KANDUH at the NTCIR-17 Transfer Task

Tomoya Hashiguchi  
KDDI Corporation  
Japan  
tohashiguchi1995@gmail.com

Ryota Mibayashi  
University of Hyogo  
Japan  
af22h007@guh.u-hyogo.ac.jp

Huu-Long Pham  
University of Hyogo  
Japan  
af23a009@guh.u-hyogo.ac.jp

Wakana Kuwata  
University of Hyogo  
Japan  
ad23g022@guh.u-hyogo.ac.jp

Yuka Kawada  
University of Hyogo  
Japan  
ad22l019@guh.u-hyogo.ac.jp

Yuya Tsuda  
University of Hyogo  
Japan  
ad22o041@guh.u-hyogo.ac.jp

Takehiro Yamamoto  
University of Hyogo  
Japan  
t.yamamoto@sis.u-hyogo.ac.jp

Hiroaki Ohshima  
University of Hyogo  
Japan  
ohshima@ai.u-hyogo.ac.jp

ABSTRACT
The KANDUH team participated in the Transfer subtasks 1 and 2 of NTCIR-17. In this paper, we report on our approach to solving the problem and the results. Subtasks 1 and 2 address the dense vector search task, respectively. In both subtasks 1 and 2, we used BM25 to filter documents, followed by dense vector retrieval. The method with the highest nDCG@20 was 0.4339 the one that first finetuned DeBERTa-v2 with MSMARCO and then additionally finetuned with NTCIR-1 data. On the other hand, the method with the lowest nDCG@20 was 0.0751 the one that fine-tuned only MSMARCO data.

KEYWORDS
BM25, BERT, DeBERTa-v2, GPT-3

TEAM NAME
KANDUH

SUBTASKS
Dense First Stage Retrieval  
Dense Reranking

1 INTRODUCTION
The Transfer task of NTCIR-17 is intended to develop dense search technology. Subtask 1 is a task to perform dense search for NTCIR-2 utilizing a model trained using NTCIR-1 document and query data as existing data. Subtask 2 is a task that utilizes the same data as Subtask 1 to re-rank pre-ranked documents using dense search techniques [1].

2 DATASETS
We used NTCIR1 data and MSMARCO data [2] for fine tuning. For fine tuning, we set a positive example for a RELVANCE greater than 1 and a negative example for all other cases, so as to fine tune the classification task. The training and validation data were generated by random splitting the fine tuning data set so that the ratio of training/validation = 9:1.

3 PROPOSED METHOD
We addressed subtasks 1 and 2 in the following sequence.
- Filtered documents for search by BM25
- Dense vector search by query against filtered documents

The data from Subtask 2 was used to filter the data in BM25. The following model was used to create the dense vectors.
- Vectorize text using BERT model
- Vectorize text using DeBERTa-v2 model
- Vectorize text using text-embedding-ada-002 model
- Fine tuning of DeBERTa-v2 model with cross-encoder technique
- Fine tuning of DeBERTa-v2 model with bi-encoder technique

The Azure embedding model text-embedding-ada-002 with api-version of 2022-12-01 was used.

3.1 BM25
We used BM25 for document filtering in 1st subtask and 2nd subtask in common. The top 1000 documents provided in 2nd subtask were used for document filtering using BM25 in 1st subtask. As shown in the figure 1, even when there are less than 1,000 target documents, no documents are added. Therefore, the data and methods handled are the same for 1st subtask and 2nd subtask, with the only difference being the number of top-ranked cases to be obtained. Since the nDCG@1000 results for the NTCIR-1 Ad-Hoc/CLIR Test Collection 83 topics, when creating document rankings using BERT for vectorization and cosine similarity, improved from 0.10 to 0.639, it was decided to perform filtering using BM25.

3.2 Embedding Model
Run id 1 and 2 were employed as a baseline. The methods used were the general BERT and OpenAI’s ada002, which has been the focus of much attention in recent years. Run id 3, 4, and 10 were employed to verify what would happen if hybrid results with existing keyword search were used instead of dense vector search alone. Run id

1 https://huggingface.co/cl-tohoku/bert-base-japanese-whole-word-masking
2 https://huggingface.co/ku-nlp/deberta-v2-base-japanese
5-7 and 8, 9 were employed to compare the cross-encoder and bi-
encoder handled in the dense vector search. DeBERTa-v2, which
is considered to be a highly accurate Japanese language model,
was used as the base model for fine tuning. Run id 5, 6, and 7 are
cross-encoders to verify the effect of changing or adding training
data. Run id 8 and 9 examine the effect of incorporating ada002 into
the model calculated by the bi-encoder.

Run id 1-4 were ranked based on the similarity between the
query and document vectors without model fine tuning. IndexFlatIP
from faiss ⁴, provided by Meta, was used to calculate the similarity
between queries and documents. We used FAISS because it can
output similar vectorized documents from a vectorized query with
ranked results.

For run id 1, documents and queries were vectorized using Japan-
ese BERT provided by Tohoku University. Other BERT models ⁵
provided by Tohoku University were not validated in this effort.
The query and document vectorization process was performed by the
"model.encode(text)" process using SentenceTransformers ⁶.

Run id 2 was vectorized documents and queries using the em-
bedding model text-embedding-ada-002 provided by Azure. Models
other than text-embedding-ada-002 ⁷ were not validated in this ef-
fort. The vectorization process was performed by inputting queries
and documents to the Azure API and using the output obtained. If
the input length exceeded the upper limit of 8,192 tokens, anything
over 8,192 was excluded.

Run id 3 was re-ranked using the inverse of the ranking produced
by run id 1 and the inverse of the ranking produced by BM25. Each
coefficient was validated with nDCG@1000 using the NTCIR-1 Ad-
Hoc/CLIR Test Collection 83 topics, and the ratio with the highest
score from Table 1 was used.

\[
\text{ranking} = (\alpha \cdot \frac{1}{\text{BERT}\text{\text{rank}}} + ((1 - \alpha) \cdot \frac{1}{\text{BM25}\text{\text{rank}}})
\]

Run id 4 was re-ranked using the inverse of the ranking created
by run id 2 and the inverse of the ranking created by BM25. Each
coefficient was validated with nDCG@1000 using the NTCIR-1 Ad-
Hoc/CLIR Test Collection 83 topics, and the ratio with the highest
score from Table 1 was used.

\[
\text{ranking} = (\alpha \cdot \frac{1}{\text{ada002}\text{\text{rank}}} + ((1 - \alpha) \cdot \frac{1}{\text{BM25}\text{\text{rank}}})
\]

Runs 5-10 were ranked by fine-tuning DeBERTa-v2 and ranking
the model predictions as similarity scores between queries and
documents. Run id 5 was ranked using a model of DeBERTa-v2
fine-tuned with NTCIR1 data. Fine tuning was performed as shown
in Figure 2. Similarity scores between queries and documents were
calculated and ranked using the model trained by fine tuning. Run
id 6 had a different dataset for fine tuning than run id 5, and fine
tuning was performed on MSMARCO data. Run id 7 was further
fine-tuned with NTCIR1 data on the model of run id 6.

Run id 8 is a SentenceBERT model based on DeBERTa-v2. In
conventional BERT, two sentences are input simultaneously by
connecting them with a [SEP] token, but in SentenceBERT, two
sentences are input into BERT one at a time. This time, the BERT
part was changed to DeBERTa-v2. DeBERTa-v2 is a better per-
forming model than RoBERTa, which is an improved model of
BERT. DeBERTa-v2 uses a pre-trained model published by Kyoto
University ⁸. A query and a document are input to DeBERTa-v2, re-
spectively, to obtain their respective outputs. Average pooling was
performed on those outputs. The two vectors obtained were placed
in a classification layer and a binary classification was performed
to determine whether they were related or not. The data used in
the training were the ntcir1 data with the relevance assigned as
positive examples, and the other data as negative examples. In other
words, the strength of the relevance was not taken into account.

Run id 9 is a DeBERTa-v2 based SentenceBERT model with
ChatGPT embeddings. Similar to run id 8, DeBERTa-v2 used a
pre-trained model published by Kyoto University. We input query
and document pairs, respectively, and average-pooled the resulting
output. For this 768-dimensional vector, the ChatGPT embeddings

---

Footnotes:

⁴https://github.com/facebookresearch/faiss
⁵https://huggingface.co/cl-tohoku
⁶https://www.sbert.net/s
⁸https://huggingface.co/ku-nlp/deberta-v2-base-japanese
associated with the query and document were added together. However, since ChatGPT embeddings are 1536-dimensional vectors, they were converted to 768 dimensions and then added to the vector obtained from DeBERTa-v2. The DeBERTa-v2+ChatGPT vectors were then input to the classification layer and classified as relevant or not as well.

Run id 10 was ranked using the inverse of the ranking created by run id 7 and the inverse of the ranking created by BM25. Each coefficient was validated with nDCG@1000 using the NTCIR-1 Ad-Hoc/CLIR Test Collection 83 topics, and the ratio with the highest score from Table 1 was used.

$$\text{ranking} = (\alpha \cdot \frac{1}{\text{BERTrank}}) + ((1 - \alpha) \cdot \frac{1}{\text{BM25rank}})$$

Run id 5, 6, and 7 used the fine-tuned model as shown in Figure 3 and were ranked according to their scores. Run id 8 used the fine-tuned model as shown in Figure 4 and ranked according to score. Run id 9 used the fine-tuned model as shown in Figure 5 and created a ranking according to the score.

The hyperparameters for the fine tuning of runs id 5-7 were set to the following

- Optimizer: AdamW,
- Learning rate: 2e-5,
- Batch size: 28,
- Loss function: Cross-entropy loss,
- Early stopping: patience: 5
- Tokenizer Max Length: 256
- Epoch num: 40

Since the NTCIR2 document text could not be partially processed by Juman++, DeBERTa-v2’s tokenizer, when creating the run evaluation data, the number of characters in the document data is limited to 512 for processing.

## 4 RESULTS AND DISCUSSION

The evaluation results for subtask 1 are shown in Table 2. The methods using BERT and OpenAI ada-002 (run_id 1-4) were not evaluated because no conforming documents existed. Therefore, the DeBERTa-v2 model was fine-tuned using the bi-encoder method (run_id 8 and 9) and the cross-encoder method (run_id 5, 6, 7, and 10). Comparing nDCG@20 with these two methods, the cross-encoder method was superior to the cross-encoder method. The superiority of the cross-encoder method may be related to the fact that the original BERT model was trained with the cross-encoder
method. The method with the highest nDCG@20 was the one that first finetuned DeBERTa-v2 with MSMARCO and then additionally finetuned with NTCIR-1 data. On the other hand, the method with the lowest score was the method that fine-tuned only MSMARCO data. This may be due to the fact that MSMARCO and the NTCIR-2 document set under evaluation are not similar. Given these facts, it was effective to first fine-tune the model in MSMARCO as a search task, and then perform additional fine-tuning in NTCIR-1 to recognize the domain. Since the data for both subtasks 1 and 2 were documents filtered by BM25 for ranking, it was assumed that the results would remain the same, but the method of fine-tuning the MSMARCO data alone resulted in different results. To compare the results, the results for run_id6 for each query are shown in Figure 6. From Figure 6, it can be seen that the results of some of the queries have different scores. Among them, the query ID with a higher score in subtask 1 was 118, and the query content was "distance education using videoconferencing system."

We compare nDCG@1000 and MRR, the indicators in each subtask. Looking at subtask 1, nDCG@1000, the results are correlated with nDCG@20, but the most superior results are different. However, run_id10 only changed the method of calculating the score of the best method in nDCG@20 to also consider the score of BM25, and the model itself remains the same. Next, looking at the MRR for subtask 2, the results are also correlated with nDCG@20, but the best results are different. The method in run_id5 is only a fine tuning of DeBERTa-v2 with the ntcir1 data. These suggest that fine tuning to recognize the domain was very effective, as per the considerations obtained in nDCG@20.

5 CONCLUSIONS
We participated in the Transfer subtasks 1 and 2 of NTCIR-17. In this paper, we report on our approach to solving the problem and the results. Subtasks 1 and 2 address the dense vector search task, respectively. In both subtasks 1 and 2, we used BM25 to filter documents, followed by dense vector retrieval. The method with the highest nDCG@20 was 0.4339 the one that first finetuned DeBERTa-v2 with MSMARCO and then additionally finetuned with NTCIR-1 data. On the other hand, the method with the lowest nDCG@20 was 0.0751 the one that fine-tuned only MSMARCO data. These suggest that fine tuning the model first with MSMARCO as a search task, followed by additional fine tuning with NTCIR-1 to recognize the domains, was effective.

ACKNOWLEDGEMENTS
This work was supported by JSPS KAKENHI Grant Numbers JP21H03775, JP21H03774, JP21H03554, JP22H03905.

REFERENCES