Sentence Extraction Based Single Document Summarization

Jagadeesh J, Prasad Pingali, Vasudeva Varma
Language Technologies Research Centre
International Institute of Information Technology
Hyderabad, India
{ j_jagdeesh@students.iiit.net,pvvpr@iiit.net,vv@iiit.net}

Abstract

The need for text summarization is crucial as we enter the era of information overload. In this paper we present an automatic summarization system, which generates a summary for a given input document. Our system is based on identification and extraction of important sentences in the input document. We listed a set of features that we collect as part of summary generation process. These features were stored using vector representation model. We defined a ranking function which ranks each sentence as a linear combination of the sentence features. We also discussed about discourse coherence in summaries and techniques to achieve coherent and readable summaries. The experiments showed that the summary generated is coherent the selected features are really helpful in extracting the important information in the document.

1 Introduction

A huge amount of on-line information is available on the web, and is still growing. While search engines were developed to deal with this huge volume of documents, even they output a large number of documents for a given user's query. Under these circumstances it became very difficult for the user to find the document he actually needs, because most of the naive users are reluctant to make the cumbersome effort of going through each of the documents. Therefore systems that can automatically summarize one or more documents are becoming increasingly desirable.

A summary can be loosely defined as a short version of text that is produced from one or more texts. Automatic summarization is to use automatic mechanism to produce a finer version for a given documents. Spark-Jones (1999) discussed several ways to classify summaries. The following three factors are considered to be important for text summarization.

- Input factors : text length, genre, number of documents
- Purpose factors: audience, purpose of summarization.
- Output factors: running text or headed text etc.

Summaries can be classified into different types based on dimensions, genre, and context.

- Dimensions
  Single vs. Multi-document summarization
- Genre
  Headlines, outlines, minutes, chronologies, etc.
- Context
  Generic, Query specific summaries

As pointed out in Mani and Maybury [1999] summaries can be classified in to extracts (most relevant sentences are selected from the text), and abstracts (text is analyzed, a conceptual representation is provided which in turn is used to generate sentences that form summary). Conventional text summarization systems produce summaries by using sentences or paragraphs as basic unit, giving them degree of importance (Edmundson [1969]), sorting them based on the importance, and gathering the important sentences. In this paper we have presented an extract type summary generation system.

2 Background

Most of the summarization work done till date is based on extraction of sentences from the original document. The sentence extraction techniques compute score for each sentence based on features such as position of sentence in the document[Baxendale 1958; Edmundson 1969], word or phrase frequency[Luhn 1958], key phrases (terms which indicate the importance of the sentence towards summary e.g. “this article talks about”)[Edmundson 1969]. There were some attempts to use machine learning (to identify important features), use natural language processing (to identify key passages or to use relationship between words rather than bag of words). The application of machine learning to summarization was pioneered by Kupiec, Pedersen, and Chen [1995], who developed a summarizer for scientific articles using a Bayesian classifier.

For the generation of a coherent and readable summary, one has to do significant amount of text analysis to generating good feature vector, handling discourse connectors , and refining the sentences. This system is an attempt in that direction.

3 System Description

The architecture of our summarization system is shown in Fig 1. The system has both text analysis component and summary generation component. The text analysis component is based on syntactic analysis, followed by a component which identifies the features associated with each sentence. Text normalization is applied before syntactic analysis of the text which include extracting the text from the document (format conversion, if needed), removing floating objects like figures, tables, identification of titles and subtitles, and dividing the text into sentences. After text normalization the normalized text is a passed through a feature extraction module. Feature extraction include extracting features associated with the sentence (such as sentence number, number of words in that sentence and so on) and the features associated with words (such as the named entities, the term frequency and so on). The summary generation component calculates the score for each sentence based on the features that were identified

by the feature extraction module. Sentence refinement is done on the sentences with high score, and the resulting sentences are selected for the summary in the same order as they were found in the input text document.

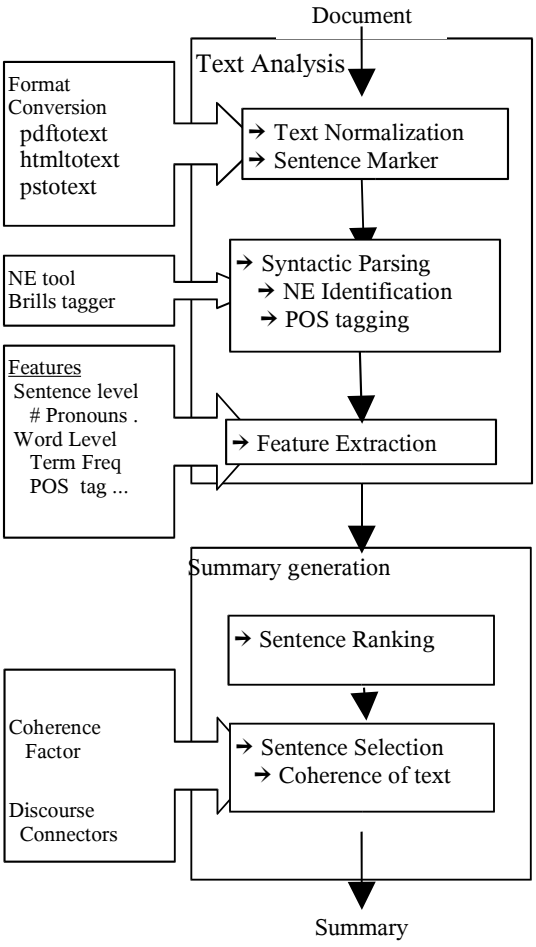

Fig 1: Architecture of the system

4 Text Analysis

As a part of summarization, we try to identify the important sentences which represent the document. This involves considerable amount of text analysis. We assume that the input document can be of any document format (ex. Pdf, html ...), hence the system first applies document converters to extract the text from the input document. In our system we have used document converters that could convert PDF, MS Word, post-script and HTML documents into text.

4.1.1 Text Normalization

The text normalization is a rule based component which removes the unimportant objects like figures, tables, identifies the headings and subheadings and handling of non-standard words like web URL's and emails and so on. The text is then divided into sentences for further processing.

4.1.2 Sentence Marker

This module divides the document into sentences. At first glance, it may appear that using end-of-sentence punctuation marks, such as periods, question marks, and exclamation points, is sufficient for marking the sentence boundaries. Exclamation point and question mark are somewhat less ambiguous. However, dot '(.') in real text could be highly ambiguous and need not mean a sentence boundary always. The sentence marker considers the following ambiguities in marking the boundary of sentences.

1. Non standard word like web urls, emails, acronyms, and so on, will contain '.'
2. Every sentence starts with an uppercase letter
3. Document titles and subtitles can be written either in upper case or title case

4.2 Syntactic Parsing

This module analyzes the sentence structure with the help of available NLP tools such as Brills tagger [Brill], named entity extractor, etc. .

A named entity extractor can identify named entities (persons, locations and organizations), temporal expressions (dates and times) and certain types of numerical expressions from text. This named entity extractor uses both syntactic and contextual information. The context information is identified in the form of POS tags of the words and used in the named entity rules, some of these rules are general and while the rest are domain specific.

4.3 Feature Extraction

The system extracts both the sentence level and the word level features which are used in calculating the importance or relevance of the sentence towards the document. The sentence level features include

1. Position of the sentence in input document
   The sentence number is normalized to the scale of 0 to 1. The weights corresponding to the sentence position [Yohei Seki], is shown in Figure 2.

2. Presence of the verb in the sentence
   Based on the assumption that a complete sentence contains verb, this feature will help in deciding the candidate sentence to generate the summary.

3. Referring pronouns
   The score for a sentence is attributed by the words that are present in the sentence. During this process most of the IR/IE systems neglect the stop words that occur in the document. The referring pronouns are also neglected as stop words. But to get the actual sentence score one should also consider the proper nouns to which these pronouns are referring to.

4. Length of the sentence
   Since long sentences contain more number of words, they usually get more score. This factor needs to be considered while calculating the score of the sentence. In our system we normalize

Sentence position	*0 < x <= 0.1*	*0.1< x<= 0.2*	*0.2<x<=0.3*	*0.3<x<=0.4*	*0.4<x<=0.5*
Distributed Probability	0.17	0.23	0.14	0.08	0.05
Sentence position	*0.5 < x <= 0.6*	*0.6< x<= 0.7*	*0.7<x<=0.8*	*0.8<x<=0.9*	*0.9<x<=1.0*
Distributed Probability	0.04	0.06	0.04	0.04	0.15

Fig:2 Distributed probability of Important Sentence

the sentence score by the number of words in that sentence, which is the score of the sentence per word.

The word level features include

1. Term frequency tf(w)
   Term frequency is calculated using both the unigram and bigram frequency. We considered only nouns while computing the bigram frequencies. A sequence of two nouns occurring together denotes a bigram. The unigram/bigram frequency denotes the number of times the unigram/bigram occurred in the document. Typically the bigrams occur less number of times than the unigrams, so we used a factor that convert the bigram frequency to unigram frequency as a word level feature. All the bigrams in which the word occurs are taken, and normalized to unigram scale. Finally the maximum of the unigram and normalized bigram frequency is taken as the term frequency of the word.

2. Length of the word l(w)
   Smaller words occur more frequently than the larger words, In order to negate this effect we considered the word length as a feature.

3. Parts of speech tag p(w)
   We used Brills tagger[Brill] to find the POS tag of the word. We ranked the tags and assigned weights, based on the information that they contribute to the sentence.

4. Familiarity of the word f(w)
   Familiarity, derived from the WordNet[Miller], denotes how general the word is across all the documents. This also indicates the ambiguity of the word. Words with less familiarity were given more weightage. We have used a sigmoid function in calculating the weightage of the word. Weightage of the word is given by
   $$\frac{1}{1+e^{-8(\frac{1}{fam}-0.5)}}$$

5. Named entity tag NE(w)
   We ranked the NE tags depending on the frequency of their occurrence and the type of tag. The weights are assigned to the words based on the NE tag.

6. Occurrence in headings or subheadings O(w)
   The words which occur in the headings and subheadings are treated as important and are given more weightage over other words.

7. Font style F(w)
   Finally the font in which the word is written is also stored as a feature. Currently we are storing whether the word is written in upper case, title case or lower case. The preference for the words is given in the same order.

All the above features are normalized on a 0-1 scale. A weighted combination of all these features is used in calculating the score of sentence

5 Summary Generation

Summary generation include tasks such as calculating the score for each sentence, selecting the sentences with high score, and refinement of the selected pool of sentences.

5.1 Sentence Ranking

Some of the word features are dependent on the context in which it occurs, i.e they depend on the sentence number also(ex. POS tag, familiarity, ..). So the score of the word is also dependent on the sentence number. Once the feature vector for each sentence is extracted, the score of a sentence is the sum of score of individual words influenced by sentence level features.

$$Score(l,w)=\prod_i f_i(w)$$

$$Score(l)=\sum_i Score(l,w_i)$$

where l, denotes the sentence number and w denotes the word that occurred in the sentence, and $f_i(w)$ denotes the value of i^{th} feature value.

In order to compute effect of referring pronouns on sentence score, we assumed that pronouns in a given sentence are referring to nouns in immediate preceding sentence. We made this assumption only if the pronoun occurred in the first half of the sentence, otherwise it is assumed that the pronoun is referring to noun within the sentence. Based on the above assumption the actual score of the sentence (if those nouns were existing in the same sentence) can be calculated as

$$Score(l) \leftarrow Score(l) + (No.\ of\ coreferrents \times SPW(l-1))$$

$$SPW(l) \leftarrow \frac{Score(l)}{length(l)}$$

Where $SPW(l)$ denote the score per word, of the sentence l. The SPW is multiplied by the positional value of the sentence, to get the final score of the sentence.

5.2 Sentence Selection

After the sentences are scored, we need to select the sentences that make good summary. One strategy is to pick the top N sentence towards the summary, but this creates the problem of coherence. The selection of sentence is dependent upon the type of the summary requested.

The process of selecting the sentences for final summary can be viewed as a Markov process. This is to say, the selection of the next sentence for summary is dependent on already selected sentences for summary. This approach is important to get a meaningful and coherent summary.

5.2.1 Coherence Score CS

This is a measure of the amount of information that is common to the set of sentences that are already selected and the new sentence that is going to be selected. We have used a bag of words technique to calculate the coherence of the information flow CS.

Let s_w represent the set of words present in the sentencess that are already selected, and l_w be the set of word in the new sentence, then coherence score is the sum of the scores of the words that are in common to both s_w and l_w Now the score of the new sentence is given by

$$CF \times CS(l) + (1 - CF) \times SPW(l)$$

where CF is the Coherence Factor, which is user defined parameter.

5.3 Summary Refinement

Sentence selection module will give a set of sentences which satisfies the user criteria. Further to increase readability of the summary the following transformations were used, in the specified order:

- Add sentences to the pool so as to avoid dangling discourse relations. We have a list of discourse connectors that are commonly used to connect different sentences in a document. For example if a sentence starts with “afterwards” or “but”, then this sentence is marked as dependent on the previous sentence at the discourse level. At this stage the system can optionally
  - mark preceding sentence as important and as well add to the pool of selected sentences.

    or
  - remove this sentence from the selected list.

- Some sentences are removed depending on the length of the desired summary. If a short length summary is requested, then it is good to select many short sentences and remove very long sentences. If the length of summary is comparable with the length of the document then sentences which are less than some threshold are removed from the pool.

- Remove questions, title and subtitles from the set of sentences.

- Rewrite sentences by deleting marked parenthetical units.
- If a coreferent is found in the given sentence, then the previous sentence is also included in the set of selected sentences.

In the final step, we order the sentences based on their occurrence in the document and generate the summary by concatenating the ordered sentences.

6 Evaluation

The system was evaluated on news articles, with 20% condensed rates. The system is evaluated with human ranking from the best (5 points) to the worst(0 points). The results are shown as follows.

Article ID	Score	Coherence
20010101	3	4
20010102	4.5	4.5
20010103	3.5	4
20010104	3	3.8
20010105	2.5	3.5
20010106	4	4.2
20010107	1.5	3.5
20010108	4	4.5
Avg	3.25	4

In summaries the article:20010107 got very low score because, these documents are from finance domain while rest of the documents are from the cricket domain. The Named entity tool is more customized to cricket domain and identified most of the named entities. The article ID:20010108 belongs to mobile phones category, but got high score because this document contained large number of country names, which are identified by the NE tool. This shows us that the system is dependent on the performance of the NE system.

As expected the summaries generated are very coherent. Even though we considered some of the ambiguities in marking the sentence boundaries, we still are not able to get good sentence marker. This sometimes caused the inconsistency in the summary generated.

Because of the sentence refinement techniques the number of sentence with missing antecedent came down drastically, but currently we are neglecting multi-word referents (ex. the middle-order batsman), which was found to be a place for improvement.

7 Conclusion and Future Work

In this paper we presented a sentence extraction based single document summarization system. We used shallow text processing approaches as opposed to semantic approaches related to natural language processing. We presented a detailed architecture and internal working of our system while discussing some of the challenges we came across in generating readable and coherent summaries. While the evaluation that we

have presented here is subjective to the user, we would like to evaluate our system in the environments like DUC, where the evaluation is done using automated systems like ROUGE.

In our system we have come up with arbitrary weights by trial and error method. We plan to implement machine learning techniques to learn these weights automatically from training data. We would like to use more NLP tools such as word sense disambiguation and co-reference resolution modules to obtain precise weights for the sentences in the document. We also plan to extend this system to perform deeper semantic analyses of the text and add more features to our ranking function. We would like to extend this system to be able to generate multi-document summaries.

References

1. Baxendale, P. B. 1958. Man-made index for technical literature--An experiment. *IBM Journal of Research and Development*, 2(4):354-361
2. Eric Brill. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. Computational Linguistics, 21(4):543-565, 1995
3. D. Macru, Discourse-Based Summarization in DUC-2001, in *Document Understanding Conference*, 2001
4. E. D' Avanzo, B. Magnini, A. Vallin, ITC-irst. Keyphrase Extraction for Summarization purposes: The LAKE system at DUC-2004, *Document Understanding Conference.*
5. Edmundson, H. P. 1969. New methods in automatic extracting. *Journal of the Association for Computing Machinery*, 16(2):264-285.
6. Kupiec, Julian, Jan O. Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *Research and Development in Information Retrieval*, pages 6873.
7. Luhn, H. P. 1958. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159-165.
8. Mani, Inderjeet and Mark Maybury, editors. 1999. Advances in Automatic Text summarization. MIT Press, Cambridge. Marcu, Daniel. 1997a. From discourse structures to text summaries. In *Proceedings of the ACL '97/EACL '97 Workshop on Intelligent Scalable Text Summarization*, Madri, July 11, pages 82-88
9. **Miller** G. Nouns in WordNet: A Lexical Inheritance System. Five Papers on WordNet Princeton University
10. Spark Jones, Karen. 1999. Automatic summarizing: Factors and directions. In I. Mani and M.T. Maybury, editors, *Advances in Automatic Text Summarization*. MIT Press, Cambridge, pages 1-13.
11. Yohei Seki, *Sentence Extraction by tf/idf and position weighting from Newspaper Articles,* Proceedings of the third NTCIR workshop.