A Template-based Abstractive Summarization of Meeting Conversations

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng
Department of Computer Science, University of British Columbia, Vancouver, Canada
{toya, mehdad, carenini, rng}@cs.ubc.ca

1. Introduction

The most common approaches to automatic meeting summarization have been extractive. Since extractive approaches require no natural language generation techniques, they are arguably simpler to apply and have been extensively investigated. However, a user study conducted by (Murray et al., 2010) indicates that people prefer automatically generated abstractive meeting summaries to extractive ones. In this paper, we introduce a fully automatic abstractive summarization system for meeting conversations. The main contribution of this paper is that we have successfully applied a multi-sentence fusion algorithm for generating generalized templates. In addition, we demonstrate an approach to generate abstract sentences from such templates.

2. Approach

Our framework consists of two components: 1) a Template Generation module, which generalizes collected human abstract summaries and creates templates; and 2) a Summary Generation module, which extracts important phrases from a transcripted meeting and creates a summary of the meeting by filling these phrases into templates. The next two sections describe each of these components in detail.

3. Template Generation Module

Templates are derived from training portion of human-authored abstractive summary sentences provided in the AMI corpus (Carletta et al., 2005), which contains 139 transcripted meetings and their summaries. We first identify all noun phrases in the human summary sentences and label the phrases with their hypernyms using WordNet. In WordNet, hyponyms are organized into hierarchies starting from the most abstract to the most specific. For our work, we utilize the fourth most abstract hypernyms. We then treat these hypernym-labeled sentences as templates and the phrases as blanks.

Next, we further generalize these templates by utilizing a clustering and a multi-sentence compression algorithm. To cluster them based on their similarity, we first create a fully-connected graph where each node represents a root verb in a template and each edge represents a similarity score between the connected verbs calculated by their hypernym path distance in WordNet. We then apply a Normalized Cuts algorithm (Shi and Malik, 2000) to cluster the root verbs based on similarity. Finally, all the templates are organized into the groups created by their root verbs.

A novel multi-sentence compression algorithm introduced by (Filippova, 2010) is then used to create generalized templates. For each clustered group, the algorithm first constructs a word graph and ranks the paths in the graph.

A word graph is a directed graph with words or blanks as nodes and edges connecting them. The graph is constructed by first creating a start and end node then iteratively adding templates to it. When a new template is added, a word or blank from this template is mapped onto a node in the graph if the node consists of the same word or blank and if it has not been mapped to any word or blank in this template. When more than one node refer to the same word or blank in the template, or when more than one word or blank in the template can be mapped to the same node in the graph, for disambiguation purpose, the algorithm checks the neighboring nodes in the graph and the preceding and the following words or blanks in the template. Then it selects the mapping based on the overlaps in context.

To select the concise and generalized templates, for each path that connect the start and end nodes, we create a path ranking model. Our ranking model is inspired by the one used in (Yashar et al., 2013) and considers: 1) Fluency measured by a language model trained on all templates and 2) Edge weight which is defined so that the paths that are informative and those that contain salient words and blanks are selected. We select the top ten scored templates for each group.

As an illustration, we show a word graph in Figure 1 obtained from the following three templates.

- After introducing [situation.n.01], [speaker] then discussed [content.n.05] .
- [speaker] discussed [act.n.02] and [content.n.05] for [artifact.n.01] .
- [speaker] briefly discussed [content.n.05] of [artifact.n.01] and [material.n.01] .

The best scored path is shown in bold, which clearly shows that the algorithm is selecting a generalized template.

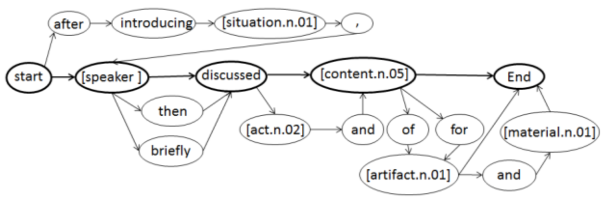

Figure 1 A word graph generated from related templates and the best path selected

4. Summary Generation Module

In this section, we introduce our summary generation module. First, it is important for a summary to cover all topics discussed in a meeting. Therefore, given a meeting transcript to be summarized, we segment the transcript employing a topic segmenter, LCSeg (Galley et al., 2003).

Second, all salient phrases are extracted from each topic segment in the same way as we did in the first module: 1) extract all noun phrases; and 2) label each with their hypernym. All extracted phrases are subsequently scored and ranked based on the sum of their word frequencies.

Third, we select the ideal templates to be used for generating summary sentences for each topic segment.

In the AMI corpus, all human-authored abstractive summary sentences in the training data have links to the subsets of their source transcripts which convey the information in the abstractive sentences. These subsets are called communities. Since each community is used to create one summary sentence, we hypothesize that each community covers one specific topic.

Thus, to find the best templates for each topic segment, we first find the communities in the training set that are similar to the topic segment and identify the templates derived from the summary sentences linked to these communities.

This process is done in two steps, by: 1) Associating the communities in the training data with the groups containing templates that were created in our template generation module; and 2) Finding templates for each topic segment by comparing the similarities between the segments and all sets of communities associated with the template groups. Below, we describe the two steps in detail.

1) Recall that in the template creation module, we label human-authored abstractive summary sentences in training data with hypernyms and cluster them into similar groups. Thus, we first associate all sets of communities in the training data into these groups by determining to which groups the summary sentences linked by these communities belong.

2) Next, for each segment, we compute average cosine similarity between the segment and all communities in all of the groups. At this stage, each community is already associated with a group that contains ranked templates. In addition, each segment has a list of average-scores that measures how similar the segment is to the communities in each group. Hence, the templates used for each segment are decided by selecting the ones from the groups with higher scores.

Our system now has a set of ranked phrases, and ideal templates for each segment. Thus, in the next step, templates and phrases are selected based on their score. Then, the candidate sentences are generated by filling the templates with the phrases having the same hypernym to blanks.

Finally, all the sentences are ranked using a ranking model that considers: 1) Fluency measured by a language model; and 2) Coverage that gives additional reward to the sentence containing high scored phrases. For each segment, the sentence with the highest score is selected and included in the final summary.

5. Experiments and Preliminary Results

For our preliminary tests, we use manual transcripts of meeting records from the AMI corpus. The templates are created on 85 meetings and tested on 30 meetings. Preliminary results on the test meetings indicate that our system creates informative and readable summaries. We are currently planning a more comprehensive evaluation to compare our system with various baselines.

References

J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner. 2005. The AMI meeting corpus: A pre-announcement. In *Proceeding of MLMI 2005*, Edinburgh, UK, pages 28–39.

Christiane Fellbaum 1998. *WordNet, An Electronic Lexical Database.* The MIT Press. Cambridge, MA.

Katja Filippova. 2010. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics, COLING '10*, pages 322–330, Stroudsburg, PA, USA. Association for Computational Linguistic

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003).* 562–569. Sapporo, Japan.

Ravi Kondadadi, Blake Howald and Frank Schilder. 2013. A Statistical NLG Framework for Aggregated Planning and Realization. In *Proceeding of the Annual Conferene for the Association of Computational Linguistic (ACL 2013).*

Marie-Catherine de Marneffe, Bill MacCartney and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06).*

Yashar Mehdad, Giuseppe Carenini, Frank Tompa. 2013. Abstractive Meeting Summarization with Entailment and Fusion. In *Proceedings of the 14th European Natural Language Generation (ENLG - SIGGEN 2013),* Sofia, Bulgaria.

Rata Mihalcea and Paul Tarau 2004. TextRank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, July.

Gabriel Murray, Giuseppe Carenini, and Raymond T. Ng. 2010. Generating and validating abstracts of meeting conversations: a user study. In *INLG 2010*.

Gabriel Murray, Giuseppe Carenini and Raymond Ng. 2012. Using the Omega Index for Evaluating Abstractive Community Detection, *NAACL 2012, Workshop on Evaluation Metrics and System Comparison for Automatic Summarization,* Montreal, Canada.

Vasin Punyakanok and Dan Roth. 2001. The Use of Classifiers in Sequential Inference. *NIPS (2001)* pp. 995-1001.

Jianbo Shi and Jitendra Malik. 2000. Normalized Cuts & Image Segmentation. *IEEE Trans. of PAMI,* Aug 2000.

David C. Uthus and David W. Aha. 2011. Plans toward automated chat summarization. In *Proceedings of the Workshop on Automatic Summarization for Different Genres, Media, and Languages, WASDGML'11*, pages 1-7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lu Wang and Claire Cardie. 2013. Domain-Independent Abstract Generation for Focused Meeting Summarization. In *ACL 2013*.

6. Appendix A. A Sample Output of Our System

Summary Created by Our System
project manager summarized their role of the meeting. user interface expert and project manager talks about a universal remote. the group recommended using the International Remote Control Association rather than a remote control. project manager offered the ball idea. user interface expert suggested few buttons. user interface expert and industrial designer then asked a member about a nice idea for The idea. project manager went over a weak point. the group announced the one-handed design. project manager and industrial designer went over their remote control idea. project manager instructed a member to research the ball function. industrial designer went over stability point. industrial designer went over definite points.

Human-Authored Summary
The project manager opened the meeting and had the team members introduce themselves and describe their roles in the upcoming project. The project manager then described the upcoming project. The team then discussed their experiences with remote controls. They also discussed the project budget and which features they would like to see in the remote control they are to create. The team discussed universal usage, how to find remotes when misplaced, shapes and colors, ball shaped remotes, marketing strategies, keyboards on remotes, and remote sizes. team then discussed various features to consider in making the remote.

Figure 2: A comparison between a summary created by our system and a human-authored abstractive summary