A Study of Text Summarization Techniques for Generating Meeting Minutes

Tu My Doan, Francois Jacquenet(✉), Christine Largeron, and Marc Bernard

Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d'Optique Graduate School,
Laboratoire Hubert Curien UMR 5516, 42023 Saint-Etienne, France
{Francois.Jacquenet,Christine.Largeron,
Marc.Bernard}@univ-st-etienne.fr

Abstract. A lot of research has been conducted all over the world in the domain of automatic text summarization and more specifically using machine learning techniques. Many state of the art prototypes partially solve this problem so we decided to use some of them to build a tool for automatic generation of meeting minutes. In fact, this was not an easy work and this paper presents various experiments that we did using Deep Learning, GANs and Transformers to achieve this goal as well as dead ends we have encountered during this study. We think providing such a feedback may be useful to other researchers who would like to undertake the same type of work to allow them to know where to go and where not to go.

Keywords: Text summarization · Text generation · GAN · Deep learning · Meeting summarization

1 Introduction

According to a Microsoft survey[1], employees globally spend an average of 5.6 hours a week in meetings and, one of the most sensitive and unpleasant things to do in a meeting is probably to write its minutes. However, in any organization, writing meeting minutes is essential for the life of project teams in order to keep track of what was said throughout the projects, the decisions made, the tasks performed, etc. Generating meeting minutes is within the scope of text summarization [8,11,15], a subfield of natural language processing. Two main approaches can be distinguished: *extractive summarization* and *abstractive summarization*. In the first approach, the main idea is to extract the top-k sentences of an input text, that is, the most important ones, and merge them to produce the summary as an output. In the second approach, the ultimate goal is

This work has been carried out as part of the REUS project funded under the FUI 22 by BPI France, the Auvergne Rhône-Alpes Region and the Grenoble metropolitan area, with the support of the competitiveness clusters Minalogic, Cap Digital and TES.

[1] https://tinyurl.com/yx2bz6pv.

to understand the meaning of the most important ideas developped in the input text and then produce, as an output, the summary of those ideas using words or sentences that may not be in the input text.

Obviously, abstractive summarization is far more difficult than extractive summarization. Nevertheless, the latter approach doesn't fit the task of summarizing meetings. Indeed, meetings are made up of dialogues, especially follow-up project meetings that we want to address. Therefore, the structure of a meeting is very different from the structure of normal texts and this can be a difficult problem. The issue with a dialogue is that an extractive method just copies parts of the input text without understanding the important underlying relationships between them and thus the underlying meaning of the whole conversation. This is why we focus on abstractive summarization techniques and the reader interested in a detailed survey on that domain can refer to [6]. Thus, the aim of this paper is twofold. Firstly, it presents experiments we did to design a tool for generating automatically the minutes of meetings. Secondly, it describes the dead ends encountered to solve this issue.

2 Supervised Deep Learning Approaches

2.1 Using the Bottom-Up Abstractive Summarization System

In the context of our study of text summarization techniques for meeting minutes generation, we decided to use the Bottom-up Abstractive Summarization prototype presented by Gehrmann et al. in [3]. It combines extractive and abstractive summarization techniques. In a first part, the system is trained to learn an extractive summarizer that is used to extract the most significant parts of the text. In a second part, the system uses the selected parts of the text to build an abstractive summary. It integrates the most advanced techniques of the moment (seq2seq, attention, pointer-generator, language models, etc.) and can be considered as an improved version of the system presented by See et al. in [16].

There are many parameters to tune for the training step[2] to efficiently run the prototype. We followed the authors' recommendations and used a 128-dimensional word-embedding, and 512-dimensional one layer LSTM. On the encoder side, as the prototype uses a bidirectional LSTM, the 512 dimensions were split into 256 dimensions per direction. The maximum norm of the gradient was set to 2, and a renormalization was done if the gradient norm exceeded this value and there was no dropout.

To apply this supervised approach, we need as an input some pairs of texts with their associated summaries. Various datasets have been used in that context such as the DUC dataset [14], the Gigaword dataset [13] or the CNN/Daily Mail dataset [7,12] to cite the most famous ones but for meeting summarization, there exists only one dataset called the AMI dataset [1]. It is made up of 142 written transcriptions of meetings with their associated abstractive summaries

[2] More details can be found here: http://opennmt.net/OpenNMT-py/Summarization.html.

written by humans. Those meetings are real meetings that were held as role-playing games, each of them having an approximate average duration of one hour. We mainly carried out three experiments, the results of which are presented in Table 1 in terms of the ROUGE measure [9]. First we trained the system on 80% of the AMI dataset and tested it on the remaining 20% (first column). Second we trained the system on CNN/DM and tested it on AMI (second column). Finally we trained the system on a part of CNN/DM and tested it on this dataset (third column) to confirm the results presented in [3].

Table 1. Performance of the system trained and tested on the AMI and CNN/DM datasets

Metric	AMI AMI	CNN/DM AMI	CNN/DM CNN/DM
ROUGE-1 Recall	0.21788	0.03991	0.43652
ROUGE-1 Precision	0.69117	0.23391	0.41798
ROUGE-1 F-measure	0.32103	0.06540	0.41323
ROUGE-2 Recall	0.07578	0.00676	0.19465
ROUGE-2 Precision	0.25072	0.04199	0.18575
ROUGE-2 F-measure	0.11241	0.01145	0.18376
ROUGE-L Recall	0.20664	0.03490	0.40367
ROUGE-L Precision	0.65332	0.20982	0.38710
ROUGE-L F-measure	0.30406	0.05748	0.38239

In the first column we can observe that the precision is quite good at 69.11% but the recall is only equal to 21.78% according to ROUGE-1, this is due to a training step on a really small dataset that does not provide enough vocabulary and an important variety of sentences. This is why we tried to solve this issue by training the system on CNN/DM and the second column of values shows that the results were very poor in terms of recall. In fact, the vocabulary and the style of sentences in CNN/DM have nothing to do with those of the AMI dataset, that explains the fact that the system is not able to generate interesting sentences. The third column confirms that the system has quite good performances (with a F-measure equals to 0.41 compared to 0.065 or 0.32 obtained previously) when trained on CNN/DM and tested on CNN/DM the most popular dataset for text summarization research that is quite large.

2.2 Enlarging the Dataset

Based on those unsatisfying results, we then thought it was necessary to find a way to increase the size of the AMI dataset. Thus, we first investigated the GAN approach [4] and more specifically the LeakGAN [5] system which is the state of the art in the domain of GANs for generating texts. Unfortunately, LeakGAN performed badly on the AMI dataset. Thus, we decided to explore another approach consisting of paraphrasing texts. Unfortunately, to build a model for paraphrasing, we need a dataset composed of pairs *(source/target)* where *source* is an original sentence and *target* is a paraphrase, but AMI was not designed for that purpose. To tackle the above problem, we decided to use the ParaNMT-50M dataset [17] to train a model that can paraphrase any sentence into another semantically similar sentence. By using this model on the AMI dataset we have been able to generate new meetings similar to the ones of the AMI dataset.

Table 2 shows a comparison of the ROUGE score of the abstractive bottom-up summarization model built on the AMI dataset and on the AMI + paraphrased AMI dataset.

Table 2. ROUGE scores of the Bottom-up model learned on AMI (col. 2) or AMI+ParaAMI (col. 3) datasets and BertSUM (col. 4)

Metric	AMI	AMI+ParaAMI	BertSUM
ROUGE-1 Recall	0.21788	**0.30832**	0.32509
ROUGE-1 Precision	0.69117	0.49864	0.48588
ROUGE-1 F-measure	0.32103	**0.37215**	0.37622
ROUGE-2 Recall	0.07578	**0.08704**	0.09100
ROUGE-2 Precision	0.25072	0.13995	0.13852
ROUGE-2 F-measure	0.11241	0.10463	0.10684
ROUGE-L Recall	0.20664	**0.28506**	0.29171
ROUGE-L Precision	0.65332	0.46423	0.44001
ROUGE-L F-measure	0.30406	**0.34554**	0.33987

It can be easily seen that there are some significant improvements in terms of recall but obviously the precision decreases. Nevertheless, if we consider the F-measure, we can observe a very significant improvement of this one for ROUGE-1 and ROUGE-L.

2.3 Using Transformers

As text summarization has been recently addressed with models based on transformers such as BERT [2] or an extended version, BertSUM [10] able to generate abstractive text summaries, we decided to use this last one in the context of meeting minutes generation.

As for the use of Bottom-up, we used the AMI+ParaAMI dataset split into training, validation and test sets as previously. Table 2 Column 2 shows the results we obtained on this dataset with BertSUM and we can see that the results obtained with BertSUM are quite similar to those obtained with Bottom-up which is a little bit disappointing given the promises about transformers in the literature.

In fact, as all meetings of the AMI+ParaAMI dataset are very long, we thought that it was difficult for BertSUM (as well as for Bottom-up) to capture the entire content of each meeting. To overcome this problem and try to improve the results provided by BertSUM, we decided to preprocess each meeting of the AMI+paraAMI dataset to select the more relevant sentences given its associated summary. So first, we split each meeting in the training set of AMI+paraAMI into sentences, tokenized them and learned their representation vector using the BERT model. For each meeting summary, we also learned a representation vector using BERT. Then, for each meeting of the training set, we compared the representation vector of each sentence with the representation vector of its associated summary using the cosine similarity measure. Finally, we kept any sentence that had a cosine similarity value greater than a given threshold. After this preprocessing, we ran the BertSUM system on these new training sets. Table 3 presents the results obtained with AMI and ParaAMI preprocessed with the most significant values of the threshold, that is 0.7, 0.75 and 0.8. The column labelled BertSUM recalls the results of BertSUM applied on the AMI+ParaAMI dataset without any preprocessing while the three columns on its right show the results of BertSUM applied on the AMI+ParaAMI dataset after the preprocessing step where the threshold has been fixed respectively to 0.7, 0.75 and 0.8.

Table 3. ROUGE scores for the BertSUM and improved BertSUM models, trained and tested on AMI+ParaAMI

Metric/Model	BertSUM	BertSUM (0.7)	BertSUM (0.75)	BertSUM (0.8)
ROUGE-1 Average_R	0.32509	0.30876	**0.35090**	0.32499
ROUGE-1 Average_P	0.48588	**0.51284**	0.50482	0.50407
ROUGE-1 Average_F	0.37622	0.37579	**0.40448**	0.38385
ROUGE-2 Average_R	0.09100	0.09331	**0.10237**	0.09682
ROUGE-2 Average_P	0.13852	**0.15712**	0.14665	0.14280
ROUGE-2 Average_F	0.10684	0.11425	**0.11737**	0.11081
ROUGE-L Average_R	0.29171	0.27736	**0.31751**	0.29574
ROUGE-L Average_P	0.44001	**0.46260**	0.45962	0.45628
ROUGE-L Average_F	0.33987	0.33801	**0.36729**	0.34840

As we can see, the best values in terms of the recall and F1 ROUGE measures are obtained with a threshold equals to 0.75. In that context, the BertSUM system obtains a value of F1 measure improved by 10% compared to the results obtained without any preprocessing of the AMI+ParaAMI dataset (0.40448 versus 0.37622 for ROUGE-1 for example). Moreover, when we read the content of the summaries that are generated, we can observe that the information provided is more accurate and the sentences are more grammatically correct.

3 Conclusion

A tool that can generate meeting minutes would be invaluable to companies today because of the importance of meetings in their daily lives and the human cost that this task can have if done carefully. This is why we tried to build such a tool using state of the art technologies. We hope we have convinced the reader that the task is not easy but we have provided guidance on what kind of approach can work and what kind of limitations such state of the art tools may have. We think it is clear that this research field still needs a great amount of work before we can provide an operational tool.

References

1. Carletta, J., et al.: The AMI meeting corpus: a pre-announcement. In: Renals, S., Bengio, S. (eds.) MLMI 2005. LNCS, vol. 3869, pp. 28–39. Springer, Heidelberg (2006). https://doi.org/10.1007/11677482_3
2. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019)
3. Gehrmann, S., Deng, Y., Rush, A.M.: Bottom-up abstractive summarization. In: Proceedings of the Conference on Empirical Methods in Natural, pp. 4098–4109 (2018)
4. Goodfellow, I.J., et al.: Generative adversarial nets. In: Proceedings of the Conference on Neural Information Processing Systems, pp. 2672–2680 (2014)
5. Guo, J., Lu, S., Cai, H., Zhang, W., Yu, Y., Wang, J.: Long text generation via adversarial training with leaked information. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, pp. 5141–5148. AAAI Press (2018)
6. Gupta, S., Gupta, S.K.: Abstractive summarization: an overview of the state of the art. Expert Syst. Appl. **121**, 49–65 (2019)
7. Hermann, K.M., et al.: Teaching machines to read and comprehend. In: Proceedings of the Conference on Neural Information Processing Systems, pp. 1693–1701 (2015)
8. Jones, K.S.: Automatic summarising: the state of the art. Inf. Process. Manag. **43**(6), 1449–1481 (2007)
9. Lin, C., Hovy, E.H.: Automatic evaluation of summaries using N-gram co-occurrence statistics. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 150–157. ACL (2003)
10. Liu, Y., Lapata, M.: Text summarization with pretrained encoders. In: Proceedings of the Conference on Empirical Methods in Natural, pp. 3728–3738 (2019)
11. Lloret, E., Palomar, M.: Text summarisation in progress: a literature review. Artif. Intell. Rev. **37**(1), 1–41 (2012)
12. Nallapati, R., Zhou, B., dos Santos, C.N., Gülçehre, Ç., Xiang, B.: Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: Proceedings of the Conference on Computational Natural Language Learning, pp. 280–290 (2016)
13. Napoles, C., Gormley, M.R., Durme, B.V.: Annotated Gigaword. In: Workshop Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, pp. 95–100 (2012)

14. Over, P., Dang, H., Harman, D.: DUC in context. Inf. Process. Manag. **43**(6), 1506–1520 (2007)
15. Saggion, H., Poibeau, T.: Automatic text summarization: past, present and future. In: Poibeau, T., Saggion, H., Piskorski, J., Yangarber, R. (eds.) Multi-source, Multilingual Information Extraction and Summarization, pp. 3–21. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-28569-1_1
16. See, A., Liu, P., Manning, C.: Get to the point: summarization with pointer-generator networks. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 1073–1083 (2017)
17. Wieting, J., Gimpel, K.: ParaNMT-50M: pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pp. 451–462 (2018)