Beyond utterance extraction: summary recombination for speech summarization

Jérémy Trione, Benoit Favre, Frédéric Béchet

Beyond utterance extraction: summary recombination for speech summarization

Jérémy Trione, Benoit Favre, Frédéric Béchet

Aix-Marseille Université, CNRS, LIF UMR 7279, 13000, Marseille, France

{first.last}@lif.univ-mrs.fr

Abstract

This paper describes a template filling approach for creating conversation summaries. The templates are generated from generalized summary fragments from a training corpus. Necessary pieces of information for filling them are extracted automatically from the conversation transcripts given linguistic features, and drive the fragment selection process. The approach obtains ROUGE-2 scores of 0.08471 on the RATP-DECODA corpus, which represents a significant improvement over extractive baselines and hand-written templates.

Index Terms: Summary, synopsis, template, abstractive, ROUGE, icsiboost, slots

1. Introduction

Automatic summarization is generally based on extractive methods that gather relevant text segments to make a summary. These methods are not well suited for spoken conversation summary generation, due to their spontaneous and interactive nature. By selecting only a few utterances from a conversation, extractive summaries just give a broad picture of a given point in the conversation and not a full synthetic description of what happened between the different participants.

For instance, in the call centre domain, it would be good if the summaries could inquire about the issues described by the callers and the solutions outlined by the agents. Often, issues are described in multiple speaker turns, from the caller in interaction with the agent, which is difficult to capture with an extractive approach, especially when length constraints are tight.

Template filling is a summarization approach which has shown success when the domain of the conversations is restricted [1]. It consists in filling hand-crafted templates with information extracted from the conversations transcripts. In the case of call-centre conversations, this method can be tackled for generating short narrative summaries which recount of what happened during the call. However, in addition to hand-writing templates, and annotating transcripts with template slots, this approach is limited in that it cannot handle situations that have not been imagined by the template creators.

We argue that if a database of hand-written summaries (called *synopsis* in our experimental setting) is available for training the system in the target domain, it should be possible to take advantage of this valuable data and create summaries by *recombining* fragments from existing summaries. The idea is to generate ad-hoc templates for the current conversation and fill them with the slots detected in the transcript. Technically, this approach can be qualified as extractive except that sentences are not extracted from the conversations, but rather from existing summaries. Their content is generalized to match the specificities of the conversation and only describe corresponding events.

Our contributions are the following:

- Extract meaningful concepts from the transcriptions to fill the template.
- Link concepts annotated in the manual summaries with the conversation, minimising the annotation cost.
- Dynamically generate templates from the manual summaries and the information detected in the conversation.

Section 2 relates our work to existing research, Section 3 details our approach, Section 4 describes our slot detectors, Section 5 specifies the generation approach and Section 6 shows experiment results. The paper is concluded in Section 7.

2. Related work

A good overview of the state of the art in automatic summarization can be found in [2]. Additional references for abstractive summarization can be found in [3]. Even recent work characterized as abstractive, for generating sentences from recombinations of the source documents, results in summaries far from the synthetic and structured text that they should be. For example, [4] construct novel sentences from subject noun phrases and verb phrases from the source documents. An ILP decoder both selects the phrases and creates the sentences. [5] explore semantic parsing with the AMR representation and show that it is possible to create good bag of words for use by a generation module, but they do not go as far as to actually perform generation. Semantic and syntactic parsing of speech transcripts is still challenging, making those methods less appealing for call-centre conversation summarization.

A framework which learns to alternate between copying the source and generating text can be effective for headline generation [6]. However, training such neural networks is only practical for small inputs, and requires tremendous quantity of training data. We also explore the idea of alternating between copying text from the source and generating novel text that is not in the source. However, the lack of large quantities of training data in the call-centre domain forces us to rely on a less risky approach: extract text from existing summaries.

The problem with utterance extraction is that it relies on the assumption that the style of the source documents matches that of the summaries. While this assumption is reasonable in journalistic domains, in the call-centre domain, the targeted summaries are synthetic, narratives that tell the story of what happened during a call, a style very different from that of conversation transcripts. The specifics of call-centre transcript summarization are the focus of several papers. For instance in [7], call logs are generated by filling hand-written templates thanks to information extraction models, and these templates are complemented by unstructured data extraction. [8] perform an unsupervised topic induction over the utterances from dialogues

in a set of different domains, and train an HMM that models both domain-specific and domain-wide topic sequences. Optimal utterance sequences are selected with the Viterbi algorithm. These methods require either template engineering, or rely on utterance extraction from the conversation transcripts.

Mehdad and al. [9] propose an abstractive summarization approach that fills templates which are automatically generated. The sentences in the documents are clustered then linked into a words graph. Each sentence in the summary is generated as a path in this graph. The approach looks promising but performance on meeting data is worse than that obtained by an extractive method based on sentence topic classification [10]. We follow in our work the idea of automatically generating templates. However, unlike [9] we generate these templates from a corpus of conversation summaries rather than the source documents.

3. Proposed approach

Our approach for spoken conversation summarization is based on template filling. Each slot in a template is filled depending on the analysis of conversation transcripts in order to produce a *synposis*, which is a short summary of the whole conversation. Standard template filling methods are based on manually written templates that cannot be modified to fit the specificities of a conversation. We propose in this study to generate templates dynamically thanks to a training corpus made of pairs of *conversation transcripts* and *synopses*. Our training process is made of the following steps:

1. Concept slot detection: conversation transcriptions and synopsis of the training corpus are parsed in order to detect slots corresponding to the *concepts* relevant to characterize conversations. The list of concepts used is related to the applicative domain of the corpus.
2. Sentence template generation: all sentences in the synopsis corpus are generalized by replacing concept values by labels in order to produce *sentence templates*. Examples of such templates can be found in table 1 for three different topics: *Itinerary*, *Navigo* and *Lost&Found*.
3. Concept linking: this task consists in linking concepts occurring in a summary to the same concepts in the corresponding conversation. A classifier is trained in order to predict, for all concepts detected in a given conversation, if they would occur in its corresponding summary.

Once the sentence templates and concept-linking classifier are obtained, the summarization process of a new conversation transcription is as follows:

1. Relevant concept detection: the concept-linking classifier is used in order to detect the *relevant* concepts in the conversation transcription. A concept is considered *relevant* if it would occur in the synopsis of the conversation.
2. Sentence template selection: this step consists in dynamically choosing sentence templates from the template repository according to the slots detected in the previous step.
3. Synopsis generation: all the sentence template selected are filled with the concept values found in the conversation transcription; then this set of sentences is ordered in order to produce the final synopsis.

These processes are described in more details in the next sections.

Topic	**Template**
Schedule	Query for schedules (using $TRANSPORT)? from $FROM to $TO.
Itinerary	Query for itinerary (using $TRANSPORT)? from $FROM to $TO (without using $NOT TRANSPORT)?. (Take the $LINE towards $TOWARDS from $START STOP to $END STOP.). Query for location $LOCATION.
Navigo	Query for (justification||refund||fares||receipt) for $CARD TYPE. Customer has to go to offices at $ADDRESS.
Lost&found	$ITEM lost in $TRANSPORT (at $LOCATION)? (around $TIME)?. (Found, to be retrieved from $RETRIEVE LOCATION || Not found).

Table 1: Example of templates for four different topics: Schedule, Itinerary, Navigo and Lost&Found

All the experiments presented in this study rely on the RATP-DECODA corpus. This corpus is made of 1,500 conversations recorded at the Paris Public Transport Authority (RATP) call-centre during a two-day period in 2009 [11]. Topics covered by the conversations include traffic information (status of the lines), itinerary search, schedule requests, lost-and-found, fares and monthly passes, etc.

4. Concept slot detection

Detecting slot values in conversation transcripts in order to fill templates requires training data. We consider a lightweight approach which consists in manually annotating a set of summaries with slot segments, and propagating these annotations to the conversation transcripts through alignment and matching. Then, a classifier is trained on the joint problem of determining where the slot values are in the transcripts and which slot values shall be used to fill the templates. We call this task *concept slot detection*. Table 2 shows the frequency of each of the slot type in the training data.

Slot name	**%**	**Slot name**	**%**
$TRANSPORT	42.3	$TO	27.4
$FROM	25.1	$CARD_TYPE	25.1
$INFO_TARGET	22.3	$ITEM	20.0
$ISSUE	18.3	$LINE	8.0
$LOCATION	5.1	$BUY	4.6
$TOWARDS	2.9	$END_STOP	2.9
$TIME	2.3	$NOT_TRANSPORT	2.3
$START_STOP	0.6	$FREQUENCY	0.6
$RETRIEVE_LOC	1.1		

Table 2: Frequency of slot types in the synopses. Slots correspond to important entities in the target domain.

4.1. Linking: propagation to conversation transcripts

Given a synopsis annotated with concept slots, the task consists in propagating the annotation to conversation transcripts. This linking task is performed along the following steps:

- Transcripts are automatically annotated with syntactic and semantic parses with the Macaon tool chain [12].
- Each slot in the annotated synopses is compared with all the phrases from the transcription thanks to Levenshtein alignment and a specific cost function. Text is first lowercased and diacritics are removed, the distance is computed at the character level.

- The slot value is associated with the phrase for which the alignment has lowest cost.

This method allows to align 316 slots on the 380 annotated in the synopses (83.16% alignment rate). The unaligned variables are in most cases due to manual annotation errors, too generic references that can't be detected at the word level, or a total mismatch between the synopsis and the conversation (i.e. the word does not appear in the conversation, which is the case when the author of the synopsis generalized a concept using a different word.)

4.2. Slot prediction features

The previous step leads to the creation of a corpus associating slots from the synopses and values from the transcripts. This data can be leveraged to train a slot classifier. Again we take advantage of the parses generated by Macaon [12] for feature extraction. For each phrase in a conversation, the classifier is trained to predict a type of slot among 19 available plus the NULL label indicating that the phrase is not a concept. The classifier uses the following features as input:

- **Syntactic head of the phrase:** word, lemma, part-of-speech tag, named entity tag.
- **Governor of syntactic head:** lemma, part-of-speech tag, dependency label.
- **Phrase:** length, bag of n-grams of words ($n \leq 3$), bag of n-grams of part of speech ($n \leq 3$).
- **Conversation and discourse:** number of named entities of the same type since the beginning of the conversation, number of occurrences of the head lemma since the beginning of the conversation, topic of the conversation, relative position of the phrase in the conversation, speaker role (agent or caller).

Given those features, the scores output by the classifier are passed through a *softmax* to represent probabilities between 0 and 1. For a conversation, at test time, scores for the NULL class are discarded and for each slot type, all phrases which exceed a decision threshold θ are selected for use in synopsis generation.

Using conversation and discourse features is not conventional in the concept or named entity recognition tasks. They help address the relevance of the detected concepts. For instance, a number of bus stops might be referred to in the conversation while only one is relevant for filling the template.

5. Synopsis generation

Synopsis generation is performed by combining fragments of synopses gathered in the training data, and replacing their concept slot values with those detected in the transcript. In a way, this approach can be considered as extractive except that existing synopses are leveraged instead of conversation utterances.

First, synopses from the training set are split in sentences and slot values are replaced by tokens indicating their type. Those sub-templates can be selected and filled depending on the content of a conversation. Then, slots are detected in the transcript according to the approach described in Section 4. The selection process tries to saturate the sub-templates with detected slot values which match the expected slot types, under the constraint that a slot type can only be used once. From this population of saturated sub-templates, the generated synopsis is necessarily started with a sentence which was first in its original synopsis. Then, other sub-templates are concatenated arbitrarily. We decided to rely on this heuristic because in our data the first sentence of a synopsis always contains the right description of the issue of the call.

The advantage of this approach is that sub-template selection is driven by the detected slots. This both limits the risk to accidentally instantiate sub-templates based on misdetected information, and it also allows for the approach to cope with a limited quantity of novelty in the conversations: situations that are combination of already seen situations.

6. Evaluation

Experiments are performed on 141 conversations from the RATP DECODA corpus manually annotated in synopsis. Each conversation has between 1 and 3 unique synopses for a total of 381 synopses manually annotated with slot segments and type. In the following experiments we make both use of manual transcriptions with the reference linguistic annotations (syntactic and semantic) and automatic transcriptions generated with the LIUM ASR system [13] (with a WER of 35%) with automatic linguistic annotations generated by the Macaon pipeline [12].

The corpus is split in 71 conversations for training, 43 for testing and 27 for development. All parameters of the system, including the θ threshold are set in order to maximize performance on the development set.

6.1. Results

For experiments, we compare our approach (synopsis recombination) with manual template filling and a few extractive baselines and toplines. One manual template was written for each conversation topic in the corpus in order to cover most of the information in the synopses for that topic. The topline consists in manually filling the hand-crafted template with the most relevant slot values.

For slot value predictions, we use three classifiers: adaboost [14] with 1000 rounds of boosting, a deep neural network (called DNN thereafter) implemented with Chainer[1], and the libLinear classifier [15].

We follow the experimental setup of the CCCS shared task [16] except that we have a larger test set. The length limit for synopses is 7% of the conversation words, evaluation is performed with the ROUGE-2 metric. The following systems are compared:

- **Topline**: manual templates filled with reference slots
- **Human**: the average of the performances obtained by putting aside each reference synopsis (i.e manually written) and scoring it against the other references.
- **Templates**: manual templates filled with predicted slots.
- **Recombined**: the proposed approach.
- **MMR**: maximal marginal relevance.
- **Longest**: longest speaker turn in the conversation.
- **Longest@25**: longest speaker turn in the first quarter of the conversation.

The results detailed in Table 3 show that our method (Recombined) yields better results than both the extractive baselines and the manual templates (significant at $p < .05$). This

[1] http://chainer.org – Parameters: 1 hidden layer, ReLU activations, 4 epochs of training, no dropout. Searched for from a range of configuration to maximize classification accuracy on the dev set.

System	Transcript	Slots	ROUGE-2
Topline	manual	manual	0.20491
Human	-	-	0.11848
Templates	manual	Icsiboost	0.06818
	manual	libLinear	0.03735
	manual	DNN	0.02041
Recombined	manual	Icsiboost	0.08200
	manual	libLinear	**0.08390**
	manual	DNN	0.04830
MMR	manual	-	0.03145
Longest	manual	-	0.02688
Longest@25	manual	-	0.04046
Templates	ASR	Icsiboost	0.05270
	ASR	libLinear	0.02921
	ASR	DNN	0.01775
Recombined	ASR	Icsiboost	**0.08471**
	ASR	libLinear	0.08100
	ASR	DNN	0.04033
MMR	ASR	-	0.02093
Longest	ASR	-	0.01734
Longest@25	ASR	-	0.01734

Table 3: ROUGE results on the test set for all the systems, according to the transcript source, as well as how the slots were predicted. The proposed approach is called "Recombined."

is expected because by combining sentences from multiple synopses, the system can cover situations that could not be handled by a single template per topic. This seems to also be linked to the quality of slot prediction as the topline which relies on reference slots has a much better ROUGE score. The human synopses score worse than the topline because humans tend to diverge when writing summaries even for a same conversation topic. Moreover, each human synopsis was evaluated with one-less reference than the systems.

Also, it seems that the choice of the classifier does not matter except for the DNN which is not as good as the other classifiers, probably because it is trained on so little data (note that its configuration has been optimized on the development set). Finally, an interesting outcome is that ASR output and automatic prediction of linguistic annotation does not have a large impact on performance. This comes from the fact that ASR transcripts are of relatively good quality, and that relevant slot values are generally repeated multiple times by both speakers in a conversation. The choice of the decision threshold on the development set seems appropriate, as evidenced by Figure 1.

6.2. Analysis

It is well known that ROUGE is a limited metric, especially in the framework of abstractive summarization because it does not account for paraphrasing and assumes the few reference summaries to be representative of the possible wordings. To address this problem, we performed a qualitative analysis of the output and give a few examples of instances for which the system worked well and others where it failed. In every example the slots predicted by the system are written in bold. All examples are translated from French.
Acceptable synopses:

- Information request about forgotten **glasses** in the **subway**
- Itinerary request to go to **rue d'Alger in Massy**. The agent tells him to take the **RER B to Massy**
- Query for schedules to go to **Croix de Berny**

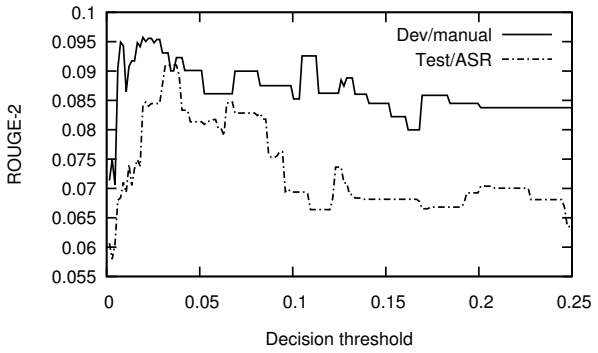

Figure 1: Variation of performance of the proposed system according to the decision threshold on the Icsiboost classifier scores. The choice of θ on the development set is relatively robust on the test set in the ASR condition.

Synopses with issues:

- The caller would like to go from the **Fischer stop** to **Fischer station**. The agent tells him to take the line to **Fischer station**. (repeated named entity, missing line name)
- The caller would like to go to Drancy station in Drancy. (did not find an itinerary)
- Information request about issue on the **line**. Found, go to **terminus** for retrieval (Wrong name issue, and wrong template selection)

A quick analysis of the failed synopses shows that our approach can lead to different types of errors: usage of the same value in consecutive slots types, sub-template selection errors, and coverage of situations which did not occur in the training data. That last problem has to be tackled with a completely different approach such as detecting deviant situations, and processing extracted utterances so that they match the expected style of the synopses.

7. Conclusion

We presented a method based on filling a template generated from fragments of synopses and informations detected in the conversation thanks to linguistic, interaction and discourse features. The proposed approach outperforms both extractive baselines and hand-crafted template filling by a large margin. The approach is still limited by the coverage of situations in the training data which we hope to address by combining extractive and abstractive frameworks.

8. Acknowledgements

- The research leading to these results has received funding from the European Union - Seventh Framework Programme (FP7/2007-2013) under grant agreement 610916 – SENSEI.
- The Tesla K40 used for this research was donated by the NVIDIA Corporation.
- The authors would like to thank Yannick Esteve and the LIUM team for sharing their ASR transcriptions of the RATP-DECODA corpus.

9. References

[1] M. White, T. Korelsky, C. Cardie, V. Ng, D. Pierce, and K. Wagstaff, "Multidocument summarization via information extraction," in *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, 2001, pp. 1–7.

[2] A. Nenkova, S. Maskey, and Y. Liu, "Automatic summarization," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts of ACL 2011*. Association for Computational Linguistics, 2011, p. 3.

[3] N. Salim, "A review on abstractive summarization methods," *Journal of Theoretical and Applied Information Technology*, vol. 59, no. 1, pp. 64–72, 2014.

[4] L. Bing, P. Li, Y. Liao, W. Lam, W. Guo, and R. J. Passonneau, "Abstractive multi-document summarization via phrase selection and merging."

[5] F. Liu, J. Flanigan, S. Thomson, N. Sadeh, and N. A. Smith, "Toward abstractive summarization using semantic representations," p. 1077–1086, 2015.

[6] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," *arXiv preprint arXiv:1603.06393*, 2016.

[7] R. J. Byrd, M. S. Neff, W. Teiken, Y. Park, K.-S. F. Cheng, S. C. Gates, and K. Visweswariah, "Semi-automated logging of contact center telephone calls," in *Proceedings of the 17th ACM conference on Information and knowledge management*. ACM, 2008, pp. 133–142.

[8] R. Higashinaka, Y. Minami, H. Nishikawa, K. Dohsaka, T. Meguro, S. Takahashi, and G. Kikui, "Learning to model domain-specific utterance sequences for extractive summarization of contact center dialogues," in *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*. Association for Computational Linguistics, 2010, pp. 400–408.

[9] Y. Mehdad, G. Carenini, and R. T. Ng, "Abstractive summarization of spoken and written conversations based on phrasal queries." in *ACL (1)*, 2014, pp. 1220–1230.

[10] N. Garg, B. Favre, K. Reidhammer, and D. Hakkani Tür, "Clusterrank: a graph based method for meeting summarization," in *Interspeech*, 2009.

[11] F. Bechet, B. Maza, N. Bigouroux, T. Bazillon, M. El-Beze, R. De Mori, and E. Arbillot, "Decoda: a call-centre human-human spoken conversation corpus." in *LREC*, 2012, pp. 1343–1347.

[12] T. Bazillon, M. Deplano, F. Bechet, A. Nasr, and B. Favre, "Syntactic annotation of spontaneous speech: application to call-center conversation data." in *LREC*, 2012, pp. 1338–1342.

[13] C. Lailler, A. Landeau, F. Béchet, Y. Estève, and P. Deléglise, "Enhancing the RATP-DECODA corpus with linguistic annotations for performing a large range of NLP tasks," in *LREC*, 2016.

[14] B. Favre, D. Hakkani-Tür, and S. Cuendet, "Icsiboost," http://code.google.come/p/icsiboost, 2007.

[15] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[16] B. Favre, E. Stepanov, J. Trione, F. Béchet, and G. Riccardi, "Call centre conversation summarization: A pilot task at multiling 2015," in *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2015, p. 232.