Research on Text Summarization

Ying Qu

Abstract

The techniques of automatic summarization helps people extracting the crucial information from a huge amount of unimportant or redundant information. Text summarization is the process of distilling the most salient information from a text to produce an abridged version for a particular task and user (definition adapted from Mani and Maybury(1999)). Extract summarization is the process choosing original concepts in multi-documents. It is necessary to figure out the similarity between sentences, and we tried to calculate the similarity by using the approach of adjacent vectors notion for the representation of sentences semantically to compare with the Lin approach. On a standard dataset with ROUGE evaluation measures, we evaluate sentence and get the results. It turns out that there are some advantages of the method of adjacent word distribution vector for automatic summarization.

1.Introduction

Text summarization systems are generally described by their solutions to the following three problems: content selection, which is to decide what sentences to be extracted into summarization; information ordering, which is to structure the extracted units, sometimes we just use the original order for single document extractive summarization; sentence realization, which is to clean up the extracted units to be fluent in new context, merging information or fixing coherence for simplify sentence. Summarization is to get a brief salient information from a vast volumes of text or multiple documents, and to present a representative summary consequence. If we have a high quality of summary, it obviously will significantly decrease the workload to absorb or understand the specific documents for users. Currently, the best way to have an accurate summary is still made by humans. However, with the development of natural language processing, conditions changed. It is not necessary to be handmade for all types of summaries. For multi-document summarization, we estimate an approach of adjacent word distribution vector. According to the word embeddings on a standard dataset using the ROUGE， we experimented different sentence compositions for building representative sentences. The experiments show an improved performance and indicate beneficial behavior of using adjacent word distribution vector.

2. Deep learning for word embedding

The general method used in deep learning for word vector is distributed representation rather than one-hot representation. The dimension is general between 50 to 100. In my opinion, the significant contribution of distributed representation is to shorten the distance of relative or similar word. The distance measure can be represented by Euclidean distance or Cosine angle.

The concept of distributed representation is firstly come out from Hinton's paper Learning Distributed Representations of concepts in 1986 and to gain the attention since 2000. It is also called word representation or word embedding. To use the low dimension of word embedding escape the disaster of solving some complex tasks, for instance, building language model from Bengio et al 2003. It is impossible to use the characteristic of high dimension because of the huge volume workload.

2.1 Related Work

Words in documents are generally seen as discrete units. Bengio et al introduced word embeddings. A word embedding is a continuous vector representation which retrieve semantic and syntactic information about a word. Word embedding in a Euclidean space which reflects a priori notion of similarity between them. Therefore a text document can be viewed as a bag-of-embedded-words (BoEW): a set of real valued vectors.

The Vector Space Model (VSM) representation is at the root of topic models such as Latent Semantic Indexing (LSI) (Deerwester, 1988), Probablistic Latent Semantic Analysis (PLSA) (Hofmann, 1999) or Latent Dirichlet Allocation (LDA) (Blei et al., 2003).

A seminal work in this word embedding is the one by Collobert and Weston (Collobert and Weston, 2008) where a neural network is trained by stochastic gradient descent in order to minimize a loss function on the observed n-grams.

Socher et al introduced merging word representations into representations for phrases and sentences. The authors indicate a recursive neural network architecture (RvNN) which is for jointly learn parsing and phrase or sentence representations. The model is fast and easy. To derive the merged representations, the model only uses one neural network to be used recursively in a binary parse tree. The back draw is that labeled data is necessary for the system.

Google's pagerank is one of the most popular ranking algorithms. It is a graph-based ranking algorithm to decide the salient of a node within the graph by investigating the global information recursively rather than depending just on the local node specific information. Erkan and Radev first reported the application of PageRank in extractive summary. To generate links and define link strength, the similarity between two sentence was applied.

Bonzanini, Martinez, Roelleke introduced an algorithm which to begin with all sentences in the summary and iteratively selecting the most unimportant information to remove from the all sentences set. The experiment shows a good result on the Opinosis dataset.

Vector space word representations are good at capturing syntactic and semantic regularities in natural language processing.

2.2 Bengio three-layer neural network model

Bengio et al presents a three-layer neural network to build a language model in 2001 of paper A Neural Probabilistic Language Model. See figure 1

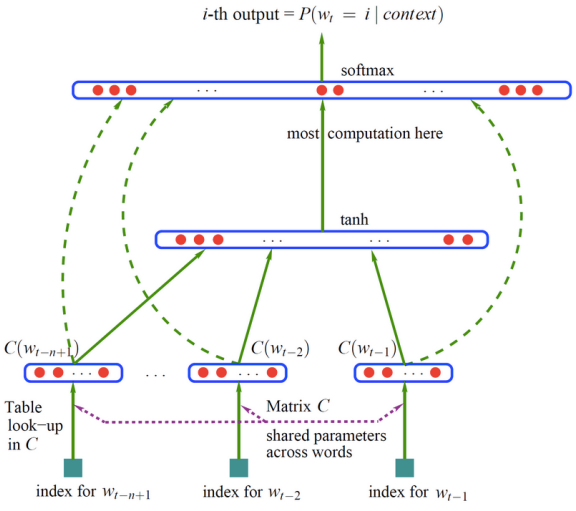

figure 1 Bengio three-layer neural network model

The problem is that we need to predict the next word wt according to the previous n-1 words: wt−n+1,…,wt −2,wt−1. C(w) is the word representation of word w existing in the matrix C (|v|*m) where |v| is the size of the word table and m is the dimension of word representation.

The first layer of the network is the input layer, to connect the n-1 vectors: C(wt−n+1),…,C(wt−2),C(wt −1) from head to tail, to get a (n-1)*m dimension vector, represented as x.

The second layer of the network is the hidden layer, similar to a normal neutral network, computing by d +Hx where d is an offset coefficient. Use tanh as activation function.

The third layer is the output layer consisting of |v| nodes. For each node yi is the next word I's log probability. And to use softmax as the activation function to summarize y as probability. The formula of y is y = b + Wx + U tanh(d + Hx), where U (a |V|*h) is the argument from hidden layer to output layer. There is another matrix W (|V|*(n-1)m) contains the straight edges from input layer to output layer. Though straight edge cannot improve the performance of model, it can reduce the nearly half of the iterative times.

Finally Bengio implement the model by Stochastic gradient descent method.

2.3 Mikolov RNNLM

Mikolov present the recurrent neural network based language model in 2010, the general principle is shown in figure 2.

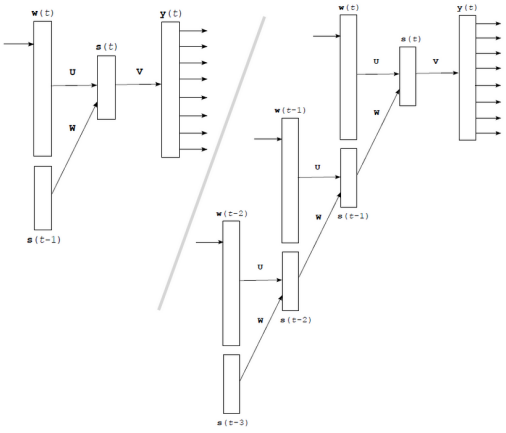

figure 2 RNNLM

The abstract of network structure is on the left side. RNNLM is used on chronological ordering. w(t) is the one-hot representation vector of the t word in sentence, therefore w is a very long vector with only one element of "1". s(t-1) is a vector in hidden layer. The formula to compute hidden layer is

$$s(t) = \text{sigmoid}(Uw(t) + Ws(t-1)). \quad (1)$$

We can figure out how RNNLM is expansion from the right side of figure 2. Each time coming a new word, it is combined with the last hidden layer to compute the following hidden layer, recursively to remain the latest status. Since w(t) is one-hot representation of a specific word, then Uw(t) is representative as choosing one column from matrix U, which is the corresponding word representation of the word.

The advantage of using RNNLM is that it makes full use of the context information to predict the next word.

Mikolov also presents an interesting idea in 2013. He finds that we can get the relationship of two word vectors by directly getting the difference of the two word vectors. For instance, C(king)−C(queen)≈C(man)−C(woman). C(king)−C(man)+C(woman)is approximate to C(queen). To analyze the characteristic of word vector, Mikolov use analogy method for testing. Given a,b,c, to see whether C(a) - C(b) + C(c) is approximating to d.

2.4 Vector addition phrase embeddings

Mikolov et al introduced the model for representing short phrases

$$\mathbf{x}_p = \sum_{\mathbf{x}_w \in \{\text{sentence}\}} \mathbf{x}_w \quad (2)$$

where xp is a phrase embedding, and xw is a word embedding. This is a simple equation to get the sum of all word embedding as a consequence of the phrase embedding which is used in our experiments.

3.Summarization

The automatic summarization techniques are basically divided into extractive and abstractive types. The goal of extractive summarization is to construct summaries by selecting original salient phrases or sentence from the input documents, whereas abstract summarization is to use different words to conclude the contents of the documents, and thus to be considered as a much more difficult problem.

We only estimate extractive multi-document summarization, which is to produce a condensation of the content of the entire group. There are basically two issues, the summarization framework and the similarity methods for getting comparison sentences in the extractive summarization techniques. The definition of the concepts and standard criteria applied to judge the importance of the phrases or sentence are to be considered. A simple and majority method applied in summarization is to extract key textual elements as weighted by tf*idf score. Phrases are selected according to the rank of the weight score.

3.1 Traditional Similarity Measure

The core technique for the majority of extractive summarization systems is the method of sentence similarity measures Sim(I,j). Lin and Bilms measure similarity between sentences by tf-idf (Salton and McGrill,1986) vectors sentence representation and to measure the cosine angle between vectors.

For the tf-idf, for each term I that occurs in the sentence evaluated, we compute its count in the current document tfi,j and multiply the inverse document frequency over the whole collection idefi:

$$\text{weight}(wi) = \text{tfi},j * \text{idfi} \quad (3)$$

Each sentence is represented by a word vector w = (w1, . . . , wN) where N is the size of the vocabulary.In sentence I of word k, weights wki is matching to the tf-idf value. The similarity measure is :

$$\text{Sim}(i,j) = \frac{\sum_{w \in i} \text{tf}_{w,i} \times \text{tf}_{w,j} \times \text{idf}_w^2}{\sqrt{\sum_{w \in i} \text{tf}_{w,i}^2 \times \text{idf}_w^2} \sqrt{\sum_{w \in j} \text{tf}_{w,j}^2 \times \text{idf}_w^2}} \quad (4)$$

where tfw,i and tfw,j are the number of occurrences of w in sentence i and j respectively, and idfw is the inverse document frequency (idf) of w.

The summarization algorithms computes this weight for every sentence, and then ranks all sentences by their score. The extracted summary consists of the top ranked sentences.

There must be an overlap of high tf-idf score words between two sentences in order to have a high similarity between sentences. In order to get higher

similarity, we tried the use of adjacent extract-worthy words vectors to measure the similarity between sentences.

3.2 Submodular Optimization

By monotone nondecreasing submodular set functions, Lin and Bilmes(2011) formulated the extractive summarization as an optimization problem. A submodular function F on the set of sentences V satisfies: for any A ⊆ B ⊆ V \{v}, F(A + {v}) − F(A) ≥ F(B + {v}) − F(B) where v ∈ V. This is a diminishing returns property which is to decrease the sentence set progressively and to retrieve the intuition by putting a sentence to a small set like summary rather than to a larger set. And then we need to maximize the coverage field of the input text and the diversity of the summary sentences by using the function F(S) = L(S) + λR(S) where S is the summary, L(S) is the coverage of the input text, R(S) is a diversity reward function. The λ is a coefficient to be defined to change the importance ratio between coverage of the input text with diversity of the summary sentences. If the objective function is submodular, we can use a fast scalable algorithm which turns out to be guaranteed approximation. Lin and Bilmes choose the submodular function as:

$$\mathcal{L}(S) = \sum_{i \in V} \min\{\sum_{j \in S} \mathrm{Sim}(i,j), \alpha \sum_{j \in V} \mathrm{Sim}(i,j)\} \quad (5)$$

where $\{\sum_{j \in S} \mathrm{Sim}(i,j),$ represents the similarity between sentence i and the summary S, $\alpha \sum_{j \in V} \mathrm{Sim}(i,j)$ represents the similarity between sentence i and the rest of the input V. Sim(i, j) is the similarity between sentence i and sentence j and 0 ≤ α ≤ 1 is a threshold coefficient.

4. Word2Vec

There are two models of word2vec: CBOW(Continuous Bag-of-Word Model) and Skip-gram model(Continuous Skip-gram Model). For each model, we can use hierarchical softmax strategy and negative sampling strategy. See figure 3 and 4.

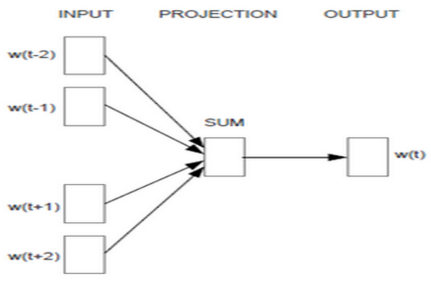

Figure 3 CBOW

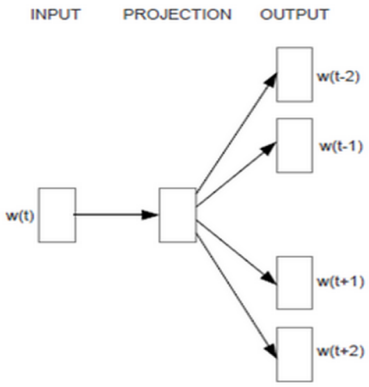

Figure 4 Skip-gram

Both models are three layers neuron network, including input, projection and output. CBOW is to predict the word wt by knowing the context of wt (wt-1,wt-2,wt+1,wt+2). On the opposite, Skip-gram is to predict the context information of wt. The number of the context words are flexible to set, for example 4 words in figure 3 and 4.

5.Experiments

We did experiments and comparison phrase embeddings with tf-idf based vectors.

5.1Experimental Settings

To evaluate the quality of the generated summaries, we choose an automatic summary evaluation metric ROUGE. ROUGE is a recall-based metric for fixed length summaries. It bases on N-gram co-occurrence and compares the system created summaries to the gold standard human summaries. Word2Vec vector addition is to be used for the configurations. It is compared by cosine similarity and Euclidean distance. W2Vcos for Word2Vec vectors combined using vector addition and compared using cosine similarity. W2VEuc for Word2Vec vectors combined using vector addition and compared using Euclidean distance.

5.2 Dataset and Evaluation

The Opinosis dataset (Ganesan et al., 2010) consists of short user reviews in 51 different topics. Each topic contains 50-575 sentences which are the user reviews made by different authors about a hotel, car or a product. As each sentence is considered as a single

document, this dataset is appropriate to be used for multi-document summarization. Four to five gold-standard summaries are built by human for each topic. To count the word overlaps between gold standard summaries and generated summaries, the summary is evaluated with ROUGE-1(unigram-based). Recall measures fraction of a gold standard summary which is captured, precision measures fraction of the generated summary which corresponds to gold standard. F-score = 2*Recall*Precision/(Recall +Precision).

We get the results by running implementation of Lin-Bilmes submodular optimization summarization.

5.3 Similarity measurement formula

5.3.1Cosine Similarity measurement

Using two vector similarity measure for the phrase embeddings summarization, the following presents a cosine similarity measure of the interval between 0 and 1:

$$\mathrm{Sim}(i,j) = \left(\frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_j\|\|\mathbf{x}_j\|} + 1\right)/2 \quad (6)$$

where x denotes a phrase embedding.

5.3.2Euclidean distance similarity measurement:

$$\mathrm{Sim}(i,j) = 1 - \frac{1}{\max_{k,n}\sqrt{\| \mathbf{x}_k - \mathbf{x}_n \|^2}}\sqrt{\| \mathbf{x}_j - \mathbf{x}_i \|^2} \quad (7)$$

ROUGE-1

	Recall	Precision	F-score
Optimal ROUGE	58.43	19.37	29.09
W2VCos	33.17	16.39	21.94
W2VEuc	29.08	16.25	20.85
Origianl	23.99	19.03	21.22

6.Conclusion and Future Work

The results of using word and phrase embeddings in extractive summarization show a higher score of some field than Lin-Bilmes, which proves that it is efficient for semantically aware summarization. Phrase embeddings capture useful information for extractive summarization. The embeddings can fundamentally make a decision of extracting the word worthy text and throwing away the redundant information from the input text. R-1 with a maximum recall of 58.43% is the result.

By using phrase embeddings of Word2Vec vector for summarization, the performance get improved comparing to the Lin and Bilmes methods. The phrase embeddings are computed from standard topic models showing significant improvements with respect to the original topic models. For the future work, I will try to enlarge the comparison by testing different rouges, different vectors to get a comparison of the results and find out what might be better to be used. By using a larger set similarity measures and summarization frameworks to get a detailed comparison for a much more persuasiveness result.

References

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 510-520. ACL.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. ArXiv preprint arXiv: 1301.3781.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. **A neural probabilistic language model**. Journal of Machine Learning Research (JMLR), 3:1137–1155, 2003.

Andriy Mnih & Geoffrey Hinton. **Three new graphical models for statistical language modelling**. International Conference on Machine Learning (ICML). 2007.

Andriy Mnih & Geoffrey Hinton. **A scalable hierarchical distributed language model**. The Conference on Neural Information Processing Systems (NIPS) (pp. 1081–1088). 2008.

Mikolov Tomáš. **Statistical Language Models based on Neural Networks**. PhD thesis, Brno University of Technology. 2012.

Eric Huang, Richard Socher, Christopher Manning and Andrew Ng. **Improving word representations via global context and multiple word prototypes**. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. 2012.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. **Linguistic regularities in continuous space word representations**. Proceedings of NAACL-HLT. 2013.

Marco Bonzanini, Miguel Martinez-Alvarez, and Thomas Roelleke. 2013. Extractive summarisation via sentence removal: Condensing relevant sentences into a short summary. In Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '13, pages 893–896. ACM.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In IEEE International Conference on Neural Networks, volume 1, pages 347–352. IEEE.

Gerard Salton and Michael J. McGill. 1986. Introduction to Modern Information Retrieval. McGrawHill, Inc., New York, NY, USA.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 384–394. ACL.