ditlab at the NTCIR-16 QA Lab-PoliInfo-3

Yuuki Tachioka
Denso IT Laboratory
Japan
tachioka.yuki@core.d-itlab.co.jp

Atsushi Keyaki*
Hitotsubashi University
Japan
a.keyaki@r.hit-u.ac.jp

ABSTRACT

The ditlab team participated in the QA Alignment and Question Answering task of the NTCIR-16 QA Lab-PoliInfo-3 task. First, we developed a QA Alignment system that associates each question with its answer by using heuristic rules to make paragraphs composed of related sentences and then matches them. Heuristic rules were optimized for government minutes. We prepared four types of features for matching. Second, we built a QA system that uses a similarity measure to find the original question of which contents are similar to that of the question summary. It then identifies the answers associated with the original question by using the results of the QA Alignment described above. A Text-to-Text Transfer Transformer (T5) was used to summarize the associated answer.

KEYWORDS

BM25, BERT, Gale-Shapley algorithm, T5

TEAM NAME

ditlab

SUBTASKS

QA Alignment task (Japanese)
Question Answering task (Japanese)

1 INTRODUCTION

The ditlab team participated in the QA Alignment and Question Answering tasks of the NTCIR-16 QA Lab-PoliInfo-3 task [3]. First, we proposed heuristic rules to make a paragraph composed of related sentences for question and answer. We prepared three types of features to calculate similarities between question and answer paragraphs: BM25 [6], Bidirectional Encoder Representations from Transformers (BERT) [1], and Wikipedia2Vec [7].

In this task, government minutes are composed of sentences. Each sentence has a Q/A/O tag. For the QA Alignment subtask, it is necessary to make paragraphs and associate a question with its answer. QA Alignment is performed in three steps (shown in Fig. 1). First step finds the corresponding part from the entire minutes by date and questioner ID. Second step combines multiple related sentences with “Q” or “A” tags to form a paragraph. Third step matches question and answer paragraphs based on the similarities between them.

We also developed a QA system that utilizes the results of the QA Alignment, as shown in Fig. 2. The first step is associating a *question summary* (input of the system) with the original question asked in the Tokyo Metropolitan Assembly. Then, the corresponding answer of the original question is identified. Lastly, the *answer summary* is generated using a summarizer.

*This work was conducted while the author was at Denso IT Laboratory.

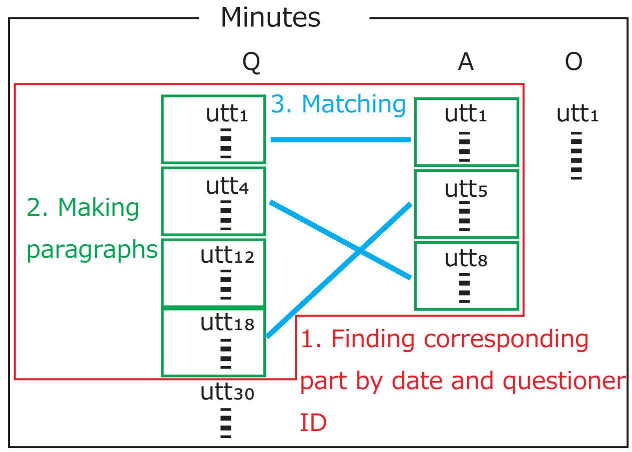

Figure 1: Linking questions and answers.

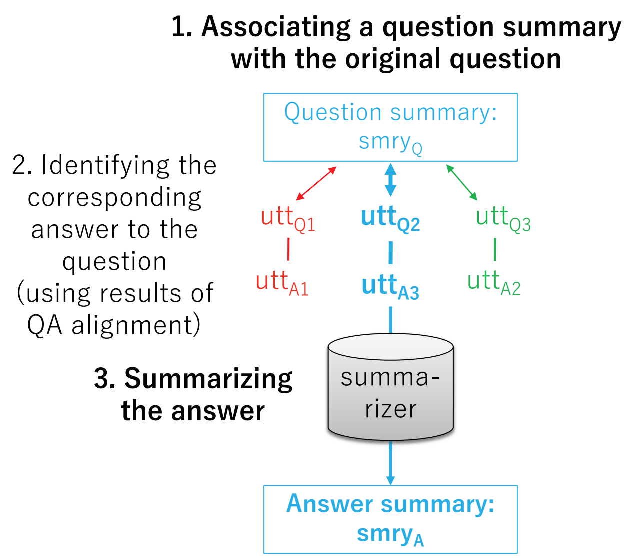

Figure 2: Overview of the QA part.

2 METHODS (QA ALIGNMENT)

2.1 Heuristic rules to make paragraphs

We can accurately combine sentences by regular expressions that are optimized for the minutes because questions and answers in the Diet have a fixed format.

2.1.1 Fixed phrases at the beginning or the ending of the sentence that start paragraphs. New paragraphs start when the pattern

```
r`^まず[^は]|^最初に|^初めに|^次に|^次いで|^続いて|^続きまして|^最後(に|で)|^終わりに|^なお，[^，]+質問|^[一二三四五六七八九十]+(点|問)目|^[^，]+についてで(す|あります|ございます)(が|けれど)|^終わり(ま|で)す．$|^以上で|^ありがとうございま'
```

matches the beginning of the sentence or the pattern

```
r`について再?質問します．$|^[^，]{,10}でございます．$|に(関して|ついて)(のご質問|のお尋ね|のご指摘)?(で|が)(あり|ござい)ま(した|す)．$|一言申し上げます|'
```

matches the ending of the sentence.

2.1.2 Fixed phrase at the ending of the sentence that terminates paragraphs. Paragraphs terminate when the pattern below matches the ending of the sentence.

```
r`(お?伺い|お尋ね)を?(いたし|し|し?たいと思い)?ます．$|問います．$|伺いたい．$|(お示し|答えて|お答え|お述べ|ご説明|示して|お聞かせ)(ください|願います)．$|(見解|所見|答弁|認識)を(お願いし|求め|伺い)ます．$|(いかがお考えで|いかがで|どうで)(しょうか|すか)．$|(要望を?(して|しておき|し|させていただき)|求めておき)ます．$|(対応する|認識している|どう[^，]+)のですか．$|(対応し|認識し|どう[^，]+)(てい)?ますか．$|必要ではないでしょうか．$|ではありませんか．$|.+質問を(終わり|終了し)ます．$|所存でございます．$'
```

2.1.3 One-sentence paragraph. A sentence that matches the pattern below makes an isolated paragraph.

```
r`^また，.+(いかがで|どうで)(しょうか|すか)．$'
```

2.1.4 Eliminating unnecessary sentences. The sentences that match the pattern below are neither a question nor an answer.

```
r`ありがとうございました．|^(.{,5}の)*(見解|所見|答弁)を(お伺いし|求め)，再?質問を終わります．$|^以上.*(です|ます|でした|ました)．$|^(終了し|終わり)ます．|質疑を終えます．|(他|その他|残余)のご?質問|議員の(一般|代表|ご)?質問にお答えを?(申し上げ|いたし)ます．$|再質問を留保して，?質問を終わります．$|再質問に(ついて)?お答えを?いたします．$|^(.{,5}に|.+を代表して?.*)?，?再?質問を?(いた|留保)?します．$|以上で再?質問を終わります．$|お静かに願います．$|^ご清聴|議長，よろしくお願いします．$|発言する者あり'
```

2.1.5 Merging paragraphs. Finally, we merge paragraphs, because based on the rules above, too many one-sentence paragraphs are made. When a one-sentence paragraph matches the pattern below, this paragraph is merged with the next paragraph.

```
r`(^まず|^最初に|^初めに|^次に|^次いで|^最後に|^終わりに|^そこで).*(伺い[^，]*|関連?して伺い|お尋ね[^，]*|申し上げ)ます．$|.+について.*(です．|伺い[^，]*ます．|お(尋ね|答え)(いたし|し|させていただき)ます．|質問(を|にお答え)?(いた)?します．)|(についてで|質問[^，]*)ございま(した|す)．$'
```

When a one-sentence paragraph matches the pattern below, this paragraph is merged with the previous paragraph.

```
r`(こう|そう)した|お答えください．$|(お?伺い|お尋ね)を?(いたし|し|し?たいと思い)?ます．$|(いかがで|どうで)(しょうか|すか)．$|ではありませんか．$'
```

2.2 Features for matching

2.2.1 n-gram. The baseline provided by the task organizers uses character n-grams. In addition to this, we prepared word n-grams by morphological analysis, processed by MeCab[1]. Word n-grams are a set of morpheme n-grams excluding tokens. Similarities were calculated by counting the number of common n-grams between question and answer paragraphs. For the three features below, we used the same word n-grams.

2.2.2 BM25. BM25 models [6] were constructed on the morphemes excluding tokens, auxiliary verbs, and post-positional particles. BM25 values are high-dimensional sparse vectors that only have BM25 values at the existing morpheme. Cosine similarities between sparse vectors were used.

2.2.3 BERT. BERT [1] converts words in the sentence into embedded vectors. At the beginning of the sentence, a special token "CLS" is added, which represents the meaning of the sentence. The effectiveness of this vector was shown in the document classification task[1]. Here, cosine similarities between vectors corresponding to the "CLS" token were used.

2.2.4 Wikipedia2Vec. Wikipedia2Vec [7] can acquire the embedded vectors of words and entities appearing in Wikipedia by considering their similarities. It has been widely used for various tasks and particularly for QA tasks [4] because of its higher performance than word2vec. To convert word-wise vectors into a paragraph-wise vector, we took their average and then used the cosine similarities between averaged vectors.

2.3 Matching algorithm

We used the hospital and resident[2] [2] matching algorithm, which is the most basic algorithm, for matching question and answer paragraphs.

3 METHODS (QUESTION AND ANSWERING)

We took the approach of generating an answer summary from the original answer in minutes. Thus, we first tried to find the original question asked in the Tokyo Metropolitan Assembly from a question summary as input, and then used the results of QA Alignment to find the original answer.

3.1 Associating a question summary with the original question

Again, we used MeCab to tokenize both a question summary and candidate questions of the original question. For the calculation

[1]https://taku910.github.io/mecab/
[2]https://pypi.org/project/matching/

Table 1: Scores in terms of way of making paragraphs.

	F-value	Precision	Recall
Baseline	0.6166	0.5991	0.6437
New heuristic rules	0.7458	0.7606	0.7349

of the similarity between a question summary and questions, we reused the n-gram, BM25, and BERT features introduced in section 2.2. We add the Jaccard index as a new (fourth) feature. Additionally, we applied two-stage retrieval using BM25 and BERT. We first filtered top-ranked questions in descending order of BM25 scores and then reranked them by BERT.

3.2 Summarizing the answer

We utilized a commonly used transfer learning model, Text-to-Text Transfer Transformer (T5)[5], as the model of the summarizer. The pre-trained model of the summarizer was **sonoisa/t5-base-japanese** trained with a 100 GB Japanese corpus. We fine-tuned the model using answer–answer summary pairs extracted from the training data.

4 EXPERIMENTS (QA ALIGNMENT)

4.1 Experimental setup

We evaluated the performance by using “formal run” of the NTCIR-16 QA Lab-PoliInfo-3. BM25 models were trained using the distributed data[3] for “Himawari” derived from the minutes of the plenary session and budget committee of the national Diet. For BERT, we used a pre-trained Japanese model[4]. To obtain useful embedded vectors for document classification, it is necessary to fine-tune the pre-trained model with document classification tasks. We fine-tuned the pre-trained model using a news classification task[5] because we couldnot prepare an appropriate document classification task for this application. Wikipedia2Vec was trained according to the procedure in the website[6]. We prepared 100 and 300-dimensional vectors.

4.2 Results and discussion

Table 1 lists the scores in terms of the way of making paragraphs. The feature for matching was a character n-gram for both cases. Refinement of the heuristic rules was very effective, demonstrating a 13-point improvement in the F-value score.

Table 2 lists the scores in terms of the features for matching. Word n-gram improved the F-value by 2.2 points. BM25 significantly improved the F value by 8.9 points, which was the best performance among the “formal run” participants. BM25 was effective presumably because this model was trained on the minutes of the national Diet. BERT degraded the performance because the fine-tuning task was inappropriate for this application. Although

[3]https://csd.ninjal.ac.jp/lrc/index.php
[4]https://www.nlp.ecei.tohoku.ac.jp/news-release/3284/
[5]https://www.rondhuit.com/download.html#ldcc
[6]https://wikipedia2vec.github.io/wikipedia2vec/

Table 2: Scores in terms of features for matching.

	F-value	Precision	Recall
word n-gram	0.7677	0.7861	0.7533
BM25	**0.8348**	**0.8739**	**0.8045**
BERT	0.5968	0.6187	0.5799
W2V (100 dim)	0.6821	0.7139	0.6575
W2V (300 dim)	0.7382	0.7659	0.7160

Table 3: Scores in terms of QA.

	ROUGE-1 F-measure
word n-gram	0.2992
Jaccard index	0.2732
BM25	**0.3013**
BERT	0.1423
BM25 + BERT	0.1715

Wikipedia2Vec (W2V in the table) was worse than n-gram, the performance using a 300-dim vector outperformed that using a 100-dim vector, which demonstrates that a higher dimensional vector is effective.

5 EXPERIMENTS (QUESTION AND ANSWERING)

5.1 Experimental setup

We also evaluated the performance of the QA task by using a “formal run” of the NTCIR-16 QA Lab-PoliInfo-3. BM25 and BERT models were the same as in the previous experiment.

5.2 Results and discussion

The results in Table 3 show that BM25 performed the best among the compared methods. BERT did not work as well as for the QA Alignment task. Similarly, the two-stage retrieval approach (BM25 + BERT) was worse than BM25 only.

6 CONCLUSION

6.1 QA Alignment

In order to associate each question with its answer, we refined heuristic rules that make a paragraph and prepared three types of features for matching question and answer paragraphs. The refinement of heuristic rules improved the F-value by 13 points. Word n-gram and BM25 improved the F-value by 2.2 and 8.9 points, respectively. BERT and Wikipedia2Vec degraded the performance because training data were inappropriate.

6.2 Question and answering

We generated an answer summary from the original answer in minutes. In this method, we first find the original question from a question summary using similarity calculation, then identify the answer using the results of QA Alignment, and lastly summarize the answer with T5 to generate an answer summary. Experimental results showed that BM25 was the best term-weighting scheme.

REFERENCES

[1] 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*, Vol. 1. 4171–4186.

[2] D Gale and L Shapley. 1962. College Admissions and the Stability of Marriage. *Americam Mathematical Monthly* 92 (1962), 261–268.

[3] Yasutomo Kimura, Hideyuki Shibuki, Hokuto Ototake, Yuzu Uchida, Keiichi Takamaru, Madoka Ishioroshi, Masaharu Yoshioka, Tomoyoshi Akiba, Yasuhiro Ogawa, Minoru Sasaki, Kenichi Yokote, Kazuma Kadowaki, Tatsunori Mori, Kenji Araki, Teruko Mitamura, and Satoshi Sekine. 2022. Overview of the NTCIR-16 QA Lab-PoliInfo-3 Task. *Proceedings of The 16th NTCIR Conference* (6 2022).

[4] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics, Online, 803–818. https://doi.org/10.18653/v1/2020.findings-emnlp.71

[5] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. In *arXiv*.

[6] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Journal Foundations and Trends in Information Retrieval* (2009), 333–389.

[7] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, 23–30.