www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 12 December 2014, Page No. 9528-9532

A survey on various summarization techniques

Divya Vidyadharan[1], Anju CR[2]

[1]KMCT College of Engineering, Calicut University
Kozhikode, Kerala, India
divyav.email@email.com

[21]KMCT College of Engineering, Calicut University
Kozhikode, Kerala, India
anjucr@email.com

Abstract: ***Text Summarization condenses a document or multiple documents into a smaller version by preserving its information content and the meaning. It is very difficult for a person to manually summarize large documents of text. TextSummarization methods can be classified into extractive and abstractive summarization. There are lots of techniques developed over the years for summarization. In this paper, some of the techniques are studied and reviewed.***

Keywords: summarization, lexical association, Bernoulli model, context indexing**.**

1. Introduction

Document summarization is an information retrieval task, which aims at extracting a condensed version of the original document. Readers will decide whether to read a complete document only after going through the summary. Even scientific people read a document only after reading the summary. Summarization has become an important tool.

The summary will provide the main details of the document. The main goal of a summary is to present the main idea in a document or a set of documents in a short and readable paragraph. Summaries can be produced either from a single document or many documents. The summaries produced from multiple documents are called multi-document summarizer.

A summary can be employed in an indicative way as a pointer to some parts of the original document, or in an informative way to cover all relevant information of the text. Text Summarization methods can be classified into extractive and abstractive summarization. An extractive summarization method consists of selecting important sentences, paragraphs etc. from the original document and concatenating them into shorter form. The importance of sentences is decided based on statistical and linguistic features of sentences. An abstractive summarization method consists of understanding the original text and re-telling it in fewer words. It uses linguistic methods to examine and interpret the text and then find the new concepts and expressions to best describe it by generating a new shorter text that conveys the most important information from the original text document.

There are various techniques of summarization. Some of them rely on centroid based techniques, semantic analysis. Some of the techniques will be biased to a particular topic, as in the case of query based summarization. These techniques actually retrieve documents related to a particular query. The QCS system (query, cluster, and summarize) retrieves relevant documents in response to a query, clusters these documents by topic and produces a summary for each cluster. Opinion summarization is another application of text summarization. Topic summarization deals with the evolution of topics in addition to providing the informative sentences.

The major issues for multi-document summarization are as follows: first of all, the information contained in different documents often overlaps with each other, therefore, it is necessary to find an effective way to merge the documents while recognizing and removing redundancy. Another issue is identifying important difference between documents and covering the informative content as much as possible .

2. Previous work

There are several techniques developed till date. Single document summarization focus on summarizing a single document. But in a single document summary, it will contain the summary of that document only. Multiple document summary has an advantage that it is a collective summary of all the documents together. One doesn't need to read summary of every document when it is a multi-document summary.

2.1 Centroid-based summarization of multiple documents

A multi-document summarizer, called MEAD [1] generates summaries using cluster centroids produced by topic detection and tracking system. The process of identifying all articles on an emerging event is called Topic Detection and Tracking (TDT). The entry in the official TDT evaluation, called CIDR , uses modified TF*IDF to produce clusters of news articles on the same event.

A new technique for multi-document summarization, called centroid-based summarization (CBS) which uses as input the centroids of the clusters produced by CIDR to identify which sentences are central to the topic of the cluster, rather than the individual articles. It is implemented CBS in a system, named MEAD.

A key feature of MEAD is its use of cluster centroids, which consist of words which are central not only to one article in a cluster, but to all the articles.

2.1.1. Centroid-based algorithm: MEAD decides which sentences to include in the extract by ranking them according to a set of parameters. The input to MEAD is a cluster of articles (e.g., extracted by CIDR) and a value for the compression rate r. For example, if the cluster contains a total of 50 sentences (n = 50) and the value of r is 20%, the output of MEAD will contain 10 sentences. Sentences are laid in the same order as they appear in the original documents with documents ordered chronologically. There are timestamps associated with each document.

$$\text{SCORE}(s) = \Sigma_i (w_c C_i + w_p P_i + w_f F_i) \qquad (1)$$

where i ($1 \leqslant i \leqslant n$) is the sentence number within the cluster.

2.1.2. Redundancy-based algorithm: A redundancy penalty (R_s) is subtracted for each sentence which overlaps with sentences that have higher SCORE values.

$$\text{SCORE}(s) = \Sigma_i (w_c C_i + w_p P_i + w_f F_i) - w_R R_s \qquad (2)$$

For each pair of sentences extracted by MEAD, we compute the cross-sentence word overlap according to the following formula:

$$R_s = 2 * (\#\text{ overlapping words}) / (\#\text{ words in sentence 1} + \#\text{ words in sentence 2}) \qquad (3)$$

$$w_R = \text{Max}_s (\text{SCORE}(s))$$

Rs = 1 when the sentences are identical and $R_s = 0$ when they have no words in common. After deducting R_s, we re-rank all sentences and possibly create a new sentence extract. We repeat this process until re-ranking doesn't result in a different extract.

2.2 Multi-Document Summarization via Sentence-Level Semantic Analysis and Symmetric Matrix Factorization

The multi-document summarization framework is based on sentence-level semantic analysis and symmetric non-negative matrix factorization is introduced [2]. First sentence-sentence similarities using semantic analysis and construct the similarity matrix is calculated. Then symmetric matrix factorization is done, which is equivalent to normalized spectral clustering, is used to group sentences into clusters. Finally, the most informative sentences are selected from each group to form the summary.

A framework based on sentence-level semantic analysis (SLSS) and symmetric non-negative matrix factorization (SNMF) is introduced. SLSS can better capture the relationships between sentences in a semantic manner, it is used to construct the sentence similarity matrix. Based on the similarity matrix, the SNMF algorithm is used cluster the sentences. Finally the most informative sentences are selected in each cluster considering both internal and external information.

Given a set of documents, these documents are cleaned by removing formatting characters. In the similarity matrix construction phase] the set of documents are decomposed into sentences, and each sentence is parsed into frame(s) using a semantic role parser. Pairwise sentence semantic similarity is calculated based on both the semantic role analysis and word relation discovery using WordNet. Once the pairwise sentence similarity matrix is obtained, symmetric matrix factorization is done to group these sentences into clusters in the second phase. In each cluster, the most semantically important sentences are identified using a measure combining the internal information (e.g., the computed similarity between sentences) and the external information (e.g., the given topic information).The selected sentences finally form the summary.

2.3 Bayesian Query-Focused Summarization

BAYESUM (for "Bayesian summarization") [3], is a model for sentence extraction in query-focused summarization. BAYESUM leverages the common case in which multiple documents are relevant to a single query. BAYESUM can be used on large data sets and results in a state-of-the-art summarization system.

The key requirement of BAYESUM is that multiple relevant documents are known for the query in question. This task is very similar to the standard ad-hoc IR task, with the important distinction that we are comparing query models against sentence models, rather than against document models.

The model is applicable to any problem for which multiple relevant documents are known for a query, the model is formulated in terms of relevance judgments. For a collection of D documents and Q queries, we assume we have a D × Q binary matrix r, where rdq = 1. In multidocument summarization, rdq will be 1 exactly when d is in the document set corresponding to query q; in search-engine summarization, it will be 1 exactly when d is returned by the search engine for query q.

2.3.1. *Language Modeling for IR*: BAYESUM is built on the concept of language models for information retrieval. The idea behind the language modeling techniques used in IR is to represent either queries or documents (or both) as probability distributions, and then use the standard probabilistic techniques for comparing them. These probability distributions are always "bag of words" distributions that assign a probability to words from a fixed vocabulary.

2.3.2. *Bayesian Statistical Model*: A sentence appears in a document because it is relevant to some query, because it provides background information about the document. The model assumes that each word can be assigned a discrete, exact source.

2.4 Automated Text Summarization in SUMMARIST

SUMMARIST [4] is an attempt to create a robust automated text summarization system, based on the 'equation': summarization = topic identification + interpretation +

generation. The task of a Summarizer is to produce synopsis of any document (or a set of documents) submitted to it. These synopses may range from a list of isolated keywords that indicate the major content of the document(s), through a list of independent single sentences that express the major content, all the way up to a coherent, fully planned and generated paragraph that compresses the document. The more sophisticated a synopsis, the more effort it generally takes to produce.

Producing an abstract requires stages of topic fusion and text generation not needed for extracts. The goal of SUMMARIST is to provide both extracts and abstracts for arbitrary English (or any other other-language) input text.

2.4.1. *Identification*: The input is selected and filtered to determine the most important, central, topics. A text can have many (sub)-topics, and that the topic extraction process can be parameterized to include more or fewer of them to produce longer or shorter summaries. The topic identiification is done by methods based on position, cue phrases, word frequency and discourse segmentation.

2.4.2. *Interpretation*: The second step in the summarization process is that of concept interpretation. A collection of extracted concepts are 'fused' into their one (or more) higher-level unifying concept(s). Fusing topics into one or more characterizing concepts is the most difficult step of automated text summarization. Here, too, a variety of methods can be employed. All of them associate a set of concepts (the indicators) with a characteristic generalization (the fuser or head). The challenge is to develop methods that work reliably and to construct a large enough collection of indicator-fuser sets to achieve effective topic reduction.

3) Summary Generation: The final step in the summarization process is to generate the summary, consisting of the fused concepts. The output will contain three generation modules, associated as appropriate with the various levels for various applications. It can be a simple list of the summarizing topics, extract of the noun phrases and clauses from the input text, by following links from the fuser concepts through the words that support them back into the input text, well formed, fluent, summaries, taking as input the fuser concepts and their most closely related concepts.

2.5 Opinion Summarization with Integer Linear Programming Formulation for Sentence Extraction and Ordering

Opinion summarization [5] technique takes into account both content and coherence, simultaneously. A summary is considered as a sequence of sentences and acquire the optimum sequence from multiple review documents by extracting and ordering the sentences. This is achieved with a novel Integer Linear Programming (ILP) formulation. This is a powerful mixture of the Maximum Coverage Problem and the Traveling Salesman Problem. This is widely applicable to text generation and summarization tasks. Each candidate sequence is scored according to its content and coherence. The content score can be defined by opinions and the coherence score is developed in training against the review document corpus.

This is a multidocument summarizer. A set of documents is given to the summarizer. A graph is constructed whose nodes are the sentences and there will be links between the nodes. The content score and coherence score is collectively combined to form the summary. The source document set includes set of concepts e. Each concept e is covered by one or more of the sentences in the document set.

The two parameters content score and coherence score are calculated first. Opinion is adopted as a concept. Opinion can be defined as $e = <t, a, p>$ as the tuple of target t, aspect a and its polarity $p = \{-1, 0, 1\}$. The t is the target of the sentence, a is the features of the target, p tells whether the sentence is a positive, negative or neutral sentence.

The coherence score is denoted as c. The coherence scores of sentence pairs (local coherence) is calculated and their sum is taken as the global coherence. The local coherence score $c_{i,j}$ of two sentences $x = \{s_i, s_j\}$ and their order $y = <s_i, s_j>$ is represented as

$$c_{i,j} = w \,.\phi(x, y) \qquad (4)$$

w is a parameter vector and $\phi(x, y)$ is a feature vector of the two sentences s_i and s_j . the Passive-Aggressive algorithm to find w. The Passive-Aggressive algorithm is an online learning algorithm that updates the parameter vector by taking up one example from the training examples and outputting the solution that has the highest score under the current parameter vector.

Using ILP (Integer Linear Programming) formulation the objective function can be decoded as:

$$\text{Max } \{\lambda \, \Sigma_e \, w_i e_i + (1-\lambda) \, \Sigma_a \, c_{i,j} a_{i,j} \} \qquad (5)$$

Eq.5 attempts to cover as much of the concepts included in input document set as possible according to their weights w and orders sentences according to discourse coherence c. λ is a scaling factor to balance w and c.

The summary meets the condition of maximum summary size and also avoids redundancy. The concepts introduced helps in the avoidance of same sentence, arc or the concepts twice.

3. New system

3.1 A Context-Based Word Indexing Model for Document Summarization

Existing models for document summarization mostly use the similarity between sentences in the document to extract the most salient sentences. The documents as well as the sentences are indexed using traditional term indexing measures, which do not take the context into consideration. Therefore, the sentence similarity values remain independent of the context.

A context sensitive document indexing model based on the Bernoulli model of randomness. The Bernoulli model of randomness has been used to find the probability of the cooccurrences of two terms in a large corpus. The lexical association between terms is used to give a context sensitive weight to the document terms. The resulting indexing weights are used to compute the sentence similarity matrix. This sentence similarity measure has been used with the baseline graph-based ranking models for sentence extraction.

The main motivation behind using the lexical association is the central assumption that the context in which a word appears provides useful information about its meaning. Cooccurrence measures observe the distributional patterns of a term with other terms in the vocabulary and have applications in many tasks pertaining to natural language understanding such as word classification, knowledge acquisition, word sense disambiguation, information retrieval, sentence retrieval , and word clustering. the Bernoulli model of randomness to find the probability of the cooccurrences of two terms in a corpus and use the classical semantic information theory to quantify the information contained in the cooccurrences of these two terms.

The lexical association metric is used to make a context-sensitive document indexing model. The idea is implemented using a PageRank-based algorithm to iteratively compute how informative each document term is. Sentence similarity calculated using the context sensitive indexing reflect the contextual similarity between two sentences. This will allow the two sentences to have different similarity values depending on the context. The hypothesis is that an improved sentence similarity measure would lead to improvements in the document summarization.

3.1.1. Exploring lexical association for text summarization:

The terms encountered in it can either be topical or nontopical. It is difficult to decide about the topicality of a term only on the basis of a single document, the patterns of term cooccurrence over a larger data set can be helpful. Lexical association measures use the term cooccurrence knowledge extracted from a large corpus. Nontopical terms appear randomly across all the document while topical terms appear in bursts. Therefore, when computed on a sufficiently large corpus, the lexical association value between two topical terms should be higher than the lexical association between two nontopical terms or a pair of topical and nontopical terms.

3.1.2. Bernoulli Model of Randomness

Bernoulli model of randomness is used to find the distribution between the terms in the document. The probability of each term in the document is calculated. Also the probability of terms occurring in different documents is also calculated. Stirling’s approximation is used to approximate the factorials included in the computation. Bernoulli model will finally give the self-information of the cooccurrences of term ti in Nj documents.

3.1.3. Context-Based Word Indexing

After the lexical association measure between two terms in a document, the next task is to calculate the context sensitive indexing weight of each term in a document. A graph-based iterative algorithm is used to find the context sensitive indexing weight of each term. Given a document Di, a document graph G is built. Let G = (V,E) be an undirected graph to reflect the relationships between the terms in the document Di. V denotes the set of vertices, where each vertex is a term appearing in the document. E is a matrix of dimensions |V x V|. Each edge correspond to the lexical association value between the terms corresponding to the vertices vj and vk. The lexical association between the same terms is set to 0. The context-sensitive indexing weight of each word vj in a document Di, denoted by indexWt(vj) is calculated. It can be found in a recursive way using the Page-rank-based algorithm.

3.1.4. Sentence Similarity Using the Context-Based Indexing

A context-sensitive indexing weight to each document term. The next step is to use these indexing weights to calculate the similarity between any two sentences. Given a sentence in the document, the sentence vector is built using the indexWt(). The sentence vector is calculated such that if a term appears in a sentence, it is given a weight indexWt(); otherwise, it is given a weight 0. The similarity between two sentences is computed using the dot product.

3.2 System architecture

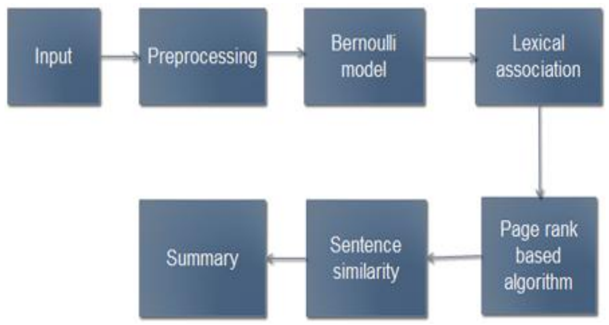

Figure 1: Overview of the architecture

The figure 1 shows the step-by-step view of the architecture. The input to the system is a document. The preprocessing techniques are done to find the different terms of the document. The Bernoulli model of randomness will distinguish the topical and nontopical terms. The lexical association between the terms obtained can be used for the construction of the graph which is based on the Page Rank based algorithm. The sentence similarity is calculated to obtain the sentences to be included in the summary.

4. Conclusion

The Bernoulli model of randomness has been used to develop a graph-based ranking algorithm for calculating how informative is each of the document terms. Using this model the association between the terms can be found. This greatly improves the summary than any of the previous works so far. The previous works did not take context into account. As a result, the summary is a set of some sentences that contain the prominent terms. So the resulting summary might not cover the entire meaning of the sentence. The context sensitive model provides much more better summary than others.

References

[1] D.R. Radev, H. Jing, M. Sty□s, and D. Tam, “Centroid-Based Summarization of Multiple Documents,” Information Processing and Management, vol. 40, pp. 919-938, http://portal.acm.org/ citation.cfm?id=1036118.1036121, Nov. 2004.

[2] D. Wang, T. Li, S. Zhu, and C. Ding, "Multi-Document Summarization via Sentence-Level Semantic Analysis and Symmetric Matrix Factorization," Proc. 31st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 307-314, http://doi.acm.org/10.1145/1390334.1390387, 2008.
[3] H. Daume´ III and D. Marcu, "Bayesian Query-Focused Summarization," Proc. 21st Int'l Conf. Computational Linguistics and the 44th Ann. meeting of the Assoc. for Computational Linguistics, pp. 305-312, http://dx.doi.org/10.3115/1220175.1220214, 2006.
[4] E. Hovy and C.-Y. Lin, "Automated Text Summarization and the Summarist System," Proc. Workshop Held at Baltimore, Maryland (TIPSTER '98), pp. 197-214, http://dx.doi.org/10.3115/1119089. 1119121, 1998.
[5] H. Nishikawa, T. Hasegawa, Y. Matsuo, and G. Kikui, "Opinion Summarization with Integer Linear Programming Formulation for Sentence Extraction and Ordering," Proc. 23rd Int'l Conf. Computational Linguistics: Posters, pp. 910-918, http://portal.acm.org/ citation.cfm?id=1944566.1944671, 2010.
[6] Pawan Goyal, Laxmidhar Behera and Thomas Martin McGinnity, "A Context-Based Word Indexing Model for Document Summarization"
Transactions on knowledge and data engineering, pp. 1693 – 1705, http://ieeexplore.ieee.org/

Author Profile

Divya Vidyadharan is a student pursuing MTech in Computer Science & Engineering at KMCT College of Engineering, Calicut university, Kerala. She has done BTech. in computer science & engineering from Calicut university, Kerala in 2012.

Anju CR is Assistant Professor in Computer Science & Engineering at KMCT College of Engineering, Calicut university, Kerala. She has done MTech. in computer science & engineering from KMCT College of Engineering, Calicut university, Kerala in 2014. She has done BTech. in computer science & engineering from SNS college of Technology, Anna university, Tamil Nadu in 2007.