# CSE 5243 Lab 1 Report

Young Suk Cho

Ph.D. student in Computer Science Engineering

cho.648@osu.edu

## 1. Goal

The goal of the lab 1 is to generate a feature matrix that consists of a feature vector information of each document in the given document set files. There are three important tagged texts such as <topic>, <body>, and <place> in the document. Therefore, what I have done is parse the information from the given document, then organized all the information as a table shaped feature matrix.

## 2. Feature vector generation

Matlab was used for the development. There are five program files (.m) as described below

a. **textPreProc.m**: Main driver function. It reads the text file and send the raw texts to stemmer.m so they are translated to their original forms. After the stemming it generates TF-IDF matrix and the feature vector.
b. **stemmer.m**: Stems each term in all the body documents to convert it into its original form (past, past participle → present, plural → singular)
c. **extIndcs.m**: Extracts all the index information for each tag and link each data element with its ID number.

Basically, I composed the feature matrix such that each row represents each article in all the document sets and each column represents a feature word. Each feature word is either a topic word or a word chosen from its corresponding body text that has the highest TF-IDF value among those of all the words in the same body text. My system works in order of the following steps.

a. Read all the Reuter text files. Each text file except the last one contains 1000 articles in it.
b. Locate every index of <topic>, <body>, <places>.
c. Stem all the selected words in body text using external dictionary website. The criteria to select words to be stemmed are specified at the item 3-d below. The information regarding the external dictionary website is described at the item 3-a below.
d. Generate TF and IDF matrices based on the following formulas.

$$\mathrm{tf}(t,d) = 0.5 + \frac{0.5 \times \mathrm{f}(t,d)}{\max\{\mathrm{f}(t,d) : t \in d\}}, \quad \mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|},$$

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D)$$

   e. Starts generating a feature matrix using topic words. If there is any article does not have a topic word then picks most important word of that article based on TF-IDF value of that article.

3. **Challenges encountered and approaches to solve**
   At the beginning, I was trying to use python to program supported by python natural language toolkit (NLTK). However, it was unsuccessful since the system must be able to run on stdlinux and it requires a sudo privilege to import the library by installing the toolkit. Therefore, I had to choose Matlab as an alternative. This impacted
   Regarding the dataset, I have found six challenging points during the feature vector development. Thus, I addressed each problem described as below:
   a. There articles that have missing either <title>, <topic>, or even <body> information in the document set intermittently.
   b. Since I had changed my direction from python to Matlab very early, I was not able to exploit NLTK stemmer. Therefore, I had to use an external dictionary named "dictionary.com". To get the original form of each word, I made a crawler that queries every word to the website and extract the original word from the retrieved web page.
   c. The number of retrieved features was too many if I wanted to take all the terms as features. Thus, I had to lose the terms that are not frequently appears either in each document or over all the document sets. I removed the terms in IDF matrix that appear less than 19 times and the terms appear less than twice in each article.
   d. There were lots of terms that includes not only letters and numbers, but also includes special symbols. Moreover, there were the terms that were consisted with only numbers. Therefore I did not included a word to the feature vector if it includes any special characters (other than alphabet letters) or consist only with numbers.
   e. Even after I removed many unnecessary words, there were still some articles or prepositions such as "the", "of", and etc. have TF-IDF values high enough to get into the feature vector. Thus, I removed some words that have too high TF values to reduce the size of feature matrix.

4. **Feature vector sample**
   a. Topic feature vector
      i. **Selected features**: 'acq' 'alum' 'austdlr' 'barley' 'bfr' 'bop' 'can' 'carcass' 'castoroil' 'castorseed' 'citruspulp' 'cocoa' and etc.
   b. Body feature vector
      i. **Selected features**: 'bank' 'year' 'with' 'billion''cts' 'share' 'as' 'company' 'hā' 'market' 'an' 'trade' and etc.
   c. Place feature vector
      i. **Selected features**: 'elsalvador usa u...' 'usa' 'usa' 'usa brazil' 'usa' 'argentina' and etc.

5. **Miscellaneous**
    a. I adopted some utility functions that are built-in functions of the latest version of Matlab but not provided by stdlinux Matlab from open source websites as listed below:
    http://www.mathworks.com/matlabcentral/fileexchange/21710-string-toolkits/content/strings/strjoin.m
    https://github.com/mim/mimlib