centrality[[3]]))
colnames(df) <- c("page_rank", "betweeness", "eigen_centrality")

df %>% gt() %>%
  tab_header(paste0("Top 5 words based on centralities based on word association network for
             words ",target_word1 , " and ",target_word2))
```

Top 5 words based on centralities based on word association network for words BOOK and DICTIONARY

page_rank	betweeness	eigen_centrality
WORDS	SCHOOL	LIBRARY
SCHOOL	VOCABULARY	WORDS
PAPER	SPELL	CHAPTER
FOOD	MEANING	ENCYCLOPEDIA
ENCYCLOPEDIA	SPANISH	THESAURUS

```r
vertex_size <- 2.5 + degree(g)/1.5
cex_size <-2 + degree(g)/12

ggraph(word_association_network, layout = "fr")
```

```
# plot(word_association_network,
#      layout=layout_with_dh(word_association_network),
#      vertex.color= ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
#                           rgb(243/255,243/255, 15/255, 0.6),
#                           rgb(0, 0, 0, 0.2)),
#      vertex.size = ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
#                           10,
#                           5),
#      vertex.label.cex=ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
#                           0.3,
#                           0.2),
#      vertex.label.dist=0,
#      edge.curved=0.2,
#      edge.width=0.4,
#      edge.arrow.size=0.01,
#      edge.color= rgb(0.5,0.5,0.5, 0.3))
```

```
target_word1 <- "ABDOMEN"
target_word2 <- "STOMACH"

check_cue_words(target_word1, target_word2)
```

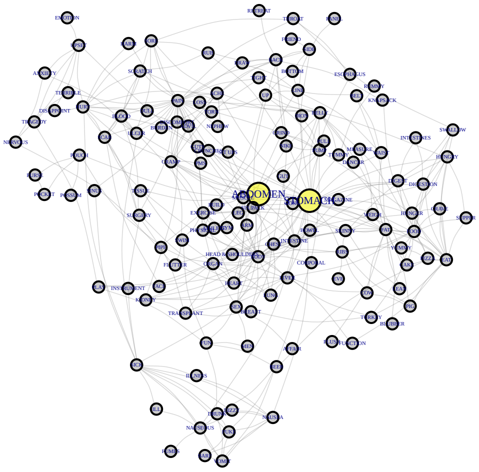
```
## Both target words are cue words
```

```
out <- random_walk_topic_network(g, c(target_word1, target_word2), 3, 100, "all", 10)

Vertices_in_word_association <- V(g)[name %in% out]

word_association_network <- induced.subgraph(g, Vertices_in_word_association)

centrality = centralities(word_association_network)

df <- tibble(names(centrality[[1]]) , names(centrality[[2]]), names(centrality[[3]]))
colnames(df) <- c("page_rank", "betweeness", "eigen_centrality")

df %>% gt() %>%
  tab_header(paste0("Top 5 words based on centralities based on word association network for
             words ",target_word1 , " and ",target_word2))
```

Top 5 words based on centralities based on word association network for words ABDOMEN and STOMACH

page_rank	betweeness	eigen_centrality
PAIN	BODY	SICK
FOOD	PAIN	ILL
HURT	SEX	ILLNESS
SICK	FAT	NAUSEOUS
FAT	BACK	PAIN

```
plot(word_association_network,
     layout=layout_with_dh(word_association_network),
     vertex.color= ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          rgb(243/255,243/255, 15/255, 0.6),
                          rgb(0, 0, 0, 0.2)),
     vertex.size = ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          10,
                          5),
     vertex.label.cex=ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          0.3,
                          0.2),
     vertex.label.dist=0,
     edge.curved=0.2,
     edge.width=0.4,
     edge.arrow.size=0.01,
     edge.color= rgb(0.5,0.5,0.5, 0.3)
)
```



```
target_word1 <- "HEAD"
target_word2 <- "HEART"

check_cue_words(target_word1, target_word2)
```

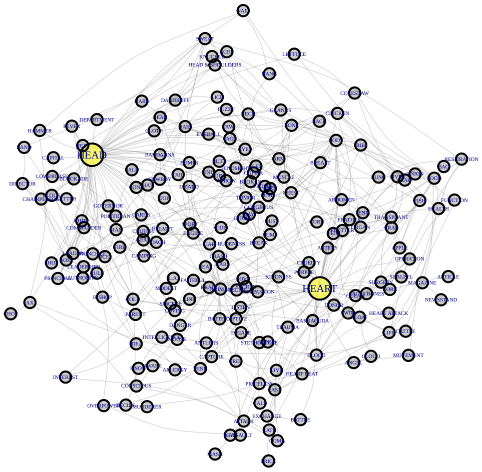
```
## Both target words are cue words
```

```
out <- random_walk_topic_network(g, c(target_word1, target_word2), 3, 100, "all", 10)

Vertices_in_word_association <- V(g)[name %in% out]

word_association_network <- induced.subgraph(g, Vertices_in_word_association)

centrality = centralities(word_association_network)

df <- tibble(names(centrality[[1]]) , names(centrality[[2]]), names(centrality[[3]]))
colnames(df) <- c("page_rank", "betweeness", "eigen_centrality")

df %>% gt() %>%
  tab_header(paste0("Top 5 words based on centralities based on word association network for
             words ",target_word1 , " and ",target_word2))
```

Top 5 words based on centralities based on word association network for words HEAD and HEART

page_rank	betweeness	eigen_centrality
HAIR	BODY	NAIL
CAR	CAPTIVE	HAMMER
BLOOD	MAN	CLAW
LOVE	IDEA	AX
BRAIN	BREAK	HAT

```
plot(word_association_network,
     layout=layout_with_dh(word_association_network),
     vertex.color= ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          rgb(243/255,243/255, 15/255, 0.6),
                          rgb(0, 0, 0, 0.2)),
     vertex.size = ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          10,
                          5),
     vertex.label.cex=ifelse(V(word_association_network)$name %in% c(target_word1, target_word2),
                          0.3,
                          0.2),
     vertex.label.dist=0,
     edge.curved=0.2,
     edge.width=0.4,
     edge.arrow.size=0.01,
     edge.color= rgb(0.5,0.5,0.5, 0.3)
)
```



```r
#fast greedy community detection
fc <- cluster_fast_greedy(word_association_network)
#louvain community detection
lv <- cluster_louvain(word_association_network)
#walktrap community detection
wt <- cluster_walktrap(word_association_network)

create_community_table <- function(clustering_data){
  strings <- NULL
  lengths <- c()
  for(i in 1:length(clustering_data)){
    string = ""
    for(word in clustering_data[[i]]){
      string = paste0(string, "'", word, "',")
    }

    strings <- c(strings, string)
    lengths <- c(lengths, length(clustering_data[[i]]))
  }

  df <- tibble(strings, lengths) %>%
  arrange(desc(lengths))
  colnames(df) <- c("Cluster","Size")

  return(df)
}
```

```
View(create_community_table(fc))

View(create_community_table(lv))

View(create_community_table(wt))
```