From Extractive to Abstractive Meeting Summaries: Can It Be Done by Sentence Compression?

Fei Liu and Yang Liu
Computer Science Department
The University of Texas at Dallas
Richardson, TX 75080, USA
{feiliu, yangl}@hlt.utdallas.edu

Abstract

Most previous studies on meeting summarization have focused on extractive summarization. In this paper, we investigate if we can apply sentence compression to extractive summaries to generate abstractive summaries. We use different compression algorithms, including integer linear programming with an additional step of filler phrase detection, a noisy-channel approach using Markovization formulation of grammar rules, as well as human compressed sentences. Our experiments on the ICSI meeting corpus show that when compared to the abstractive summaries, using sentence compression on the extractive summaries improves their ROUGE scores; however, the best performance is still quite low, suggesting the need of language generation for abstractive summarization.

1 Introduction

Meeting summaries provide an efficient way for people to browse through the lengthy recordings. Most current research on meeting summarization has focused on extractive summarization, that is, it extracts important sentences (or dialogue acts) from speech transcripts, either manual transcripts or automatic speech recognition (ASR) output. Various approaches to extractive summarization have been evaluated recently. Popular unsupervised approaches are maximum marginal relevance (MMR), latent semantic analysis (LSA) (Murray et al., 2005a), and integer programming (Gillick et al., 2009). Supervised methods include hidden Markov model (HMM), maximum entropy, conditional random fields (CRF), and support vector machines (SVM) (Galley, 2006; Buist et al., 2005; Xie et al., 2008; Maskey and Hirschberg, 2006). (Hori et al., 2003) used a word based speech summarization approach that utilized dynamic programming to obtain a set of words to maximize a summarization score.

Most of these summarization approaches aim for selecting the most informative sentences, while less attempt has been made to generate abstractive summaries, or compress the extracted sentences and merge them into a concise summary. Simply concatenating extracted sentences may not comprise a good summary, especially for spoken documents, since speech transcripts often contain many disfluencies and are redundant. The following example shows two extractive summary sentences (they are from the same speaker), and part of the abstractive summary that is related to these two extractive summary sentences. This is an example from the ICSI meeting corpus (see Section 2.1 for more information on the data).

Extractive summary sentences:
Sent1: um we have to refine the tasks more and more which of course we haven't done at all so far in order to avoid this rephrasing
Sent2: and uh my suggestion is of course we we keep the wizard because i think she did a wonderful job

Corresponding abstractive summary:
the group decided to hire the wizard and continue with the refinement...

In this paper, our goal is to answer the question if we can perform sentence compression on an extractive summary to improve its readability and make it more like an abstractive summary. Compressing sentences could be a first step toward our ultimate goal of creating an abstract for spoken documents. Sentence compression has been widely studied in language processing. (Knight and Marcu, 2002; Cohn and Lapata, 2009) learned rewriting rules that indicate which words should be dropped in a given context. (Knight and Marcu, 2002; Turner and Charniak, 2005) applied the noisy-channel framework to predict the possibilities of translating a sentence to a shorter word sequence. (Galley and McKeown, 2007) extended the noisy-channel approach and proposed a head-driven Markovization formulation of synchronous context-free grammar (SCFG) deletion rules. Unlike these approaches that need a training corpus, (Clarke and Lapata, 2008) encoded the language model and a variety of linguistic constraints as linear inequalities, and employed the integer programming approach to find a subset of words that maximize an objective function.

Our focus in this paper is not on new compression algorithms, but rather on using compression to bridge the gap of extractive and abstractive summarization. We use different automatic compression algorithms. The first one is the integer programming (IP) framework, where we also introduce a filler phrase (FP) detection

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 261–264,
Suntec, Singapore, 4 August 2009.

module based on the Web resources. The second one uses the SCFG that considers the grammaticality of the compressed sentences. Finally, as a comparison, we also use human compression. All of these compressed sentences are compared to abstractive summaries. Our experiments using the ICSI meeting corpus show that compressing extractive summaries can improve human readability and the ROUGE scores against the reference abstractive summaries.

2 Sentence Compression of Extractive Summaries

2.1 Corpus

We used the ICSI meeting corpus (Janin et al., 2003), which contains naturally occurring meetings, each about an hour long. All the meetings have been transcribed and annotated with dialogue acts (DAs), topics, abstractive and extractive summaries (Shriberg et al., 2004; Murray et al., 2005b). In this study, we use the extractive and abstractive summaries of 6 meetings from this corpus. These 6 meetings were chosen because they have been used previously in other related studies, such as summarization and keyword extraction (Murray et al., 2005a). On average, an extractive summary contains 76 sentences[1] (1252 words), and an abstractive summary contains 5 sentences (111 words).

2.2 Compression Approaches

2.2.1 Human Compression

The data annotation was conducted via Amazon Mechanical Turk[2]. Human annotators were asked to generate condensed version for each of the DAs in the extractive summaries. The compression guideline is similar to (Clarke and Lapata, 2008). The annotators were asked to only remove words from the original sentence while preserving most of the important meanings, and make the compressed sentence as grammatical as possible. The annotators can leave the sentence uncompressed if they think no words need to be deleted; however, they were not allowed to delete the entire sentence. Since the meeting transcripts are not as readable as other text genres, we may need a better compression guideline for human annotators. Currently we let the annotators make their own judgment what is an appropriate compression for a spoken sentence.

We split each extractive meeting summary sequentially into groups of 10 sentences, and asked 6 to 10 online workers to compress each group. Then from these results, another human subject selected the best annotation for each sentence. We also asked this human judge to select the 4-best compressions. However, in this study, we only use the 1-best annotation result. We would like to do more analysis on the 4-best results in the future.

[1]The extractive units are DAs. We use DAs and sentences interchangeably in this paper when there is no ambiguity.

[2]http://www.mturk.com/mturk/welcome

2.2.2 Filler Phrase Detection

We define filler phrases (FPs) as the combination of two or more words, which could be discourse markers (e.g., I mean, you know), editing terms, as well as some terms that are commonly used by human but without critical meaning, such as, "for example", "of course", and "sort of". Removing these fillers barely causes any information loss. We propose to use web information to automatically generate a list of filler phrases and filter them out in compression.

For each extracted summary sentence of the 6 meetings, we use it as a query to Google and examine the top N returned snippets (N is 400 in our experiments). The snippets may not contain all the words in a sentence query, but often contain frequently occurring phrases. For example, "of course" can be found with high frequency in the snippets. We collect all the phrases that appear in both the extracted summary sentences and the snippets with a frequency higher than three. Then we calculate the inverse sentence frequency (ISF) for these phrases using the entire ICSI meeting corpus. The ISF score of a phrase i is:

$$isf_i = \frac{N}{N_i}$$

where N is the total number of sentences and N_i is the number of sentences containing this phrase. Phrases with low ISF scores mean that they appear in many occasions and are not domain- or topic-indicative. These are the filler phrases we want to remove to compress a sentence. The three phrases we found with the lowest ISF scores are "you know", "i mean" and "i think", consistent with our intuition.

We also noticed that not all the phrases with low ISF scores can be taken as FPs ("we are" would be a counter example). We therefore gave the ranked list of FPs (based on ISF values) to a human subject to select the proper ones. The human annotator crossed out the phrases that may not be removable for sentence compression, and also generated simple rules to shorten some phrases (such as turning "a little bit" into "a bit"). This resulted in 50 final FPs and about a hundred simplification rules. Examples of the final FPs are: 'you know', 'and I think', 'some of', 'I mean', 'so far', 'it seems like', 'more or less', 'of course', 'sort of', 'so forth', 'I guess', 'for example'. When using this list of FPs and rules for sentence compression, we also require that an FP candidate in the sentence is considered as a phrase in the returned snippets by the search engine, and its frequency in the snippets is higher than a pre-defined threshold.

2.2.3 Compression Using Integer Programming

We employ the integer programming (IP) approach in the same way as (Clarke and Lapata, 2008). Given an utterance $S = w_1, w_2, ..., w_n$, the IP approach forms a compression of this utterance only by dropping words and preserving the word sequence that maximizes an objective function, defined as the sum of the signifi-

cance scores of the consisting words and n-gram probabilities from a language model:

$$\max \lambda \cdot \sum_{i=1}^{n} y_i \cdot Sig(w_i) + (1-\lambda) \cdot \sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} x_{ijk} \cdot P(w_k|w_i, w_j)$$

where y_i and x_{ijk} are two binary variables: $y_i = 1$ represents that word w_i is in the compressed sentence; $x_{ijk} = 1$ represents that the sequence w_i, w_j , w_k is in the compressed sentence. A trade-off parameter λ is used to balance the contribution from the significance scores for individual words and the language model scores. Because of space limitation, we omitted the special sentence beginning and ending symbols in the formula above. More details can be found in (Clarke and Lapata, 2008). We only used linear constraints defined on the variables, without any linguistic constraints.

We use the lp_solve toolkit.[3] The significance score for each word is its TF-IDF value (term frequency $\times$ inverse document frequency). We trained a language model using SRILM [4] on broadcast news data to generate the trigram probabilities. We empirically set λ as 0.7, which gives more weight to the word significance scores. This IP compression method is applied to the sentences after filler phrases (FPs) are filtered out. We refer to the output from this approach as "FP + IP".

2.2.4 Compression Using Lexicalized Markov Grammars

The last sentence compression method we use is the lexicalized Markov grammar-based approach (Galley and McKeown, 2007) with edit word detection (Charniak and Johnson, 2001). Two outputs were generated using this method with different compression rates (defined as the number of words preserved in the compression divided by the total number of words in the original sentence).[5] We name them "Markov (S1)" and "Markov (S2)" respectively.

3 Experiments

First we perform human evaluation for the compressed sentences. Again we use the Amazon Mechanical Turk for the subjective evaluation process. For each extractive summary sentence, we asked 10 human subjects to rate the compressed sentences from the three systems, as well as the human compression. This evaluation was conducted on three meetings, containing 244 sentences in total. Participants were asked to read the original sentence and assign scores to each of the compressed sentences for its informativeness and grammaticality respectively using a 1 to 5 scale. An overall score is calculated as the average of the informativeness and grammaticality scores. Results are shown in Table 1.

[3] http://www.geocities.com/lpsolve

[4] http://www.speech.sri.com/projects/srilm/

[5] Thanks to Michel Galley to help generate these output.

For a comparison, we also include the ROUGE-1 F-scores (Lin, 2004) of each system output against the human compressed sentences.

Approach	Info.	Gram.	Overall	R-1 F (%)
Human	4.35	4.38	4.37	-
Markov (S1)	3.64	3.79	3.72	88.76
Markov (S2)	2.89	2.76	2.83	62.99
FP + IP	3.70	3.95	3.82	85.83

Table 1: Human evaluation results. Also shown is the ROUGE-1 (unigram match) F-score of different systems compared to human compression.

We can see from the table that as expected, the human compression yields the best performance on both informativeness and grammaticality. 'FP + IP' and 'Markov (S1)' approaches also achieve satisfying performance under both evaluation metrics. The relatively low scores for 'Markov (S2)' output are partly due to its low compression rate (see Table 2 for the length information). As an example, we show below the compressed sentences from human and systems for the first sentence in the example in Sec 1.

Human: we have to refine the tasks in order to avoid rephrasing
Markov (S1): we have to refine the tasks more and more which we haven't done in order to avoid this rephrasing
Markov (S2): we have to refine the tasks which we haven't done order to avoid this rephrasing
FP + IP: we have to refine the tasks more and more which we haven't done to avoid this rephrasing

Since our goal is to answer the question if we can use sentence compression to generate abstractive summaries, we compare the compressed summaries, as well as the original extractive summaries, against the reference abstractive summaries. The ROUGE-1 results along with the word compression ratio for each compression approach are shown in Table 2. We can see that all of the compression algorithms yield better ROUGE score than the original extractive summaries. Take Markov (S2) as an example. The recall rate dropped only 8% (from the original 66% to 58%) when only 53% words in the extractive summaries are preserved. This demonstrates that it is possible for the current sentence compression systems to greatly condense the extractive summaries while preserving the desirable information, and thus yield summaries that are more like abstractive summaries. However, since the abstractive summaries are much shorter than the extractive summaries (even after compression), it is not surprising to see the low precision results as shown in Table 2. We also observe some different patterns between the ROUGE scores and the human evaluation results in Table 1. For example, Markov (S2) has the highest ROUGE result, but worse human evaluation score than other methods.

To evaluate the length impact and to further make

	All Sent.				Top Sent.		
Approach	Word ratio (%)	P(%)	R(%)	F(%)	P(%)	R(%)	F(%)
Original extractive summary	100	7.58	66.06	12.99	29.98	34.29	31.83
Human compression	65.58	10.43	63.00	16.95	34.35	37.39	35.79
Markov (S1)	67.67	10.15	61.98	16.41	34.24	36.88	35.46
Markov (S2)	53.28	11.90	58.14	18.37	32.23	34.96	33.49
FP + IP	76.38	9.11	59.85	14.78	31.82	35.62	33.57

Table 2: Compression ratio of different systems and ROUGE-1 scores compared to human abstractive summaries.

the extractive summaries more like abstractive summaries, we conduct an oracle experiment: we compute the ROUGE score for each of the extractive summary sentences (the original sentence or the compressed sentence) against the abstract, and select the sentences with the highest scores until the number of selected words is about the same as that in the abstract.[6] The ROUGE results using these selected top sentences are shown in the right part of Table 2. There is some difference using all the sentences vs. the top sentences regarding the ranking of different compression algorithms (comparing the two blocks in Table 2).

From Table 2, we notice significant performance improvement when using the selected sentences to form a summary. These results indicate that, it may be possible to convert extractive summaries to abstractive summaries. On the other hand, this is an oracle result since we compare the extractive summaries to the abstract for sentence selection. In the real scenario, we will need other methods to rank sentences. Moreover, the current ROUGE score is not very high. This suggests that there is a limit using extractive summarization and sentence compression to form abstractive summaries, and that sophisticated language generation is still needed.

4 Conclusion

In this paper, we attempt to bridge the gap between extractive and abstractive summaries by performing sentence compression. Several compression approaches are employed, including an integer programming based framework, where we also introduced a filler phrase detection module, the lexicalized Markov grammar-based approach, as well as human compression. Results show that, while sentence compression provides a promising way of moving from extractive summaries toward abstracts, there is also a potential limit along this direction. This study uses human annotated extractive summaries. In our future work, we will evaluate using automatic extractive summaries. Furthermore, we will explore the possibility of merging compressed extractive sentences to generate more unified summaries.

References

A. Buist, W. Kraaij, and S. Raaijmakers. 2005. Automatic summarization of meeting data: A feasibility study. In *Proc. of CLIN*.

E. Charniak and M. Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proc. of NAACL*.

J. Clarke and M. Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

T. Cohn and M. Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*.

M. Galley and K. McKeown. 2007. Lexicalized markov grammars for sentence compression. In *Proc. of NAACL/HLT*.

M. Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Proc. of EMNLP*.

D. Gillick, K. Riedhammer, B. Favre, and D. Hakkani-Tur. 2009. A global optimization framework for meeting summarization. In *Proc. of ICASSP*.

C. Hori, S. Furui, R. Malkin, H. Yu, and A. Waibel. 2003. A statistical approach to automatic speech summarization. *Journal on Applied Signal Processing*, 2003:128–139.

A. Janin, D. Baron, J. Edwards, D. Ellis, G. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. 2003. The ICSI meeting corpus. In *Proc. of ICASSP*.

K. Knight and D. Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139:91–107.

C. Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proc. of ACL Workshop on Text Summarization Branches Out*.

S. Maskey and J. Hirschberg. 2006. Summarizing speech without text using hidden markov models. In *Proc. of HLT/NAACL*.

G. Murray, S. Renals, and J. Carletta. 2005a. Extractive summarization of meeting recordings. In *Proc. of INTERSPEECH*.

G. Murray, S. Renals, J. Carletta, and J. Moore. 2005b. Evaluating automatic summaries of meeting recordings. In *Proc. of ACL 2005 MTSE Workshop*.

E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, and H. Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) corpus. In *Proc. of SIGdial Workshop on Discourse and Dialogue*.

J. Turner and E. Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proc. of ACL*.

S. Xie, Y. Liu, and H. Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *Proc. of IEEE Workshop on Spoken Language Technology*.

[6] Thanks to Shasha Xie for generating these results.