LIAT Team’s Extractive Summarizer 
at NTCIR-15 QALab PoliInfo-2

Kouta Nakayama  
RIKEN AIP, Japan  
kouta.nakayama@riken.jp

Satoshi Sekine  
RIKEN AIP, Japan  
satoshi.sekine@riken.jp

ABSTRACT
This is the report of the summarization system that our team LIAT submitted to the dialog summarization task in NTCIR-15 QALab PoliInfo-2. We designed an extractive summarizer by dividing the task into three parts and training a model for each. Analysis from the scores showed that the line level extractive summarizer that we created did not suit the task.

TEAM NAME
LIAT

SUBTASKS
QALab PoliInfo-2 (Dialog summarization task)

1 INTRODUCTION
NTCIR-15 QALab PoliInfo-2[2] is a shared task that deals with political documents in Japan. The dialog summarization task we participated in is a task to parse and summarize the dialogue structure of local councils. For more information on the task, see the PoliInfo-2 organizer’s paper[2]. Note that the formal submission was found to contain a bug, so this paper describes the content of the late submission. Therefore, there are no manual evaluation results.

2 SYSTEM DESCRIPTION
We created three models, border detector, topic matcher and summarizer, to solve the task. We use BERT[1] for those models. We explain the role of those models below.

2.1 Border Detector
The dialogue summarization task needs to detect a range of question or answer sentences at first. The border detector estimates the range of question or answer sentences by determining their boundaries. A boundary line is predicted by a binary classification of whether the boundary line is between two sentences. The training data for border detectors is generated from segmented training data. The segment boundaries are treated as positive and the line boundaries inside the segment as negative. We sampled up to two boundaries from inside each segment to avoid bias towards negative examples. Sentence 1 and sentence 2 are combined with a special token as follows and passed to BERT.

[CLS] Sentence 1 [SEP] Sentence 2 [SEP]

BERT then predicts whether the sentence boundary is the beginning or end of the question or answer from the output corresponding to the [CLS] token. In more detail, we have linear layers of \( f_{\text{beginning}} \), \( f_{\text{end}} \), \( f_{\text{beginning answer}} \), and \( f_{\text{end answer}} \), each of which is connected to the output of BERT for binary classification.

2.2 Topic Matcher
The topic matcher matches segments and subtopics. We use segmented training data for training, which is positive if the segment is about a subtopic and negative otherwise. BERT will be passed the subtopic and the beginning and end of the segment as following.

[CLS] Sub topic [SEP] Beginning of the segment [SEP]  
End of the segment [SEP]

If the input length is exceeded, we cut out the beginning and end of the segment appropriately. BERT predicts from the output corresponding to [CLS] whether a segment is valid for a given subtopic or not. BERT predicts whether a segment corresponds to a given subtopic. Specifically, we connect a linear layer \( f_{\text{bc tp}} \) to the output of BERT for binary classification. Let us denote the output of BERT for [CLS] token as \( x_{\text{cls}} \). We get the final output using the softmax function \( f_{\text{softmax}} \) as follows.

\[Pr(y|x_{\text{cls}}) = f_{\text{softmax}}(f_{\text{bc tp}}(x_{\text{cls}}))\]

Since more than one segment may be assigned for a subtopic, we select one where \( Pr(y = 1|x_{\text{cls}}) \) is higher than the threshold \( t \). If no segment exceeds the threshold, then the segment with the largest \( Pr(y = 1|x_{\text{cls}}) \) is applied.

2.3 Summarizer
We create an extraction summarizer at the line level. Therefore, it is necessary to identify the training data lines in advance that could be the answer. We took the line that shares the most nouns with the gold summary as the correct answer. Also, we used the results of the topic matcher on the unsegmented training data as training data. The characteristics of a sentence that could be a summary may be very different for a question and an answer. Therefore, we trained a different summarizer with questions and answers. We define the input to BERT as follows. Herein, the number of lines \( n \) to be passed simultaneously is determined by the maximum input length of BERT.

[CLS] Sub topic [SEP] [Mask] Line 1 [MASK] Line 2  
... [MASK] Line n [SEP]

We get prediction results for [MASK] tokens placed in front of each line. Specifically, we connect the common linear layer \( f_{\text{bc s}} \) for binary classification to the outputs of BERT, which correspond to each [MASK] token. Let \( x_{\text{mask i}} \) be the prediction of BERT for
We present our results with the development data in Table 1. We see that the [MASK] token connected to line $i$. Using the softmax function $f_{soft}$, the final output is obtained as follows.

$$Pr(y_i|x_{mask_i}) = f_{soft}(f_{bc}(x_{mask_i}))$$

We use the threshold as well as the topic matcher to select the outputs, as we allow multiple lines to be a summary.

### 3 EXPERIMENTS

#### 3.1 Data Preprocessing

We use MeCab[3] to tokenize the corpus. And, we use Juman[4] as a dictionary for MeCab. During tokenization, we also collect the nouns that are used to create training data for summarizer.

#### 3.2 Experimental settings

We use the pre-trained parameters\(^1\) for BERT. We used the hyper-parameters in Table 2 for training unless mentioned otherwise and train the models with mixed precision floating point arithmetic [5]. We used the minutes of two meetings with recent dates in the segmented data as development data.

**Topic Matcher.** We set the batch size to $b = 128$ and the threshold for output selection to $t = 0.9$.

**Summarizer.** We set the batch size to $b = 8$ and the threshold for output selection to $t = 0.5$.

#### 3.3 Results

We present our results with the development data in Table 1. We see that the border detector and the topic matcher score are relatively high. In other words, our system seems to be able to handle data that doesn’t have segments. However, the summarizer scores are quite low. Because the summary lines are identified by noun matching with each gold summary, they may contain a lot of noise. Given that segmentation is working well, perhaps we should create a generative summarizer.

The scores on the leaderboard are shown in Table 3. ROUGE-1-R is a macro average of the content words. JRIRD is the best result on the leaderboard and seems to be a generative summary. Our summarizer is greatly inferior. Line level extractions are likely to contain irrelevant parts and are likely to have lower scores.

### 4 CONCLUSIONS

This paper describes the system submitted to PoliInfo-2. Analysis from the scores shows that line level extractive summarizers do not seem to match the task very well. Future research will include detailed analysis and the creation of a generative summarizer.

### REFERENCES


---

\(^1\)We selected a model that does not use BPE. https://alaginrc.nict.go.jp/nict-bert/index.html

---

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epoch</td>
<td>10</td>
</tr>
<tr>
<td>Batch size</td>
<td>32</td>
</tr>
<tr>
<td>Gradient steps</td>
<td>1</td>
</tr>
<tr>
<td>Sequence length</td>
<td>512</td>
</tr>
<tr>
<td>Weight decay</td>
<td>0.01</td>
</tr>
<tr>
<td>Max grad norm</td>
<td>1</td>
</tr>
<tr>
<td>Hidden dropout</td>
<td>0.3</td>
</tr>
<tr>
<td>Attention dropout</td>
<td>0.1</td>
</tr>
</tbody>
</table>

**Table 1: Common hyperparameters.**

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Border detector</td>
<td>0.928</td>
<td>0.967</td>
<td>0.947</td>
</tr>
<tr>
<td>Topic matcher</td>
<td>0.915</td>
<td>0.874</td>
<td>0.894</td>
</tr>
<tr>
<td>Summarizer (question)</td>
<td>0.594</td>
<td>0.632</td>
<td>0.612</td>
</tr>
<tr>
<td>Summarizer (answer)</td>
<td>0.607</td>
<td>0.591</td>
<td>0.599</td>
</tr>
</tbody>
</table>

**Table 2: Scores on development data.**

<table>
<thead>
<tr>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>JRIRD (formal submission)</td>
<td>0.321</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours (late submission)</td>
<td>0.095</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 3: Scores on leader board.**