Learning to Rank Utterances for Query-Focused Meeting Summarization

Xingxian Liu, Yajing Xu[*]
Pattern Recognition & Intelligent System Laboratory
Beijing University of Posts and Telecommunications, Beijing, China
{liuxingxian,xyj}@bupt.edu.cn

Abstract

Query-focused meeting summarization(QFMS) aims to generate a specific summary for the given query according to the meeting transcripts. Due to the conflict between long meetings and limited input size, previous works mainly adopt extract-then-summarize methods, which use extractors to simulate binary labels or ROUGE scores to extract utterances related to the query and then generate a summary. However, the previous approach fails to fully use the comparison between utterances. To the extractor, comparison orders are more important than specific scores. In this paper, we propose a **Ranker-Generator** framework. It learns to rank the utterances by comparing them in pairs and learning from the global orders, then uses top utterances as the generator's input. We show that learning to rank utterances helps to select utterances related to the query effectively, and the summarizer can benefit from it. Experimental results on QMSum show that the proposed model outperforms all existing multi-stage models with fewer parameters.

1 Introduction

Query-focused meeting summarization(QFMS) aims to summarize the crucial information for the given query into a concise passage according to the meeting transcripts. By responding to the query, QFMS can meet the user's need to focus on a specific aspect or topic of the meeting (Litvak and Vanetik, 2017; Baumel et al., 2018). Unlike the generic summary, QFMS requires the summary depending on both the given query and meeting transcripts.

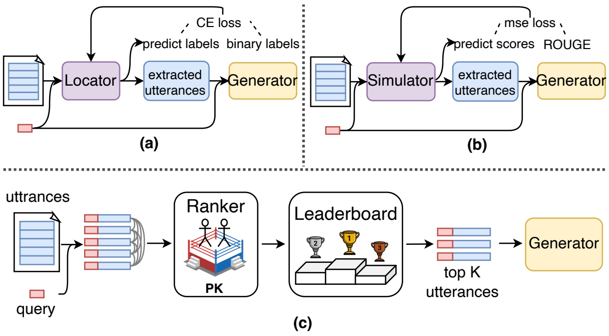

Figure 1: (a) **Locator-Generator** framework, it predicts a binary label and uses Cross-Entropy loss to update parameters. (b) **Simulator-Generator** framework, it simulates the ROUGE score and uses Mean Squared Error loss to update parameters. (c) **Ranker-Generator** framework proposed in this paper, it learns to rank utterances from the relative order between utterances. The top K utterances can be passed to the generator.

Previous works consist of end-to-end and two-stage frameworks. The end-to-end models take the whole long meeting as the input. Although some works such as HMNet (Zhu et al., 2020) and HATBART (Rohde et al., 2021) use hierarchical attention mechanism to alleviate the rapid growth in computational complexity, it's still faced with difficulties in training efficiency. The two-stage models extract utterances related to the query and then pass the concatenation of them to the generator. For QFMS, the key information related to the query scatters in certain parts of the meeting. Therefore, the two-stage framework is considered as a practical approach to balance experimental performance and computational efficiency in the long-input problems.

The two-stage framework mainly includes the Locator-Generator and the Simulator-Generator approaches. As shown in Figure 1, in the first stage, the Locator-Generator (Zhong et al., 2021b) framework considers it as a binary classification task. It predicts a binary label of whether the utterance is relevant to the query and uses cross-entropy loss to update parameters. But the hard binary labels can not reflect the relative quality. Especially when the training data is limited by scarcity, the binary classification will have a large margin between positive and negative samples. So the Simulator-Generator (Vig et al., 2022) framework considers

[*] Yajing Xu is the corresponding author.

Findings of the Association for Computational Linguistics: ACL 2023, pages 8496–8505
July 9-14, 2023 ©2023 Association for Computational Linguistics

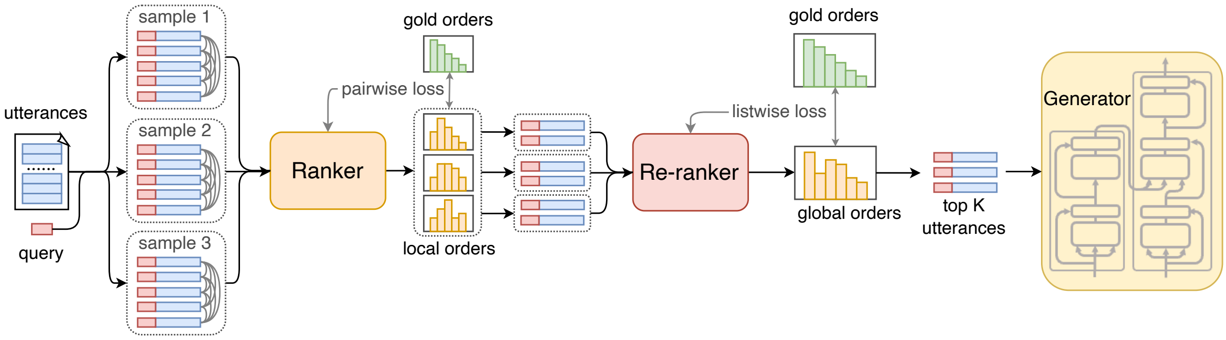

Figure 2: The overall model structure

it as a ROUGE score regression task. It simulates the ROUGE score and uses MSE loss to update parameters. However, there is a gap between the extractor's ultimate objective and the objective of minimizing the absolute error between predicted scores and ROUGE scores. In fact, rather than specific scores, we care more about the relative orders of utterances.

To make full use of the comparison information between samples, we propose a Ranker-Generator framework in this paper. To balance experimental effectiveness and computational efficiency, the framework contains three steps. First, the utterances would be divided into samples. We conduct pairwise ranking to get an order for each sample. Second, the top utterances in different samples would be fed into the re-ranker, which would conduct listwise ranking to get a global order. Finally, the top K utterances would be concatenated and passed to the generator.

To summarize, our contributions are as follows: (1) This paper demonstrates that, by enhancing the accuracy of extracting query-relevant utterances, the generator can make the summary more related to the query. (2) We propose a Ranker-Generator framework to extract query-relevant utterances by learning to rank discourse to improve the quality of the generated summaries. (3) Experimental results show that the proposed model outperforms existing multi-stage models with fewer model parameters.

2 Method

The architecture of our method is illustrated in Figure 2. Our model consists of a two-stage ranking step and a generating step. The utterances would be ranked by the Sample Pairwise Ranking module and the Global Listwise Re-ranking module, and top of them can be passed to the generator to produce the final summary.

2.1 Two-Stage Ranking

The utterance ranking orders for a brief meeting can be efficiently obtained using the single-stage ranking paradigm. However, the computing complexity of full-pairwise ranking grows at a square rate as the number of utterances grows. Therefore, we adopt a two-stage ranking framework. In the first stage, we propose sample pairwise ranking to reduce computational complexity. But sample pairwise ranking can only evaluate the relative quality within samples. It performs poorly when applied to utterances from various samples, e.g., the top utterances in sample 1 may be ranked lower in sample 2. To overcome the above problem, we apply global listwise re-ranking and concentrate on the top-k utterances in the second stage. Utterances that are unlikely to appear in the generator are filtered out by the pairwise ranking model, then global listwise ranking is conducted to get better top-k orders.

2.2 Sample Pairwise Ranking

In this paper, the ROUGE (Lin, 2004) scores between utterances U and the gold summary S^* are considered as the measure of query-relevance. The utterances from one meeting are divided into various samples. In one sample, the utterances would be ordered by the ROUGE scores. The ranker should be encouraged to assign higher relevance scores to these top utterances in the order. By learning to rank in pairwise, the model can distinguish the utterances that are more relevant to the query from the comparison. Following the previous work (Zhong et al., 2020), the loss is as follows:

$$L = \sum_{i} \sum_{j>i} max(0, f(U_j) - f(U_i) + \lambda_{ij}) \quad (1)$$

$$\lambda_{ij} = (j - i) * \lambda \quad (2)$$

where U_i and U_j are the i-th and j-th utterances in gold ranking orders, ROUGE(U_i, S^*)>ROUGE(U_j, S^*), $\forall i, j, i < j$, λ is the base margin. $f(U_i)$ is the predicted query-relevance score given by a cross-encoder model.

2.3 Global Listwise Re-ranking

As shown in Figure 2, the top utterances in different samples are gathered in the re-ranking module. The gold orders would be determined by ranking the utterances according to the ROUGE scores. To obtain a more precise top-ranking order, we would perform a refined global sort on these top utterances from various samples using listwise re-ranking. Inspired by ListNet (Cao et al., 2007), we optimize the permutation probability distribution between predicted scores s and the gold scores s^*. The permutation probability is defined as

$$P_s(\pi) = \prod_{j=1}^{n} \frac{\phi(s_{\pi(j)})}{\sum_{t=j}^{n} \phi(s_{\pi(t)})} \tag{3}$$

π is a permutation on the n objects, and $\phi(.)$ is an increasing and strictly positive function.

But different with ListNet, we optimize the top-k permutation probability rather than top-1 probability. The top-k permutation probability is as follows:

$$P_s^k(\pi) = \prod_{j=1}^{k} \frac{\phi(s_{\pi(j)})}{\sum_{t=j}^{n} \phi(s_{\pi(t)})} \tag{4}$$

For example, the top-3 permutation probability of $\pi = \langle 1, 2, 3, 4, 5 \rangle$ is as follows:

$$P_s^3(\pi) = \frac{\phi(s_1)}{\sum_{i=1}^{5} \phi(s_i)} \cdot \frac{\phi(s_2)}{\sum_{i=2}^{5} \phi(s_i)} \cdot \frac{\phi(s_3)}{\sum_{i=3}^{5} \phi(s_i)} \tag{5}$$

The predicted top1-to-topk distribution is $P_s = (P_s^1, P_s^2, \cdots, P_s^k)$, the gold top1-to-topk distribution is $P_{s^*} = (P_{s^*}^1, P_{s^*}^2, \cdots, P_{s^*}^k)$ We use KL-divergence to reduce the gap between the above two distributions.

$$L = KL(P_{s^*} || P_s) \tag{6}$$

$$KL(P_{s^*} || P_s) = \sum_{i=1}^{k} P_{s^*}^i \cdot \log \frac{P_{s^*}^i}{P_s^i} \tag{7}$$

2.4 Generator

As shown in Figure 2, after the two-stage ranking, top-k of the utterances would be concatenated and fed into the generator. In the generation stage, the objective is to minimize the cross-entropy loss:

$$L = -\sum_i p_{gt}(S_i | S_{<i}^*, U) \log p(S_i | S_{<i}^*, U) \tag{8}$$

$$p_{gt}(S_i | S_{<i}^*, U) = \begin{cases} 1 & S_i = S_i^* \\ 0 & S_i \neq S_i^* \end{cases} \tag{9}$$

U is the generator's input, S^* is the gold summary.

3 Experiments

3.1 Setup

3.1.1 Implementation Details

Models are implemented using the PyTorch framework. The pre-trained BART[*] from the Transformers (Wolf et al., 2020) library is used as the base abstractive model. The pre-trained MiniLM[†] from the sentence-transformers (Reimers and Gurevych, 2019) library is used as the pairwise ranking model and the listwise re-ranking model.

All experiments are conducted on NVIDIA RTX 3090 GPU(24G memory). The generator model is trained for 10 epochs. For one model training, the average running time is around 2 hours. Weight hyperparameter λ is 0.01 in Equation 2. The generator's max length of the input is 1024, max length of the output is 256. Learning rate is 5e-6.

Models were evaluated using the ROUGE metrics (Lin, 2004) in the SummEval toolkit (Fabbri et al., 2021) and each pair of results was subjected to t-test to confirm the effectiveness of our method.

3.1.2 Datasets Details

QMSum (Zhong et al., 2021b) is a query-focused meeting summarization dataset consisting of 1,808 query-summary pairs over 232 meetings from product design, academic, and political committee meetings. Additionally, QMSum contains manual annotations such as topic segmentation and relevant spans related to the reference summary.

3.1.3 Baselines Details

We compare the proposed method with several baselines. **TextRank** (Mihalcea and Tarau, 2004) is an extractive summarization method with a graph-based ranking model. **PGNet** (See et al., 2017) uses pointer mechanism to copy tokens from source texts. **BART** (Lewis et al., 2020) is a pre-trained encoder-decoder Transformer model with a denoising objective, which achieves advanced performance on several summarization datasets(i.e. CNN/DailyMail (Hermann et al., 2015) and Xsum

[*] The checkpoint is "facebook/bart-large", containing around 400M parameters.

[†] The checkpoint is "cross-encoder/ms-marco-MiniLM-L-12-v2", containing around 134M parameters.

Models	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**Extractor Size(M)**
TextRank (Mihalcea and Tarau, 2004)	16.27	2.69	15.41	-
PGNet (See et al., 2017)	28.74	5.98	25.13	-
BART (Lewis et al., 2020)	29.20	6.37	25.49	-
LEAD + BART	32.06	9.67	27.93	-
HMNet (Zhu et al., 2020)	32.29	8.67	28.17	-
Longformer (Beltagy et al., 2020)	34.18	10.32	29.95	-
DialogLM (Zhong et al., 2021a)	33.69	9.32	30.01	-
SUMM^{N} (Zhang et al., 2022)	34.03	9.28	29.48	-
DYLE (Mao et al., 2022)	34.42	9.71	30.10	501
Pointer Network + PGNet (Zhong et al., 2021b)	31.37	8.47	27.08	440
Pointer Network + BART (Zhong et al., 2021b)	31.74	8.53	28.21	440
RELREG-TT (Vig et al., 2022)	33.02	10.17	28.90	329
RELREG (Vig et al., 2022)	34.91	11.91	30.73	1372
Oracle	43.80	19.63	39.10	
Locator-Generator	31.47(-3.77)	8.53(-3.70)	28.21(-3.07)	134
Simulator-Generator	32.92(-2.59)	9.46(-2.77)	28.93(-2.35)	134
Ranker-Generator	**35.51**	**12.23**	**31.28**	134
RankSUM(w/o re-ranking)	33.02(-2.49)	9.73(-2.50)	29.15(-2.13)	134

Table 1: ROUGE-F1 scores for different models on QMSum dataset.

Models	**Top 5**			**Top 10**		
	R-1	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**
Gold	26.32	7.58	24.43	20.55	5.15	19.29
LEAD	11.15	0.99	10.17	12.11	1.11	11.10
RELREG	18.02	2.46	15.30	15.02	2.35	13.23
Locator	16.89	2.24	13.97	14.10	1.97	12.75
Simulator	17.06	2.36	14.88	14.44	2.14	13.06
Ours	**20.07**	**3.69**	**17.78**	**17.08**	**3.01**	**15.48**

Table 2: ROUGE-F1 scores between the gold summary and top-5/top-10 utterances for different models on QMSum.

(Narayan et al., 2018)). **LEAD+BART** uses the beginning utterances as the BART's input. **HMNet** (Zhu et al., 2020) uses a hierarchical attention mechanism and cross-domain pre-training for meeting summarization. **Longformer** (Beltagy et al., 2020) replaces the quadratic self-attention mechanism with a combination of local attention and sparse global attention. **DialogLM** (Zhong et al., 2021a) is a pre-train model using intra-window denoising self-reconstruction pre-training task and intra-block inter-block mixing attention. **SUMM^{N}** (Zhang et al., 2022) is a multi-stage summarization framework for the long-input summarization task. **DYLE** (Mao et al., 2022) treats the extracted text snippets as the latent variable and jointly trains the extractor and the generator. **Point Network+PGNet** and **Point Network+BART** (Zhong et al., 2021b) adopt a two-stage approach of locate-then-summarize for long meeting summarization. **RELREG-TT** (Vig et al., 2022) and **RELREG** (Vig et al., 2022) considers extracting as a ROUGE regression model using bi-encoder and cross-encoder.

3.2 Results & Analysis

The ROUGE score (Lin, 2004) is adopted as the evaluation metric. The performances of our method and baselines are summarized in Table 1. Experimental results show that our method significantly outperforms the baselines ($p < 0.05$) on QMSum dataset with fewer parameters.

To have a fair comparison among the three frameworks, we design an experiment to evaluate the performance of these frameworks using the same backbone as the extractor and the same generator. The experimental results show that the proposed model significantly outperforms Locator-Generator and Simulator-Generator, which demonstrates that the ranker can obtain meeting utterances that are more suitable for the generator by learning to rank utterances.

To verify the effectiveness of the two-stage ranking paradigm, we conduct an ablation experiment. Our model significantly outperforms the model without re-ranking module ($p < 0.05$). Experimental results show that the model without re-ranking module reduces 2.49 ROUGE-1, 2.50 ROUGE-2, 2.13 ROUGE-L scores, which demonstrates the importance of the re-ranking module. By listwise ranking, we can get a more precise top-ranking order.

We have an interesting observation. Unlike the ROUGE score regression model, the ranker is less sensitive to the model size. We believe this is because learning the relative order by comparison is easier than fitting ROUGE scores separately. It reduces the ranker's reliance on the model size by

Models	**Flu.**	**QR.**	**FC.**
Gold	4.88	4.90	4.92
BART	4.48	3.78	3.64
RELREG	4.51	4.12	4.07
Locator-Generator	4.45	3.90	3.83
Simulator-Generator	4.48	4.01	4.02
Ours	**4.52**	**4.40**	**4.21**

Table 3: Human evaluation on Fluency (Flu.), Query Relevance (QR.) and Factual Consistency (FC.) for QMSum.

making full use of the comparison between samples. As a training task for extractors, learning to rank is a more suitable objective. Since to the extractor, it is the relative order that matters rather than the absolute error in fitting the ROUGE score.

3.3 Extractor Performance

We conduct experiments to evaluate the performance of the extractor, which help to explore the impact of the extractor on the quality of the generated summaries. The lexical overlap metric between the extracted utterances and the gold summary is used to measure the relevance of the meeting utterances to the summary/query. The experimental results show that the ranker significantly outperforms the baselines in extracting relevant utterances. It demonstrates that by learning to rank utterances, the ranker is able to extract the utterances that are more relevant to the summary/query.

3.4 Human Evaluation

We further conduct a manual evaluation to assess the models. We randomly select 50 samples from QMSum and ask 5 professional linguistic evaluators to score the ground truth and summaries generated by 5 models according to 3 metrics: fluency, query relevance and factual consistency. Each metric is rated from 1 (worst) to 5 (best) and the scores for each summary are averaged.

As shown in Table 3, the proposed model significantly outperforms all the baselines on query relevance, which benefits from the extractor's improvement on selecting the relevant utterances. Besides, the factual consistency score is also improved. We think that by comparing the relevance between utterances and the summary/query, the top utterances are more relevant to each other, which may help to improve factual consistency. In the aspect of fluency, the proposed model has only slight improvement compared to the baselines.

4 Conclusion

This paper proposes a new multi-stage framework for QFMS. It learns to rank the meeting utterances by pairwise and listwise comparison between them. By selecting the utterances with high query-relevance scores as the generator's input, the generator can produce high-quality summaries that are more relevant to the query. The experiments demonstrate the effectiveness of the Ranker-Generator framework.

5 Acknowledgements

This work was supported by MoE-CMCC ”Artifical Intelligence” Project No. MCM20190701 and the National Natural Science Foundation of China (NSFC No.62076031).

We thank the anonymous reviewers for valuable feedback and helpful suggestions.

Limitations

This paper mainly focuses on the Query-focused Meeting Summarization(QFMS) task. Besides, We have explored the performance of the Ranker-Generator framework on the long-input summarization task. But the results do not show a significant improvement. Although QMSum dataset is also faced with the long-input challenge, the QFMS task only summarizes specific parts of the original text, so it can take these parts as the input. While the goal of the long-input summarization task is to generate an overall summary, which needs to have a global view on the original text. So we think the extract-then-generate framework is unsuitable for the long-input summarization task. The previous work SUMM^N (Zhang et al., 2022) is more suitable for the long-input summarization task.

In addition, the multi-stage approach has a performance disadvantage over the end-to-end approach. However, the computational complexity of the multi-stage approach is much lower than that of the end-to-end approach. The multi-stage approach can balance experimental performance and computational complexity. So it is worthy of exploration as well as the end-to-end approach.

Ethics Statement

In this paper, all experiments are conducted on **QMSum** (Zhong et al., 2021b), which is open-source and obeys MIT license. The meeting transcripts data doesn't contain any privacy information(such as password, phone number and trade secrets) or offensive content.

References

Tal Baumel, Matan Eyal, and Michael Elhadad. 2018. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *International Conference on Machine Learning*.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Marina Litvak and Natalia Vanetik. 2017. Query-based summarization using MDL principle. In *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, pages 22–31, Valencia, Spain. Association for Computational Linguistics.

Ziming Mao, Chen Henry Wu, Ansong Ni, Yusen Zhang, Rui Zhang, Tao Yu, Budhaditya Deb, Chenguang Zhu, Ahmed Awadallah, and Dragomir Radev. 2022. DYLE: Dynamic latent extraction for abstractive long-input summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1687–1698, Dublin, Ireland. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Tobias Rohde, Xiaoxia Wu, and Yinhan Liu. 2021. Hierarchical learning for generation with long source sequences. *ArXiv*, abs/2104.07545.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Jesse Vig, Alexander Fabbri, Wojciech Kryscinski, Chien-Sheng Wu, and Wenhao Liu. 2022. Exploring neural models for query-focused summarization. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1455–1468, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^n: A multi-stage summarization framework for long input dialogues and documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, Online. Association for Computational Linguistics.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021a. Dialoglm: Pre-trained model for long dialogue understanding and summarization. In *AAAI Conference on Artificial Intelligence*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021b. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 194–203, Online. Association for Computational Linguistics.

A Case Study

Query: Why did the industrial designer disagree with the marketing when talking about some new ideas about future products?

Gold: The marketing thought they should not trust too much the trend because it wouldn't last long - fruits and vegetables were in fashion this year but maybe next year it would be insects. However, the industrial designer believed it was good to follow the flow now and then since electronic devices were not long life products: if people changed, the product appearance should change as well.

Previous: Industrial Designer thought that it would be interesting to see if people would buy this one, and maybe add some features to it after. However, he thought it would not be very productive to go through all the process from the beginning to the end.

Ours: The industrial designer thought it was good to follow the flow and make it now and after, if the people change their mind, you change also the product.

🟨 About future trend

Query: What did Barry Hughes think about the targets regarding performance measure?

Gold: Barry Hughes stated that they did not have targets and only prosecuted with reasonable prospects. As for the police, there were no targets for the police anymore, except for some individuals.

Previous: Barry Hughes thought that there was a shortage of registered intermediaries in Wales, and the Ministry of Justice had taken action to deal with that, and they had had a number of people who were now in a position to act as intermediaries. Now, of course, if they were to decide not to do that anymore, they may have a problem, but, in turn, they would be looking to recruit more people into those positions. So, it had the potential to serve as a barrier, but in practice, he didn't think it would be a barrier.

Ours: Barry Hughes thought that the police were much more sensible these days than they may have been 20 years ago, in terms of trying to get cases charged in order to meet some notional target. In the CPS, they did not have targets for securing convictions or not. Clearly, they prosecuted if they thought that there was a reasonable prospect, a realistic prospect, of conviction and it was in the public interest.

🟨 About the performance measure

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
The Limitations section.

☑ A2. Did you discuss any potential risks of your work?
The Ethics Statement section.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
The Abstract and Section 1.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Yes, the References section.

☑ B1. Did you cite the creators of artifacts you used?
Yes, the References section.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
The Ethics Statement section.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
The Limitations section.

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
The Ethics Statement section.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix B.

C ☑ Did you run computational experiments?

Section 3.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix A.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix A.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A.

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Section 3.4.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Section 3.4.

☑ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
The Ethics Statement section.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.