DME LAB 2

January 2011. Krzysztof Gorgolewski
February 2012. Victor Hernandez-Urbina
School of Informatics, University of Edinburgh.

LATENT DIRICHLET ALLOCATION WITH MALLET

In the first part of this lab we are going to have a look at MALLET - a machine learning toolbox that implements (among other algorithms) LDA which you should recall from the lectures. In the second part, we will do a small -but instructive- example of LSA in R. Let's start!

1. First you need to download mallet and create an alias to make your life easier

   ```
   wget http://mallet.cs.umass.edu/dist/mallet-2.0.6.tar.gz
   tar zxvf mallet-2.0.6.tar.gz
   alias mallet={write here the path where you extracted the file}/mallet-2.0.6/bin/mallet
   ```

   Also have a look the the MALLET documentation realted to topic analysis.

2. Data set you will be working on today can be found here. Remember to uncompress this file before moving on.

3. First you need to import the dataset into a format understandable by mallet (this may take a while):

   ```
   mallet import-file --input imdb-reviews.txt --output imdb-reviews.mallet --keep-sequence
   ```

4. Now we are ready to train our model. We will use 50 topics for 100 iterations. Additionally we are going to save the final set of topics to imdb-reviews-topics.txt and the Inferencer object to inferencer.mallet. We are going to use it later for inferring topics from new documents (this may also take a while).

   ```
   mallet train-topics --input imdb-reviews.mallet --num-topics 50 --inferencer-filename inferencer.mallet --num-iterations 100 --output-topic-keys imdb-
   ```

5. From the generated topics you can see that clearly there is something wrong. Some of the words should not be included in the analysis. You can exclude common words during import by setting '--remove-stopwords' flag or provide your own list '--stoplist-file' (one word per line). So, you should import the file once again to continue.

6. Try playing with different number of topics and iterations. Do the topics make sense? Can you interpret them and assign labels?

7. Go to http://www.imdb.com and pick a movie synopsis. Save it (just the content) to a text file (everything should be in one line) and import into mallet by typing:

   ```
   mallet import-file --input new_sample_review.txt --output new_sample_review.mallet  --keep-sequence --remove-stopwords --use-pipe-from imdb-reviews.ma
   ```

8. Now you can try to infer which topics are prevalent in the review you have picked.

   ```
   mallet infer-topics --input new_sample_review.mallet  --inferencer inferencer.mallet --output-doc-topics inferred_topics.txt
   ```

   The results will be saved in inferred_topics.txt.

LATENT SEMANTIC ANALYSIS WITH R

Now, let's turn our attention to LSA in R. Open the R console as you did in the previous lab. Once, you are there, you must install the proper package to do LSA in R.

1. In the console type the following commands to install and load the LSA package.

   ```
   install.packages("lsa")
   library(lsa)
   ```

   When the system ask you if you would like to create a personal library for R, answer yes.

2. Next, download the documents to which you will perform LSA. Follow this link, then uncompress the file to a folder named "docs/" in your current working directory.

3. We will create a matrix containing the word frequencies in each of the documents of our data. Type:

   ```
   matrix<-textmatrix("docs/", stopwords=c("the","a","an","in","of","for","to","and"))
   ```

   And then, inspect the contents of this new variable. (Don't forget to specify -in the above command- the name of the folder in which you uncompressed the files!)

4. Now, create a latent semantic space by doing:

   ```
   LSAspace<-lsa(matrix,dims=dimcalc_raw())
   ```

   You should take a look at what the function dimcalc_raw() does. Also, inspect the contents of the LSA space. Do you understand what you see? If not, take a look at the lecture notes and to the help article of the function lsa(). Remember that LSA is performing a PCA transformation on the data.

5. The lsa() function is really a thin wrapper over a singular value decomposition. Let's try two ways to see this:

   - Run:

     ```
     svd(matrix)
     ```

Compare this to the output of lsa(). What do you notice?

- The function for matrix multiplication in R is called %*%. If A and B are matrices, then A %*% B returns their product. Try:

```
round(LSAspace$tk %*% diag(LSAspace$sk) %*% t(LSAspace$dk))
```

What do you notice here? What does the remind you from the lecture?

6. Next, try this slight change on the LSA space:

```
newLSAspace<-lsa(matrix, dims=2)
```

This last command is identical to the last one, however now we are specifying the dimension of the LSA space.

7. Compare both LSA spaces. What are their differences? How many topics can we find in each of them?
8. Now, we will reconstruct the original space based on this second LSA space. Do:

```
newMatrix<-round(as.textmatrix(newLSAspace),2)
```

And take a look at it and compare it to the first matrix. Do you notice anything strange in this reconstruction?

9. Next, we will find close terms in the textmatrix. The function associate() returns those terms above a threshold close to the input term, sorted in descending order of their closeness. Try:

```
associate(matrix,"computer")
```

And then,

```
associate(newMatrix,"computer")
```

What do you see? Try this with other terms.

10. Now, let's make a simple version of the plot that we saw in the lecture. Type:

```
t.locs<-newLSAspace$tk %*% diag(newLSAspace$sk)
```

What do you see in this plot? If this is not clear, then try running the following two commands and try again to interpret this plot.

```
> plot(t.locs,type="n")
> text(t.locs, labels=rownames(newLSAspace$tk))
```

11. Ok. That's all for this lab. Now, it's up to you whether to use LSA or LDA if you are interested in doing some topic modelling. Now, you might close R by typing

```
q()
```

12. Once again, if you happen to have some spare time, take a look to the following links:

- Misha Glenny on why hackers should be hired.
- Eli Pariser: on the effects of online filtering.
- Read here about a car that drives itself by using algorithms of AI.

Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, Scotland, UK
Tel: +44 131 651 5661, Fax: +44 131 651 1426, E-mail: school-office@inf.ed.ac.uk
Please contact our webadmin with any comments or corrections.