KASYS at the NTCIR-16 WWW-4 Task

Kota Usuha
University of Tsukuba
s2221635@s.tsukuba.ac.jp

Kohei Shinden
University of Tsukuba
s2221648@s.tsukuba.ac.jp

Makoto P. Kato
University of Tsukuba
mpkato@acm.org

Sumio Fujita
Yahoo Japan Corporation
sufujita@yahoo-corp.jp

ABSTRACT

The KASYS team participated in the English subtask of the NTCIR-16 WWW-4 task. This paper describes our approach of generating NEW runs, and REV runs in the NTCIR-16 WWW-4 task. We applied BERT machine reading comprehension model to the WWW-4 task for generating NEW runs. We investigated the effectiveness of reading comprehension model in the ad-hoc Web document retrieval task. The evaluation results showed that our run outperformed the baseline in the gold relevance assessment for the four runs we submitted. The evaluation results of REV runs showed that our runs in WWW-3 still well performed in WWW-4.

TEAM NAME

KASYS

SUBTASKS

English

1 INTRODUCTION

The KASYS team participated in the English subtask of the NTCIR-16 WWW-4 task [8]. This paper describes our approach of generating NEW runs, and REV runs in the NTCIR-16 WWW-4 task.

We applied our approach AIRRead, which uses machine reading comprehension model for relevance estimation in NEW runs. AIRRead generates questions from the query and estimates relevance by whether the answers to the questions are contained in the document. We investigated the effectiveness of reading comprehension model in the ad-hoc Web document retrieval task. The evaluation results showed that our runs outperform baseline on the gold relevance assessments except KASYS-CD-NEW-5.

We also applied our run submitted to the NTCIR-15 WWW-3 task [9]. We generate the REV RUN using the same approach that used in WWW-3 and the evaluation results showed that our runs in WWW-3 still well performed in WWW-4.

The remainder of this paper is organized as follows. Section 2 describes the approarch of NEW runs. Section 3 and 4 presents the WWW-4 result of NEW and REV runs, respectively. Finally, in Section 5, we conclude this paper.

2 NEW RUNS

The ad-hoc retrieval task, which is the central task of information retrieval, is the problem of sorting documents in order of increasing fitness for a given query. It captures the user's information request from the query and estimates the documents that satisfy the information request as high conformity. There is a different task called 'machine reading task'. This is a task that extracts the answer to a question from passage. For example, given a question 'What is the capital of Vietnam? and a passage 'Vietnam is a country in Southeast Asia, and its capital is Hanoi', the reading comprehension model is expected to extract 'Hanoi' as the answer to the question from the given passage.

machine reading comprehension has received attention in recent years compared to ad-hoc retrieval task, which is a problem that has been addressed for many years. in ad-hoc retrieval task, workshops on information retrieval, such as Trec Web Track and NTCIR, have been held since the 1990s. In recent years, datasets such as the MS MARCO (a large scale MAchine Reading COmprehension dataset) and the TREC Deep Learning Track are publicly available. On the other hand, with the advent of large datasets such as MS MARCO and SQuAD (Stanford Question Answering Datase), which are datasets for machine reading tasks, machine reading models have greatly improved their performance in resent years. A model based on fine-tuned BERT even achieved a better performance than human on the SQuAD dataset [4].

Although these problems have been tackled and developed separately, they share the same goal in terms of retrieving content that matches a given string. In ad-hoc retrieval, the goal is to retrieve documents from a set of documents that match a given query, and in machine reading tasks, the goal is to retrieve answers from text that match a question.

In this paper, we focus on the similarities between these two different problems and raise a question about whether one model can be used to solve the other, and in particular, we propose a method for solving ad-hoc retrieval tasks by using a reading comprehension model. To solve ad-hoc retrieval tasks with a machine reading model, we propose AIRRead (**A**d-hoc **I**nformation **R**etrieval model based on machine **Read**ing comprehension and question generation). AIRRead estimates the relevance of a document by generating the underlying question from a search query and determining whether it contains the answer to the question using a machine reading model. If one model can achieve the other's problem with sufficient accuracy, the development of one model would directly contribute to the other's problem, and more efficient technological progress could be made in the both fields. In this study, a reading comprehension model is used for information retrieval, and we expect the improvement of the performance of the reading comprehension model lead directly to that of the performance of the ad-hoc retrieval task. In addition, if a machine reading model can solve the ad-hoc retrieval task with sufficient accuracy, we can obtain ad-hoc retrieval models for languages for which there is no

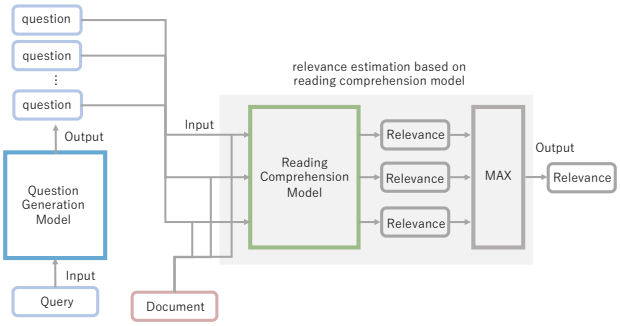

Figure 1: outline drawing of our framework

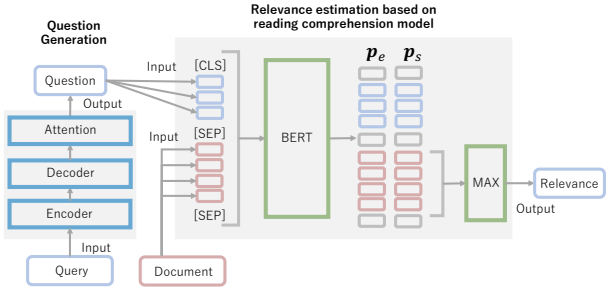

Figure 2: flow of the proposed method

ad-hoc retrieval dataset, as long as there is a reading comprehension model for those languages.

As a machine reading model, we use a pre-trained BERT model fine-tuned with SQuAD2.0, a dataset for machine reading and question answering tasks; SQuAD2.0 not only needs to extract answers in the machine reading task, but also to determine if they are answerable [6]. Under the assumption that a document is relevant if the answer to a question behind a query is likely to be contained in the document, we input a question and a document into the machine reading model, and used the estimated probability of the answer in the text as the degree of relevance. Since the method of generating questions from search queries is a string-to-string conversion, we considered it as a translation task and used an existing machine translation model. To train the machine translation model, we constructed a dataset in which queries and questions are paired by generating queries from questions. The questions used to build the dataset were derived from MS MARCO, a dataset for machine reading tasks. The ranking of documents by the machine reading model was done on the top 10 documents in the list ranked by BM25.

2.1 Model

In this section, we describe the problem of generating a query into a question and performing a relevance estimation by pre-trained reading comprehension model.

2.2 Problem Setting

Let D be document collection, We estimate the relevance score s_i of documents d_i based on given q_r and rank document in order of relevance score. However, We use trained reading comprehension model for estimating relevance score and return the top-K documents. We do not train reading comprehension model as ad-hoc retrieval task.

2.3 Framework

Figure 1 illustrates the framework we propose for solving ad-hoc retrieval task by reading comprehension model.

Given a query, we first retrieve an initial ranked list of documents D_{BM25} from the document set D with a search model that can be rapidly retrieved by indexing, such as BM25. The query q_r is transformed into a question q_s for input to the reading comprehension model, and the relevance score is estimated for D_{BM25} by the machine reading model f. Questions and documents are input to the reading comprehension model to obtain the relevance score s_i. The questions are generated based on the query by the question generation model g. When estimating the relevance of the ith document and query by the reading comprehension model and the question generation model, the relevance is estimated as follows.

$$q_s = g(q_r)$$

$$s_i = f(q_s, d_i)$$

Note that when translating a query into a question, the information need must be shared between query and question in order to capture the original information need. Also, if the information need of the query is ambiguous, it is possible to generate multiple questions. For example, given the query 'Wakayama tourism,' possible information need include 'I want to know about tourist attractions in Wakayama Prefecture,' 'I am looking for a page of the tourism department of Wakayama city office', 'I want to know about the tourism faculty of Wakayama University,' and so on.

In such a case, the relevance of a document is estimated for multiple questions, and the multiple relevance is aggregated into one, which is the final relevance score. When multiple questions are generated from a query and the maximum value is used as the relevance score as following.

$$Q_s = g(q_r)$$

$$s_i = \max_{q \in Q_s} f(q, d_i)$$

where Q_s is the set of generated questions. Generate a set of questions from the query with $g(q_r)$, and let $\max_{q \in Q_s} f(q, d_i)$, the maximum value of relevance obtained for each of the generated questions, be the relevance score of the document d_i to the query q_r. The final document list is created by sorting the documents based on the obtained relevance score.

2.4 AIRRead

In this section, we describe detail of AIRRead. Figure 2illustrates the flow of the proposed method.

Algorithm 1 Generate a query from a question

```
tokens ← Tokenize(question)
length ← GetLength()
length ← max(length, Length(question))
query ← {}
for all i ← 1 ... length do
  token ← arg max_{token∈tokens} IDF(token)
  query ← query ∪ {token}
end for

return query
```

2.5 Question Generation

In this paper, we generate questions from queries in order to use a pre-trained reading comprehension model. Several methods have been proposed to generate questions from queries, such as generating question templates from query logs and using LSTM for text transformation [2] [10] [5]. We regard the process of generating a question from a query as a text-to-text translation process [5]. Therefore, the machine translation model is used to translate the query into a question. We use Encoder-Decoder with attention mechanism for translation model. In order to train a machine translation model, paired data of queries and questions are required. In this paper, we constructed a dataset in which queries and questions are paired by generating queries from questions. The procedure for generating a query from a question is shown in Algorithm 1. Note that the idf of the token t was calculated as follows:

$$idf(t) = \log \frac{|\text{Questions}|}{|\{q_s : t \in \text{tokens}\}|}$$

where |Questions| is the total number of questions in the dataset, and tokens is the set of questions q_s divided into tokens. In order to generate a query from a question, The first step is to split the question into tokens (Tokenize(question)). Queries are generally shorter than questions, with the average length of a web query being 2.4 and about 76% of queries being 3 words or less [11]. Therefore, the length of the query to be generated is randomly determined to be uniformly distributed in the range of 1 to 3 (GetLength()). Note that the length of the generated query does not exceed the question. Since the query to be generated must share the information need of the question, the words to be used as the query are extracted from the words included in the question, and the words to be extracted are determined by the idf of the word under the assumption that the lower the frequency of occurrence of the word, the more information the word has. Then, for the number of times of the determined query length, one of the tokens in the question with the highest IDF value is extracted ($\underset{token \in tokens}{\arg\max}$ IDF(token)) and added to the set of queries (query ← query ∪ {token}).

2.6 Relevance Estimation Based On Reading Comprehension

In this section, we propose a method for estimating relevance in ad hoc search using a pre-trained reading comprehension model.

Typically, a reading comprehension model is used for a question answering task, which takes a passage and a question as input and extracts the answer to the question from the given passage [3]. The answer is presented as an span in the text, and the reading comprehension model outputs, for each token in the text, the probability that the answer span starts with that token, and the probability that the answer ends with that token.

In this paper, we use BERT as a reading comprehension model, where we input a sequence of questions and passages and output the probability of the beginning and end of the answer span for each input token. When inputting a question q_s and a passage a, we add [SEP] between the question and the passage and at the end of the input sequence, and [CLS] at the beginning of the input sequence. Let length of the question is $\text{len}(\text{q}_\text{s})$ and the passage length $\text{len}(\text{a})$, the length of input sequence will be $l = \text{len}(\text{q}_\text{s}) + \text{len}(\text{a}) + 3$. In this case, the output will be two probability distributions of length l. The probability that one starts the answer interval with the token $\mathbf{p_s} = (p_{s,1}, p_{s,2}, \ldots, p_{s,l})(\in \mathbb{R}^l)$, and the probability that the other ends the answer interval with the token $\mathbf{p_e} = (p_{e,1}, p_{e,2}, \ldots, p_{e,l})(\in \mathbb{R}^l)$.

In this paper, we compute the relevance score as the maximum of the probabilities corresponding to the passage of $\mathbf{p_s}$ obtained under the assumption that passage that can be judged to have an answer to a question are relevant. Compute the $\mathbf{p_s}$ corresponding to the token in the passage as $\mathbf{p_a} = (\mathbf{p_{s,j}}, \mathbf{p_{s,j+1}}, \ldots, \mathbf{p_{s,j+\text{len}(a)}})(j = \text{len}(\text{q}_\text{s}) + 2)$, the relevance score of the passage a in q_s, $s_{q_s,a}$, is calculated as follows.

$$s_{q_s,a} = \max_{1 \le i \le \text{len(a)}} p_{a,i}$$

In this paper, we fine-tune BERT with SQuAD2.0 [6]. SQuAD2.0 is a dataset for a reading comprehension task, but unlike SQuAD1.0, it includes questions that are unanswerable from the passage. Therefore, if a question is determined to be unanswerable, BERT is trained so that the position of the [CLS] token at the beginning of the input sequence becomes the answer interval. This means that if there is a high probability that the answer is unanswerable, the probability of guessing that the answer span starts at the [CLS] position will increase and the probability of guessing that the answer span starts at other positions will decrease. As already explained, the answer is computed from the probability corresponding to the tokens in the passage, so if the reading comprehension model can determine that there is no answer, the computed relevance score will also be low.

In an ad hoc information retrieval task, we need to compute the relevance of a document for a given query, but since the BERT reading comprehension model has an upper limit on the length of the input sequence, we cannot directly estimate the relevance of a document if the length of the input sequence from the query and the document exceeds the upper limit. To solve this problem, we divide the document into sentences so that the length of the input sequence does not exceed the upper limit, and estimate the relevance by inputting each sentence and question to the machine reading model. Since multiple relevance score are generated for one

Table 1: The statistics of the constructed query-question dataset

Data size	Average query length	Average question length
727,858	1.99	6.38

Table 2: actual example of query-question dataset.

Query	Question
stalin eastern why	why did stalin want control of eastern europe
nails rusty	why do nails get rusty
depona ab	depona ab
is world	is the atlanta airport the busiest in the world

question-document pair, this paper uses the maximum relevance score of each sentence as the relevance score of the document for the query. When we divide a question q_s and a document d_i into a set of sentences A and estimate the relevance, the relevance score $s_{q_s,i}$ of d_i is as follows.

$$s_{q_s,i} = \max_{a \in A} f(q_s, a)$$

In this paper, $s_{q_s,i}$ is used as s_i.

2.7 Experiments

In this section, we first explain the dataset to be used. Since the training of the question generation model involved the construction of a dataset, we describe the statistical information. Then we describe the baseline method, and finally we show the experimental results.

2.8 Dataset

In order to learn a model that translate a query into a question, we constructed a dataset of (query-question) pairs using MS MARCO questions. The construction procedure is as described in 2.5. The statistics of the constructed dataset is shown in Table 1 and a part of the actual data is shown in Table 2. As shown in Table 2, for questions with a question length of 3 or less, the generated query may match the question.

To evaluate the proposed method, we use NTCIR WWW-2 and WWW-3 as dataset.

2.9 Experimental Settings

As a baseline method in our experiments, we used BM25(www), which is provided as a baseline in NTCIR WWW-2 and WWW-3, BM25(our), which is a ranking method in BM25 used in the proposed method, and Birch, which is the method that achieved the best performance in NTCIR15 WWW-3 English SubTask. Birch is an ad-hoc retrieval model using BERT. Birch is a BERT-based ad-hoc retrieval model that estimates document relevance by dividing documents by sentences [1]. We use three evaluation metrics: MSnDCG@10, Q@10, and nERR@10. Q is a metric proposed by Sakai to evaluate a ranked list of documents, and Q is an evaluation metric that incorporates cumulative gain into the average precision and extends it to a form that can be used with multi-level conformance [7]. nERR is a normalized version of ERR, which is based on reciprocal rank and incorporating cascade user model, which the higher the relevance of a document, the more likely the user is to be satisfied with the document and to leave the search. For the question generation model, we used Encoder-Decoder with an attention mechanism and trained on the dataset with paired questions and queries described in 4.1. We generated multiple questions and performed relevance estimation, but since this did not contribute to the performance improvement, we generated one question from the query when generating questions. The relevance estimation by the reading comprehension model was done with 16 Batches.

As a variation of AIRRead, use the following configuration.

Partially A method to apply AIRRead only to queries that contain tokens that are proper nouns. For queries that do not contain proper noun tokens, the ranking of the documents in BM25 is output as is; details of the conditions for applying AIRRead are given in 2.10.1.

Handmade In order to measure the performance when questions are created by hand, we manually transformed the queries given in the English subtask of NTCIR15 WWW-3 and WWW-2 into questions, and then used the questions for relevance estimation. The transformation was done by the author, who read the description field describing the information need of the query and transformed the query into a question. This method is expected to measure the ability of ad-hoc retrieval on a trained reading comprehension model, independent of the performance of the question generation model.

WhatIs A method that a question to be input into a reading comprehension model is made into a question by adding 'What is' to the beginning of a given query, instead of being created by a question generation model. For example, if 'blue note' is given as a query, the question becomes 'What is blue note'.

$$s = \alpha \cdot s_{\text{AIRRead}} + (1 - \alpha) \cdot s_{BM25(\text{our})} \quad (1)$$

The hyperparameter α was fine-tuned in www-3 and determined to be $\alpha = 0.005$.

2.10 Experimental Results

Figure3 shows the experimental results of the baseline method and our proposed method. For both the WWW-2 and WWW-3 datasets, AIRRead outperformed BM25(www) on all the metrics compared, but underperformed Birch. However, when we look at the results for BM25 (our), we see that BM25 (our) performed better than AIRRead on all metrics. This suggests that AIRRead did not improve the performance of ad-hoc information retrieval by the reading comprehension model in AIRRead, since the final ranking of documents is output by reordering the ranking obtained by BM25 with the reading comprehension model.

Table4 shows the experimental result of AIRRead variation described in 2.9.

AIRRead (Handmade) was lower than AIRRead for all metrics in WWW-3, but higher than AIRRead for all metrics in WWW-2. Compared to the AIRRead (WhatIs) results, the WWW-3 results

Table 3: Experimental result of the baseline method and our proposed method.

Method	WWW-3			WWW-2		
	nDCG	Q	nERR	nDCG	Q	nERR
BM25(www)	0.575	0.585	0.676	0.326	**0.304**	0.478
BM25(our)	0.628	0.639	0.744	0.317	0.291	0.459
Birch	**0.694**	**0.712**	**0.796**	**0.334**	0.300	**0.486**
AIRRead	0.627	0.636	0.735	0.303	0.281	0.424

Table 4: the experimental result of AIRRead variation. * means the value when fine-tuned the method on the dataset.

Method	WWW-3			WWW-2		
	nDCG	Q	nERR	nDCG	Q	nERR
AirRead	0.627	0.636	0.735	0.303	0.281	0.424
AirRead(Handmade)	0.621	0.631	0.719	0.309	0.284	0.447
AirRead(Partially)	0.627*	0.635*	0.745*	**0.320**	**0.295**	**0.474**
AirRead(WhatIs)	0.621	0.632	0.713	0.308	0.287	0.436
AirRead(Handmade ∧ Partially)	0.629*	0.639*	0.749*	0.319	**0.295**	0.472
AirRead(WhatIs ∧ Partially)	0.627*	0.635*	0.737*	0.319	**0.295**	0.469

Table 5: Percentages of interrogatives in questions handmaded by the authors.

Interrogative	WWW-3	WWW-2
what	0.875	0.684
where	0.063	0.127
who	0.025	0.0633
when	0.013	0.0
how	0.013	0.038
which	0.013	0.089

were below AIRRead (WhatIs) for Q and nERR, and similar for MSnDCG; the WWW-2 results were also above AIRRead (WhatIs) for MSnDCG and nERR, except for Q. However, for both WWW-2 and WWW-3, the difference between AIRRead (WhatIs) and AIRRead (Handmade) for each indicator is small. Table 5 shows the percentage of interrogatives per dataset for the questions created by the authors. From this table, we can see that the majority of the questions created by the authors started with what, since the question with what is the most common question in both WWW-2 and WWW-3, and the difference from the next highest question is as large as 0.875 in WWW-3 and 0.684 in WWW-2. Of these, 87% of the questions in WWW-2 and 70 % in WWW-3 started with 'what is'. This suggests that the small difference in ratings between AIRRead (WhatIs) and AIRRead (Handmade) may be due to the fact that the questions used to estimate the relevance were largely similar.

AIRRead (BM25) outperforms AIRRead on all evaluation metrics for both WWW-2 and WWW-3. This result indicates that incorporating the BM25 scores into the scores obtained from the machine-readable model's conformance estimation contributes to the improved performance of AIRRead's ad hoc search. Although we rerank top 10 documents retreived with BM25, by explicitly including the word-based matching information in the score, the reranking of documents can reflect the word-based matching information that cannot be done by the machine reading model, which may result in higher performance.

AIRRead (Handmade ∧ Partially) is an adaptation of AIRRead (Handmade) only for queries containing tokens that are proper nouns, while AIRRead (WhatIs ∧ Partially) is an adaptation of AIRRead (WhatIs) only for queries containing tokens that are proper nouns. Comparing AIRRead (Handmade ∧ Partially) and AIRRead (WhatIs ∧ Partially), AIRRead (Handmade ∧ Partially) outperforms or equals AIRRead (WhatIs ∧ Partially) in all evaluation metrics for both WWW-2 and WWW-3. This result suggests that question improvement may contribute to the ranking results in the reranking of proper nouns.

From the ranking of documents in BM25 (our), we examine for each query how much the reranking by the reading comprehension model improves the ranking. We define the improvement rate of reranking by the reading comprehension model as follows.

$$\text{improvement rate} = \frac{\text{MSnDCG}_{\text{RC}}}{\text{MSnDCG}_{\text{BM25}}}$$

. where MSnDCG_{RC} is the MSnDCG of the ranking of documents by the reading comprehension model, and MSnDCG_{BM25} is the MSnDCG of the ranking of documents by BM25_{our}. When MSnDCG_{BM25} is 0, MSnDCG_{RC} is also 0, so the improvement rate is 0.

2.10.1 Partial adaptation of AIRRead. This section describes a method for partially adapting the AIRRead performed in AIRRead (Partially).

For the WWW-3 queries, we sorted the AIRRead (Handmade) rankings in descending order by improvement rate and examined the percentage of parts of speech in the top 20 and bottom 20 queries. Table 6 shows the results of sorting in descending order by the percentage of parts-of-speech in the overall, top 20 and bottom 20 queries and the difference between the percentage of each part-of-speech in the top 20 and bottom 20 queries. Since the difference between the top 20 and the bottom 20 in the improvement rate

Table 6: Percentage of parts-of-speech in the top 20 queries when the queries are sorted in descending order by improvement rate. The difference is the percentage of parts of speech in the top 20 queries minus there in the bottom 20 queries.

POS	all	top-20	bottom-20	top-20 - bottom-20
PROPN	0.238	0.486	0.245	+0.241
ADV	0.022	0.057	0.019	+0.038
OTHER	0.006	0.029	0.000	+0.029
VERB	0.066	0.057	0.038	+0.019
ADJ	0.077	0.086	0.094	-0.008
PART	0.028	0.000	0.019	-0.019
CCONJ	0.006	0.000	0.019	-0.019
DET	0.022	0.000	0.019	-0.019
ADP	0.050	0.029	0.075	-0.046
NOUN	0.459	0.257	0.472	-0.215

Table 7: Improvement rate of AIRRead and queries of WWW-3.

Query	Improvement	Has PROPN
kangaroo	1.498	✓
george washington university	1.440	✓
scorpions	1.339	
Pirates of the Caribbean	1.317	✓
zeus	1.285	
Texas Hold'em	0.841	✓
akron beacon journal	0.784	✓
Smart home	0.686	
Movies about animals	0.657	
internet pros and cons	0.565	

for proper nouns is the largest, we can consider that reranking by AIRRead is effective for queries containing proper nouns. For nouns, the difference between the top 20 and the bottom 20 is the smallest, but considering that the ratio of nouns to all queries in WWW-3 is as high as 0.459, more research is needed to conclude that AIRRead reranking has a negative effect on performance improvement for queries that contain nouns.

In our experiments on WWW-3, we observed that a relatively large percentage of queries containing proper nouns were in the top of the improvement rate. According to AIRRead (Partially) and AIRRead (Handmade ∧ Partially) in Table 4, the effectiveness of partial adaptation of AIRRead by proper nouns is also confirmed on WWW-2. Comparing AIRRead and AIRRead (Partially) on WWW-2, AIRRead (Partially) outperforms AIRRead in all evaluation metrics. Similarly, AIRRead (Handmade ∧ Partially) outperforms AIRRead (Handmade) on all metrics in WWW-2. AIRRead (Partially) also outperforms BM25 (our) on all metrics on WWW-2 in Table 3. This indicates that reading comprehension model contributes to the improvement of performance by selective reranking based on queries.

Table 7 shows the queries and improvement rates for the top five and bottom five in the list of AIRRead improvement rates on WWW-3, sorted in descending order. In the 'Has PROPN' column, ✓ is placed in the row if the query contains proper nouns. The table shows that there are queries do not contain proper nouns in the top five queries and do contain proper nouns in the bottom five queries. Since some of the queries with the highest improvement rates do not contain proper nouns, further improvement in performance can be expected if the queries can be classified so that they can be reranked by the machine reading model when they are given. Similarly, the fact that the bottom five queries include proper nouns indicates that outputting BM25 rankings for these queries as-is would improve the performance of AIRRead by reducing the negative impact of the machine reading model on the document ranks.

3 WWW4 RESULTS

Table 8 shows the results of the baseline method and our NEW runs in WWW4 on the gold assessment. We prepare a method to adapt the proposed method to the top 100 documents retrieved by BM25 as TOP100. We can see that our new runs are outperformed baseline except for KASYS-CO-NEW-5 from this table. KASYS-CO-NEW-4, which was created with question generation model and partial adaptation, achieved the highest score at all ealuation metrics among our runs. Similar to the results with WWW-2 and WWW-3, the method of partial adaptation with WWW-4 has the highest performance. On the other hand, Comparing KASYS-CO-NEW-3 and KASYS-CO-NEW-2, KASYS-CO-NEW-3 is a partial adaptation of KASYS-CO-NEW-2, but the improvement of performance is not clear.

Our method worked well for the gold assessment, but not for the bronze assessment. Table 9 shows the results of NEW runs in WWW4 on the bronze assessment. none of our approach outperformed the baseline in terms of nDCG and nERR. In contrast with the result of the gold assessment, KASYS-CO-NEW-1 and KASYS-CO-NEW-3 outperformed KASYS-CO-NEW-4 for the bronze assessment.

4 REV RUN

In English subtask, we submitted one revived run. To generated this run, we kept the same process and parameters as the WWW-2 run (KASYS-E-CO-NEW-1). Table 10 shows the results of our REV run (KASYS-CO-REV-6) and SLWWW REP (SLWWW-CO-REP-1) run on the gold assessment, and Table 11 shows the results on the bronze assessment. THUIR-CO-NEW-2 is the most successful run in terms of Mean Q, Mean nERR in gold relevance assessment, and all evaluation measure in bronze relevance assessment except Mean iRBU. We can see that our REV run is still well performed in WWW-4 and perform similarly with SLWWW REP run on all evaluation measure.

5 CONCLUSIONS

The KASYS team participated in the English subtask of the NTCIR-16 WWW-4 Task. Our NEW runs are based on a BERT based machine reading comprehension model, and outperformed the baseline. The result of REV runs suggest that our approach in WWW-3 still work well in WWW-4.

Table 8: result of the baseline method and our proposed method in WWW4 on the gold assessment.

Run	Method	nDCG	Q	nERR	iRBU
baseline		0.3205	0.2473	0.4541	0.7327
KASYS-CO-NEW-1	AIRRead	0.3294	0.2548	0.4769	0.7351
KASYS-CO-NEW-2	AIRRead(Handmade)	0.3273	0.2539	0.4747	0.7343
KASYS-CO-NEW-3	AIRRead(Handmade ∧ Partially)	0.3280	0.2538	0.4733	0.7348
KASYS-CO-NEW-4	AIRRead(Partially)	**0.3312**	**0.2566**	**0.4971**	**0.7351**
KASYS-CO-NEW-5	AIRRead(Handmade ∧ TOP100)	0.2879	0.2086	0.4580	0.7206

Table 9: result of the baseline method and our proposed method in WWW4 on the bronze assessment.

Run	Method	nDCG	Q	nERR	iRBU
baseline		0.5170	0.4806	0.6711	0.8920
KASYS-CO-NEW-1	AIRRead	0.5147	0.4842	0.6519	0.8905
KASYS-CO-NEW-2	AIRRead(Handmade)	0.5090	0.4733	0.6427	0.8902
KASYS-CO-NEW-3	AIRRead(Handmade ∧ Partially)	0.5130	0.4799	0.6629	0.8922
KASYS-CO-NEW-4	AIRRead(Partially)	0.5025	0.4658	0.6384	0.8912
KASYS-CO-NEW-5	AIRRead(Handmade ∧ TOP100)	0.4097	0.5666	0.5666	0.8399

Table 10: result of the REV run and our proposed method in WWW4 on the gold assessment.

Run	nDCG	Q	nERR	iRBU
THUIR-CO-NEW-2	0.3670	0.2944	0.5289	0.7544
SLWWW-CO-REP-1	0.3686	0.2886	0.5098	0.7840
KASYS-CO-REV-6	0.3682	0.2890	0.5098	0.7811

Table 11: result of the REV run and our proposed method in WWW4 on the bronze assessment.

Run	nDCG	Q	nERR	iRBU
THUIR-CO-NEW-2	0.6249	0.5857	0.7967	0.9028
SLWWW-CO-REP-1	0.5846	0.5629	0.7537	0.9397
KASYS-CO-REV-6	0.5931	0.5743	0.7634	0.9424

REFERENCES

[1] Zeynep Akkalyoncu Yilmaz, Wei Yang, Haotian Zhang, and Jimmy Lin. 2019. Cross-Domain Modeling of Sentence-Level Evidence for Document Retrieval. 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)*. Association for Computational Linguistics, 3490–3496.

[2] Gideon Dror, Yoelle Maarek, Avihai Mejer, and Idan Szpektor. 2013. From Query to Question in One Click: Suggesting Synthetic Questions to Searchers. In *Proceedings of the 22nd International Conference on World Wide Web* (Rio de Janeiro, Brazil) *(WWW '13)*. Association for Computing Machinery, New York, NY, USA, 391–402. https://doi.org/10.1145/2488388.2488423

[3] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1* (Montreal, Canada) *(NIPS'15)*. MIT Press, Cambridge, MA, USA, 1693–1701.

[4] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[5] Adarsh Kumar, Sandipan Dandapat, and Sushil Chordia. 2018. Translating Web Search Queries into Natural Language Questions. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan. https://aclanthology.org/L18-1151

[6] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, 784–789. https://doi.org/10.18653/v1/P18-2124

[7] Tetsuya Sakai. 2004. New Performance Metrics Based on Multigrade Relevance: Their Application to Question Answering. *NTCIR-4 Proceedings*.

[8] Tetsuya Sakai, Sijie Tao, Zhumin Chu, Maria Maistro, Yujing Li, Nuo Chen, Nicola Ferro, Junjie Wang, Ian Soboroff, and Yiqun Liu. 2022. Overview of the NTCIR-16 We Want Web with CENTRE (WWW-4) Task. In *Proceedings of NTCIR-16*. to appear.

[9] Kohei Shinden and Atsuki Maruta. [n.d.]. KASYS at the NTCIR-15 WWW-3 Task.

[10] Zhao Shiqi, Wang Haifeng, Li Chao, Liu Ting, and Guan Yi. 2011. Automatically Generating Questions from Queries for Community-based Question Answering. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, Chiang Mai, Thailand, 929–937.

[11] Amanda Spink, Dietmar Wolfram, Major B. J. Jansen, and Tefko Saracevic. 2001. Searching the web: The public and their queries. *Journal of the American Society for Information Science and Technology* 52, 3, 226–234. https://doi.org/10.1002/1097-4571(2000)9999:9999<::AID-ASI1591>3.0.CO;2-R