Boğaziçi University

Cmpe 492

Senior Project in Computer Engineering

Extractive Text Summarization

Author:
Hilal Dönmez

Supervisor:
Prof. Tunga Güngör

June 3, 2018

Contents

1 Introduction

There have been huge amounts of text materials. Enormous amount of work is needed to review the large amounts of text data on a given topic. Therefore, there is a need to reduce the large text data to shorter text.

The aim of the text summarization is to extract the most important pieces of information from the text. Another substantial aim is to summarize the text with consistent, representative information. Summaries by hand have been quite successful. However, creation of summaries by hands is not possible because of huge amounts of text data. Thus, it is necessary that the text data is summarized automatically. Also, the automatic summarization takes into account the aims of summarization by hands. Automatic text summarization enables users to gain brief, consistent and representative summaries. Moreover, it provides that the reading time for the users decreases and the selection process of the documents is easier for users.

Automatic summarization pays attention to the similarity between sentences in the documents. Sentences are represented as semantically to measure similarity. In my project, continuous vector representations of sentences are used and evaluated on a dataset.

On the other hand, there are two types of automatic text summarizations which are extractive and abstractive summarizations. Extractive summarization obtains summary sentences from input document directly whereas abstractive summarization creates new sentences for summary with regard to the input document and summary compatibility.

Furthermore, texts which have been summarized consists of either single document or multi-documents. In the project, extractive multi-document summarization has been studied on the dataset.

2 Problem Description

The aim of the project is to summarize multiple documents automatically. The summary generated consists of the sentences from the original text, that type of summary is called extractive summarization. Summary which is formed should have overall meaning of the document and consist of distinctive sentences from the document. Also, summary should not very long. The number of sentences in the summary is fixed to 2 sentences in this project. Finally, summaries which are generated automatically are evaluated with

human reference summaries via ROUGE measurements.

3 Related Work

This projects is considered as two parts that are extractive summarization and summarization with deep learning.

3.1 Extractive Summarization

There are many approaches about extractive text summarization. First widely used approach is Maximum Marginal Relevance (MMR) which is a greedy approach to select sentences. Summary is produced by overlapping the documents and reducing the redundancy via MMR that is a linear combination of measurements of relevance and novelty in the document independently [1]. MMR gives scores sentences considering combination of relevance and redundancy with sentences in the summary. After that, summaries are created with the sentences that maximizes the MMR score.

Another approach is Integer Linear Programming (ILP) formulation of the problem. ILP produces a solution representing an optimal selection of sentences considering a summary length constraint. It uses efficient branch and bound algorithms for finding the optimal solution. This summarization takes into account information and redundancy at sentence level. The score of a summary is equal to the sum of the relevance scores of the sentences it contains minus the sum of the redundancy scores of each pair of these sentences. When generating the summary, this function should be maximized subject to the summary length [2].

On the other hand, there are various learning based summarization approaches. One of them is Support Vector Regression (SVR). It is used to estimate the importance of the sentences in the document. Regression model using SVR is created and this model learns continuous functions that estimate the importance of the sentences in the document thanks to training data. After that, final summaries are obtained via the model [3].

3.2 Summarization with Deep Learning

Text is summarized with deep learning techniques. There are various approaches. One of them uses RNN to rank sentences for multi-document

summarization. The aim of the ranking scores of the sentences and words is to select informative and non-redundant sentences in order to generate summaries. Sentence ranking task is described as a hierarchical regression process which measures the salience of a sentence in the parsing tree. This process is modeled by RNN. An advantage of the RNN is that RNN learns the ranking features automatically with hand-crafted feature vectors of words as input. In addition, getting consistent ranking scores from word to sentence with RNN makes the sentence selection more accurate. Finally, this model achieves the state-of-the-art performance [4].

Another approach is about a language model with Convolutional Neural Network. In this approach, sentences are projected in distributed representations. The language model generates the sentence representation with hierarchical neural network and combines it with the representations of context word to estimate the next word. This prediction based language model makes the similarity between sentences easier. After that, sentence selection process is considered as optimization problem with respect to the dissimilar sentences with each other and salient sentences in the documents [5].

At the end, automatic summarization is achieved by continuous vector representations for semantically aware representations of sentences. That is a base on sentence similarity. Distinct word vector representations are used and different combinations for producing sentence vector representations are created. Therefore, sentences are considered according to the semantic aspects thanks to sentence vector representation. In addition to the word vectors representations such as Word2Vec and Collobert & Weston, RNN is used in order to consider order of the words and grammar of the sentences. (Kågebäck et al., 2014) [6] trains the model with this perspective. Finally, this model achieves the state-of-the-art performance.

4 Dataset

Opinosis Dataset (Ganesan et al.,2010) have been selected for the extractive multi-document text summarization[7]. The dataset consists of 51 different topics. Each topic includes the collection of user reviews and between 50 and 575 sentences in the collection. This dataset is suitable for multi-document summarization. On the other hand, each topic has 5 gold summaries. Each gold summary consists of different summary of the same topic from others. Also, a summary in the gold summary is created by humans, not chosen by

the topic [6].

5 Methods and Progress

Extractive multi-document automatic text summarization is divided into two parts which are summarization framework and sentence representation for measuring sentence similarity.

5.1 Summarization Framework

In this project, submodular function optimization of Lin and Bilmes and Lexrank are implemented as a base of multi-document extractive summarization.

5.1.1 Submodular Function Optimization of Lin and Bilmes

This submodular function is run on all sentences of the document which is denoted as **V**. Intuition of the function for the summarization is that adding a sentence to small set of sentences have a greater effect than adding a sentence to larger one. The objective function of Lin and Bilmes have a target on finding a summary that maximizes the coverage of the input text and has more diverse sentences [6]. This objective function as follows:

$$F(S) = L(S) + \lambda R(S)$$

where S is the summary, *L(S)* measures the coverage of the summary set S to the document, *R(S)* rewards diversity in S, and λ is a trade-off coefficient which is between 0 and 1[8].

$$L(S) = \sum_{i \epsilon V} \min(\sum_{j \epsilon S} \text{Sim}(i,j), \alpha \sum_{j \epsilon V} \text{Sim}(i,j))$$

First summation in min function measures how similar S is to sentence i. If the result of the min function is equal to the $\alpha \sum_{j \epsilon V} \text{Sim}(i,j))$, it means that i is saturated by S and adding new sentence j cannot advance the coverage of the summary[8].

$$R(S) = \sum_{k=1}^{K} \sqrt{\sum_{j \epsilon S \cap P_k} \frac{1}{N} \sum_{i \epsilon V} \text{Sim}(i,j)}$$

where N is the number of sentences in the document, K is the number of clusters which K-means clustering algorithm generates. P_k is the set of sentences that are generated by clustering the document with K-means clustering algorithm. *R(S)* function implies that selecting a sentence from a cluster which the summary has no sentence from has greater effect on summary that selecting a sentence from other clusters which the summary has any sentence. In the project, K is set to 0.2N considering the experiments from Lin and Bilmes[8]. Also, α is a threshold coefficient between 0 and 1.

In this project, summary length is fixed to 2 sentences. Therefore, effect of *R(S)* function on summary is quite less than effect of *L(S)* function when maximizing the objective function.

5.1.2 Lexrank

Lexrank is a stochastic graph-based method for computing sentence importance in the document. A connectivity matrix with intra-sentence cosine similarity is generated in graph representation of sentences. This model states that the sentences which are similar to many of other sentences in a cluster are salient in the topic. Thus, sentence similarity is defined at first. In original model , similarity between two sentences are computed with tf-idf cosine measurement. In this experiment, we calculate the similarities with tf-idf and phrase embeddings with cosine measurement. Then, cosine similarity matrix is created for each type of measurements. In this model, each sentence is a node and each similarity pair is an edge in graph representation. After that, the overall centrality of a sentence given its similarity to other sentences is calculated. In order to do that, low similarity values in cosine similarity matrix are eliminated according to the threshold. Then, the number of similar sentence for each sentence are counted for evaluating sentence centrality. This method defines the degree centrality of a sentence as the degree of the corresponding node in the similarity graph. Also, degree centrality of each node provides a node with a vote to determine the overall centrality value of each node. However, sentence importance is not only determined with the number of related sentences, but also depends on which sentences are related with. Therefore, the centrality value of each node is distributed to the neighbor nodes. Then, transition matrix is generated in order to determine the transition probability from current node to next node in the corresponding Markov chain. A Markov chain with the stochastic matrix eventually converges to a stationary distribution. This new sentence

similarity measurement is called LexRank. In our experiment, two sentences having highest lexrank score are selected for the summarization [9].

5.2 Sentence Representation

This project focus on the different combinations of sentence representations in order to generate sentence representation. Firstly, the words are converted into vectors , which is word embeddings. Secondly, sentences are represented as vectors , which is called as phrase embeddings. Finally, similarity between two sentence vectors is calculated. Distinct combinations of sentence representations are obtained by using different models and methods in each processing steps.

5.2.1 Word Embeddings

Vector representation of the word is called as word embedding. Words are mapped to the vectors by several methods such as Continuous Bag-of-Words model (CBOW), Continuous Skip-Gram model, Collobert and Weston etc. It extracts semantic and syntatic information about the word thanks to vector representation. Semantically similar words are mapped to nearby points in the vector space. In this project, Continuous Skip-Gram model is implemented as word embedding.

Continuous Skip-Gram Continuous Skip Gram is a model that is used to produce word embeddings. This model predicts context words from given target word. This model is implemented in Word2Vec tool. In this project, "gensim" library is used for Continuous Skip Gram model. The model is trained with "Google News Dataset".

5.2.2 Phrase Embeddings

It is necessary that sentences are compared with each other for summarization. Therefore, sentences are converted into vectors after preparing the vector representation from words. In this project, three different phrase embedding methods are implemented. They are vector addition, average of vector addition and Doc2Vec.

Vector Addition The simplest way for vector representation of sentences is vector addition. In this methods, all word vectors in the sentence are added without considering the order of the word in the sentence.

Average of Vector Addition After all word vectors in the sentence are added, result vector is divided by the number of words in the sentence.

Doc2Vec Doc2Vec is used to produce sentence to vector. In this project, "gensim" library is used for Doc2Vec. This model is trained with both our dataset and "CNN Dataset" [10].

5.2.3 Measuring Similarity

After sentences are converted into vectors, the similarity between the sentences are regarded for summarization. Cosine similarity is implemented as similarity measurement in this project. "Sim(i,j)" means the similarity measurement between sentence i and sentence j.

6 Evaluation

The quality of automatic summaries are evaluated with the human reference summaries via ROUGE. ROUGE measures take into account of the overlapping units such as n-grams between the automatic summary and reference summaries. This project considers the Rouge-1 , Rouge-2 that counts the match in unigrams and bigrams respectively. Recall, precision and f scores are reported for each topic with multi references. Firstly, Rouge-N scores between a automatic summary and every reference summary of a topic are calculated. Then, the maximum of these Rouge-N scores is called as final Rouge-N score. After the evaluation of each topic result individually, the average of each topic scores are calculated and reported. Finally, the evaluation of the automatic summarization are obtained via ROUGE [11].

7 Results

In this project, word embeddings and phare embeddings are combined in order to produce summary. `W2V` is represented as Word Embedding. `Add` is

described as Vector Addition and `Avg` is called as Average of Vector Addition. Also, `S2V` is represented as Doc2Vec. As a result, $\texttt{W2V\_Add}_{Cos}$ expresses the summary with Word Embedding, Vector Addition and cosine similarity. On the other hand, two types of sentence selection methods which are Submodular Function Optimization and Lexrank are implemented. Each model with these selection methods produces a summary. Finally, summaries that are generated with these all combinations are evaluated with ROUGE. Results as follows:

- Summary with Submodular Function Optimization

Rouge-1			
	R	P	F
$\texttt{W2V\_Add}_{Cos}$	39.9	14.08	19.86
$\texttt{W2V\_Avg}_{Cos}$	41.52	14.38	20.49
$\texttt{S2V}_{-Cos}$	38.32	13.28	19.17

Rouge-2			
	R	P	F
$\texttt{W2V\_Add}_{Cos}$	11.30	2.49	3.95
$\texttt{W2V\_Avg}_{Cos}$	11.84	2.54	4.04
$\texttt{S2V}_{-Cos}$	10.62	2.24	3.63

Rouge-l			
	R	P	F
$\texttt{W2V\_Add}_{Cos}$	29.72	8.64	9.26
$\texttt{W2V\_Avg}_{Cos}$	29.91	9.05	9.72
$\texttt{S2V}_{-Cos}$	27.54	8.44	9.02

- Summary with Lexrank

Rouge-1			
	R	P	F
`tf-idf`	30.44	25.83	26.79
`W2V_Add`$_{Cos}$	41.53	14.38	20.57
`W2V_Avg`$_{Cos}$	41.14	14.39	20.42
`S2V`$_{\_Cos}$	33.11	17.29	21.78

Rouge-2			
	R	P	F
`tf-idf`	14.07	8.91	9.55
`W2V_Add`$_{Cos}$	13.50	3.09	4.66
`W2V_Avg`$_{Cos}$	13.41	2.94	4.54
`S2V`$_{\_Cos}$	10.37	4.07	5.60

Rouge-l			
	R	P	F
`tf-idf`	26.24	19.32	19.49
`W2V_Add`$_{Cos}$	31.46	9.06	9.47
`W2V_Avg`$_{Cos}$	31.76	9.02	9.55
`S2V`$_{\_Cos}$	25.08	12.81	13.83

At the end of the experiment, `S2V`$_{\_Cos}$ is more suitable for extractive multi-document text summarization in terms of continuous vector space. `S2V`$_{\_Cos}$ in the tables are trained with our own dataset because CNN dataset is not compatible domain for our dataset. Thus, I see that domain compatibility is very important for the training process. Results from `S2V`$_{\_Cos}$ pretrained with CNN dataset in Submodular Function Optimization are followings:

`S2V`$_{\_Cos}$			
	R	P	F
Rouge-1	40.10	9.71	15.38
Rouge-2	11.64	1.59	2.73
Rouge-l	28.35	6.18	6.46

8 Conclusion and Future Work

I have studied on the effect of word embedding and phrase embedding on extractive text summarization. Up to now, whereas Word2Vec as a word embedding is implemented, sentence vectors are obtained via vector addition, average of vector addition and Doc2Vec. Also, the similarity between sentences are found with cosine similarity. After getting similarity between each sentence with others, summaries are obtained with summarization framework. Finally, automatic summaries are evaluated with human reference summaries via ROUGE.

This project will also consider the paragraph to vector with deep learning. After the combination of the models are implemented, the differences between them will be evaluated. On the other hand, cosine similarity for sentence similarity is used up to now. Euclidean distance as a similarity measurement will be implemented with the summarization framework. Finally, words will be represented with CW vectors by Collobert & Weston method. The difference between Word2Vec and CW vectors will be considered on the summarization.

References

[1] Jaime Carbonell and Jade Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, New York, NY, USA, 1998. ACM.

[2] Dan Gillick and Benoit Favre. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, ILP '09, pages 10–18, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[3] You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. Applying regression models to query-focused multi-document summarization. *Inf. Process. Manage.*, 47(2):227–237, March 2011.

[4] Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. Ranking with recursive neural networks and its application to multi-document

summarization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2153–2159. AAAI Press, 2015.

[5] Wenpeng Yin and Yulong Pei. Optimizing sentence modeling and selection for document summarization. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 1383–1389. AAAI Press, 2015.

[6] Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. Extractive summarization using continuous vector space models. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)*, pages 31–39, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[7] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 340–348. Association for Computational Linguistics, 2010.

[8] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceeding HLT 11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, volume 1, page 514. Association for Computational Linguistics, 2011.

[9] `https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume22/erkan04a-html/erkan04a.html`.

[10] `https://cs.nyu.edu/~kcho/DMQA/`.

[11] Chin-Yew Lin. Rouge: a package for automatic evaluation of summaries. July 2004.