Document Retrieval System with Automatic Summarization and Multiple Format Support

Yun Da Tsai
CSIE
NTU
Taipei, Taiwan
b04902103@ntu.edu.tw

Yan Ci Su
CSIE
NTU
Taipei, Taiwan
b04902108@ntu.edu.tw

Bo Han Shih
CSIE
NTU
Taipei, Taiwan
b04902025@ntu.edu.tw

Ray Chang
CSIE
NTU
Taipei, Taiwan
b04902093@ntu.edu.tw

***Abstract*—In this paper, we proposed a document retrieval system that aims to help find documents on the computer in a very fast manner. The system supports multiple file format and automatic summarization for users to quickly examine the content of file. We investigate several recent state-of-the-art deep learning approaches. In our desinged system, we implement multiple summarization algorithms and a novel algorithm that leverages BERT, a pretrained transformer model, with k-means clustering to make extractive summaries. Please visit Demo website : http://140.112.187.116:8787 which is not include in the presentation.**

I. Introduction

In a complex computer file system, there are thousands of documents. We all have once ran into an awkward situation that we have forgotten where an important file is placed. Current systems can only search files with their filenames and cannot provide a better usability to find documents according to the content and find relevant documents. Imagine a company with a huge number of documents and very likely these documents are collected for over decades. Many documents may be out of management and many of them cannot be found. Our designed system comes in for the rescue! Our system supports multiple file format and provide a interface to upload new files into the system and integrate automatic summarization to provide users a fast way to examine the content of the files.

II. Proposed Framework

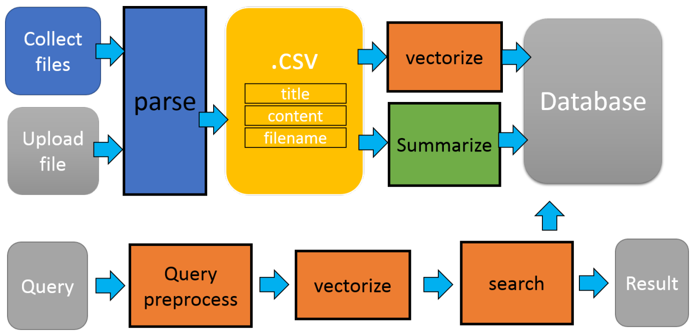

Fig. 1. System Overview

The system is mainly composed of four parts with a website as the interface.

1) multiple format parsing
2) document vectorization
3) document summarization
4) fast similarity search

III. Data Collection

Its nowadays very convenient for us to get data from web using web crawler, so we make a crawler using 'scrapy' to crawl ptt Gossip board. Our system is composed of around 10000 ptt pages and news data.

IV. Document Preprocessing

The data we have collected by the crawler is very messy so we have to clean it up to a more readable format.

A. Phrasing

Some data is relatively formal, news especially, and some is used in oral language. The former often phrased with commas and periods, and the latter phrased with new line character. Its necessary to set up some rules to deal with them.

B. Indexing

We did Chinese word segmentation with jieba, stop words removal, and then make the inverted file.

V. Multiple Format Support

Our system supports files of type .pdf and .docx.

VI. Retrieval Model

We try to retrive the most relevant documents based on their contents and titles. The followings are several models and methods, which we learned on IR classes, we've tried.

A. TFIDF + okapi + doc length normalization

$$TF(t,d) = \frac{(k+1)c(t,d)}{c(t,d)+k(1-b+b\frac{|d|}{avdl})}, k = 1.75, b = 0.75$$

where c(t,d) is the frequency count of term t in doc d.
$IDF(t) = \log(\frac{n}{k})$
where n is the total numbers of documents, and k is the numbers of docs with term t(doc frequency).
$Weight(t,d) = TF(t,d)IDF(t)$

B. Doc2Vec

The difference between Doc2Vec and Word2Vec is that besides the word vectors we also include paragraph embeddings to capture the paragraph, and we thought it would be helpful to make the vector more representative.

C. Similarity

From each file we derive a vector, we use cosine similarity to get the similarity between any documents or queries.

VII. Summarization

We use automatic summarization to provide user a quick overview of the content. Recent works focus on deep learning architectures and reinforcement learning. However, since we have difficulties collecting labeled data, we use only the extractive summarization methods. The followings are several algorithms we have implemented on the system.

A. TextRank

TextRank [1] is a graph-based ranking model for text processing. TextRank algorithm is modified as undirected and weighted graph and gives the following formula:
$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$
where d is a damping factor that can be set between 0 and 1, which has the role of integrating into the model the probability of jumping between vertex.
The algorithm consists of the following main step:

1) Pre-process the text: remove stop words and stem the remaining words.
2) Create a graph where vertices are sentences.
3) Connect every sentence to every other sentence by an edge. The weight of the edge is how similar the two sentences are.
4) Run the PageRank algorithm on the graph.
5) Sort the vertices(sentences) with the highest PageRank score.

B. LexRank

LexRank [2] is another stochastic graph-based method for computing relative importance of textual units. LexRank uses IDF-modified Cosine as the similarity measure between two sentences. LexRank also incorporates an intelligent post-processing step and Degree centrality scores which makes sure that top sentences chosen for the summary are not too similar to each other.

C. LSA

Latent Semantic Analysis [3] applied the singular value decomposition (SVD) to generic text summarization. The process starts with creation of a term by sentences matrix A = $[A_1, A_2, , A_n]$ with each column vector A_i, representing the weighted term-frequency vector of sentence i in the document under consideration. Metrics V^T after applying the SVD describes an importance degree of each topic in each sentence.

D. SummaRuNNer

SummaRuNNer [4] is a Recurrent Neural Network (RNN) based sequence model for extractive summarization of documents. Each sentence is visited sequentially in the original document order and a binary decision is made in terms of whether or not it should be included in the summary. The decision at each sentence depends on the content richness of the sentence, its salience with respect to the document, its novelty with respect to the accumulated summary representation and other positional features. We did not implement this one.

$$\begin{aligned} P(y_j = 1 | \mathbf{h}_j, \mathbf{s}_j, \mathbf{d}) = \sigma(W_c \mathbf{h}_j & \quad \texttt{\#(content)} \\ + \mathbf{h}_j^T W_s \mathbf{d} & \quad \texttt{\#(salience)} \\ - \mathbf{h}_j^T W_r \tanh(\mathbf{s_j}) & \quad \texttt{\#(novelty)} \\ + W_{ap} \mathbf{p}_j^a & \quad \texttt{\#(abs. pos. imp.)} \\ + W_{rp} \mathbf{p}_j^r & \quad \texttt{\#(rel. pos. imp.)} \\ + b), & \quad \texttt{\#(bias term)} \end{aligned}$$

Fig. 2. SummaRuNNer Objective

E. Transformer Extractive Summarizer

Recently, the transformer architecture has out performed RNN in multiple NLP tasks. Transformer has also been used on the summarization task [5], [6]. Here we make use of the pretrained bert model and calculate the similarities between sentences in the graph-based ranking algorithm with the bert embeddings. Using the default pre-trained BERT model, one can select multiple layers for embeddings. Using the [cls] layer of BERT produces the necessary N x E matrix. We use k-means clustering to rank the sentences and select k sentences closest to the centroids as our summary.

VIII. Contribution

Name	A work
Dada Tsai	Summarization
Ray chen	Retrieval model
Paul Shih	Data collection and Parsing
Chi Su	Web and pipeline

References

[1] Mihalcea, Rada, and Paul Tarau. "Textrank: Bringing order into text." Proceedings of the 2004 conference on empirical methods in natural language processing. 2004.

[2] Erkan, Gnes, and Dragomir R. Radev. "Lexrank: Graph-based lexical centrality as salience in text summarization." Journal of artificial intelligence research 22 (2004): 457-479.

[3] Steinberger, Josef, and Karel Jezek. "Using latent semantic analysis in text summarization and summary evaluation." Proc. ISIM 4 (2004): 93-100.

[4] Nallapati, Ramesh, Feifei Zhai, and Bowen Zhou. "Summarunner: A recurrent neural network based sequence model for extractive summarization of documents." Thirty-First AAAI Conference on Artificial Intelligence. 2017.

[5] Liu, Peter J., et al. "Generating wikipedia by summarizing long sequences." arXiv preprint arXiv:1801.10198 (2018).

[6] Miller, Derek. "Leveraging BERT for Extractive Text Summarization on Lectures." arXiv preprint arXiv:1906.04165 (2019).