Various Uses of Pre-trained Language Model for a Knowledge-grounded Task Oriented Dialogue System

Junhee Cho[1], Taesuk Hong[2], Choongwon Park[1], Jaeah You[1], Youngjoong Ko[1]*, Kwanyoung Son[3]

[1]Sungkyunkwan University, [2]Sogang University, [3]LG Electronics
{chojunhee7003, lino.taesuk, pcw1102, youjaeah}@gmail.com
yjko@skku.edu, kwanyoung.son@lge.com

Abstract

The tenth dialog system technology challenge (DSTC10) Track2 sub-track#2 is the knowledge-grounded task-oriented dialogue modeling based on spoken conversations, and is equivalent to Track1 of DSTC9 except that spoken dialogs are used. Unlike the existing standardized database, the external knowledge of the challenge is organized in the form of frequently asked questions (FAQs). Therefore, participants need to model a dialogue system that effectively finds connections between external knowledge and conversations. In this study, we present several methods for using a pre-trained language model to solve the problem . For task#1, RoBERTa determines whether to use external knowledge as a binary classification. Task#2 has been divided into three steps. RoBERTa and BERT perform the tasks of multi-class classification, machine reading comprehension, and sentence similarity. Finally, in task#3, we generated a natural language response using BART. Our proposed method achieved the best performance for F1-score of task#1 and ranked fifth in objective evaluation and fourth in the human evaluation of sub-track#2 of DSTC10's Track2.

1 Introduction

The task-oriented dialogue system is closely related to real life and helps users complete certain tasks in certain areas, such as restaurant and hotel reservations. Task-oriented dialogues are generally configurations in which users ask questions and systems respond in a manner similar to frequently asked questions (FAQs). Furthermore, to respond fluently and correctly to the user's questions, the system must generate a response by drawing on information about external knowledge (or a database) related to the topic of the conversation. In this case, the system should access and use information from the external knowledge that is suitable for the topic and flow of the conversation. Track2 of DSTC10, "Knowledge-grounded Task-Oriented Dialogue Modeling based on Spoken Conversations" encourages you to access external knowledge and maintain high-quality language during the conversation.

Sub-track#2 of DSTC10's Track2 divides a series of dialog processes into three tasks, shown in Figure 1. The first task is to determine whether to use the external knowledge information when the system answers the user's last utterance (question). Here, the external knowledge consists of FAQs related to a specific entity, such as hotel and restaurant names, and to a specific domain. The second task occurs only when it is determined that the external knowledge information is being used. In this task, the system must select the most appropriate knowledge among the candidates by considering the conversation of the knowledge-seeking turn. Finally, in the third task, the knowledge selected in the second task is used to generate a fluent and informative response (answer). Unlike the previous version of DSTC9 Track 1, where the dialog data is only in written form, DSTC10 Track 2 the validation and test data is the dialog of automatic speech recognition (ASR) output.

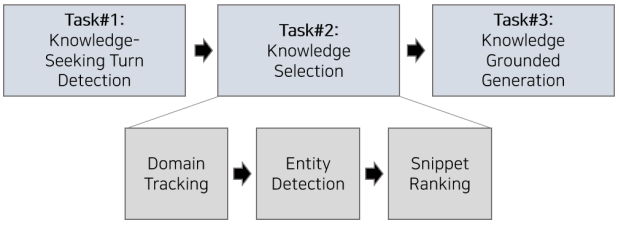
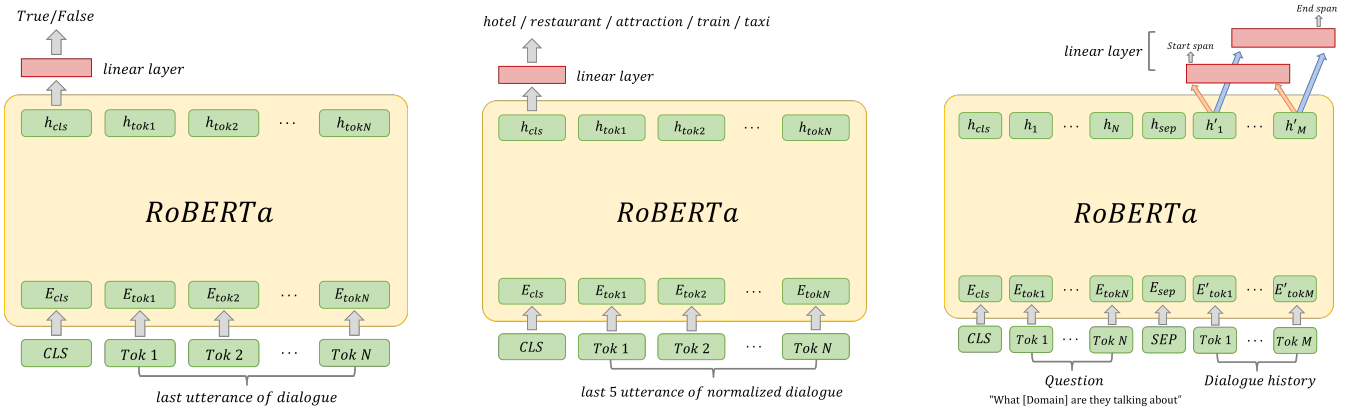


Figure 1: Dialogue process in Track2 sub-track#2. Our approach split task#2 into three steps.

Therefore, the focus of this challenge is to solve the problem appropriately for each task and develop a model that is robust for spoken data. For knowledge-seeking turn detection (Task#1), the binary classification problem should be solved using a simple pre-trained language model such as RoBERTa. Moreover, the results of this task can be modified according to the results of entity detection or snippet ranking. In knowledge selection (Task#2), it is challenging to determine the required knowledge from a large number of knowledge candidates. Therefore, we create three independent steps, entity detection, domain tracking and snippet ranking, to identify the entity, snippet and the domain of the dialogue, respectively. The predicted results of the first two steps are in turn used to reduce the search scope of the knowledge candidates. Our knowledge selection model then selects the most suitable knowledge from the given knowledge candidates. Finally, knowledge was selected using var-

*Corresponding author



(a) KSTD model (b) Domain tracking model (c) Entity detection model

Figure 2: Various inputs and tasks using three RoBERTa models

ious ranking methods, including using the cosine similarity score and the logits of [CLS] tokens from RoBERTa. Consequently, in knowledge grounded generation (Task#3), a denoising auto encoder for pre-training the sequence-to-sequence model, BART was used to generate knowledge-grounded responses. Our proposed method achieved the best performance for F1 score of task#1, ranking fifth in the objective evaluation, and was ranked fourth in the human evaluation of sub-track#2 of DSTC10's Track2.

2 Methodology

This section describes the proposed method in detail. It consists of five subsections and describes which model and method were used for each task.

2.1 Knowledge-seeking turn detection (KSTD)

Task#1 requires a determination of whether to use external knowledge to generate the last system response for a given dialog. Since all the subsequent tasks after task#1 (KSTD) depend on task#1, this task should be treated carefully, and two steps are developed for this task. 1) Binary classification step (see section 'Binary classification'), and 2) filtering depending on whether the entity appears, and where it comes from. The two steps are explained in detail below.

Binary classification To decide whether external knowledge should be used, we found that the model needs to focus only on the last utterance of the dialogue rather than the entire dialog history, since the last utterance of a knowledge-seeking turn dialog is always the user's question about the topic or entity. The rest of the utterances before the last utterance were more likely to be distractions, rather than helpful. Furthermore, since the external knowledge used in the training and validation/test phases is the same, we can expect it to work correctly in the validation and test phases if the system can learn patterns between the last utterance and the knowledge snippets. We use RoBERTa (Liu et al. 2019), which performs well among pre-trained language models, to solve a binary classification problem that determines whether to use the external knowledge for the last utterance as true or false. The last utterance is given as a sequence of tokens for the input of the model, as shown in Figure 2a. The output hidden vector of the first token (token with the meaning of [CLS] token from BERT (Devlin et al. 2019)) is used as input to the classifier to select knowledge-seeking turn.

Additional Filtering Additional filtering is a method to withdraw that decision from the previous binary classification of the use of external knowledge by certain conditions. If entity detection does not find a matching entity among the offered entities or finds an entity that occurs in an incorrect source, the use of external knowledge is aborted. Even if the correct entity is found, but the snippet related to the last utterance cannot be selected, the use of external knowledge is also aborted. The detailed processes and conditions for each method are described in the entity detection and snippet ranking sections.

2.2 Domain Tracking

External knowledge has different entities in each of the five domains and multiple pairs of question answers (knowledge snippets). Therefore, determining the domain of the dialog is the first way to select the correct knowledge.

To begin domain tracking, we first balanced the amount of data in each domain. The five domains covered in DSTC10 are Hotel, Restaurant, Attraction, Train, and Taxi. Domain tracking model should completely distinguish the dialog domain from the five domains in the domain tracking step. However, if the amount of data prepared for each domain differs significantly in the training data, the model can be biased toward domains with a lot of data. In the training data, the amount of data contained in the attraction domain is much smaller than the data contained in the other four domains. We needed to adjust the data amount of the other four domains to be close to the amount of data contained in the attraction domain. Furthermore, noisy utterances which can occur in all domains were removed from the dialogs to distinguish the domains well. For example, a dialog turn where the user and system ask question and answer regarding "zip, zip code, phone, and phone number" was removed.

Algorithm 1: Make Alias

Input: ground-truth entity e
Output: list of entity alias $alias$
Requires: list of words that represent domain dom

Let $e' \leftarrow normalized\ entity$
$alias$.append(e)
$alias$.append(e')
for d in dom **do**
  if e contain a word "d" **then**
    $alias$.append(d + e')
    $alias$.append(e' + d)
  end if
end for
return $alias$

In the dialog history of the training data, dialogs do not have only one domain, and several domains may occur. Certainly, the domain that appeared earlier in the dialog and the domain that appeared later may be different. Therefore, we tried to identify the domain only by the last five utterances of the dialogue.

For domain tracking, the 5-domain classification task was performed using RoBERTa. The last five utterances of the dialog in the form of tokens, are used as input and the output hidden vector of the [CLS] token is used as the input of the above classifier (Figure 2b).

2.3 Entity Detection

The second step in selecting appropriate external knowledge is to determine the entity to which the user will eventually refer. In this case, the entity is the name of a hotel, restaurant, or attraction. The knowledge snippets in the train and taxi domains are domain-wide, where each knowledge snippet has no related entity. Therefore, entity detection occurs only when the domain tracking model predicts hotels, restaurants, and attractions, that are domains containing entity-specific knowledge snippets.

Since the domain can change in a conversation, multiple entities can be referred to in a single dialogue. For instance, when a user asks, *"Can you recommend a restaurant that serves Korean-food?"* The system can recommend a variety of restaurants, but only one of them is the restaurant that the user chooses. So instead of paying attention to all the entities that appear in the dialog, we need to determine the most important entities that appear in the dialog.

We solved this problem by considering entity detection as a machine reading comprehension task (MRC) (Zeng et al. 2020). The MRC is a task that takes questions and documents as inputs and finds a span that corresponds to the answer in the document. MRC is more robust than the string match method because it detects the position of a possible entity, allowing it to detect an entity with typos or aliases. The entity detection data was preprocessed for input to the MRC task. The start and end positions of the gold entity span are given as the label. In cases where the entity only appears as an alias in the dialogue history, the start and end positions of the alias are provided as labels. For the input, the document part was constructed by appending the dialog history, and the question was created by generating a sentence "What **[DOMAIN]** are they talking about?". Then, the predicted result from domain tracking model was inserted in the **[DOMAIN]**. Therefore, the entity detection model can carefully search only the entities of the predicted domain, even if there are multiple domain entities in the dialog. Similar to the conventional MRC models that use a pre-trained language model, RoBERTa is given a concatenated string of the question and dialog history as input. Then RoBERTa's last layer hidden vector of each token go into two linear layers that score the start and the end span, respectively (Figure2c).

Alias of entity When an entity appears in a dialog, it does not always appear exactly as the ground-truth entity (especially because of spoken data, making it more vulnerable to proper nouns). Therefore, it is difficult to automatically label the entity-detection training data or match the model's results with a ground-truth entity. To solve this problem, we created an alias in the ground-truth entity. The process of creating an alias is presented in Algorithm 1. *"dom"* is a list of words that directly represent a domain. For instance, 'hotels,' 'guest houses,' and 'motels', are words that directly represent the hotel domain. Likewise, "restaurants" and "cuisine" can be the elements of *"dom"* for the restaurant domain. Normalized entities are ground-truth entities without the above words so that only the entity's unique words (or proper nouns) remain. Using the method of Algorithm 1, various alias reflecting the domain information for each entity were created.

Levenshtein Distance Levenshtein distance (Levenshtein 1966) is a string distance metric that defines the similarity between two words or sentences. The levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another. In addition, we can calculate the similarity ratio based on the levenshtein distance. We use this ratio to match the output of the entity detection model with a ground-truth entity. Because the given data is spoken data, there can be many typos or mistakes. As a result, we provide some flexibility in detecting an entity using fuzzy string matching.

Additional filtering to KSTD Furthermore, some of the detected entities may not be present in the knowledge.json[1] but exist only in db.json[2]. Therefore, if the model predicts the entity that exists only in db.json, the question cannot be answered using external knowledge. In this case, the result of the KSTD is modified to "false." This method can mitigate the weakness of KSTD in determining whether to use external knowledge based on the characteristics of the only last utterance regardless of the entity.

2.4 Snippet Ranking

The final step in knowledge selection is to select up to five knowledge snippet candidates. Since the domain and entity

[1]containing the unstructured knowledge sources, used in sub-task#2

[2]database of entities, used in sub-task#1

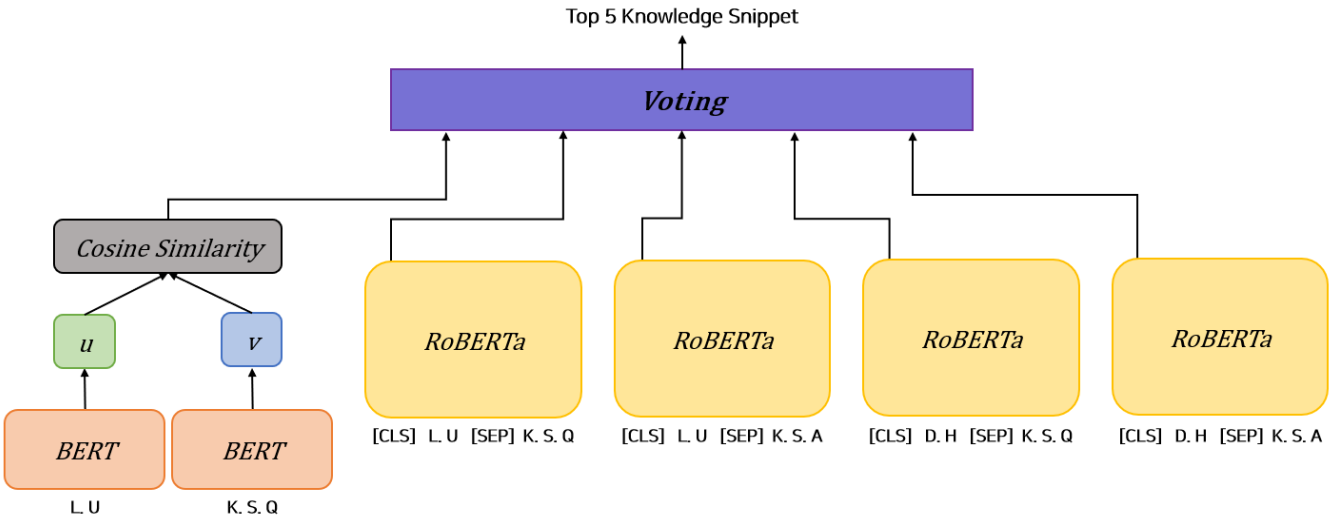


Figure 3: Overall structure of ensemble for snippet ranking.'L.U,' 'D.H,' 'K.S.Q,' and 'K.S.A' are the abbreviations for 'Last Utterance,' 'Dialogue History,' 'Knowledge Snippet Question,' and 'Knowledge Snippet Answer' respectively.

were determined in the previous step, the candidates can be selected within a narrow range of the predicted domain and entity. We created a final model with an ensemble of five models. One of them is a BERT-based model in the form of a bi-encoder provided by the Sentence-Transformers (Reimers and Gurevych 2019) library. The remaining four models were developed by RoBERTa, and each model had different types of input. The final top five snippets were determined by the voting results of the five models.

Finally, the model attempts to recheck the top five ranks. RoBERTa was used for this re-ranking task, which takes the last utterance and each of the top five knowledge snippets as input. Using the [CLS] token, we predict "true" if the two sentences are similar and "false" if they are not similar. Finally, the first knowledge snippet predicted to be "true" by the re-ranker becomes the final selected knowledge snippet.

Suppose that the entity is not specified in the entity detection at the time of reference. In this case, the similarity with the last utterance is computed for all snippets included in the domain predicted by domain tracking. However, it is very time-consuming to run all large language models for all snippets. Therefore, we pre-encoded the entire knowledge snippet using the *spacy sentence encoder* (Honnibal and Montani 2017), calculated the similarity with the last utterance, and selected the top ten snippets to include in the final model.

Ensemble For the final snippet ranking model, we ensemble five different models. Figure3 illustrates the overall structure of the ensemble. The sentBERT model is shown on the left-hand side of the model. The sentBERT is pre-trained with a question intent matching task, where the model determines if the intent of the two questions is the same. In sentBERT, two separate BERTs encode each of the given question pairs, and the similarity score is calculated based on the cosine similarity of the two BERT outputs. The Quora question pair dataset was used as the pre-training dataset.

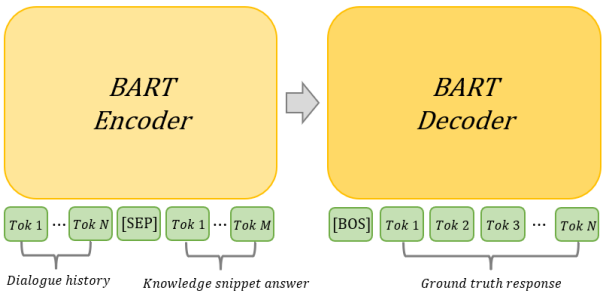


Figure 4: BART for response generation.

Different input pairs were used in different ways for training the four models using RoBERTa. The input pairs consist of (last utterance, knowledge snippet question), (last utterance, knowledge snippet answer), (dialogue history, knowledge snippet question), (dialogue history, knowledge snippet answer). The logits, i.e., meaning a correlation between two inputs, of the [CLS] token of each model is used as the similarity score. By combining the above four different models, the final model can capture not only the features of the last utterance relating to the knowledge snippet, but also the entire dialogue history.

The top five snippets were determined by voting based on the rank assigned by each model. Each of the five models selects top5 knowledge snippet candidates. And by examining the knowledge snippet which has top score of each model, the one snippet with the most votes is determined as the final first rank. In this way, to select the final Nth knowledge snippets, the knowledge snippets from first to Nth of each model are investigated and the final Nth rank is determined through the number of votes.

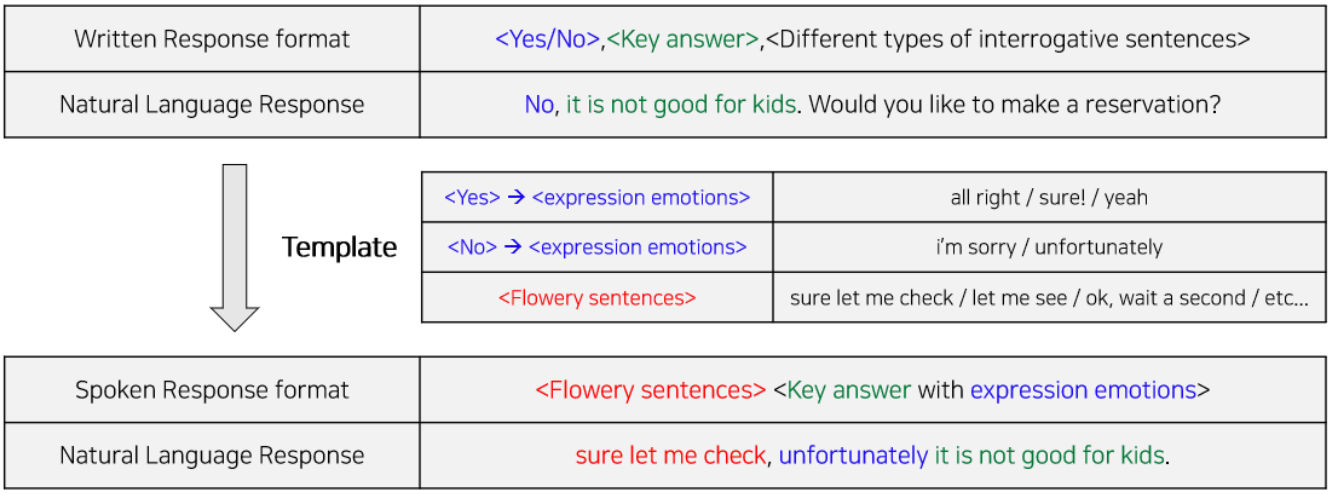


Figure 5: An example how to modify the written response to spoken response through template. Expression emotions can replace meaning of ⟨Yes/No⟩. Interrogative sentences is deleted from written response and flowery sentences is added to spoken response

Additional filtering to KSTD If the entity was not found during the entity detection, ten knowledge snippet candidates were constructed using the *the spacy sentence encoder*. Ten candidates were selected only if the similarity score was higher than a threshold. Therefore, if there is no knowledge snippet that exceeds the threshold, the KSTD result of the dialogue is corrected to "false" because there are no candidates to rank.

2.5 Response Generation

We used a pre-trained BART (Lewis et al. 2020) to determine the system response. The structure of the model is shown in Figure 4. To capture the information of the dialogue, the ⟨sep⟩ token between the dialogue history and the predicted knowledge snippet answer are given as input to the BART encoder. The decoder generates a fluent and appropriate response based on the information implied by the encoder. We modified the data so that BART can generate a response that corresponds to the spoken dialogue. The existing data consists of sentences in the form of "⟨Yes/No⟩, ⟨key answer⟩, ⟨different types of interrogative sentences⟩." However, in the spoken data, sentences are structured as "⟨flowery sentences⟩ ⟨key answer⟩." The flowery sentences were extracted and regenerated from the response of the validation dataset of DSTC10. Therefore, as shown in Figure 5, we modified the knowledge snippet answer used for learning according to the specified templates.

3 Experiments

3.1 Data and Settings

We used the training, validation, and test dataset of DSTC9 (Kim et al. 2021) Track1 as the training dataset for subtrack#2 of DSTC10's Track#2. Since the validation dataset of this track contains a part of the spoken dialogue data of DSTC9's test dataset, redundant data was excluded from training. The number of datasets, the reason for data configuration, and model settings for each task and step are as follows:

	DSTC10 DEV
	Accuracy (%)
RoBERTa-large	91.35
w/ normalized dialogue	94.23

Table 1: Performance of Domain Tracking. Normalized dialogue refers to dialogue data in which utterances containing unnecessary information for domain classification are deleted, as mentioned in section 2.2.

- **Knowledge-seeking turn detection**: We used 30,724 dialog datasets. This is the first step which affects the subsequent steps. To filter out only dialog that do not use external knowledge for surely, the dialog data with a knowledge-seeking turn of "true" should be more than dialog data with a knowledge-seeking turn labeled with "false". (true: negative = 3.4:1). Thus, a model can be trained biasly that predicts "true" more than "false." The RoBERTa-large was used for this task.
- **Domain tracking**: A total of 1,643 dialog were used. In the data for which the label of KSTD is "true," the number of dialog in the attraction domain is 243, which is significantly less than that of the other domains. Therefore, to flatten the ratio for each domain, only 350 data of the other 4 domains were used. RoBERTa-large was used for this step.
- **Entity detection**: We used 11,439 dialog data in this step. From the data where the label of KSTD is "true," dialog with domains train and taxi were excluded, and only dialogues that can be automatically labeled with start/end spans with entity alias list were used. RoBERTa-large

ID	Task1: Turn Detection			Task2: Knowledge Selection			Task3: Response Generation							
	Precision	Recall	F1	MRR@5	Recall@1	Recall@5	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-1	ROUGE-2	ROUGE-L
1	0.8592	0.9473	0.9011	0.5758	0.5474	0.624	0.2603	0.1746	0.0897	0.0287	0.3022	0.3299	0.1527	0.305
2	0.9214	0.9268	0.9241	0.6021	0.5737	0.6511	0.2631	0.1767	0.0904	0.0291	0.3062	0.3301	0.1544	0.308
3	0.9214	0.9268	0.9241	0.6155	0.5927	0.6511	0.2652	0.1777	0.091	0.0298	0.3092	**0.3361**	**0.1558**	0.3102
4(Final submitted)	**0.9214**	**0.9268**	**0.9241**	**0.6464**	**0.6204**	**0.6847**	**0.2705**	**0.1818**	**0.0936**	**0.0298**	**0.3167**	0.3324	0.1543	**0.3184**

Table 2: Performance of system according to applied methods. ID4 system is final model which is submitted

was used for this step.

- **Snippet ranking**: In this step, a total of 144,082 last utterance-snippet question pairs were used. Negative sampling was used to construct the input data pairs. The BM25 score between the last utterance and the snippets was calculated, and the top seven highest scores were used as negative samples. sentBERT and RoBERTa-base were used.
- **Response generation**: A total of 23,838 dialog were used. All data for which the KSTD is "true" were used. The BART-large was used for this task.

3.2 Evaluation Metrics

Knowledge-seeking turn detection (Task#1) is a binary classification task that determines whether external knowledge should be used for the system response. Precision, recall, and F1-score were used as evaluation methods. Knowledge selection (Task#2) is a task that rank the top five knowledge snippets used in system responses, Recall@1, Recall@5, and MRR@5 are used as evaluation methods. In response generation (Task#3), BLEU (Papineni et al. 2002), ROUGE (Lin 2004), and METEOR (Banerjee and Lavie 2005) were used to evaluate the response generated by the system in natural language.

3.3 Experiments Results

Table1 shows the performance improvement when utterances that are not useful for domain classification are removed. When the model was trained with normalized dialog, performance increased by 2.88%p compared to training with the original data. This performance improvement means that the deleted utterance pair interfered with domain classification.

Table 2 shows the performance of the entire system. The four systems used the same model and method for binary classification for knowledge-seeking turn detection (task#1), domain tracking, and response generation.

- **ID 1**: No additional filtering to KSTD. No rerank model.
- **ID 2**: Additional filtering to KSTD. No rerank model.
- **ID 3**: Additional filtering to KSTD and use of the rerank model.
- **ID 4**: Additional filtering to KSTD and use of the rerank model. Ensemble five snippet ranking models.

ID4 is the final system submitted to DSTC10, and all the proposed methods are applied to it. As shown in the Table 2, ID4 showed the best performance in all the evaluation metrics except Rouge-1 and Rouge-2 in response generation. Note that the performance of ID1 for turn detection is lower than the other models because it does not implement additional filtering to KSTD. However, a relatively high recall score was achieved for ID1 since it was trained to be biased towards predicting the "true" value. From ID2 onwards, the performance of turn detection was improved by additional filtering to KSTD. As the performance of Task1 increased, the performance of Task2 and Task3 was also higher than that of ID1. The difference between ID2 and ID3 is whether the rerank model is used or not. Recall@1 of ID3 increased by 1.9%p compared to ID2.This result proves that the rerank model improved the snippet selection performance by providing additional checking. Finally, the ensemble of the five models significantly improved the overall knowledge selection performance.

	Task1: Turn Detection		
	Precision	Recall	F1
Ours	**0.9214**	0.9268	**0.9241**
Objective first rank	0.8503	**0.9810**	0.9109

Table 3: Our system of best performance in Task1. Comparing ours with the first ranker in objective evaluation

Our proposed system achieved state-of-the-art performance in the turn detection task in the DSTC10 challenge. Overall, we ranked fifth in the objective evaluation, and achieved fourth place in the human evaluation (Table 4).

Rank	Team ID	Accuracy	Appropriateness	Average
Ground-truth		3.5769	3.4814	3.5292
1	B10	3.4947	3.3523	3.4235
2	B04	3.3356	3.3021	3.3189
3	B08	3.3433	3.2559	3.2996
4	B14(Ours)	3.2935	3.2815	3.2875

Table 4: Results of human evaluation of sub-track#2 of DSTC10 track2

4 Conclusion

In this paper, we introduce and explain how we solve the problem of knowledge-grounded task-oriented dialogue modeling on spoken conversations with our proposed method. An interesting point is that after training the task#1 model with a slight bias to one side, we were able to increase the performance of task#1 with additional filtering processing. Additionally, we divided task#2 into three step which allowed the system can select the appropriate knowledge snippet by narrowing the search scope. Our approach achieved the best F1-score on task#1 and ranked fourth in the final human evaluation.

Acknowledgements

This work was supported in part by LG Electronics (CTO/AI lab) and in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2020R1A2C2100362).

References

Banerjee, S.; and Lavie, A. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 65–72. Ann Arbor, Michigan: Association for Computational Linguistics.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805.

Honnibal, M.; and Montani, I. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Kim, S.; Eric, M.; Hedayatnia, B.; Gopalakrishnan, K.; Liu, Y.; Huang, C.-W.; and Hakkani-Tur, D. 2021. Beyond Domain APIs: Task-oriented Conversational Modeling with Unstructured Knowledge Access Track in DSTC9. arXiv:2101.09276.

Levenshtein, V. 1966. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10: 707.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7871–7880. Online: Association for Computational Linguistics.

Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, 74–81. Barcelona, Spain: Association for Computational Linguistics.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. Cite arxiv:1907.11692.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics.

Reimers, N.; and Gurevych, I. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Zeng, C.; Li, S.; Li, Q.; Hu, J.; and Hu, J. 2020. A Survey on Machine Reading Comprehension: Tasks, Evaluation Metrics and Benchmark Datasets. arXiv:2006.11880.