Project 4 - Paper 4

Ken Chew, Terry Li, Yue Jin

04/12/2017

This file is currently a template for implementing one of the suggested papers, Han, Zha, & Giles (2005). Due to the nature of the method, we only implement the method on a subset of the data, "AKumar.txt". In your project, you need to work on the whole dataset. You should follow the same structure as in this tutorial, but update it according to the papers you are assigned.

Step 0: Load the packages, specify directories

```
if (!require("pacman")) install.packages("pacman")
pacman::p_load(text2vec, dplyr, qlcMatrix, kernlab, knitr)

setwd("~/Dropbox/Project4_WhoIsWho/doc")
# here replace it with your own path or manually set it in RStudio
# to where this rmd file is located
```

Step 1: Load and process the data

For each record in the dataset, there are some information we want to extract and store them in a regular form: canonical author id, coauthors, paper title, publication venue title. You may need to find regular matched in the input string vectors by using regex in R. Here is a tutorial for regular expression in R, which might help you https://rstudio-pubs-static.s3.amazonaws.com/74603_76cd14d5983f47408fdf0b323550b846.html

```
# AKumar <- data.frame(scan("../data/nameset/AKumar.txt",
#                           what = list(Coauthor = "", Paper = "", Journal = ""),
#                           sep=">", quiet=TRUE), stringsAsFactors=FALSE)
# # This need to be modified for different name set
#
# # extract canonical author id before "_"
# AKumar$AuthorID <- sub("_.*","",AKumar$Coauthor)
# # extract paper number under same author between "_" and first whitespace
# AKumar$PaperNO <- sub(".*_(\\w*)\\s.*", "\\1", AKumar$Coauthor)
# # delete "<" in AKumar$Coauthor, you may need to further process the coauthor
# # term depending on the method you are using
# AKumar$Coauthor <- gsub("<","",sub("^.*?\\s","", AKumar$Coauthor))
# # delete "<" in AKumar$Paper
# AKumar$Paper <- gsub("<","",AKumar$Paper)
# # add PaperID for furthur use, you may want to combine all the nameset files and
# # then assign the unique ID for all the citations
# AKumar$PaperID <- rownames(AKumar)
```

Step 2: Feature design

Following the section 3.1 in the paper, we want to use paper titles to design features for citations.

TF-IDF is a numerical statistics that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval, text mining, and user modeling. The TF-IDF value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

$$\begin{aligned}
\text{TF}(t) &= \frac{\text{Number of times term } t \text{ appears in a document}}{\text{Total number of terms in the document}} \\
\text{IDF}(t) &= \log \frac{\text{Total number of documents}}{\text{Number of documents with term } t \text{ in it}} \\
\text{TF-IDF}(t) &= \text{TF}(t) \times \text{IDF}(t)
\end{aligned}$$

To compute TF-IDF, we first need to construct a document-term matrix (DTM). In other words, the first step is to vectorize text by creating a map from words to a vector space. There are some good packages you could use for text mining (probably you have tried during first project, you don't need to follow my code if you are familiar with other package), e.g. *text2vec, tm, tidytext*. Here, we are going to use *text2vec* package. A good tutorial can be found here, https://cran.r-project.org/web/packages/text2vec/vignettes/text-vectorization.html.

Let's first create a vocabulary-based DTM. Here we collect unique terms from all documents and mark each of them with a unique ID using the `create_vocabulary()` function. We use an iterator to create the vocabulary.

Running LDA on dtm

```r
library("topicmodels")
load("../output/dtm_list.RData")


# LDA parameters
burnin <- 4000
iter <- 1000
thin <- 200
seed <-list(2003,2,68,100001,765)
nstart <- 5
best <- TRUE
delta=0.1


# number of topics
k <- 10
alpha<-50/k


# run lda on each name
theta_list<-list(name=NULL,theta=NULL)
phi_list<-list(name=NULL,phi=NULL)

for (i in 1:length(query.list)){
  dtm<-dtm_list[[i]][[2]]
  rowTotals <- apply(dtm , 1, sum)
  dtm  <- dtm[rowTotals> 0, ]
  # Run LDA using Gibbs sampling
  ldaOut <-LDA(dtm, k, method="Gibbs",  control=list(nstart=nstart,
                                                  seed = seed,
                                                  best=best,
                                                   burnin = burnin,
                                                   alpha=alpha,
                                                   delta=delta,
                                                   iter=iter,
                                                   thin=thin))
  # theta_dj;
  # Probabilities associated with each topic assignment for each record
  theta <- as.data.frame(ldaOut@gamma)
  # phi_mj
  # the probability of using word m in topic j
  phi <- posterior(ldaOut)$terms

  temp1<-list(name=query.list[i], theta=theta)
  theta_list[[i]]<- temp1
  temp2<-list(query.list[i], phi)
  phi_list[[i]] <- temp2
}

## rerun LDA on combined dtm
load("../output/dtm_list_combined.RData")
dtm_combined<-dtm_list_combined
rowTotals <- apply(dtm_combined , 1, sum)
dtm_combined  <- dtm_combined[rowTotals> 0, ]
# Run LDA using Gibbs sampling
ldaOut_combined <-
  LDA(dtm_combined, k, method="Gibbs", control=list(nstart=nstart,
                                                   seed = seed,
                                                   best=best,
                                                   burnin = burnin,
                                                   alpha=alpha,
                                                   delta=delta,
                                                   iter=iter,
                                                   thin=thin))
  # theta_dj;
  # Probabilities associated with each topic assignment for each record
theta_combined <- as.data.frame(ldaOut_combined@gamma, rownames(dtm_combined))
  # phi_mj
  # the probability of using word m in topic j
phi_combined <- posterior(ldaOut_combined)$terms

save(theta_combined,file="../output/theta_combined.RData")
save(phi_combined,file="../output/phi_combined.RData")
```

i think there's an error, right now in the last column of the theta_combined, there there are only only variations of YChen so when im filtering, it creates a problem. Im not sure where the other names went. There are 8000+ rows which is right, but the labeling is off. @keng

```r
load("../output/theta_combined.RData")
load("../output/theta_list.RData")


#clustering on the combined theta list (from 1 dtm)
#create a column of names (14 distinct ones) to filter dtm by names later
theta_combined$name<-  sub("(.*?)_.*", "\\1", row.names(theta_combined))
```

```r
name.list <- unique(theta_combined$name)
cluster.list.combined=NULL
for (i in name.list){
  print(i)
  Theta <- filter(theta_combined, theta_combined$name == i)
  #Theta<- as.matrix(as.data.frame(theta_combined[[3]][2]))
  Theta<- as.matrix(as.data.frame(Theta[,-ncol(Theta)]))
  distance<-dist(Theta, method = "euclid")
  cluster <- hclust(distance, method = "complete")
  #cut tree at h= 0.15, differ from the 0.05 proposed. read explanation in doc
  cut <- cutree(cluster, h=0.15)
  unique(cut)
  cluster.list.combined[[which(name.list== i)]] <- cut
}

save(cluster.list.combined,file="../output/cluster_list_combined.RData")




#clustering on the theta formed from the non-combined dtm
cluster.list.separate=NULL
for (i in 1:length(theta_list)){
  print(i)
  Theta<- as.matrix(as.data.frame(theta_list[[i]][2]))
  Theta<- Theta[,-11]
  distance<-dist(Theta, method = "euclid")
  cluster <- hclust(distance, method = "complete")
  cut <- cutree(cluster, h=0.15)
  unique(cut)
  cluster.list.separate[[i]] <- cut
}
save(cluster.list.separate,file="../output/cluster_list_separate.RData")
```

Step 4: Evaluation

To evaluate the performance of the method, it is required to calculate the degree of agreement between a set of system-output partitions and a set of true partitions. In general, the agreement between two partitioins is measured for a pair of entities within partitions. The basic unit for which pair-wise agreement is assessed is a pair of entities (authors in our case) which belongs to one of the four cells in the following table (Kang et at.(2009)):

Let M be the set of machine-generated clusters, and G the set of gold standard clusters. Then. in the table, for example, a is the number of pairs of entities that are assigned to the same cluster in each of M and G. Hence, a and d are interpreted as agreements, and b and c disagreements. When the table is considered as a confusion matrix for a two-class prediction problem, the standard "Precision", "Recall","F1", and "Accuracy" are defined as follows.

$$\text{Precision} = \frac{a}{a+b}$$
$$\text{Recall} = \frac{a}{a+c}$$
$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$
$$\text{Accuracy} = \frac{a+d}{a+b+c+d}$$

```r
source('../lib/evaluation_measures.R')
load("../output/cluster_list_separate.RData")
load("../output/cluster_list_combined.RData")
name_list<-unlist(strsplit(row.names(theta_combined),"_"))
name_ID <- data.frame(matrix(name_list, ncol=3, byrow=T)[,1:2])
names(name_ID)<-c("Name", "AuthorID")

performance.list.separate=list()
for (i in 1:length(theta_list)){
  G<-name_ID$AuthorID[name_ID$Name==query.list[i]]
  M<-cluster.list.separate[[i]]
  matching_matrix_separate <- matching_matrix(G,M)
  performance <- performance_statistics(matching_matrix_separate)
  performance.list.separate[[i]]<-performance
}

performance.separate <- data.frame(matrix(unlist(performance.list.separate), ncol=3, byrow=T))

save(performance.list.separate,file="../output/performance.list.separate.RData")
performance.separate<-data.frame(matrix(unlist(performance.list.separate),ncol=4,byrow=TRUE))
names(performance.separate)<-c("precision","recall","f1","accuracy" )
save(performance.separate,file="../output/performance_separate_df.RData")


performance.list.combined=list()
for (i in 1:length(theta_list)){
  G<-name_ID$AuthorID[name_ID$Name==query.list[i]]
  M<-cluster.list.combined[[i]]
  matching_matrix_combined <- matching_matrix(G,M)
  performance <- performance_statistics(matching_matrix_combined)
  performance.list.combined[[i]]<-performance
}

save(performance.list.combined,file="../output/performance.list.combined.RData")
performance.combined<-data.frame(matrix(unlist(performance.list.combined),ncol=4,byrow=TRUE))
names(performance.combined)<-c("precision","recall","f1","accuracy" )
save(performance.combined,file="../output/performance_combined_df.RData")


perf_s<-colMeans(performance.separate)
perf_c<-colMeans(performance.combined)
compare_df <- data.frame(perf_s,perf_c)
names(compare_df)<-c("Separate","Combined")
kable(compare_df,caption="Comparision of performance for two LDA methods",digits = 2)
```

Table 1: Comparision of performance for two LDA methods

	Separate	Combined
precision	0.26	0.29
recall	0.24	0.26
f1	0.22	0.25
accuracy	0.79	0.81