JRIRD at the NTCIR-15 QA Lab-PoliInfo-2 Task: An Abstractive Dialog Summarization System for Japanese Assembly Minutes

Kazuma Kadowaki
The Japan Research Institute, Limited
kadowaki.kazuma@jri.co.jp

ABSTRACT

The JRIRD team participated in the Dialog Summarization subtask of the NTCIR-15 QA Lab-PoliInfo-2 task. This paper describes my approach for the topic-aware summarization of assembly member speeches. The system consists of three modules: (1) a pre-processor that retrieves speeches from minutes, (2) a BERT-based sentence extractor that extracts candidate sentences by predicting the topic-aware importance of each sentence in a speech without annotations, and (3) a UniLM-based summary generator that generates a summary from the extracted sentences while controlling the length of the summary. Results show that my system achieved an outstanding performance among all of the participants in the task, both in the evaluation using ROUGE scores and in human evaluations.

TEAM NAME

JRIRD

SUBTASKS

Dialog Summarization

1 INTRODUCTION

Generating a short summary from long minutes plays an important role in helping the fact-checking of speaker utterances as well as understanding of their policies. The Dialog Summarization subtask of the NTCIR-15 QA Lab-PoliInfo-2 task [5] (hereinafter referred to as "PoliInfo-2") aims to automatically generate a short summary from the Tokyo Metropolitan Assembly minutes. In this subtask, given the assembly minutes and one of the topics discussed in a speech, participants of the subtask were asked to generate a topic-aware summary corresponding to the speech, keeping in mind that each speech mentions several topics.

Ogawa et al. [10], being inspired by the previous NTCIR-14 QA Lab-PoliInfo tasks [6] (hereinafter referred to as "PoliInfo"), proposed an approach that first used a rule-based segmentation method to extract a paragraph that mentioned the given topic, and then used decision trees to reduce unnecessary words and to produce a short summary. However, in contrast to the previous PoliInfo Segmentation subtask, reference summaries were not given in the PoliInfo-2 Dialog Summarization subtask, making it rather challenging to extract from the entire speech an appropriate paragraph for a given topic.

Hiai et al. [4], using another approach, extracted important sentences without importance scores attached to each sentence, by using support vector regression that predicted similarities between a summary and each of the sentences in a speech. However, their scope was limited to extraction from a segmented paragraph, so their approach could not be simply applied to the current PoliInfo-2 Dialog Summarization subtask, where multiple topics may be mentioned in a speech.

I extended this idea of extracting important sentences related to a topic, and tackled the PoliInfo-2 Dialog Summarization subtask [5] as a member of the JRIRD team. My approach uses a regression model that predicts a topic-aware importance score. It eliminates not only the need to annotate the importance of each sentence, but also the need to split the text of a speech into segments taking their topics into account. This also enables us to achieve a better performance by training our models using a large quantity of the minutes and reference summaries without the effort of further annotations such as segmentation, extraction, or allocating importance scores to each sentence. My model makes use of a strong pre-trained language model, BERT [2].

I then generate summaries from the extracted sentences. I use another strong pre-trained language model, UniLM [3]. I propose modifications for this abstractive generation method to control the length of a generated summary within a given desired length while containing a sufficient amount of information.

As a result, my models achieved the best performance among all the participants in the subtask, both in the human evaluations of all five metrics and in the ROUGE-based automatic evaluations.

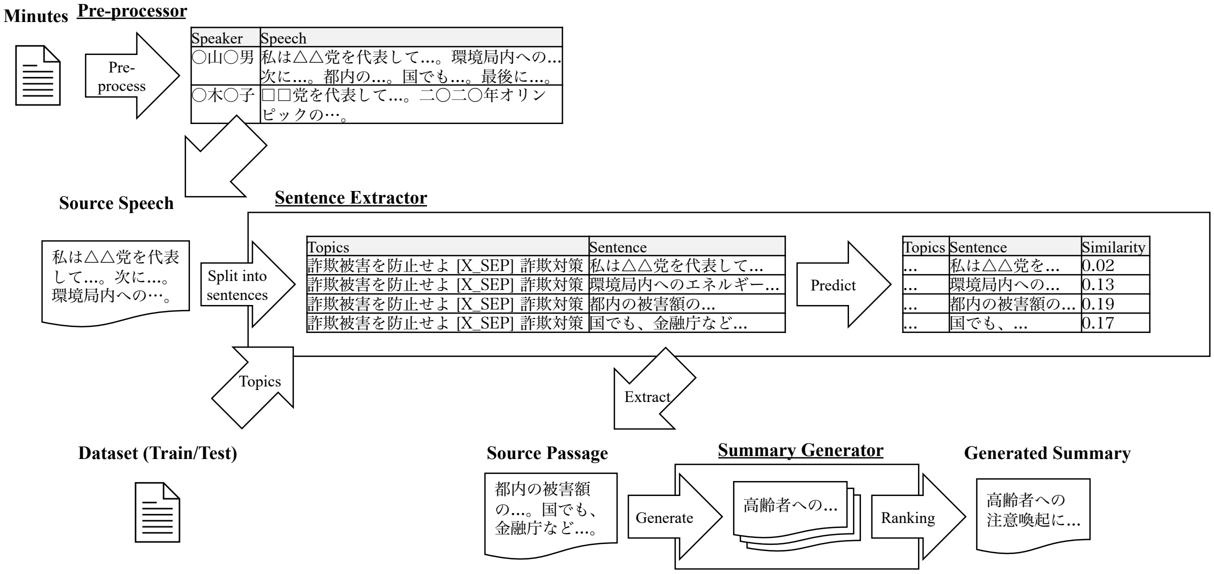
In this paper, I describe my approach in the Dialog Summarization subtask and its results. We also discuss the performances and limitations of our settings through additional experiments. The reminder of this paper is organized as follows: Section 2 describes my approach. Section 3 explains the details of my implementations. The results of the formal run are described in Section 4. In Section 5, we discuss the limitations of our approach and future research directions. Finally, Section 6 concludes my paper.

2 MY SYSTEM

The overview of my system is shown in Figure 1. My system generates summaries from the Tokyo Metropolitan Assembly minutes, and consists of three modules: a pre-processor, a BERT-based sentence extractor, and a UniLM-based summary generator. In this section, I briefly outline the task settings and then describe each of three modules.

2.1 Task Settings

The outlines of the Dialog Summarization subtask are described in the task overview paper [5]. In this section, I only describe the details of the settings.



Speaker	Speech
○山○男	私は△△党を代表して…。環境局内への…次に…。都内の…。国でも…。最後に…。
○木○子	□□党を代表して…。二○二○年オリンピックの…。

Topics	Sentence
詐欺被害を防止せよ [X_SEP] 詐欺対策	私は△△党を代表して…
詐欺被害を防止せよ [X_SEP] 詐欺対策	環境局内へのエネルギー…
詐欺被害を防止せよ [X_SEP] 詐欺対策	都内の被害額の…
詐欺被害を防止せよ [X_SEP] 詐欺対策	国でも、金融庁など…

Topics	Sentence	Similarity
…	私は△△党を…	0.02
…	環境局内への…	0.13
…	都内の被害額の…	0.19
…	国でも、…	0.17

Figure 1: Overview of my system.

Assembly minutes are transcripts of all the utterances made in a *meeting*. A questioner, a member of the assembly, makes a *speech*[1], asking several *questions* on various topics. After his or her questions are all posed, the Governor and superintendents make their speeches to answer the questions in turn, after which a next questioner starts his/her speech. Unlike in other prefectural assemblies in Japan, each questioner in the Tokyo Metropolitan Assembly makes only one speech in a meeting, and that speech may contain multiple topics and questions. The assembly minutes are, however, not segmented by (sub-)topics nor annotated, hence they contain no metadata other than the speakers' names and the speeches themselves.

A *newsletter* (called *Togikai-dayori*) is published after each meeting. It contains the speakers' names, main- and sub-topics, and summaries (but not the speeches themselves). To compose a newsletter, each questioner's speech is segmented so that all of the sentences in a segment are about a *sub-topic*, which is expressed in a few words. Responses that correspond to questioners' speeches are also segmented using the same (i.e., questions') sub-topics. A *summary* is written for each of the segments (both for questions and answers) and contains about 50 characters or a few sentences. In addition, each questioner's speech is abstracted into a pair of two short sentences, called a *main topic*, although it does not contain the complete opinion of the questioner.

The goal of the Dialog Summarization subtask was to automatically generate such a summary, given a main topic, a sub-topic, and an entire speech in the assembly minutes made by the same speaker in the meeting. In the PoliInfo-2 task, participants were also given a desired number of characters for each input, and the generated summary was required to be of this length or shorter. I mainly used the datasets prepared by the task organizers [5, 7], and the generated summaries were evaluated using ROUGE-1 [9] recall of their content words[2], as well as human evaluations. The task overview paper [5] outlines the detailed format and an example of the dataset. Section 3.2 outlines details of our datasets.

2.2 Pre-process

The first module, the pre-processor, takes a summary (or its placeholder for the test dataset) as an input, and retrieves the entire speech made by each of the speakers who mention a sub-topic in the meeting.

I first developed a person-role map for each meeting. This was necessary as those responding are distinguished by their names (e.g., 石原慎太郎, Shintaro Ishihara) in the assembly minutes, but by their roles (e.g., 知事, the Governor) in the newsletter. I extracted the mapping from remarks (e.g., 知事石原慎太郎君登壇, Governor Shintaro Ishihara appeared on the platform) in the assembly minutes[3].

I then split a meeting into sub-meetings, so that each sub-meeting contained one questioner's speech and at least one response. Moreover, all of the speeches made after a questioner appeared on the platform and before either the meeting was closed, or another questioner appeared on the platform, were considered to be in the same sub-meeting. Such a split helped me to identify a questioner's speech that corresponded to a response, as responders (like the Governor) often make several speeches in a meeting.

Then, for each sub-topic in the newsletter, given a list of its questioner and respondents as well as person-role maps, I chose

[1] Hereinafter, we assume that a *speech* is represented in text.

[2] The summaries are tokenized using MeCab-UniDic [1] for this automatic evaluation.

[3] For a few exceptional examples where the roles are abbreviated in the newsletter, I also mapped each of the roles to a role in the assembly minutes that contains all of its characters. For example, "オリンピック・パラリンピック準備局長" (Director General of Bureau of Olympic and Paralympic Games Tokyo 2020 Preparation), a role in a remark in the assembly minutes, contains all the characters in "オリパラ局長" (Director General of Bureau of Oly-Para), an abbreviated role in the newsletter, and I added this pair to the mapping as well.

a sub-meeting and a set of corresponding speeches. Here, the sub-topic and speeches exhibit a one-to-many correspondence, as we do not yet take the sub-topic itself into account, only the speakers. We call each of the speeches in this set a *source speech* hereafter, and assume that each summary can somehow be generated from a source speech (i.e., without any external knowledge base).

2.3 Sentence Extraction

The second module, the sentence extractor, reduces the entire source speech (35.80 sentences or 4,895.59 characters long on average) into a few sentences, taking its sub-topic into account. We call a set of extracted sentences a *source passage* and assume that only this passage in the speech is essential to generate a summary.

We also assume that a summary and its source passage are lexically similar. In other words, given a summary, the higher the ROUGE score [9] between the summary and a sentence in the speech, the more likely the sentence to be a part of the summary's source passage. Therefore, I calculated the ROUGE-1 F-measure (for surface form) for each of the sentence-and-summary pairs and extracted sentences with higher scores as a source passage. Here, the number of sentences were determined so that the entire passage did not exceed 150 subwords[4]. Also, the sentences in a passage are not necessarily continuous in the speech nor in the assembly minutes.

A challenge here is that, unlike the previous PoliInfo Segmentation subtask [6], a summary is not given for the test dataset in the PoliInfo-2 task, so that a ROUGE-1 score cannot be simply calculated. To deal with this problem, I built a BERT-based [2] regression model to predict the ROUGE-1 score. The model outputs a value in $[0.0, 1.0]$ which we regard as a (predicted) likelihood that the sentence will be a part of its source passage.

The model takes as an input a pair of two segments: a summary's metadata and a sentence from the minutes. The metadata is presented in a sequence of three sentences concatenated with a special out-of-vocabulary separator: two sentences of the questioner's main-topic for that meeting, and the sub-topic. To train the model, as well as for prediction, I used all possible combinations of sub-topics and sentences in the same speech.

Moreover, each summary in the training (segmented) dataset has a `StartingLine` and an `EndingLine` annotated, so that we know that sentences outside this range are not essential to generate the summary. For the training examples with this information available, I regard the ROUGE-1 scores for sentences outside this range to be zero.

2.4 Summary Generation

The last module, the summary generator, abstracts the source passage. It generates an abstractive summary using UniLM [3], a BERT-based model that outputs token sequences. I did not adopt a sentence reduction (i.e., an extractive summarization) approach that often generated unnatural sentences in the previous task [11].

For each of the summaries (or its placeholder for the test dataset), I prepared an input to the model. It had a single segment and was presented in a sequence of concatenated sentences with a special out-of-vocabulary separator: two sentences of a main-topic, a sub-topic, a source passage, and a special out-of-vocabulary token that distinguished whether the summary was a question or an answer.

To control the length of the generated summary, I made some minor modifications to the UniLM implementations. First, all of the examples whose summaries were shorter than 20 subwords were discarded from the training dataset. Second, I forced the model to generate at least 10 subwords during the beam search, by replacing the probabilities that an EOS token was output before that threshold with zero. Third, I used a modified metric to choose a sequence from the results of a beam search, instead of the calculated probability that the sequence was generated. Moreover, I first prioritized all of the hypotheses shorter than the given desired length, and then, chose the hypothesis that had the highest ROUGE-1 recall with regard to the source passage, assuming that all the words in the source passage were more likely to appear in the summary (i.e., the summary was lexically similar to the source passage).

3 IMPLEMENTATION DETAILS

This section explains my model implementation in detail.

3.1 Pre-trained Models

To train my models, I used the NICT BERT Japanese Pre-trained Model[5] (32k vocabulary version) as a starting point, provided to the public by the Data-driven Intelligent System Research Center, National Institute of Information and Communications Technology (NICT). This model was pre-trained for 1.1 million steps (one million steps with a maximum sequence length of 128 and 100 thousand more steps with that a maximum sequence length of 512) with a batch size of 4,096, resulting in 16 times more epochs than other publicly-available models trained using Japanese Wikipedia.

I further pre-trained this model for UniLM [3], with the same corpus (i.e., Japanese Wikipedia) and the same settings, except for a batch size of 256. As a result, my models had the same number of parameters as the $\text{BERT}_{\text{BASE}}$ [2] (i.e., 12 layers, 768 hidden state size, etc.).

3.2 Datasets

To train my models, I used all the training datasets provided by the task organizers [5, 7]: (1) the segmented dataset, whose summary had `StartingLine` and `EndingLine` annotated, created for meetings held in 2011 and 2012, and (2) the unsegmented dataset, without such annotations, created for those meetings in 2014. I used the test dataset provided by the task organizers, which was created for those meetings in 2013.

Additionally, in one of my experiments, I also used (3) my own datasets, without annotations for `StartingLine` and `EndingLine`, that were created for those meetings from 2001–2011 and 2015–2019.

[4] I used byte-pair-encoding [12] and MeCab-Jumandic [8] as I describe later in Section 3.

[5] https://alaginrc.nict.go.jp/nict-bert/index.html

Table 1: Statistics of the datasets.

	Assembly minutes			Newsletters		
Dataset	Questioners	Speeches (Q & A)	Sentences	Sub-topics	Summaries	Date
(1) Train (Segmented)	123	560	20,284	438	993	Jun 2011 – Nov 2012
(2) Train (Unsegmented)	91	363	12,756	325	693	Mar 2014 – Dec 2014
(3) Train (Mine)	1,177	5,089	180,596	4,565	9,878	Sep 2001 – Feb 2011, Feb 2015 – Dec 2019
Test	74	293	10,310	254	533	Feb 2013 – Dec 2013

Table 2: Quality question scores in the formal run.

	Content		Well-	Non-twisted		Sentence	Dialog
	X = 2	X = 0	formed	All	Evaluable	goodness	goodness
ID 185	1.014	0.900	1.830	1.220	1.581	1.042	0.848
ID 189	**1.082**	**0.975**	**1.858**	**1.316**	**1.712**	**1.129**	**0.937**
Baseline [10]	0.748	0.671	1.582	1.011	1.658	0.730	0.488

Table 3: ROUGE scores in the formal run.

		Recall							F-measure						
		N1	N2	N3	N4	L	SU4	W1.2	N1	N2	N3	N4	L	SU4	W1.2
Surface Form	ID 185	0.503	0.221	0.134	0.087	0.415	0.252	0.199	0.373	0.158	0.096	0.061	0.303	0.174	0.193
	ID 189	**0.517**	**0.241**	**0.146**	**0.093**	**0.429**	**0.267**	**0.206**	**0.387**	**0.175**	**0.106**	**0.069**	**0.317**	**0.188**	**0.202**
	Baseline [10]	0.405	0.130	0.076	0.046	0.338	0.169	0.160	0.308	0.099	0.058	0.036	0.253	0.123	0.159
Stem	ID 185	0.511	0.224	0.137	0.091	0.421	0.258	0.202	0.379	0.161	0.098	0.064	0.308	0.178	0.196
	ID 189	**0.526**	**0.247**	**0.152**	**0.098**	**0.437**	**0.277**	**0.210**	**0.394**	**0.180**	**0.110**	**0.073**	**0.323**	**0.194**	**0.206**
	Baseline [10]	0.425	0.144	0.087	0.055	0.355	0.185	0.171	0.323	0.109	0.066	0.042	0.266	0.134	0.169
Content Word	ID 185	0.298	0.128	0.061	0.024	0.281	0.154	0.180	0.215	0.090	0.041	0.017	0.202	0.091	0.158
	ID 189	**0.321**	**0.149**	**0.077**	**0.034**	**0.302**	**0.171**	**0.192**	**0.237**	**0.109**	**0.056**	**0.027**	**0.222**	**0.106**	**0.172**
	Baseline [10]	0.244	0.105	0.051	0.024	0.233	0.123	0.150	0.185	0.079	0.038	0.019	0.177	0.080	0.139

The statistics of the datasets[6] are shown in Table 1. Here, for the number of speeches I only counted the ones related to question-and-answers in the newsletter, reducing the speeches made by chairpersons and others. I used the meetings held on the last day of each of the provided training datasets (i.e., Nov 30, 2012, and Dec 18, 2014; roughly 7% of the provided training datasets), for validation.

Following the NICT BERT Japanese Pre-trained Model's instruction, I used MeCab-Jumandic [8] to tokenize the Japanese sentences, and the model's vocabulary to tokenize them into subwords using byte-pair-encoding [12].

3.3 Hyper Parameter Selection

I used HuggingFace's Transformers [13] implementation to fine-tune my BERT models, and s2s-ft[7] implementation to fine-tune my UniLM models.

For fine-tuning my BERT model, I attempted every combination of the epochs of {1, 2, 3} and a learning rate of {2e-5, 3e-5, 5e-5}, and then chose a model whose outputs (i.e., source passages) achieved the best ROUGE-1 recall with regard to the reference summaries on the validation datasets. The batch size remained fixed at 32 throughout my experiments.

For fine-tuning my UniLM model, I attempted every combinations of the mask probability of {0.7, 0.9}, epochs of {10, 15, 20}, and a learning rate of {5e-5, 7e-5}, and then chose a model that achieved the best ROUGE-1 recall for content words on the validation datasets. Other hyperparameters remained fixed throughout my experiments: a batch size of 16, a beam size of 20, and a label smoothing rate of 0.1.

4 RESULTS OF THE FORMAL RUN

I submitted two formal runs[8] in the PoliInfo-2 task. The ID 185 system was a model that was trained only using the training datasets provided by the task organizers (i.e., the segmented and unsegmented datasets in Table 1). Meanwhile, the ID 189 system used all of the available training datasets.

Our official formal run results are shown in Tables 2 and 3, where Table 2 shows the human evaluation results and Table 3 shows the evaluation using ROUGE scores (see the task overview paper [5] for metrics used in the human evaluation). Bolded scores indicate the best results among all the participants (including the participants not shown in this paper)[9], and underlined scores indicate the next best results.

[6] The number of summaries is not twice the number of questions because more than one person (e.g., the Governor and superintendents) may answer a single question. Also, the gaps in dates in Table 1 (e.g., Mar to May 2011) are there when no meetings were held, while I used all the data from Sep 2001 to Dec 2019.

[7] https://github.com/microsoft/unilm/tree/master/s2s-ft

[8] In fact, I submitted three results to the leaderboard: IDs 185, 189, and 195. Since the first and the last submissions are from the same run and have identical results, I eliminate from this paper any results of the last submission.

[9] I only show the comparison with Baseline [10] in Tables 2 and 3 to avoid redundancy. See the task overview paper [5] for the results of other participants.

Table 4: Performance of each of my modules in the ID 189 system.

Text	Output by	Recall w.r.t. ref. summaries	Characters per summary
Source speech	Pre-processor	0.818	4,895.59
Source passage	Sentence extractor	0.423	117.65
Generated summary	Summary generator	0.320	57.76
Reference summary	-	-	38.69

Table 5: Human evaluation results for single- and multiple-sentence summaries.

		Content		Well-	Non-twisted	Sentence	Dialog
		X = 2	X = 0	formed	All	goodness	goodness
ID 189	all	1.082	0.975	1.858	1.316	1.129	0.937
	single	1.103	0.988	1.863	1.329	1.154	0.969
	multiple	0.995	0.921	1.841	1.266	1.028	0.768

The results show that my approach (ID 185) achieved better performance than other approaches in most of the metrics, especially in content, sentence goodness, and dialog goodness metrics.

Further, we can observe that adding my training datasets contributed to performance improvement (see IDs 185 vs. 189) and that the ID 189 system achieved the best performance among all systems in terms of all of the metrics. As my approach did not necessarily need annotated (i.e., segmented) data to train the model, I could use as many assembly minutes and newsletters as required, without additional effort.

5 DISCUSSIONS

5.1 Performance of Each Module

In this section, we evaluate the performance of each of my modules. Table 4 shows the ROUGE-1 recall for content words, with regards to the reference summaries of the output of each of my modules in the ID 189 system[10], along with the average length of the outputs.

The results show that 81.8% of the content words in a reference summary originated from the source speech. In other words, 18.2% of the content words in a reference summary need some external knowledge or similar. On the other hand, my sentence extractor and summary generator successfully extracted 51.7% (42.3%/81.8%) and 75.7% (32.0%/42.3%), respectively, of the available content words from the previous module's outputs. The reason for the lower performance of my sentence extractor may be as a result of the source speeches being too long—i.e. only 2.4% (117.65/4,895.59) of the characters needed to be extracted as the source passage using the current settings.

5.2 Unimplemented Features in My System

While we attempted to summarize the Tokyo Metropolitan Assembly minutes, my approach was not designed to fully cover the nature of the minutes. In this section, we consider two possible weaknesses in my approach and discuss future research directions.

First, I evaluated the performance of my model on each of single- and multiple-sentence summaries. Although the summaries in the newsletter were segmented by sub-topics, some of the summaries consisted of multiple sentences, posing several questions on the same sub-topic.

The results are shown in Table 5. The results show that the performance of my system was slightly worse for summaries with multiple sentences than for those with a single sentence. One possible reason is that my approach does not implement any special architecture to deal with the current settings. If, for example, our model could predict the number of questions, and the lengths of a source passage, and each summary sentence was adequately controlled, the generated summaries might be an improvement on the current approach.

Second, my approach achieved a lower dialog goodness score than sentence goodness score. One of the reasons might be that I generate a summary without taking relationships between questions and answers into account. If our model could consider such context, the dialog goodness score might be an improvement on the current approach.

I will address these weaknesses of my current system in future work.

5.3 Model Generalization for Future Meetings

To investigate whether my model can generate summaries for future meetings or not, I conducted additional tests using the same training datasets but with test datasets from different years. Figure 2 shows the ROUGE scores for content words of my ID 185 model that was trained using datasets from 2011, 2012, and 2014.

The results show that, even though my model was trained using minutes and newsletters between 2011 and 2014, we do not observe any performance degradation after 2013. This suggests that my model is robust enough for changes in topics discussed in the assembly.

However, the performance is poorer for meetings before 2010, especially in 2009. It might be possible that the editor of the newsletter who summarizes the assembly minutes may have changed, so that the resulting summaries were of a different nature before 2010. For our future work, we should investigate the reason for this observation in detail and find ways to mitigate or detect such performance degradation.

[10] The score for the generated summary is slightly different from the official result shown in Table 3 because I used my own evaluation script here.

Table 6: Details of human evaluation results on the ID 189 system.

	Content	Well-formed	Non-twisted	Sentence goodness	Dialog goodness
Grade A	29.7%	88.0%	60.5%	42.8%	29.3%
Grade B	38.0%	9.8%	10.6%	27.3%	35.0%
Grade C	26.9%	2.2%	28.9%	29.9%	35.6%
Grade X	5.3%				

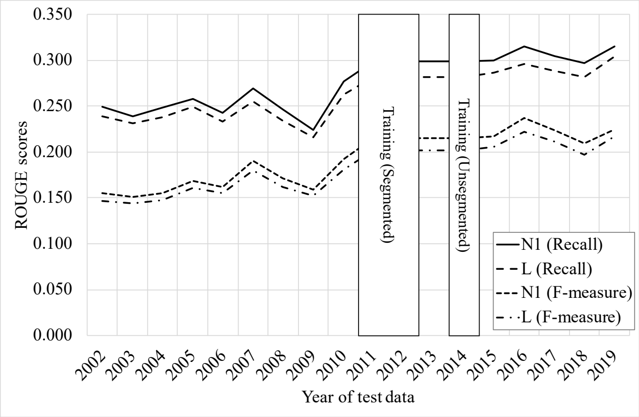


Figure 2: ROUGE scores on test datasets from different years (ID 185).

5.4 Detailed Results of the Human Evaluation

In this section, we examine the details of the human evaluation for the ID 189 system that achieved the best scores. Table 6 shows the number of instances for each grade (refer to the task overview paper [5] for metrics used).

While my system performs the best among all of the participants, only 29.7% and 29.3% of the instances were graded A for their summary contents and dialog goodness, respectively, and only 64.3% (29.3% + 35.0%) were graded A or B for their dialog goodness. Furthermore, 28.9% of generated summaries were judged to be twisted. We cannot but conclude that any systems submitted to this subtask, including mine, may not always be helpful to fact-check the opinions of the assembly members. In future work, the task settings should be revised so that the systems can output more reliable results for fact-checking.

6 CONCLUSIONS

I participated in the Dialog Summarization subtask of the PoliInfo-2 task. My contributions are summarized as follows:

- I proposed a system that generates summaries from the Tokyo Metropolitan Assembly minutes. My system consists of an extractor and a generator and achieved the best performance among all the participants. I also confirmed that my models could generate summaries for future meetings.
- I used a BERT-based model and a UniLM-based model to implement my extractor and generator, respectively.
- I proposed several modifications to the UniLM implementation in order to control the length of a generated summary.

In future work related to my methods, I will attempt to improve the dialog goodness score by adding a mechanism to consider a context, rather than generating summaries for questions and answers independently. I will also attempt to apply my models to other real-world tasks, including business conversations.

For our future work in the Dialog Summarization subtask, better task setting should be designed to help people fact-check the utterances of a speaker based on the generated summaries.

As another research direction, I will investigate automatic summarization from noisy minutes that could be generated by automatic speech recognition systems. It would be useful to apply such a system in a real-time assembly, or even in business meetings that typically lack minutes, to summarize conversations in a similar way.

REFERENCES

[1] Yasuharu Den, Toshinobu Ogiso, Hideki Ogura, Atsushi Yamada, Nobuaki Minematsu, Kiyotaka Uchimoto, and Hanae Koiso. 2007. The Development of an Electronic Dictionary for Morphological Analysis and Its Application to Japanese Corpus Linguistics. In *Japanese Linguistics*, Vol. 22. 101–123. (in Japanese).

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186.

[3] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 13063–13075.

[4] Satoshi Hiai, Yuka Otani, Takashi Yamamura, and Kazutaka Shimada. 2019. KitAi-PI: Summarization System for NTCIR-14 QA Lab-PoliInfo. In *NTCIR-14 Conference: Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies*. 159–166.

[5] Yasutomo Kimura, Hideyuki Shibuki, Hokuto Ototake, Yuzu Uchida, Keiichi Takamaru, Madoka Ishioroshi, Teruko Mitamura, Masaharu Yoshioka, Tomoyoshi Akiba, Yasuhiro Ogawa, Minoru Sasaki, Kenichi Yokote, Tatsunori Mori, Kenji Araki, Satoshi Sekine, and Noriko Kando. 2020. Overview of the NTCIR-15 QA Lab-PoliInfo-2 Task. In *Proceedings of the 15th NTCIR Conference*.

[6] Yasutomo Kimura, Hideyuki Shibuki, Hokuto Ototake, Yuzu Uchida, Keiichi Takamaru, Kotaro Sakamoto, Madoka Ishioroshi, Teruko Mitamura, Noriko Kando, Tatsunori Mori, Harumichi Yuasa, Satoshi Sekine, and Kentaro Inui. 2019. Final Report of the NTCIR-14 QA Lab-PoliInfo Task. In *Lecture Notes in Computer Science*, Vol. 11966. 122–135.

[7] Yasutomo Kimura, Keiichi Takamaru, Takuma Tanaka, Akio Kobayashi, Hiroki Sakaji, Yuzu Uchida, Hokuto Ototake, and Shigeru Masuyama. 2016. Creating Japanese Political Corpus from Local Assembly Minutes of 47 Prefectures. In *Proceedings of the 12th Workshop on Asian Language Resources (ALR12)*. The COLING 2016 Organizing Committee, Osaka, Japan, 78–85.

[8] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying Conditional Random Fields to Japanese Morphological Analysis. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Barcelona, Spain, 230–237.

[9] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*. Association for Computational Linguistics, Barcelona, Spain, 74–81.

[10] Yasuhiro Ogawa, Yasutomo Kimura, Hideyuki Shibuki, Tomoyoshi Akiba, Ken-Ichi Yokote, Hokuto Ototake, and Madoka Ishioroshi. 2020. NTCIR-15 QA Lab-PoliInfo-2 *ni okeru* Dialog Summarization. In *Proceedings of the Twenty-sixth Annual Meeting of the Association for Natural Language Processing*. Ibaraki, Japan,

945–948. (in Japanese).

[11] Yasuhiro Ogawa, Michiaki Satou, Takahiro Komamizu, and Katsuhiko Toyama. 2019. nagoy Team's Summarization System at the NTCIR-14 QA Lab-PoliInfo. In *Lecture Notes in Computer Science*, Vol. 11966. 110–121.

[12] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, 1715–1725.

[13] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv preprint arXiv:1910.03771* (2019).