A New Approach to Automatic Document Summarization

Xiaofeng Wu
National Laboratory of Pattern Recognition,
Institute of Automation,
Chinese Academy of Science
Beijing, China
xfwu@nlpr.ia.ac.cn

Chengqing Zong
National Laboratory of Pattern Recognition,
Institute of Automation,
Chinese Academy of Science
Beijing, China
cqzong@nlpr.ia.ac.cn

Abstract

In this paper we propose a new approach based on *Sequence Segmentation Models* (SSM) to the extractive document summarization, in which summarizing is regarded as a segment labeling problem. Comparing with the previous work, the difference of our approach is that the employed features are obtained not only from the sentence level, but also from the segment level. In our approach, the semi-Markov CRF model is employed for segment labeling. The preliminary experiments have shown that the approach does outperform all other traditional supervised and unsupervised approaches to document summarization.

1 Introduction

Document summarization has been a rapidly evolving subfield of Information Retrieval (IR) since (Luhn, 1958). A summary can be loosely defined as a text that is produced from one or more texts and conveys important information of the original text(s). Usually it is no longer than half of the original text(s) or, significantly less (Radev et al., 2002). Recently, many evaluation competitions (like the Document Understanding Conference DUC "http://duc.nist.gov", in the style of NIST's TREC), provided some sets of training corpus. It is obvious that, in the age of information explosion, document summarization will be greatly helpful to the internet users; besides, the techniques it uses can also find their applications in speech techniques and multimedia document retrieval, etc.

The approach to summarizing can be categorized in many ways. Some of them are: 1) *indicative*, *informative* and *evaluative*, according to functionality; 2) *single-document* and *multi-document*, according to the amount of input documents; 3) *generic* and *query-oriented*, according to applications. Yet the taxonomy currently widely employed is to categorize summarization into *abstractive* and *extractive*.

According to (Radev et al., 2002), all methods that are not explicitly extractive are categorized as abstractive. These approaches include ontological information, information fusion, and compression. Abstract-based summarization never goes beyond conceptual stage, though ever since the dawn of summarization it has been argued as an alternative for its extract-based counterpart. On the other hand, extractive summarization is still attracting a lot of researchers (Yeh et al., 2005) (Daum´e III and Marcu, 2006) and many practical systems, say, MEAD "http://www.summarization.com/mead/", have been produced. Using supervised or unsupervised machine learning algorithms to extract sentences is currently the mainstream of the extractive summarization. However, all pervious methods focus on obtaining features from the sentence granularity.

In this paper we focus on generating summarization by using a supervised extractive approach in which the features are obtained from a larger granularity, namely segment. The remainder of the paper is organized as follows: Section 2 introduces the related work concerning the extract-based summarization. Section 3 describes our motivations. Our experiments and results are given in Section 4, and Section 5 draws the conclusion and mentions the future work.

2 Related Work

Early researchers approached the summarization problem by scoring each sentence with a combination of the features like word frequency and distribution, some proper names (Luhn, 1958), sentence positions in a paragraph (Baxendale, 1958), and sentence similarity (Gong, 2001) etc. The results were comparatively good. Most supervised extractive methods nowadays focus on finding powerful machine learning algorithms that can properly combine these features.

Bayesian classifier was first applied to summarization by (Pedersen and Chen, 1995), the authors claimed that the corpus-trained feature weights were in agreement with (Edmundson, 1969), which employed a subjective combination of weighted features. Another usage of the naïve Bayesian model in summarization can be found in (Aone et al., 1997). Bayesian model treats each sentence individually, and misses the intrinsic connection between the sentences. (Yeh et al., 2005) employed genetic algorithm to calculate the belief or score of each sentence belonging to the summary, but it also bears this shortcoming.

To overcome this independence defect, (Conroy and O'leary, 2001) pioneered in deeming this problem as a *sequence labeling problem*. The authors used HMM, which has fewer independent assumptions. However, HMM can not handle the rich linguistic features among the sentences either. Recently, as CRF (Lafferty and McCallum, 2001) has been proved to be successful in part-of-speech tagging and other sequence labeling problems, (Shen et al., 2007) attempted to employ this model in document summarization. CRF can leverage all those features despite their dependencies, and absorb other summary system's outcome. By introducing proper features and making a comparison with SVM, HMM, etc., (Shen et al., 2007) claimed that CRF could achieve the best performance.

All these approaches above share the same viewpoint that features should be obtained at sentence level. Nevertheless, it can be easily seen that the non-summary or summary sentences tend to appear in a consecutive manner, namely, in segments. These rich features of segments can surely not be managed by those traditional methods.

Recently, Sequence Segmentation Model (SSM) has attracted more and more attention in some traditional sequence learning tasks. SSM builds a direct path to encapsulate the rich segmental features (e.g., entity length and the similarity with other entities, etc., in entity recognition). Semi-CRF (Sarawagi and Cohen, 2004) is one of the SSMs, and generally outperforms CRF.

3 Motivations

According to the analysis in Section 2, our basic idea is clear that we regard the supervised summarizing as a *problem of sequence segmentation*. However, in our approach, the features are not only obtained on the sentence level but also on the segment level.

Here a segment means one or more sentences sharing the same label (namely, non-summary or summary), and a text is regarded as a sequence of segments. Semi-CRF is a qualified model to accomplish the task of segment labeling, besides it shares all the virtues of CRF. Using semi-CRF, we can easily leverage the features both in traditional sentence level and in the segment level. Some features, like Log Likelihood or Similarity, if obtained from each sentence, are inclined to give unexpected results due to the small granularity. Furthermore, semi-CRF is a generalized version of CRF. The features designed for CRF can be used in semi-CRF directly, and it has been proved that semi-CRF outperforms CRF in some Natural Language Processing (NLP) problems (Sarawagi and Cohen, 2004).

In the subsections below, we first introduce semi-CRF then describe the features we used in our approach.

3.1 Semi-CRF

CRF was first introduced in (Lafferty and McCallum, 2001). It is a conditional model $P(Y|X)$, and here both X and Y may have complex structure. The most prominent merits of CRF are that it offers relaxation of the strong independence assumptions made in HMM or Maximum Entropy Markov Models (MEMM) (McCallum, 2000) and it is no victim of the label bias problem. Semi-CRF is a generalization version of sequential CRF. It extends CRF by allowing each state to persist for a non-unit length of time. After this time has elapsed, the system might transmit to a new state, which only depends on its previous one. When the system is in the "segment of time", it is allowed to behave non-Markovianly.

3.1.1 CRF vs. semi-CRF

Given an observed sentence sequence $X=(x_1,x_2,\ldots,x_M)$. The corresponding output labels are $Y=(y_1,y_2,\ldots,y_M)$, where y_i gets its value from a fixed set Ψ. For document summarization, $\Psi=\{0,1\}$. Here 1 for summary and 0 for non-summary. The goal of CRF is to find a sequence of Y, that maximize the probability:

$$P(Y \mid X,W) = \frac{1}{Z(X)} \exp(W \cdot F(X,Y)) \quad (1)$$

Here, $F(X,Y) = \sum_{i=1}^{M} \mathrm{f}(i,X,Y)$ is a vertical vector of size T. The vertical vector $\mathrm{f} = (f_1, f_2, ..., f_T)'$ means there are T feature functions, and each of them can be written as $f_t(i,X,Y) \in R, t \in (1,\cdots,T), i \in (1,\cdots,M)$. For example, in our experiment the 10th feature function is expressed as: [if the length of current sentence is bigger than the predefined threshold value]&[if the current sentence is a summary]. When this feature function is acting upon the third sentence in *text*_1 with *label_sequence*_1, the following feature equation f_{10}(3,*text*_1, *label_sequence*_1) means: in *text*_1 with *label_sequence*_1, [if the length of the third sentence is bigger than the predefined threshold value]&[if the third sentence is a summary]. W is a horizontal vector of size T that represents the weights of these features respectively. Equation (2) gives the definition of $Z(X)$, which is a normalization constant that makes the probabilities of all state sequences sum to 1.

$$Z(X) = \sum_{Y'} \exp(W \cdot F(X,Y')) \quad (2)$$

If we change the sequence vector X to $S=<s_1,s_2,\ldots,s_N>$, which means *one way to* split X into N segments, we have the semi-CRF. Each element in S is a triple: $S_j=<t_j,u_j,y_j>$, which denotes the j^{th} segment in this way of segmentation. In the triple, t_j denotes the start point of the j^{th} segment, u_j denotes its end position, and y_j is the output label of the segment (recall the example at the beginning of this subsection that there is only one output for a segment). Under this definition, segments should have no overlapping, and satisfy the following conditions:

$$\sum_{j=1}^{N} |s_j| = |X| \quad (3)$$

$$t_1 = 1, u_N = |X|, 1 \le t_j \le u_j \le |X|, t_{j+1} = u_j + 1 \quad (4)$$

Here, $|\bullet|$ denotes the length of $\bullet$.

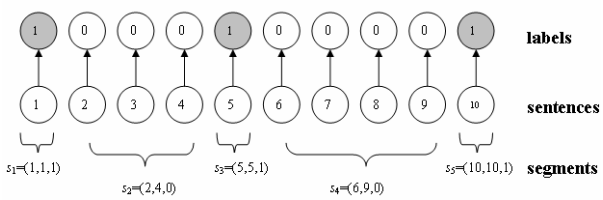

Figure 1 A 10-sentences text with label sequence

For example, one way to segment a text of 10 sentences in Figure 1 is $S=<(1,1,1),(2,4,0),(5,5,1),(6,9,0),(10,10,1)>$. The circles in the second row represent sentences, and actually are only some *properties* of the corresponding sentences.

Consequently, the feature function f in CRF converts to the segmental feature function $\mathrm{g}=(g_1,g_2,\ldots,g_{T'})$. Like f, $g_t(i,x,s) \in R$ also maps a triple (i,x,s) to a real number. Similarly, we may define $G(X,S) = \sum_{i=1}^{N} \mathrm{g}(i,X,S)$. Now we give the final equation used to estimate the probability of S. Given a sequence X and feature weight W, we have

$$P(S \mid X,W) = \frac{1}{Z(X)} \exp(W \cdot G(X,S)) \quad (5)$$

Here,

$$Z(X) = \sum_{S' \in \Delta} \exp(W \cdot G(X,S')) \quad (6)$$

Where, $\Delta = \{all - segmentations - allowed\}$.

3.1.2 Inference

The inference or the testing problem of semi-CRF is to find the best S that maximizes Equation (5). We use the following Viterbi-like algorithm to calculate the optimum path.

Suppose the longest segment in corpus is K, let $S_{1:i,y}$ represent all possible segmentations starting from 1 to i, and the output of the last segment is y. $V(i,y)$ denotes the biggest value of $P(S'|X,W)$. Note that it's also the largest value of $W \cdot G(X,S')$, $S' \in S_{1:i,y}$.

Compared with the traditional Viterbi algorithm used in CRF, the inference for semi-CRF is more time-consuming. But by studying Algorithm 1, we can easily find out that the cost is only linear in K.

Algorithm 1:
Step1. Initialization:
Let $V(i, y) = 0$, *for* $i = 0$
Step2. Induction:
for $i > 0$

$$V(i, y) = \max_{y', k=1,\ldots,K} V(i-k, y') + W \cdot \mathrm{g}(y, y', x, i-d+1, i) \quad (7)$$

Step3. Termination and path readout:
$bestSegment = \max_y V(|X|, y)$

3.1.3 Parameter estimation

Define the following function

$$L_W = \sum_l \log P(S_l \mid X_l, W) = \sum_l (W \cdot G(X_l, S_l) - \log Z(X_l)) \quad (8)$$

In this approach, the problem of parameter estimation is to find the best weight W that maximizes L_W. According to (Bishop, 2006), the Equation (8) is convex. So it can be optimized by gradient ascent. Various methods can be used to do this work (Pietra et al. 1997). In our system, we use L-BFGS, a quasi-Newton algorithm (Liu and Nocedal. 1989), because it has the fast converging speed and efficient memory usage. APIs we used for estimation and inference can be found in website "http:\\crf.sourcefourge.net".

3.2 Features

(Shen et al. 2007) has made a thorough investigation of the performances of CRF, HMM, and SVM. So, in order to simplify our work and make it comparable to the previous work, we shape our designation of features mainly under their framework.

The mid column in Table 1 lists all of the features we used in our semi-CRF approach. For the convenience of comparison, we also list the name of the features used in (Shen et al. 2007) in the right column, and name them *Regular Features*. The features in bold-face in the mid column are the corresponding features tuned to fit for the usage of semi-CRF. We name them *Extended Features*. There are some features that are not in bold-face in the mid column. These features are the same as the *Regular Features* in the right column. We also used them in our approach. The mark star denotes that there is no counterpart. We number these features in the left column.

No.	semi-CRF	CRF
1	**Ex_Position**	Position
2	**Ex_Length**	Length
3	**Ex_Log_Likelihood**	Log Likelihood
4	**Ex_Similarity_to_ Neighboring_ Segments**	Similarity to Neighboring Sentences
5	**Ex_Segment_ Length**	*
6	*Thematic*	Thematic
7	*Indicator*	Indicator
8	*Upper Case*	Upper Case

Table 1. *Features List*

The details of the features we used in semi-CRF are explained as follow.

Extended Features:

Ex_Position: is an extended version of the Position feature. It gives the description of the position of a segment in the current segmentation. If the sentences in the current segment contain the beginning sentence of a paragraph, the value of this feature will be 1, 2 if it contains the end of a paragraph; and 3 otherwise;

Ex_Length: the number of words in the current segment after removing some stop-words.

Ex_Log_Likelihood: the log likelihood of the current segment being generated by the document. We use Equation (9) below to calculate this feature. $N(w_j,s_i)$ denotes the number of occurrences of the word w_j in the segment s_i, and we use $N(w_j, D)/\sum_{w_k} N(w_k, D)$ to estimate the probability of a word being generated by a document.

$$\log P(s_i \mid D) = \sum_{w_j} N(w_j, s_i) \log p(w_j \mid D) \quad (9)$$

Ex_Similarity_to_Neighboring_Segments: we define the cosine similarity based on the TF*IDF (Frakes &Baeza-Yates, 1992) between a segment and its neighbors. But unlike (Shen et al. 2007), in our work only the adjacent neighbors of the segment in our work are considered.

EX_Segment_Length: this feature describes the number of sentences contained in a segment.

All these features above are actually an extended version used in the regular CRF (or in other supervised model). It is easy to see that, if the segment length is equal to 1, then the features will degrade to their normal forms.

There are some features that are also used in semi-CRF but we don't extend them like those features above. Because the extended version of these features leads to no improvement of our result. These features are:

Regular Features we used:

Thematic: with removing of stop words, we define the words with the highest frequency in the document to be the thematic words. And this feature gives the count of these words in each sentence.

Indicator: indicative words such as "conclusion" and "briefly speaking" are very likely to be included in summary sentences, so we define this feature to signal if there are such words in a sentence.

Upper Case: some words with upper case are of high probability to be a name, and sentences with such words together with other words which the author might want to emphasize are likely to be appeared in a summary sentence. So we use this feature to indicate whether there are such words in a sentence.

It should be noted that theoretically the number of extended features obtained from the corpus goes linearly with K in Equation (7).

4 Experiments

4.1 Corpus & Evaluation Criteria

To evaluate our approach, we applied the widely used test corpus of (DUC2001), which is sponsored by ARDA and run by NIST "http://www.nist.gov". The basic aim of DUC 2001 is to further progress of summarization and enable researchers to participate into large-scale experiments. The corpus DUC2001 we used contains 147 news texts, each of which has been labeled manually whether a sentence belongs to a summary or not. Because in (Shen et al. 2007) all the experiments were conducted upon DUC2001, we may make a comparison between the *sequence labeling models* and the *sequence segmentation mode*s we used. The only preprocessing we did is to remove some stop words according to a stop word list.

We use $F1$ score as the evaluation criteria which is defined as:

$$F1 = \frac{2*\text{Precesion}*\text{Recall}}{\text{Precesion}+\text{Recall}} \qquad (10)$$

We used 10-fold cross validation in order to reduce the uncertainty of the model we trained. The final $F1$ score reported is the average of all these 10 experiments.

All those steps above are strictly identical to the work in (Shen et al. 2007), and its result is taken as our baseline.

4.2 Results & Analysis

As we mentioned in Sub-Section 3.2, those extended version of features only work when segment length is bigger than one. So, each of these extended version of features or their combination can be used together with all the other regular features listed in the right column in Table 1. In order to give a complete test of the capacity of all these extended features and their combinations, we do the experiments according to the power set of {1, 2, 3, 4, 5} (the numbers are the IDs of these extended features as listed in Table 1), that is we need to do the test 2^5-1 times with different combinations of the extended features. The results are given in Table 2. The rows with italic fonts (1, 3, 5, 7, 9, 11, 13), in Table 2 denote the extended features used. For example, '*1+2*' means that the features Ex_Positon and the Ex_Length are **together used with** all other regular features are used.

	1	*2*	*3*	*4*	*5*
F1	0.395	0.391	0.398	0.394	0.392
	1+2	*1+3*	*1+4*	*1+5*	*2+3*
F1	0.395	0.396	0.396	0.395	0.382
	2+4	*2+5*	*3+4*	*3+5*	*4+5*
F1	0.389	0.384	0.398	0.399	0.380
	1+2+3	*1+2+4*	*1+2+5*	*1+3+4*	*1+3+5*
F1	0.398	0.397	0.393	0.403	0.402
	1+4+5	*2+3+4*	*2+3+5*	*2+4+5*	*3+4+5*
F1	0.402	0.403	0.401	0.403	0.404
	1+2 +3+4	*1+2 +3+5*	*1+2 +4+5*	*1+3 +4+5*	*2+3 +4+5*
F1	**0.407**	0.404	0.406	0.402	0.404
	All	*CRF*			
F1	0.406	0.389			

Table 2. *Experiment results.*

Other rows (2, 4, 6, 8, 10, 12, 14) give *F*1 scores corresponding to the features used.

In Table 3 we compare our approach with some of the most popular unsupervised methods, including LSA (Frakes & Baeza-Yates, 1992) and HITS (Mihalcea 2005). The experiments were conducted by (Shen et al. 2007).

	LSA	*HITS*	*Seim-CRF*
F1	0.324	0.368	0.407

Table 3 *Comparison with unsupervised methods*

From the results in Table 2 we can see that individually applying these extended features can improve the performance somewhat. The best one of these extended features is feature 3, as listed in the 2nd row, the 5th column. The highest improvement, 1.8%, is obtained by combining the features 1, 2, 3 and 4. Although a few of the combinations hurt the performance, most of them are helpful. This verifies our hypothesis that the extended features under SSM have greater power than the regular features. The results in Table 3 demonstrate that our approach significantly outperforms the traditional unsupervised methods. 8.3% and 4.9% improvements are respectively gained comparing to LSA and HITS models

Currently, the main problem of our method is that the searching space goes large by using the extended features and semi-CRF, so the training procedure is time-consuming. However, it is not so unbearable, as it has been proved in (Sarawagi and Cohen, 2004).

5 Conclusion and Future work

In this paper, we exploit the capacity of semi-CRF , we also make a test of most of the common features and their extended version designed for document summarization. We have compared our approach with that of the regular CRF and some of the traditional unsupervised methods. The comparison proves that, because summary sentences and non-summary sentences are very likely to show in a consecutive manner, it is more nature to obtain features from a lager granularity than sentence.

In our future work, we will test this approach on some other well known corpus, try the complex features used in (Shen et al. 2007), and reduce the time for training.

Acknowledgement

The research work described in this paper has been funded by the Natural Science Foundation of China under Grant No. 60375018 and 60121302.

References

C.Aone, N. Charocopos, J. Gorlinsky. 1997. An Intelligent Multilingual Information Browsing and Retrieval System Using Information Extraction. In *ANLP*, 332-339.

P.B. Baxendale. 1958. Man-made Index for Technical Literature -An Experiment. *IBM Journal of Research and Development*, 2(4):354-361.

C.M. Bishop. 2006. Linear Models for Classification, *Pattern Recognition and Machine Learning*, *chapter 4*, Springer.

J. M. Conroy and D. P. O'leary. 2001. Text Summarization via Hidden Markov Models. In *SIGIR*, 406-407.

Hal Daum´e III, and D. Marcu. 2006. Bayesian Query- Focused Summarization, In *ACL*

H. P. Edmundson. 1969. New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery*, 16(2):264-285.

W. B. Frakes, R. Baeza-Yates, 1992, *Information Retrieval Data Structures & Algorithms.* Prentice Hall PTR, New Jersey

Y. H. Gong and X. Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR,* 19-25

J. Kupiec, J. Pedersen, and F. Chen. 1995. A Trainable Document Summarizer. *Research and Development in Information Retrieval,* 68-73

J. D. Lafferty, A. McCallum and F. C. N. Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. *ICML*, 282-289.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large-scale optimization. *Mathematic Programming*, 45:503-528.

H. P. Luhn. 1958. The Automatic Creation of Literature Abstracts. *IBM Journal of Research and*

Development, 2(2): 159 -165.

A. McCallum, D. Freitag, and F. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *ICML*, 591-598

Mihalcea R. Mihalcea. 2005. Language independent extractive summarization. In *AAAI*, 1688-1689

S. D. Pietra, V. D. Pietra, and J. D. Lafferty. 1997. Inducing features of random fields. IEEE Tran. *on Pattern Analysis and Machine Intelligence*, 19(:)380–393.

D. R. Radev, E. Hovy and K. McKeown. 2002. Introduction to the Special Issue on Summarization. *Computational Linguistics*, 28(4): 399-408.

S. Sarawagi and W.W. Cohen. 2004. Semi-markov conditional random fields for information extraction.In *NIPS*

D. Shen, J. T. Sun, H. Li, Q. Yang, Z. Chen. 2007. Document Summarization using Conditional Random Fields' In *IJCAI*, 1805-1813

J. Y. Yeh, H. R. Ke, W. P. Yang and I. H. Meng. 2005. Text summarization using trainable summarizer and latent semantic analysis. *IPM*, 41(1): 75–95