akbl at the NTCIR-15 QA Lab-PoliInfo-2 Tasks

Takanori Nekomoto
Toyohashi University of Technology, Japan
nekomoto.takanori.pf@tut.jp

Daiki Shirato
Toyohashi University of Technology, Japan
shirato.daiki.at@tut.jp

Ryoto Ohsugi
Toyohashi University of Technology, Japan
ohsugi.ryoto.dv@tut.jp

Tomoyosi Akiba
Toyohashi University of Technology, Japan
akiba@cs.tut.jp

Shigeru Masuyama
Tokyo University of Science, Japan
masuyama@rs.tus.ac.jp

ABSTRACT

NTCIR-15 QA Lab-PoliInfo-2 establishes several tasks aimed at presenting pertinent information in resolving political issues. We, the akbl team, tackled the Stance Classification, the Dialog Summarization, and the Topic Detection tasks. For the Stance Classification task, we used, at first, a rule-based analyzer for extracting the opinion statements, then, for those left undetermined, we applied a BERT-based stance classifier on the debate statements. For the Dialog Summarization task, we firstly searched for the relevant segments, then we extracted the final sentence to form the output summary. For the Topic Detection task, we employed a clustering algorithm on the BERT embeddings of initial topic candidates extracted by using regular expressions, then their final topics were selected based on the centroid of each cluster.

TEAM NAME

akbl

SUBTASKS

Stance Classification Task(Japanese)
Dialog Summarization Task(Japanese)
Topic Detection Task(Japanese)

1 INTRODUCTION

The QA Lab-PoliInfo-2 (Question Answering Lab for Political Information 2) task at NTCIR-15 aims at complex real-world question answering (QA) technologies, to show summaries of the opinions of assembly members and the reasons and conditions for such opinions, from Japanese regional assembly minutes [1].

We reaffirm the importance of fact-checking because of the negative impact of fake news in the recent years. The International Fact-Checking Network of the Poynter Institute established International Fact-Checking Day on April 2 from 2017. In addition, fact-checking is difficult for general Web search engines to deal with because of the ‘ filter bubble’ whose concept was proposed by Eli Pariser [2], which keeps users away from information that disagrees with their viewpoints. For fact-checking, we should confirm primary sources such as assembly minutes. The description of the Japanese assembly minutes is a transcript of a speech, which is very long; therefore, understanding the contents, including the opinions of the members at a glance is difficult. New information access technologies to support user understanding are expected, which would protect us from fake news.

We proposed several methods for the Stance Classification Task, Dialog Summarization Task, and Topic Detection Task.

2 STANCE CLASSIFICATION

2.1 Overview

Stance classification task aims at estimating politician’s position from politician’s utterances. In PoliInfo2, a system participating in the task estimates the stances of political parties from the utterances of the members of the Tokyo Metropolitan Assembly. Given the Tokyo Metropolitan Assembly topics (agenda), member ’ s list and political denomination list, the systems classify their stance into two categories (agreement or disagreement) for each agenda [1].

We developed a rule-based binary classification method as a baseline for the aforementioned task, referring to the politicians of the regular meeting, because the representatives of each faction hold a discussion at the end of the meeting to clarify their positions. Since the rule-based approach cannot classify statements that do not clearly agree or disagree with them, we construct a politician’s statement classifier to determine whether it agrees or disagrees with a given agenda. The implementation of the classifier used a neural network with BERT for prior learning [3].

2.2 Method

First of all, we divided a speech of a politicians into two parts. Since most of the politicians reveal their party affiliation when they first express their opinions on a certain proposition, we define this oral statement as an “ opinion statements ”, the detailed contents of the subsequent statements are defined as “ debate statements ”. The Rule-based approach is used for the former and the BERT-based classifier is used for the latter to classify the agreement or disagreement.

2.3 Classification using Rule-based approach

The overall rule-based process is shown in Figure 2.1. The Rule-based approach is used to classify “opinion statements”. The minutes of the meeting are obtained in accordance with the session, and the speeches of the members are checked. If all the statements of “agreement”, “disagreement”, “representative” and “position” are included, extracts them and classifies them as the statements of opinion. The party names to which politicians with an opinion statements belongs can be found by using a dictionary of the politicians name and the party he or she belongs to, which we developed ourselves. We constructed this dictionary by providing the

names of politicians to the search engine of the Tokyo Metropolitan Assembly website, and using Beautifulsoup [4] to extract the names of past proceedings and committee meetings, etc., and then searching for the party names annexed with the members' names. In order to take into account the fact that some politicians have changed their political party affiliation, the date of affiliation to a certain party is added to the dictionary in addition to the name of the politician and his/her party.

After identifying the party names, we use cabocha, a Japanese dependency tool [5], to obtain the data for the opinion statements. We classify the data according to the following processes.

Step 1. If there is a reference to a person submitting the bill, make it clear whether she/he is the governor or a polititian.

Step 2. to see if there are references to a specific bill number.

Step 3. Check whether there are statements of approval or disapproval for a certain range of bills.

The final output is a disagreement or agreement after summarizing the agreement and agreement for each congregation in a format similar to the example in Figure 2.1. In case there is no statement or no statement on the bill, "no mention" is output.

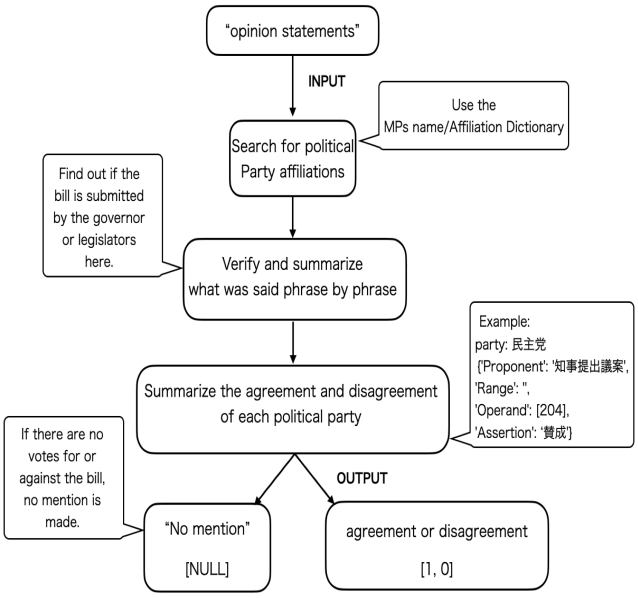

Figure 2.1: The overall rule-based process

2.4 Classification using BERT

We used the publicly available BERT Japanese Pretrained model for fine-tuning the classification task[6]. This model is trained based on the word segmentation by JUMAN [7], so we use the same morphological analyzer. In order to create the training data, from the data provided by Poliinfo2 we extracted the statements of politicians that were a cause of agreement or disagreement of the bills, we used only those statements that were in complete match with the names of the bills.

However, since these statements almost always contain content other than the agenda item in question, and the maximum number of words that can be entered into the learned BERT is 512, some pre-processing must be applied. This time, as a simple preliminary step, if a statement contains a specific conjunction, it is determined that it is unrelated to the agenda, and all statements after the conjunction are deleted. Specific conjunctions cover the following patterns.

Specific conjunctions

(次に、.*|次いで、.*|続いて、.*|最後に、.*|さて、.*)

The above pattern does not match those that appear in the middle of an utterance, but only considers those with a punctuation mark.

The total number of training data is 460, and each row consists of assertion ("agreement" or "disagreement"), labels ("0": disagreement or "1": agreement), and utterances from the beginning. In order to classify sentences, a classifier was created to perform a binary classification. We conducted an additional test to see if giving not only the debate statements(doc) of politicians but also party names that made the statement(doc+party) contributes to the improvement of the accuracy of the classification.

2.5 Results

In order to evaluate the system, we sampled 60 test data from 460 training data and examined the percentage of correct responses in the agreement/disagreement classification when only doc and doc+party were given. The percentage of correct answers is shown below.

Table 2.1: percentage of correct answers without preprocessing

	Accuracy	Correct
BERT: doc	0.633	38/60
BERT: doc+party	0.667	40/60

Table 2.2: percentage of correct answers with preprocessing

	Accuracy	Correct
BERT: doc	0.683	41/60
BERT: doc+party	0.783	47/60

The highest scores were attained for those with preprocessing and giving the party names, suggesting that there is a high correlation between the debate statements and party names.

For the formalrun submission, we used two systems based on our two classifiers. The one employed only the rule-based classifier. When the result of the classification was "no mention", it simply outputted "agreement" since it was expected in the majority. The other employed both classifiers. It firstly asked to the rule-based classifier, then when "no mention" it asked to the BERT-based classifier. When no opinion statement was found for the given agenda, it outputted "agreement" same as the former. The BERT-based classifier was trained again by using all the available

Table 2.3: Classification results by method

	Accuracy
Rule-based only	0.9498
Rule-based → BERT classification	0.9498

460 samples for the latter system. The formalrun results were shown in Table 2.3.

At the time of the rule-based approach, the percentage of correct answers was over 90%. However, the score did not change even when additional classification was implemented. This may be due to the fact that most of the politicians' statements were dominated by opinion statements that could be judged on a rule-based approach, and in this experiment we were only able to provide only six available debate statements in the test data. Also, the reason why the accuracy did not change at all could be due to the fact that the classifier's misclassification constant is as large as the number of correct answers.

2.6 Further experiments

After the formalrun submission, we tried to improve the BERT-based classifier by revising our pre-processing method on the training data and by augmenting the available training data. In this section, we describe several techniques we have used to remove unnecessary text unrelated to the agenda in the training data.

2.6.1 Preprocessing with Bag of Words.
In most cases, politicians who are given the right to speak state a wide range of opinions, agreement or disagreement, suggestions, and requests for several agendas. When the content of the agendas being referred to changes, most politicians often state the conjunctions such as "next," "following," or "finally" and so on. However, there are many politicians who suddenly start referring to different agendas, and inconsistent statements may appear in the debate, and the frequency of the words that appear before and after the statements is expected to change significantly. In addition, as mentioned in section 2.3, since only statements that exactly match the name of the bill are used, the latter part of the statement is more likely to be unrelated to the name of the bill.

Therefore, we apply Bag of Words(BoW) to the sentences of the politicians' speeches punctuated by punctuation marks, and calculate the similarity between the sentences based on the frequency of word occurrences. The comparison between sentences should be limited to the frequency of occurrence of noun phrases, and when only the billnumber is mentioned, there is a high possibility that the degree of similarity between sentences cannot be calculated. In that case, the billnumber is rewritten to be followed by the bill(e.g.『第四十四号議案について申し上げます』→『第四十四号議案(東京都議会議員の議員報酬) について申し上げます』). This is the name of the bill that has been added for comparison purposes only, so it should be deleted when adding it to the training data.

In this study, we used two methods **BoW** and **BoW'** to calculate the similarity between sentences and construct the training data, which are defined as follows using procedures.

BoW Compare the first sentence t with the $t+1$ sentence and consider each to be semantically identical if at least one word matches. Continue comparing $t+1$ and $t+2$... and so on until the end of the sentence is reached, and if there is no match, all subsequent sentences are deleted.

BoW' Starting from the first sentence t, compare it to $t+1$, $t+2$... and so on until the end of the sentence is reached, and if there is no match, all subsequent sentences are deleted.

2.6.2 Preprocessing with Dynamic Programming.
We performed text segmentation using Dynamic Programming(DP) as a method to remove statements that are not relevant to the agenda from the statements of the politician. The premise is that the smallest unit to consider a segment is a sentence. That is, one data point in sequence data is one sentence. In this case, the segmentation is performed by using a technique that uses the cohesion of words in the segment as a score and maximizes the total score through DP [8]. The implemented text segmentation procedure is as follows using procedures.

Step 1. Define a text data class and $h(t1, t2)$ to compute the score in the segment, $GetLength()$ to return the length of the text.

Step 2. Let T be the list of sentences in the text, L be the list of words in the text (using Mecab), and define an $L \times T$ scale $matrix\ F$ that represents similar words between sentences.

Step 3. The $matrix\ F$ is a sparse matrix, and the words are extracted from each sentence, and if a word in L is in T, the index of the word is taken from the list L and 1 is assigned to the row of the index in the $matrix\ F$.

Step 4. Define the product of the $matrix\ F$ and its inversion matrix as the $matrix\ D$ that represents the similarity between the sentences i and j. The same sentences are set to 0 ($D_{ii} = 0$), and this $matrix\ D$ is used to compute the score within a segment.

Step 5. The similarity of all the sentences in the segment is added together and divided by the length of the segment to obtain the score. The evaluation function J is defined with the start point of the segment as t_1 and the end point as t_2.

$$J = \frac{\sum_{i=t1}^{t2-1} \sum_{j=t1}^{t2-1} D_{ij}}{(t2 - t1)}$$

Step 6. A discussion sentence and an arbitrary number of segments are given to the recursive function of DP as an object, and the division point of the segment where J is the largest is returned as an index.

We define a series of processes as **DP**, apply them to the content of the politicians' statements, and record the first segment obtained into the training data, and delete all other segments. Since the number of segments must be given arbitrarily to the DP, the number of segments resulting from the **BoW'** is input here.

2.6.3 Further experimental results.
The proposed three methods and the baseline training data were tested to see if they contribute to improve the accuracy of the classification. As a baseline, we used training data based on the specific conjuction described in Section 2.4. We also augmented the size of the available training data for BERT-based classifier by adding the statements whose mention of agenda name matched not exactly but approximately with the given agenda. We employed edit distance for the proximity calculation. The total number of data was

457 without data augmentation and 509 with data augmentation (referred to as DA).

In order to verify the accuracy of the training data, 60 data items were randomly sampled from the training data and used for verification. These 60 data items were made to be identical to the data sampled in Section 2.5, examined the percentage of correct responses in the agreement/disagreement classification when only doc and doc+party were given. The percentage of correct responses for each proposed method was shown in Table 2.4.

Table 2.4: Percentage of correct answers for each method

method	BERT: doc		BERT: doc+party	
	Accuracy	Correct	Accuracy	Correct
Baseline	0.8000	48/60	0.7667	46/60
BoW	0.8500	51/60	0.8833	53/60
BoW'	0.9000	54/60	0.8833	53/60
DP	0.8833	53/60	0.8833	53/60
Baseline+DA	0.8167	49/60	0.8500	51/60
BoW+DA	0.8333	50/60	0.8333	50/60
BoW'+DA	0.8500	51/60	0.8167	49/60
DP+DA	0.8667	52/60	0.8667	52/60

The total number of data itself increased compared to the experiment in section 2, and the accuracy of the baseline method was higher than that of the experiment in section 2. For all the methods with data expansion using edit distance, the accuracy rate was higher than before the data augmentation, but this did not lead to a significant improvement in accuracy, and it was also observed that the addition of party did not directly contribute to the improvement in accuracy.

3 DIALOG SUMMARIZATION

3.1 Overview

Dialog summarization task aims to produce a summary, taking into account the structure of the dialogue in local councils. Given the minutes of the Tokyo Metropolitan Assembly, the topic, speaker's name, and a word limit on the summary, the system will produce a summary based on the topic and within the character limit. Segmentation was done on the given minutes and only the parts related to the summarization were used, and the final sentence extraction was used for the summary.

3.2 Method

Our system generates a summary by following the steps below.

Step 1. Collect all utterances of the speaker from the given speaker name.

Step 2. Divide the utterances into segments by finding the topic changes using regular expressions.

Step 3. From the obtained segments, select the segment related to the given topic based on the word similarity with a given topic, and create a summary sentence from the selected segment within the limited number of characters.

3.3 Segmentation

Since the speaker may be speaking about more than one topic, it is necessary to search for utterances that are relevant to a given topic. All utterances of the speaker are concatenated into one and then divided using a regular expression to create a segment for each topic. The regular expressions used are shown in Table 3.1.

Table 3.1: Regular expressions for splitting

regular expression
伺い [^、]*ます。|お尋ね [^、]*します|お答えください。|(見解|所見|答弁) を求め [^、]*ます。|(いかがで|どうで)(しょうか|すか)。|.+質問を (終わります|終了します)。

The regular expressions shown in Table 3.1 are commonly used when the questioner has finished asking a question on a topic[9].

We search for words in the topic from the created segments, and use the segment in which the word is found for summarization. At this time, the agenda is morphologically analyzed using Mecab[10], and only the words excluding particles are used for the search. If a word is found in the topic from more than one segment, all segments containing the word are arranged in order of utterances and the first one uttered is used. Also, if no words in the topic are found in any segment, all segments are arranged in order of utterances and the first one uttered is used in the summary.

One of the characteristics of speech in the Tokyo Metropolitan Assembly is that speakers speak about multiple topics in sequence, so they always finish speaking about one topic before speaking abougt the next. Therefore, if a segment is related to a particular topic, that segment can be excluded from the selection of segments related to another topic. In this way, by reducing the number of candidates when selecting the segment related to the topic, the segment can be selected more accurately. Therefore, if a word on the topic is found in only one segment during segment selection, that segment is determined as a segment related to that topic and excluded from candidates for subsequent segmentation.

3.4 Summarization

We use final sentence extraction for summarization. The last utterance of the segment obtained by segmentation is output as a summary sentence.

3.5 Results

The ROUGE score of the automatic evaluation index was 0.0621 [11]. Table 3.2 shows an example of the system output and correct answers.

From Table 3.2(a) , the output of the system is within the limited number of characters, and the content of the sentence is the same as the correct summary, but vocabulary used is significantly different. On the other hand, in the case of Table 3.2(b), the output of the system has completely different from the correct summary due to incorrect segmentation. correcting the wording of the output sentence and more accurate segmentation are our future work.

Table 3.2: Results of Dialog summarization

(a)

Summary output by the system	そこで、知事は、東京都の教育の充実に向け、特に何を重視していくのか、見解をお伺いいたします。 (Therefore, I would like to ask you for your views on what you will particularly focus on to improve education in Tokyo.)
Correct summary	教育の充実に向けた見解は。 (What are your views on how to improve education?)
character limit	50

(b)

Summary output by the system	次に、小笠原諸島の航空路問題について伺います。 (Next, I would like to ask about the air route problem to the Ogasawara Islands.)
Correct summary	食の安全・安心確保と食文化の拠点継承について知事の見解は。 (What is the governor's view on ensuring food safety and security and inheriting a base for food culture?)
character limit	50

4 TOPIC DETECTION

4.1 Overview

The purpose of the Topic Detection task was to present a "list of appropriate topics" from the minutes for each council member [1]. We considered the topics in *Togikai Dayori*[12] to be appropriate topics. From the *Togikai Dayori*, topics and summaries compiled from the minutes are posted for each council member. Since *Togikai Dayori* is created by a skilled person, the topics posted on it can be considered appropriate. Therefore, in this task, we aimed to create a system that produces output that is as similar as possible to the topics in *Togikai Dayori*.

4.2 Method

In this task, a minutes is used as input. Questions and answers are recorded in the minutes. First, we extract candidate topics from the utterances of questions and answers using regular expressions and part of speech. We found the number of the extracted candidates was more than that of *Togikai Dayori*. We also found that some candidates share a same meaning. Therefore, we clustered the candidates to summarize them. In clustering, topic candidates were converted to feature vectors by Pretrained SBERT[13]. By using these vectors, we expect topics are well classified, and a list of appropriate topics is created.

The specific process is as follows using procedures.

Step 1. Extract topic candidates from question/answer utterances
Step 2. Convert candidate topics to feature vectors with SBERT
Step 3. Cluster those feature vectors
Step 4. Select a topic from each cluster based on the centroid

The details of each step are shown below.

Step 1. Extract topic candidates from question/answer utterances. In utterances, topic are often included before "について (About)." Therefore, by extracting that part, it can be treated as a "candidate topic". We used the following regular expressions to extract candidate topics.

- **(Nouns | Prefixes | Independent Verbs | Specific Particles)+について (about)**

The first half of the regular expression means that matching of the part of speech. MeCab[?] was used to obtain the part of speech. We capture the matched part of speech and use it as a candidate topic. We also implemented filtering to reduce extraction errors. In the method, candidate topics are extracted only those preceded by a sentence breaks (punctuation marks). The idea behind that comes from the observation that topics often exist at sentence breaks. In addition, since "、(,)" is sometimes used as a delimiter to describe words in parallel, if the word before "、(,)" is a noun, "、(,)" is converted to "・" and extracted together with the word.

Step 2. Convert candidate topics to feature vectors with SBERT. These vectors will be used in the next step (clustering). We used Sentence BERT (SBERT)[14]. In clustering, elements are classified according to their Euclidean distance from each other. SBERT is a distance-learned BERT model by the Siamese Network. Distance learning means that data that are semantically close are trained to be closer in vector space and data that are semantically farther apart are trained to be apart in vector space. Therefore, the distance between the vectors of SBERT is more accurate than the distance between the vectors of normal BERT. In this system, we used the pre-trained SBERT model in Japanese[13].

Step 3. Cluster candidate topics to group similar topics. The feature vector obtained in the previous step is used. We implemented clustering using scikit-learn k-means[15]. The initial position of the cluster is determined by k-means ++ and the metric is the Euclidean distance. In k-means, it is necessary to give the number of clusters k as a parameter. k is equal to the number of topics in the final output list. We defined k = (number of cadidates topics extracted from questions and answers) x (constant C). For example, if C = 0.5, half of the extracted candidate topics will be output to the list. When we decided on C less than 0.2, topics with low relevance were grouped into the same cluster. On the other hand, when we decided on C greater than 0.2, topics with duplicate meanings were output. Therefore, we decided on C = 0.2.

Step 4. Select a candidate topic from the cluster. The closest topic is selected from the Centroids in the cluster. The metric is the Euclidean distance. The selected topic is finally output to the list as a "predictive topic".

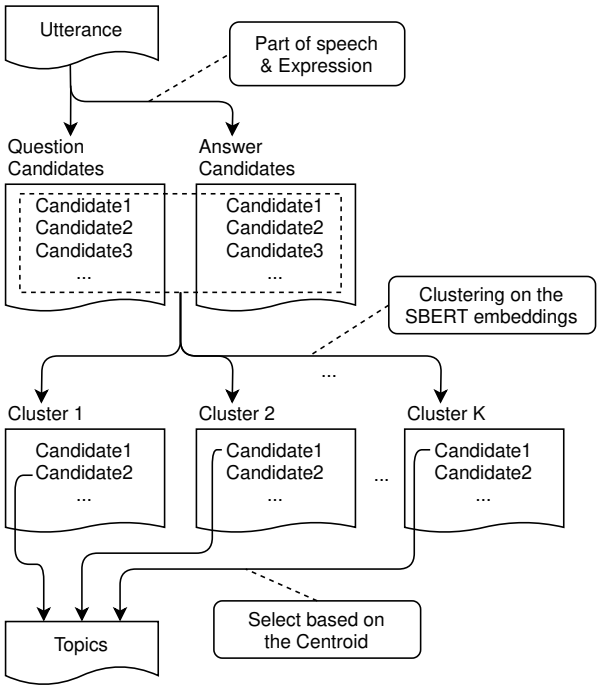

Figure 4.1: Topic Detection Task method

4.3 Previous method

At the time of formal-run, we used the method without SBERT. The differences between each step are shown below.

Step 1. The following regular expressions were used.
- **(Noun | Verb | Adverb | Adjective | Prefix | Particle)+ について (about) | に関するご?質問 (a question about) | (について)?のお?話 (speech about) (が (ga)|に (ni))**

In addition, filtering is not performed to reduce extraction errors.

Step 2. Calculate the similarity between candidate topics extracted from questions and answers and remove candidate topics with low similarity. Similarity is determined by the character match rate. This process gives results like the filtering of the previous step.

Step 3. Cluster using the similarity. The method is single-link hierarchical clustering.

Step 4. Select a topic from the cluster according to the extraction order. The candidate topic extracted earlier is selected.

4.4 Results

As an example, some of the system output topics and *Togikai Dayori* topics are shown in Table 4.1.

This task was manually evaluated. We compared the topic output from the system with the topic of *Togikai Dayori* and counted the number of matches between the two topics. Evaluation was based on the number of matches. The topic output from the system is defined as the "predicted topic", and the topic of *Togikai Dayori* is defined as the "correct topic".

Table 4.1: Results of Topic Detection Task

Topics of System Output	新型コロナウイルス対策 (Measures against new coronavirus) 多摩都市モノレールの町田方面への延伸 (Extension of Tama City Monorail toward Machida) 東京グリーンボンドの活用 (Utilization of Tokyo Green Bond) 流域対策 (Basin measures)
Topics of Togikai Dayori	新型コロナウイルス対策 (Measures against new coronavirus) 多摩都市モノレールの町田延伸 (Extension of Tama city monorail to Machida) とちょう電力プラン (Tocho power plan)

When the keyword of the predicted topic is included in the correct topic, the two topics are determined to match. Select the word that best describes the topic as a keyword. This was manually selected.

The evaluation results are shown in the table 4.2. Count the number of matching topics for each member, and calculate Precision, Recall, and F-measure from the total of all members. The comparison between the method in Section 4.2 refferd to as camera-ready version and the method in Section 4.3 refferd to as formal-run version are shown in the table 4.2.

Table 4.2: Evaluations of Topic Detection Task

version	Precision	Recall	F-measure
formal-run (without SBERT)	0.277	0.497	0.356
camera-ready (using SBERT)	0.403	0.543	0.462

Compared to the formal run version, the evaluation value was improved in the camera ready version. One of the reasons is that SBERT required more meaningful expressions and better similarity. Another reason is that k-means++ determines the distributed initial position, and the output topic list has distributed contents.

The results of the error analysis are shown below.

- Inappropriate sentences are extracted as topics. For example, there is an utterance, "基本的な考え方について知事の見解を伺うとともに…(Along with asking the governor's view on the basic idea ...)". Since this system extracts the expression appeared before "について (about)", the candidate topic "考え方 (idea)" is extracted.
- Some topics cannot be extracted by regular expressions. For example, there is a correct topic "東京ユアコイン (Tokyo Your Coin)" in *Togikai Dayori*. The first utterance of this topic was "先日、東京ユアコインの実証実験を丸の内のランチタイムに体験しました。(The other day, I experienced a demonstration experiment of Tokyo Your Coin at lunch time in Marunouchi.)" From such utterances, it is difficult to extract topics with specific expressions such as this system.

- Candidate topics not selected from the cluster may become correct topics. For example, in this system, the candidate topics "学校におけるパワーハラスメント (power harassment in schools)" and "学校体育館の空調設置 (air-conditioning installation in school gymnasiums)" are included in the same cluster. Since *Togikai Dayori* includes only the topic "学校体育館の空調設置 (air-conditioning installation school gymnasiums)", if "学校におけるパワーハラスメント (power harassment in schools)" is selected as a predicted topic, it will be incorrect.
- There are topics that cannot be summarized in *Togikai Dayori*. For example, there is the utterance "次に、オリンピック・パラリンピックについて質問します。(Next, I have a question about the Olympics and Paralympics.)" In this system, "オリンピック・パラリンピック (Olympic and Paralympic)" are extracted as candidate topics. However, there is no similar topic in *Togikai Dayori*. Therefore, if "オリンピック・パラリンピック (Olympic and Paralympic)" is selected as a predicted topic, it will be incorrect.

REFERENCES

[1] 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. Overview of the ntcir-15 qa lab-poliinfo-2 task. *Proceedings of The 15th NTCIR Conference*, 12 2020.

[2] Eli Pariser. The filter bubble: What the internet is hiding from you. 2011.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Gábor László Hajba. *Website Scraping with Python: Using BeautifulSoup and Scrapy*. Apress, 2018.

[5] Taku KUDO. Yet another japanese dependency structure analyzer. *http://chasen.org/˜ taku/software/cabocha*, 2004.

[6] Tomohide Shibata, Daisuke Kawahara, and Sadao Kurohashi. Improving the accuracy of japanese parsing with bert. *The Association for Natural Language Processing 25th Annual General Meeting*, pages 205–208, 2019.

[7] Daisuke Kawahara. Japanese morphological analysis system juman version 3.61. *http://pine.kuee.kyoto-u.ac.jp/nl-resource/juman.html*, 1999.

[8] V. Petridis A. Kehagias, P. Fragkou. Linear text segmentation using a dynamic programming algorithm. *In Proceedings of the EACL*, pages 171—-178, 2003.

[9] Katsumi Kanasaki, Jiawei Yong, Shintaro Kawamura, Shoichi Naitoh, and Kiyohiko Shinomiya. Cue-phrase-based text segmentation and optimal segment concatenation for the ntcir-14 qa lab-poliinfo task. In Makoto P. Kato, Yiqun Liu, Noriko Kando, and Charles L. A. Clarke, editors, *NII Testbeds and Community for Information Access Research*, pages 85–96, Cham, 2019. Springer International Publishing.

[10] 工藤 拓 (Kudo Taku), 山本 薫 (Yamamoto kaoru), and 松本 裕治 (Matsumoto Yuji). Conditional random fields を用いた日本語形態素解析 (applying conditional random fields to japanese morphologiaical analysis). 情報処理学会研究報告. *NL*, 自然言語処理研究会報告, 161:89–96, may 2004.

[11] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157, 2003.

[12] Tokyo metropolitan assembly togikai dayori. *https://www.gikai.metro.tokyo.jp/newsletter/*, 2020.

[13] Sentence embeddings with bert & xlnet. *https://github.com/sonoisa/sentence-transformers*, 2019.

[14] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[15] Clustering — scikit-learn. *https://scikit-learn.org/stable/modules/clustering.html*, 2007.