2011 International Conference on Advancements in Information Technology
With workshop of ICBMG 2011
IPCSIT vol.20 (2011) © (2011) IACSIT Press, Singapore

Automated Text Summarization Base on Lexicales Chain and Use of WordNet Knowledge Base

Mohsen Pourvali[1+] , Mohammad Sanee Abadeh and Mostafa etemadian

[1]Islamic Azad University of QAZVIN (qiau)

Abstract-The technology of automatic document summarization is maturing and may provide a solution to the information overload problem. Nowadays, document summarization plays an important role in information retrieval. With a large volume of documents, presenting the user with a summary of each document greatly facilitates the task of finding the desired documents. The lexical cohesion structure of the text can be exploited to determine the importance of a sentence/phrase. Lexical chains are useful tools to analyze the lexical cohesion structure in a text .In this paper we consider the effect of the use of lexical cohesion features in Summarization, And presenting a algorithm base on the knowledge base. Ours algorithm at first find the correct sense of any word, Then constructs the lexical chains, remove Lexical chains that less score than other ,detects topics roughly from lexical chains, segments the text with respect to the topics and selects the most important sentences. The experimental results on an open benchmark datasets from DUC01 and DUC02 show that our proposed approach can improve the performance compared to sate-of-the-art summarization approaches.

Keywords: Text Summarization, Data Mining, Word Sense Disambiguation

1. Introduction

The technology of automatic document summarization is maturing and may provide a solution to the information overload problem. Nowadays, document summarization plays an important role in information retrieval (IR). With a large volume of documents, presenting t[1]he user with a summary of each document greatly facilitates the task of finding the desired documents. Text summarization is the process of automatically creating a compressed version of a given text that provides useful information to users, and multi-document summarization is to produce a summary delivering the majority of information content from a set of documents about an explicit or implicit main topic (Wan, 2008). The main goal of a summary is to present the main ideas in a document in less space. If all sentences in a text document were of equal importance, producing a summary would not be very effective, as any reduction in the size of a document would carry a proportional decrease in its in formativeness. The paper (Jones, 2007) reviews research on automatic summarizing over the last decade. This paper reviews salient notions and developments, and seeks to assess the tateof-the-art for this challenging natural language processing (NLP) task. Sentence based extractive summarization techniques are commonly used in automatic summarization to produce extractive summaries. Systems for extractive summarization are typically based on technique for sentence extraction, and attempt to identify the set of sentences that are most important for the overall understanding of a given document. In our study we focus on sentence based extractive summarization. In this way we to express that The lexical cohesion structure of the text can be exploited to determine the importance of a sentence. Eliminate the ambiguity of the word has a significant impact on the inference sentence. In this article we will show that the separation text into the inside issues by using the correct concept Noticeable effect on the

[+] Corresponding author. Tel.: +988372225241.
E-mail address: Mohsen999@gmail.com.

summary text is created. The experimental results on an open benchmark datasets from DUC01 and DUC02 show that our proposed approach can improve the performance compared to sate-of-the-art summarization approaches.

2. Related Work

Generally speaking, the methods can be either extractive summarization or abstractive summarization. Extractive summarization involves assigning salience scores to some units (e.g.sentences,paragraphs) of the document and extracting the sentences with highest scores, while abstraction summarization usually needs information fusion, sentence compression and reformulation (Wan, 2008). The summarization techniques can be classified into two groups: supervised techniques and unsupervised techniques. In Supervised methods After representing each sentence by a vector of features, the classification function can be trained in two different manners (Mihalcea & Ceylan, 2007). One is in a discriminative way with well-known algorithms such as support vector machine (SVM) (Yeh, Ke, Yang, & Meng, 2005). Many unsupervised methods have been developed for document summarization by exploiting different features and relationships of the sentences – see, for example Erkan and Radev (2004) and the references therein. A query-based summary presents the information that is most relevant to the given queries (Dunlavy, O'Leary, Conroy, & Schlesinger, 2007;Wan, 2008) while a generic summary gives an overall sense of the document's content Dunlavy et al., 2007; Jones, 2007; Li et al., 2007; Wan, 2008). The QCS system (Query, Cluster, and Summarize) (Dunlavy et al., 2007) performs the following tasks in response to a query: retrieves relevant documents; separates the retrieved documents into clusters by topic, and creates a summary for each cluster. QCS is a tool for document retrieval that presents results in a format so that a user can quickly identify a set of documents of interest. In paper McDonald and Chen (2006) are developed a generic, a query-based, and a hybrid summarizer, each with differing amounts of document context. The generic summarizer used a blend of discourse information and information obtained through traditional surface-level analysis.

3. Word Sense Disambiguation

For extracting lexical chains in a document, all words and correct senses of these words should be known. Per word has a sense based on its position in the sentence In the algorithm presented in this paper, word sense are calculated locally. We use WordNet as an external source for disambiguation. In first the input text is pre-processing and the stop words is removed. Then stem of words is found and its (POS) is tagged. let w_i be a word in the document ,and w_i have n senses $\{w_{i_1}, w_{i_2}, \dots, w_{i_k}, \dots, w_{i_n}\}$.in this procedure for finding the meaning of two words related locally together and placed in the same sentence , we assume all of the possible meanings and senses of per word as the first level of the traversing word tree then we process every sense in a returning algorithm .Next , we connect all the relations for that sense as it's descendants ,and these descendants are generated through relations that are Hypernym ,... . We do this process in a returning manner for n levels. Next, every first level sense of the one word compare with all the first level senses of the other word .Afterwards, the number of equalities is considered in integer digit .the same comparison is done for another word .if there isn't any equality, for each word we choose first sense that is most common.

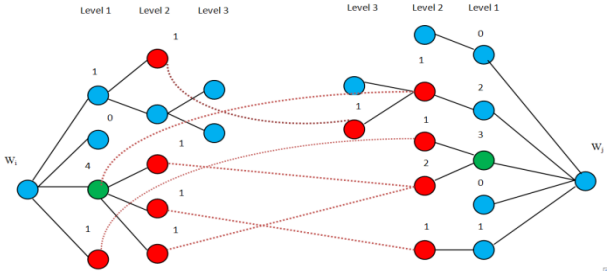

Fig. 1: Sample graph built on the 2 words

In the above figure, we illustrate the relations of the tree .the root of the tree is considered as the target word, and the first level nodes as the senses of the target words. The nodes of the second, third,...levels are senses related with the first level nodes with Hypernym ,… relations.

```
Function Hyp(ref Node t,int level)
      string[] sp;
     for  i = 0  to  EndOfFile(wn_hyp) do
          ReadLine_From_File(wn_hyp) ;  sp=Split_String_base_of('(', ',', ')')
          if t.index == sp[1] ;
             tnew=Create New Nod(sp[2]);  Call Hyp(ref tnew,level-1);
Add_New_Nod_ToList(tnew);
          end if
end for
```

Fig. 2: The algorithm for producing WordNet graph

The above algorithm is one of the functions used for producing WordNet graph. We use the great encyclopedia of Wikipedia because of the lack of special names in knowledge base of WordNet. we use creating the graph and traversing of it just for the first, middle, and last sentences, and it is useful because these sentences usually encompass concise expression of the concept of the paragraph in most of the documents .in this manner we decrease the space of interpretation and therefore the time of calculation and the space of memory because we just need to keep some highlight sentences related with each other. After clarifying the actual senses of the all words in the prominent sentences and with the similarities and relations between every pair of the words, we put them in incessant lexical chains. For each lexical chain LC_i , w_3^1 symbolizes that this word occur in the first sentence and the third sense of this word is chosen as the best sense. Lexical chains created at first are generated from highlight sentences, and we use different algorithm for putting other words of sentences in the relevant lexical chains. In this algorithm with some changes in Lesk algorithm, we use gloss concepts to represent similarities and differences of two words. let w1 , w2 are two words in text .firstly we extract senses of per word in normal Lesk algorithm from knowledge base

$$s1 \in sense\ (w1) and\ s2\ \in sense\ (w2) \tag{1}$$

Then we find overlaps between gloss concepts

$$(s1, s2) = |gloss(s1) \cap gloss(s2)| \tag{2}$$

And every two concepts that have more similarities are chosen as the target words. Moreover, we use not only uni-gram (sequence of one word) overlaps , but also bi-gram (sequence of two words) overlaps .if there is one of the senses the first word in gloss concepts of the second word, we give one special score to this two senses. We do this because two concepts may have common words that are not related with their similarities and it causes increasing in scores of those two senses and makes a mistake in choosing related word as a result. Considering the word sense in gloss concept of the second word's sense, we can award an additional chance to this sense to be chosen in process of choosing words for chains from words that are not semantically related in fact.

$$if\big(\text{s1} \in \text{gloss (s2) or } \text{s2} \in \text{gloss (s1)}\big)\ \ score\ (s1, s2) = \ score_{lest}(s1, s2) + \lambda \tag{3}$$

λ is an additional score, and considering average existed words in sense's gloss concept and experimental tests, we find that the best value for λ is 5 .

4. Clustring Lexical Chains

For each lexical chain LC_i, a sentence occurrence vector V_i is formed. $v_i = \{s_{1_i}, \dots, s_{k_i}, \dots, s_{n_i}\}$ Where n is the number of sentences in the document. Each s_{k_i} is the number of LCi members in the sentence *k*. If sentence *k* has 3 members of LCi then s_{k_i} is 3. Two lexical chains LCi and LCj go into the same cluster if their sentence occurrence vectors Vi and Vj are similar. Our clustering algorithm, starts from an initial cluster distribution, where each lexical chain is in its own cluster. Thus, our clustering algorithm starts with n clusters, where n is the number of lexical chains. Iteratively the most similar cluster pair is found and they

are merged to form a single cluster. Clustering stops when the similarity between the most similar clusters is lower than a threshold value. for this purpose we used the well known formula (4) from Linear Algebra:

$$Cos(\theta) = \frac{v_i + v_j}{||v_i||\,||v_j||} \tag{4}$$

In the equation $||v_i||$ represents the Euclidean length for the vector.

5. Sequense Extraction

In our algorithm, the text is segmented from the perspective of each lexical chain cluster, finding the hot spots for each topic. For each cluster, connected sequences of sentences are extracted as segments. Sentences that are cohesively connected are usually talking about the same topic. For each lexical chain cluster Cl_j, we form sequences separately. For each sentence S_k, if sentence S_k has a lexical chain member in Cl_j , a new sequence is started or the sentence is added to the sequence. If there is no cluster member in S_k, then the sequence is ended. By using this procedure, text is segmented with respect to a cluster, identifying topic concentration points. Fig. 3 is an example of Text Segmentation.

v_1={ 1 1 1 0 0 0 1 1 0 1 1 0 1 0 0 0 0 }
v_2={ 0 0 0 0 0 0 0 0 1 0 1 2 1 1 0 0 0 0 }

Fig. 3: An example of Text Segmentation

Each sequence is scored using the formula in Equation (5).

$$Score(Sequence_i) = Score(Cl_i) * l_i * \frac{(1+SLC_i)*PLC_i}{f^2} \tag{5}$$

Where l_i is the number of sentences in the $sequence_i$. SLC_i is the number of lexical chains that starts in $sequence_i$. PLC_i is the number of lexical chains having a member in $sequence_i$ and f is the number of lexical chains in cluster. Score of the cluster score (Cl_i), is the average score of the lexical chains in the cluster. Our scoring function tries to model the connectedness of the segment using this cluster score.

6. Expriments and Results

In this section, we conduct experiments to test our summarization method empirically. For evaluation the performance of our methods we used two document datasets DUC01 and DUC02 (http://duc.nist.gov) and corresponding 100-word summaries generated for each of documents. We use them because they are for generic single-document extraction that we are interested in and they are well preprocessed. There are many measures that can calculate the topical similarities between two summaries. For evaluation the results we use two methods. The first one is by precision (P), recall (R) and F1-measure which are widely used in Information Retrieval. For each document, the manually extracted sentences are considered as the reference summary (denoted by $Summ_{ref}$). This approach compares the candidate summary (denoted by $Summ_{cand}$) with the reference summary and computes the P, R and F1-measure values as shown in formula (6) (Shen et al., 2007)

$$P = \frac{|summ_{ref} \cap summ_{cand}|}{|summ_{cand}|} \qquad R = \frac{|summ_{ref} \cap summ_{cand}|}{|summ_{ref}|} \qquad F_1 = \frac{2PR}{P+R} \tag{6}$$

The second measure we use the ROUGE toolkit (Lin et al., 2003; Lin, 2004) for evaluation, which was adopted by DUC for automatically summarization evaluation. It measures summary quality by counting overlapping units such as the N-gram, word sequences and word pairs between the candidate summary and the reference summary. The ROUGE-N measure compares N-grams of two summaries, and counts the number of matches. The measure is defined by formula (7) (Lin et al., 2003; Lin, 2004; Svore, Vanderwende, & Burges, 2007)

$$\text{ROUGE} - \text{N} = \frac{\sum_{\text{S}\in\text{summ}_{\text{ref}}}\sum_{\text{N}-\text{gram}\in\text{S}}\text{Count}_{\text{match}}(\text{N}-\text{gram})}{\sum_{\text{S}\in\text{summ}_{\text{ref}}}\sum_{\text{N}-\text{gram}\in\text{S}}\text{Count}(\text{N}-\text{gram})} \quad (7)$$

Where N stands for the length of the N-gram, $\text{Count}_{\text{match}}$ (N-gram) is the maximum number of N-grams co-occurring in candidate summary and a set of reference–summaries. Count (N _ gram) is the number of N-grams in the reference summaries. We use two of the ROUGE metrics in the experimental results, ROUGE-1 (unigram-based) and ROUGE-2 (bigram-based). The parameters of our method are set as follows: depth of tree that is created for any word, *n=3*; extra value for *Lesk* algorithm, λ *=5*; Finally, we would like to point out that algorithm was developed from scratch in C#.net 2008 platform on a Pentium Dual CPU, 1.6 GHz PC, with 512 KB cache, and 1 GB of main memory in Windows XP environment.

6.1 Performance Evaluation and Discussion

We compared our method with four methods CRF (Shen et al., 2007), NetSum (Svore et al., 2007), Manifold–Ranking (Wan et al., 2007) and SVM (Yeh et al., 2005). Tables 1 and 2 show the results of all the methods in terms ROUGE-1, ROUGE-2, and F1-measure metrics on DUC01 and DUC02 datasets, respectively. As shown in Tables 1 and 2, on DUC01 dataset, the average values of ROUGE-1, ROUGE-2 and F1 metrics of all the methods are better than on DUC02 dataset. As seen from Tables 1 and 2 Manifold–Ranking is the worst method, In the Tables 1 and 2 highlighted (bold italic) entries represent the best performing methods in terms of average evaluation metrics. Among the methods NetSum, CRF, SVM and Manifold–Ranking the best result shows NetSum. We use relative improvement $\frac{(our\ method - other\ methods)}{our\ methods} \times 100$ for comparison. Compared with the best method NetSum, on DUC01 (DUC02) dataset our method improves the performance by 2.65% (3.62%), 4.26% (10.25%) and 1.81% (3.27%) in terms ROUGE-1, ROUGE-2 and F1, respectively.

Table 1: Average values of evaluation metrics for summarization methods (DUC01 dataset).

Methods	Av.ROUGE-1	Av.ROUGE-2	Av.F1-measure
Our method	0.47656	0.18451	0.48124
NetSum	0.46427	0.17697	0.47267
CRF	0.45512	0.17327	0.46435
SVM	0.44628	0.17018	0.45357
Manifold–Ranking	0.43359	0.16635	0.44368

Table 2: Average values of evaluation metrics for summarization methods (DUC02 dataset).

Methods	Av.ROUGE-1	Av.ROUGE-2	Av.F1-measure
Our method	0.46590	0.12312	0.47790
NetSum	0.44963	0.11167	0.46278
CRF	0.44006	0.10924	0.46046
SVM	0.43235	0.10867	0.43095
Manifold–Ranking	0.42325	0.10677	0.41657

7. References

[1] Alguliev, R. M., & Alyguliev, R. M. (2007). Summarization of text-based documents with a determination of latent topical sections and information-rich sentences. *Automatic Control and Computer Sciences , 41*, 132–140.

[2] Dunlavy, D. M., O'Leary, D. P., Conroy, J. M., & Schlesinger, J. D. (2007). QCS: A system for querying, clustering and summarizing documents. *Information Processing and Management , 43*, 1588–1605.

[3] Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research , 22*, 457–479.

[4] Jones, K. S. (2007). Automatic summarizing: *The state of the art. Information Processing and Management , 43,* 1449–1481.

[5] Lin, C. -Y. (2004). ROUGE: A package for automatic evaluation summaries. *In Proceedings of the workshop on text summarization branches out, (pp. 74–81)*. Barcelona, Spain.

[6] Lin, C. -Y., & Hovy, E. H. (2003). Automatic evaluation of summaries using N-gram co-occurrence statistics. *In Proceedings of the 2003 conference of the north american chapter of the association for computational linguistics on human language technology* (HLT-NAACL 2003), (pp. 71–78). Edmonton, Canada.

[7] Mihalcea, R., & Ceylan, H. (2007). Explorations in automatic book summarization. *In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning* (EMNLP-CoNLL 2007), (pp. 380–389). Prague, Czech Republic.

[8] [8] Navigli, R., & Lapata, M. (2010). An Experimental Study of Graph Connectivity for Unsupervised Word Sense Disambiguation. *IEEE Computer Society* , 32.

[9] Porter, M. (1980). An algorithm for suffix stripping. *Program , 14*, 130–137.

[10] [10] Radev, D., Hovy, E., & McKeown, K. (2002). Introduction to the special issue on summarization. *omputational Linguistics , 22*, 399–408.

[11] Salton, G., Singhal, A., Mitra, M., & Buckley, C. (1997). Automatic text structuring and summarization. *Information Processing and Management , 33,* 193–207.

[12] Shen, D., Sun, J. -T., Li, H., Yang, Q., & Chen, Z. (2007). Document summarization using onditional random fields. *In Proceedings of the 20th international joint conference on artificial intelligence (JCAI 2007)*, (pp. 2862–2867). Hyderabad, India.

[13] Svore, K. M., Vanderwende, L., & Burges, C. J. C. Enhancing single-document summarization by combining RankNet and third-party sources. *In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL 2007),* (pp. 448–457). Prague, Czech Republic.

[14] Wan, X. (2008). Using only cross-document relationships for both generic and topic-focused multi-document summarizations. *Information Retrieval , 11*, 25–49.

[15] Wan, X., Yang, J., & Xiao, J. (2007). Manifold-ranking based topic-focused multidocument summarization. *In Proceedings of the 20th international joint conference on artificial intelligence (IJCAI 2007)*, (pp. 2903–2908). Hyderabad, India.

[16] Yeh, J-Y., Ke, H-R., Yang, W-P., & Meng, I-H. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing and Management , 41,* 75–95.