# Single document summarization using word and sentence embeddings

Ayana[1],[2]

[1]Department of Computer Science and Technology, Tsinghua University, 100084, China

[2]Department of Computer Information Management, Inner Mongolia University of Finance and Economics, 010060, China

**Keywords:** Single-Document Summarization; Word Embeddings; Sentence Embeddings

**Abstract.** Automatic document summarization has become increasingly important in the age of big data. However, traditional summarization systems have two major weaknesses: term-vector data sparsity and semantic information deficiency. In this paper, we solve these problems by adopting word and sentence embeddings, whose distributed nature enables the relation establishment between words and sentences. Experimental result shows that our models outperform two state-of-the-art graph based summarization systems.

## Introduction

Document summarization is an automatic process of constructing a simple and coherent summary. A good summary should retain most important contents of original document and present them in shorter text as much as possible. As summary greatly shortens time of reading, it is becoming increasingly significant in the fast-growing information age. Document summarization is first introduced by Luhn[1] and have received growing interest among computational linguistics. There are two types of summarizations: abstractive summarization and extractive summarization. In this paper we focus on producing fully automated single-document extractive summary.

The pioneer work of Luhn[1] simply used word frequency driven method. Each sentence is represented as a real number, which indicates how many important words the sentence includes. Dunning[2] improved Luhn's work by applying log-likelihood ratio test. Edmundson[3] proposed that not only important content words should be considered, but also three other features: cue words, title and heading words. His work has guided many machine learning based algorithms. Lexical chain based summarization[4] concentrates on solving the soft spot that the previous work did not pay attention to words semantic relations. The main strategy is trying to capture similarity between words using man-made lexical resources.

However, all the techniques presented above suffer from their own problems. Frequency based methods rely on shallow formal clues in the text. Machine learning algorithms require human generated summaries in their training session. Lexical chain system[4] heavily rely on existing manual lexical resources such as WordNet. To address these challenges, we propose a method to combine distributed representations of words and sentences into summarization task.

## The TextRank Model

TextRank[5] is a typical graph based summarization algorithm inspired by PageRank[6]. This algorithm organizes a document into a graph where each vertex represents a sentence. An edge indicates that the two sentences corresponding to linked vertexes share common content. The edge weight is defined as the number of lexical overlaps dividing sum of two sentence lengths, which indicates the connection strength between sentence pair. The ranking process is based on the following formula:

 

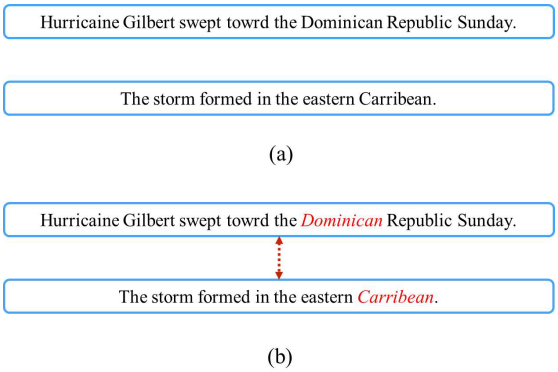


Fig. 1: Illustration of edge completion in TextRank model

$$W(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{overlap(s_i, s_j)}{\sum_{V_k \in Out(V_j)} overlap(s_j, s_k)} W(V_j) \quad (1)$$

$W(V_i)$ indicates the importance score assigned to vertex $V_i$, $In(V_i)$ stands for the set of vertices that point to $V_i$, $Out(V_j)$ denotes the set of vertices that $V_i$ point to. After several iterations, each vertex in the document is assigned with a score. Then the top-n sentences construct the summary for the document.

However, this algorithm has a serious drawback: missing semantic similarity information. If two sentences are talking about the same subject without using any same words, there will be no edge between them. We introduce distributed representations into the graph building process to solve this problem, Fig. 1 shows an example.

### Distributed Reperesentations of words and sentences

The concept of distributed representations is first introduced in 1986 by Rumelhart, Hinton and Williams[7]. Distributed word representations aim at representing words with dense, real-valued, fixed-length vectors, which is also known as word embeddings. Words with similar semantic and syntactic role will be placed near to each other in vector space. With the exploding text data on the Web and fast development of deep neural network technologies, distributed words representations have been effectively trained and used in natural language processing tasks[8][9][10][11]. Among them, the recently introduced Word2Vec toolkit[10] provides people many technical supports. In the word2vec framework, word vectors contribute to predict next word in a sentence[12].

In our work, we make use of word vectors and sentence vectors. Word vectors are derived from Word2Vec toolkit. Sentence vectors are gained from the framework of [12]. In the framework, sentence vector is concatenated with several words vectors from the sentence and predict the following word in the given context. Both word vectors and sentence vectors are trained by the stochastic gradient descent and backpropagation. At prediction time, the sentence vectors are inferred by fixing the word vectors and training the new sentences vector until convergence.

### Our models

**GoogleNews vectors Averaging.** GoogleNews vectors are trained by Skip-gram model with negative sampling. It is introduced by Miklov et al. in[9]. Skip-gram model runs as: given a sentence, select a word as input to a log-linear classifier, and predict words before and after the selected word in a fixed window. Negative sampling provides better vectors for frequent words with low dimension. GoogleNews vectors (about 100 billion words) could be easily downloaded from word2vec webpage.

It contains 300-dimensional vectors for 3 million words and phrases. To implement GoogleNews vectors into summarization, we generate each sentence vector by averaging vectors of component words. We implemente GoogleNews vectors Averaging into TextRank (GA-TR). When building graph for each algorithm, we employ cosine measure for computing sentence similarity.

**Background Corpus based Distributed Vectors.** Distributed bag of words(DBOW) model is proposed by Mikolov et al.[12]. It is similar to Skip-gram model in[9]. Each sentence in the background corpus is assigned with a sentence ID first. And sentence ID is also considered as a word. The model treats the whole background corpus along with sentence IDs as a data set, then runs the Skip-gram model on it. After the training process, every sentence and word gets an effective vector. These vectors are used to build graph for TextRank algorithm introduced in previous section. Cosine measure is adopted to compute similarities. We propose TextRank using Background Corpus based Distributed Vectors (TR-BCDV).

## Experiments

**Data.** Determined by the essence of extractive single-news-article summarization, we would make use of data set provided by DUC2002 task 1 as our test set. It is composed of 567 English news articles collected from TREC-9 and corresponding manual summaries. Each document have two corresponding manual summaries as reference summaries. In order to select proper parameters, we take DUC2001 test data set as out training set, which is composed of 309 English news article and its human generated summaries.

The distributed representations of words and sentences could be obtained through any training corpus. Nonetheless, the scale, genre and quality of training corpus will directly influence the validity of word or sentence vectors. Therefore, we constructed the training corpus for sentence vectors by only reserving English documents from DUC2001 to DUC2005 (some tasks of DUC are multi-lingual related, and those non-English documents are not included in). The training process is in accord with the framework described in[9].

**ROUGE Evaluation Metric.** ROUGE stands for Recall-Oriented Understudy of Gisting Evaluation[13]. It is widely adopted by DUC for automatic text summarization evaluation. We employ this toolkit to measure the quality of a model-generated summary, by comparing it to a human-generated "golden-standard" summary. ROUGE evaluation package calculates several kinds of scores. ROUGE-N is N-gram recall score between model summary and a set of reference summaries and the ROUGE-1 score agrees with human judgment most according to experiment results. The up-to-date version of ROUGE evaluation package ROUGE eval-1.5.5 could generate three scores (recall, precision and F-measure) for each evaluation.

**Experiments.** Table 1 shows the performances of TextRank algorithm using different distributed representations. To keep accordance with the original paper, we only report ROUGE-1 recall score under three circumstances: basic, stemmed, stemmed and no-stopwords. TR is baseline system TextRank, TR-BCDV indicates TextRank using Background Corpus based Distributed Vectors.

## Conclusion and Future Work

We have presented an approach to fuse distributed representations of words and sentences into document summarization task, and evaluated the feasibility upon two mature graph based summarization algorithms. It has been proven that either distributed representations of words or sentences can improve the performance of the two baseline systems. The background based sentence vector performed slightly inferior, because we obtained them upon small size of corpus.

In the future, we would get access to larger size of training corpus, to obtain improved sentence vector for our model. Moreover, since distributed word and sentence representations could capture

Table 1: Evaluation Results Under TextRank

| Systems | ROUGE score - Ngram(1,1) | | |
|---|---|---|---|
| | basic (a) | stemmed (b) | stemmed no-stopwords (c) |
| TR | 0.4708 | 0.4904 | 0.4229 |
| TR-BCDV | **0.4970** | **0.5187** | **0.4441** |
| GA-TR | **0.5181** | **0.5413** | **0.4643** |

the semantic relationships between words and sentences, we could apply them to multi-document summarization task.

**References**


[1] Luhn H P. The automatic creation of literature abstracts[J]. IBM Journal of research and development, 1958, 2(2): 159-165.

[2] Dunning T. Accurate methods for the statistics of surprise and coincidence[J]. Computational linguistics, 1993, 19(1): 61-74.

[3] Edmundson H P. New methods in automatic extracting[J]. Journal of the ACM (JACM), 1969, 16(2): 264-285.

[4] Brunn M, Chali Y, Pinchak C J. Text summarization using lexical chains[C]. Proc. of Document Understanding Conference. 2001.

[5] Mihalcea R, Tarau P. TextRank: Bringing order into texts[C]. Association for Computational Linguistics, 2004.

[6] Page L, Brin S, Motwani R, et al. The PageRank citation ranking: bringing order to the Web[J]. 1999.

[7] Hinton G E. Learning distributed representations of concepts[C]. Proceedings of the eighth annual conference of the cognitive science society. 1986, 1: 12.

[8] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. The Journal of Machine Learning Research, 2003, 3: 1137-1155.

[9] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]. Advances in neural information processing systems. 2013: 3111-3119.

[10] Mikolov T, Le Q V, Sutskever I. Exploiting similarities among languages for machine translation[J]. arXiv preprint arXiv:1309.4168, 2013.

[11] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[12] Le, Quoc V., and T. Mikolov. Distributed Representations of Sentences and Documents[J]. Eprint Arxiv 4(2014):1188-1196.

[13] Lin C Y. Rouge: A package for automatic evaluation of summaries[C]. Text summarization branches out: Proceedings of the ACL-04 workshop. 2004, 8.