Interactive Text Retrieval and Multi-document Text Summarization of large collection of documents

Thesis Research Advisor: Dr. Wlodek Zadrozny

1) Key Technologies

Natural Language Processing, Information Retrieval, Machine Learning, Clustering, Neural Network, Deep Learning, Text Summarization, Python

2) Background

When user enters search query to search documents or web pages, he/she gets overwhelmed with a lot of relevant documents or web pages retrieved using usual search techniques. A naive search query can be matched with multiple documents or web pages that belong to different sub-categories or sub-domains. And what if a user is interested only in one sub-category or sub-domain. So, we need a search engine (system), which can re-organize the retrieved results into different clusters of co-related documents or web pages and helps a user to get a required document or web page in the efficient and interactive fashion. Such a technique can further be made effective by providing multi-document summary for every cluster of documents or web pages.

3) Objective

The purpose of the project is to create a system that can summarize ad hoc collections of documents. There can be multiple scenarios as below:

- The user tries to find relevant prior art, and queries an IR system
- The system shows 500 most relevant and 5000 not so relevant hits
- The task is to characterize the group constituting the 500, and find a contrasting characterization of the next 5000 hits.

4) Progress status

So far, 55% task is accomplished.

5) Implementation

This project has two parts. First part is to organize retrieved documents into different clusters of cohesive documents and second part is to briefly summarize every cluster. These steps will be repeated based on cluster choice made by user.

5.1) First part

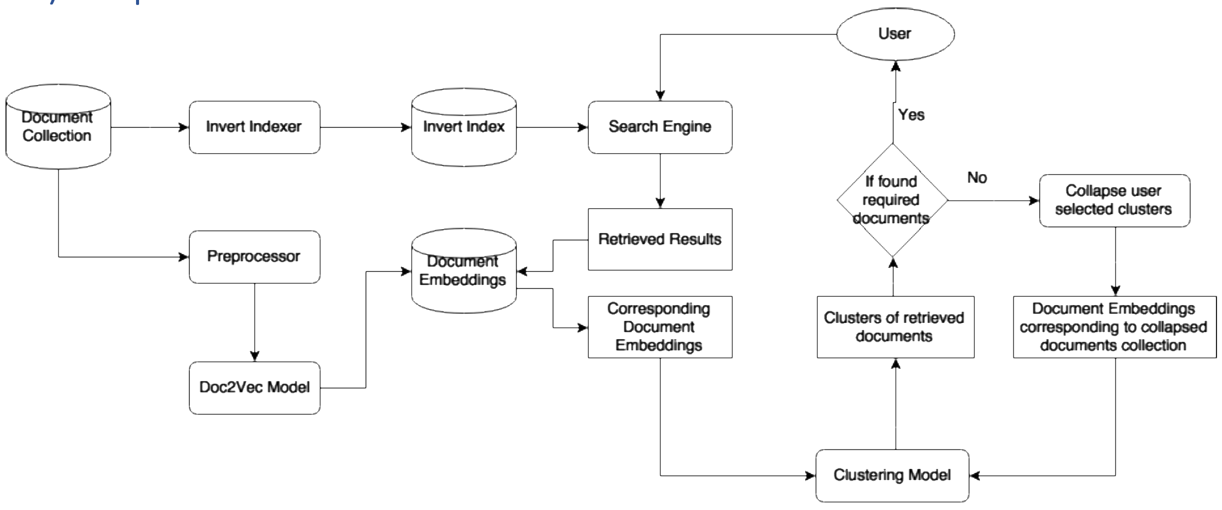
User
Document Collection
Invert Indexer
Invert Index
Search Engine
Yes
If found required documents
No
Collapse user selected clusters
Retrieved Results
Preprocessor
Document Embeddings
Corresponding Document Embeddings
Clusters of retrieved documents
Document Embeddings corresponding to collapsed documents collection
Doc2Vec Model
Clustering Model

Figure 1.0: Working flow for the first part of the thesis research

1. **Preprocessing:**
    - Replace control characters with white-spaces
    - Pad punctuation symbol with spaces on both sides
    - Remove stop-words
    - Conceptualize text using Conceptualizer model [1]
      Note: ngram limit is set to 1-8, i.e. This arrangement will generate concepts with total tokens up to 8.
    - Lemmatize conceptualized text
      Note: We use WordNet lemmatizer and lemmatize only unigrams in conceptualized text. Internally, it uses Perceptron POS tagger to identify part of speech of every unigram token. And then accordingly, it lemmatizes the conceptualized text.

2. **Patent2Vec Model:**
    - Create Patent2Vec model on top of Doc2Vec model [2] to generate document embedding
      Note: Below is the details of model's hyper-parameters,
        - ➢ Use Distributed Memory model/algorithm for Paragraph vectors with averaging context vectors.
        - ➢ Discard words with minimum word frequency < 5, which ultimately reduces the vocabulary size by disallowing not so important words
        - ➢ Document embedding size = 500
        - ➢ Context window size = 8
        - ➢ Negative sampling is set to 2, which indicates how many noise words to be drawn
    - Build vocabulary of concepts or unigrams by referring corpus of documents
    - Initialize vocabulary with concept embedding from pre-trained Conceptualizer model
    - Train Patent2Vec model
      Note: Use learning rate from 0.1 to 0.0001 and let the genism adjust it. Shuffle documents after every pass to make sure the model generalization

- Save trained model
- Store the learned document embedding into SQLite database for performing lookup and clustering afterwards

3. **Document Clustering:**
    - Create document clustering model based on Cluto library [3]
      Note: Below is the details of model's hyper-parameters,
        - Method: rbr, bisecting K-Means with global optimization
        - Criterion: I2, maximize similarity between centroid of cluster and document vectors that are assigned to
        - Similarity Metrics: Cosine similarity
        - Choice of cluster for bi-section: Pick the one such that bisecting it will optimize clustering criterion function
        - Do not normalize document embedding, as its dimensions have well distributed values
    - Get the list of document ids corresponding to the documents retrieved for the given search query
    - Perform lookup into database and retrieve the document embedding corresponding to documents ids mentioned in above step
    - Create matrix of document vectors as required by Cluto library
    - Run clustering application
    - If one gets the desired cluster of documents, stop the algorithm;
      Otherwise pick the relevant clusters, collapse documents within selected clusters and reorganize them by re-clustering till one does not find the required document or cluster of documents

5.2) Second Part

In the second part, we are thinking to use LSTM neural network to summarize every cluster of documents We are expecting to complete its implementation in Fall-2017.

6) Learning

- Natural Language Processing techniques like stop-word removal, lemmatization, POS tagging, regex matching
- Use of Gensim and Doc2Vec to generate Paragraph Vectors for documents
- Different clustering techniques which can applied to document corpus
- Text conceptualization

7) References

1. Walid Shalaby, Wlodek Zadrozny. Learning Concept Embeddings for Efficient Bag-of-Concepts Densification. Journal of Artificial Intelligence Research
2. Quoc Le, Tomas Mikolov. Distributed Representations of Sentence and Documents. 2014
3. Michael Steinbach, George Karypis and Vipin Kumar. A comparison of Document Clustering Techniques. KDD Workshop on Text Mining, 2000