Topic Networks

A mini project for Social Network Analysis

Final Version

I. Introduction

AUTOMATIC text summarization is a challenging problem whose solution would allow users to browse large document collections and quickly view highlights and drill down for details. In this project we attempt to bring the power of social network analysis to bear on this problem by developing a new approach to summarizing document collections, using topic models and bipartite graphs.

Our method first builds a topic model for a document collection using Latent Dirichlet allocation[3]. This topic model naturally defines a bipartite graph[1] between documents and the topics that appear in them. The bipartite graph is converted into a network of topics by linking topics if they appear in the same document. By analyzing a simple sample corpus, we explore properties of these *topic networks* in hopes that they will provide insights useful in summarizing the underlying document collection.

II. A Simple Example

TO demonstrate topic networks we will use a simple example consisting of texts of 102 speeches given by Barack Obama prior to his 2009 Inauguration scraped from obamaspeeches.com[5] on April 12, 2013. We'll use the **R** programming language[6] for this demo. All the steps are reproduced below and in an R source file available on GitHub[2]. We have borrowed a lot of ideas from Solomon Messing's blog post[4] on working with bipartitie/affiliation networks in **R**.

A. The Document-Term Matrix

IF the path to the directory containing the texts of Obama speeches is in the variable `corpus.source`, then we can build the corpus and document-term matrix with the following commands:

```
# Load the text mining package
require(tm, quietly=TRUE)
# Build the corpus.
corpus.source <- DirSource(corpus.directory,
  encoding="UTF-8")
corpus <- Corpus(corpus.source)
corpus.copy <-corpus
# Build the document term matrix.
corpus.dtm <- DocumentTermMatrix(
  corpus,
  control = list(
    stemming = TRUE,
    stopwords = TRUE,
    minWordLength = 3,
    removeNumbers = TRUE,
    removePunctuation = TRUE
  )
)
```

B. Frequent Words

TO gain some preliminary insight into this corpus, we'll look at a bar chart of most frequent words and a word cloud.

```
# Load required packages:
require(RColorBrewer, quietly=TRUE)
# Sort terms by frequency:
word.freq <- sort(col_sums(corpus.dtm),
  decreasing=TRUE)
# barplot with the top 20 most frequent terms
top.terms <- head(word.freq, n=20)
# complete the stems and fix missing values and errors:
completions <- stemCompletion(names(top.terms),
  dictionary=corpus.copy, type="prevalent")
completions<-ifelse(completions == "", names(top.terms),
  completions)
names(top.terms) <-completions
names(top.terms)[10] <-"every"
pdf("mostfreq.pdf")
op <- par(mar = c(4,6.1,.1,.2))
barplot(top.terms, las=2, horiz=TRUE)
dev.off()
# wordcloud
pal2 <- brewer.pal(8,"Dark2")
top.terms <- head(word.freq, n=1200)
completions <- stemCompletion(names(top.terms),
  dictionary=corpus.copy, type="prevalent")
completions<-ifelse(completions == "", names(top.terms),
  completions)
names(top.terms) <-completions
pdf("wordcloud.pdf")
par(mar = c(0,0,0,0))
wordcloud(
  words=names(top.terms),
  freq=top.terms,
  min.freq=20,
  random.order=F,
  colors=pal2,
  rot.per=.15
)
dev.off()
```

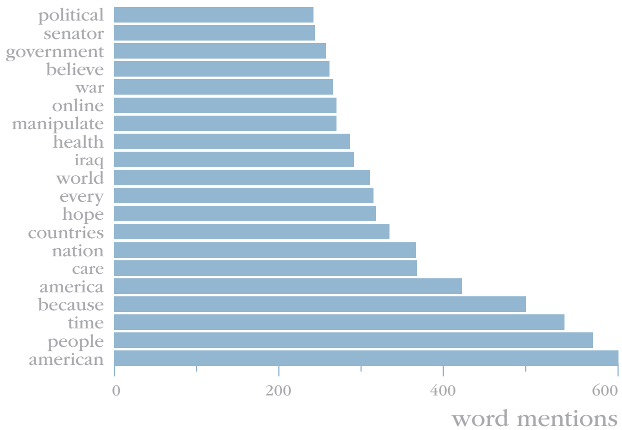

Fig. 1

Most frequent words

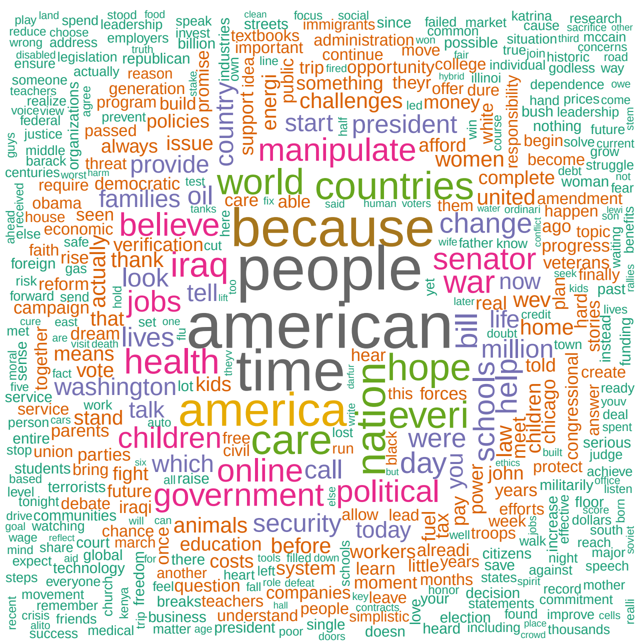

Fig. 2

Wordcloud: frequency ≥ 20

C. The Topic Model

Before building a topic model, we want to filter out "unimportant" words from the corpus. For this we will use the tf-idf matrix.

```
# Build the tf-idf matrix:
term.tfidf <- tapply(
 corpus.dtm$v/row_sums(corpus.dtm)[corpus.dtm$i],
 corpus.dtm$j, mean) *
 log2(nDocs(corpus.dtm)/col_sums(corpus.dtm > 0))
# Filter out "unimportant" terms:
dtm <- corpus.dtm[,term.tfidf >= 0.01]
dtm <- dtm[row_sums(dtm) > 0,]
```

Now we build the topic model and collect collect relevant data in a data frame.

```
require(topicmodels, quietly=TRUE)
# Build a topic model:
corpus.tm <- LDA(dtm, k = 22)
corpus.tm.terms <- terms(corpus.tm, 3)
corpus.tm.topics <- topics(corpus.tm, 4)
topic.labels <- apply(corpus.tm.terms, 2,
  function(x) paste(x, collapse=", "))
document.labels <- colnames(corpus.tm.topics)
dt.df <- data.frame(
  document=rep(document.labels, each=4),
  topic=as.vector(corpus.tm.topics)
)
# Replace doucment labels with numbers:
dt.df$document <- as.numeric(gsub(".txt","",
  dt.df$document))
# Order the data according to document number:
dt.df <- dt.df[order(dt.df$document),]
```

D. The Document-Topic Network

The first network we'll consider has two modes: documents and topics. The `igraph` package allows us to construct a network from an incidence matrix as follows.

```
require(igraph, quietly=TRUE)
# Build the incidence matrix:
dt.matrix <- as.matrix(table(dt.df))
# Build the document-topic network:
dt.network <- graph.incidence(dt.matrix)
```


```
corpus.tm <- LDA(dtm, k = 22)
corpus.tm.terms <- terms(corpus.tm, 3)
corpus.tm.topics <- topics(corpus.tm, 4)
topic.labels <- apply(corpus.tm.terms, 2,
  function(x) paste(x, collapse=", "))
document.labels <- colnames(corpus.tm.topics)
dt.df <- data.frame(
  document=rep(document.labels, each=4),
  topic=as.vector(corpus.tm.topics)
)
# Replace doucment labels with numbers:
dt.df$document <- as.numeric(gsub(".txt","",
  dt.df$document))
# Order the data according to document number:
dt.df <- dt.df[order(dt.df$document),]
```


Figure 3, which was generated with the code below, shows the document-topic network, with topic nodes displayed as large green dots to distinguish them from document nodes.

```
n.docs <- nrow(dt.matrix)
n.topics <- ncol(dt.matrix)
n.vertices <- n.docs + n.topics
V(dt.network)$color[1:n.docs] <-
  rgb(1,0,0,.4)
```

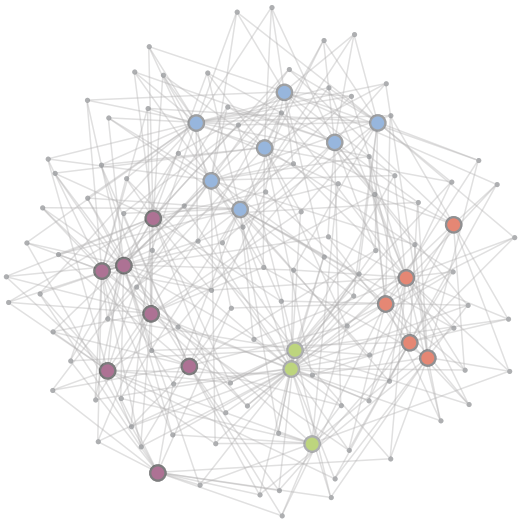

Fig. 3
Document-Topic Network

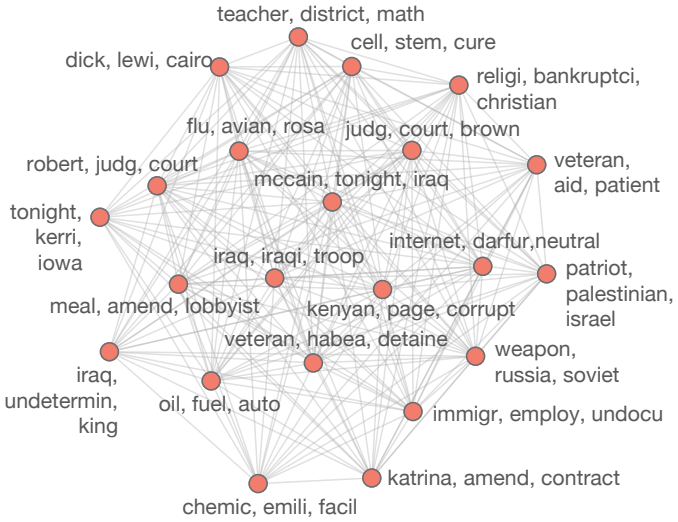

Fig. 4
Topic Network

```
V(dt.network)$color[(n.docs+1):n.vertices] <-
  rgb(0,1,0,.5)
V(dt.network)$label <- NA
V(dt.network)$size[1:n.docs] <- 2
V(dt.network)$size[(n.docs+1):n.vertices] <- 6
E(dt.network)$width <- .5
E(dt.network)$color <- rgb(.5,.5,0,.4)
pdf("dtn.pdf")
plot(dt.network,
  layout=layout.fruchterman.reingold)
dev.off()
```

E. The Topic Network

THE topic network is the one of most interest to us. To build it we create its adjacency matrix by multiplying the transpose of the document-topic matrix with itself.

```
# Create the topic network:
topic.network.matrix <- t(dt.matrix) %*%
  dt.matrix
topic.network <- graph.adjacency(
  topic.network.matrix, mode = "undirected")
```

Figure 4, which was created with the code below, shows the topic network, with topic nodes labelled by the three most probable terms in the corresponding topic distribution.

```
E(topic.network)$weight <-
  count.multiple(topic.network)
topic.network <- simplify(topic.network)
# Set vertex attributes
V(topic.network)$label <- topic.labels
V(topic.network)$label.color <- rgb(0,0,0,1)
V(topic.network)$label.cex <- .75
V(topic.network)$size <- 8
V(topic.network)$color <- rgb(0,1,0,.6)
# Set edge gamma according to edge weight
egam <- (log(E(topic.network)$weight)+.3)/
  max(log(E(topic.network)$weight)+.3)
E(topic.network)$color <- rgb(.5,.5,0,egam)
pdf("topic-network.pdf")
plot(topic.network, layout=layout.kamada.kawai)
dev.off()
```

As Figure 4 shows, the topic network is very busy, suggesting that we've not done a good job choosing the most import topics for each document. In fact, as the code below shows, the average degree, at 19.3, is very high.

```
# Compute the average degree of our network:
degrees <- V(topic.network)$degree
topic.network.average.degree <- sum(degrees) /
  length(V(topic.network))
```

This problem can probably be fixed if rather than taking the top 4 topics for each document in the document-topic network, we were to choose only those topics whose probability of appearing in the document is above a certain threshold. We plan to investigate that approach but don't have time to pursue it here.

Once we've solved the previous problem, we plan to systematically investigate the connections between important properties of the topic network and the underlying corpus. As an example, consider centrality:

```
# Betweenness centrality
V(topic.network)$btwcnt <- betweenness(topic.network)
V(topic.network)$label[order(V(topic.network)$btwcnt)]
```

This measure suggests that the most central topics in the corpus are

- tonight, mccain, page

- religion, bankruptcy, christian
- oil, fuel, auto
- iraq, iraqi, troop

F. The Overlap Network

ANOTHER interesting view of this data takes into account the percent overlap between topics, which gives rise to a directed network. To create this graph, we divide each row by the diagonal to get the adjacency matrix.

```
overlap.matrix <- topic.network.matrix /
  diag(topic.network.matrix)
overlap.network <- graph.adjacency(
  overlap.matrix, weighted=T)
```

We use the density plot show in Figure 5 to choose a reasonable cut-off value for removing some noise in the edges before plotting this network.

```
pdf("density.pdf")
plot(density(overlap.matrix), main=NA, xlab=NA)
dev.off()
overlap.matrix[overlap.matrix < 0.2] <- 0
overlap.network <- graph.adjacency(overlap.matrix,
  weighted=T)
overlap.network <- simplify(overlap.network,
  remove.multiple=FALSE, remove.loops=TRUE)
overlap.network$layout <- layout.kamada.kawai(
  overlap.network)
V(overlap.network)$label <- topic.labels
tkplot(overlap.network)
overlap.network$layout <- tkplot.getcoords(1)
```

The final step to display this network in a way to make the important properties visable requires some manual work. We'll use the tkplot gui tool to help us layout nodes so that labels are clearly visible. Once we're done arranging nodes, the layout can be saved using `tkplot.getcoords(1)`:

```
overlap.network$layout <- layout.kamada.kawai(
  overlap.network)
V(overlap.network)$label <- topic.labels
tkplot(overlap.network)
overlap.network$layout <- tkplot.getcoords(1)
```

Finally, we set a number of properties to improve the visualization:

```
V(overlap.network)$label.color <- rgb(0,0,.2,.6)
V(overlap.network)$label.cex <- .75
V(overlap.network)$size <- 6
#V(topic.network)$frame.color <- NA
V(overlap.network)$color <- rgb(0,1,0,.6)

# Set edge gamma according to edge weight
egam <- (E(overlap.network)$weight+.1)/max(
  E(overlap.network)$weight+.1)
E(overlap.network)$color <- rgb(.5,.5,0,egam)
E(overlap.network)$arrow.size <- .3
V(overlap.network)$label.cex <-
  degree(overlap.network)/(max(degree(
```

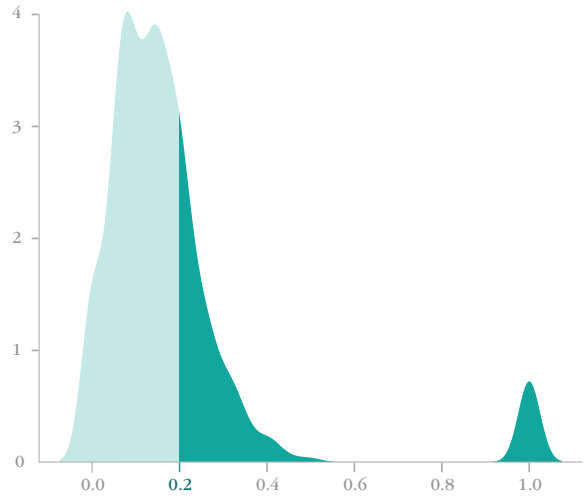

Fig. 5
DENSITY PLOT

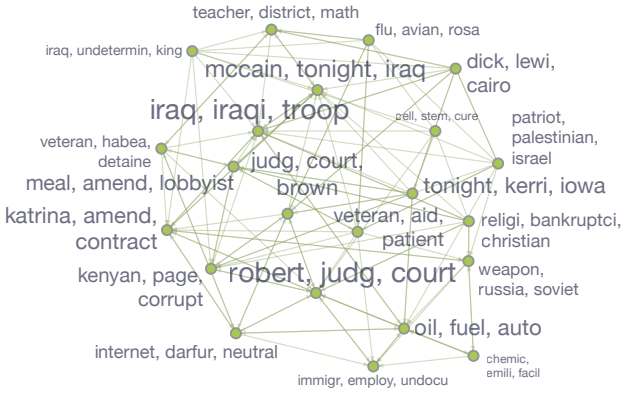

Fig. 6
TOPIC OVERLAP NETWORK

```
    overlap.network)/2))+ .3
pdf("overlap-network.pdf")
plot(overlap.network)
dev.off()
```

Figure 6 shows the results.

III. CONCLUSIONS

OUR simple example suggests that topic networks may be useful in analyzing document collections. However, we have only scratched the surface here and much more work is required to confirm this hypothesis as well as to investigate properties of topic networks and applications to document collection summarization.

REFERENCES

[1] From Wikipedia, the free encyclopedia *Bipartite graph*,

http://en.wikipedia.org/wiki/Bipartite_graph
[2] Topic Networks GitHub repository
https://github.com/bobflagg/Topic-Networks
[3] From Wikipedia, the free encyclopedia
Latent Dirichlet allocation,
http://en.wikipedia.org/wiki/Latent_Dirichlet_allocation
[4] Solomon Messing,
Working with Bipartite/Affiliation Network Data in R,
http://solomonmessing.wordpress.com/2012/09/30/working-with-bipartiteaffiliation-network-data-in-r/
[5] Best Speeches of Barack Obama through his 2009 Inauguration
http://obamaspeeches.com/
[6] R Core Team (2012), *R: A language and environment for statistical computing.*, http://www.R-project.org
[7] Jorge L. Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, *Human Activity Recognition Using Smartphones Dataset.*, Smartlab - Non Linear Complex Systems Laboratory
[8] R Core Team (2012), *R Markdown*, http://www.rstudio.com/ide/docs/authoring/using_markdown Accessed 2/12/2013
[9] Trevor Hastie, Robert Tibshirani, Jerome Friedman *The Elements of Statistical Learning*, Springer, 2011.