Adaptive Multi-Domain Dialogue State Tracking on Spoken Conversations

Jungwoo Lim, Taesun Whang, Dongyub Lee, and Heuiseok Lim

***Abstract*—The main objective of the task-oriented dialogue system is to identify the intent and needs of human dialogue. Many existing studies are conducted under the setting of written dialogue, but there always exists a difficulty in coping with real-world spoken dialogues. To this end, DSTC10 challenge organizers propose the task of building robust dialogue state tracking (DST) models on spoken dialogues. With the powerful existing DST model (i.e., MinTL), this article suggests integral components for building a dialogue state tracker; 1) Data augmentation effectively enhances the capability of the model to catch the entities that exist in the evaluation dataset. 2) Levenshtein post-processing aims to prevent the distortion in model prediction caused by automatic speech recognition errors. To validate the effectiveness of our methods, we evaluate our model on DSTC10 datasets and conduct qualitative analysis by ablating each component of the model. Experimental results show that our model significantly outperforms baselines in all evaluation metrics and took 3rd place in the challenge.**

***Index Terms*—DSTC10, dialogue state tracking, spoken dialogue.**

I. Introduction

THE task-oriented dialogue system aims to capture the intents and satisfy the needs of human dialogue. In the real-world multi-domain dialogue, there are obvious differences between the ways of speaking and writing, even for the same context and semantics of the conversations. Moreover, there always exists extra noise from disfluencies or automatic speech recognition (ASR) errors. Along with these challenging situations, DSTC10 proposes the "Knowledge-grounded Task-oriented Dialogue Modeling on Spoken Conversations" track. The objective of the track is to benchmark the robustness of the conversational models while filling the gaps between written and spoken conversations. Task 1 in this track mainly focuses on identifying the state of the given multi-domain dialogues.

The main difficulty of this challenge lies in the fact that the training corpus is not given. Since the validation set is the result of spoken conversation, most of the dialogue state tracking (DST) datasets available are mainly the written conversation corpus [1], [2], [3], [4]. Also, the entities in the training set and those from the evaluation set have significant differences. Under this situation, we set our goal as building a robust and generative dialogue system of open-vocabulary approach to comfortably manage unseen values along with the ASR errors. Moreover, we decide to adopt and implement additional modules to overcome the problem of generating inconsistent values.

To address these issues, this article proposes integral components to build applicable models to deal with the above real-world errors in spoken conversations. To show the effectiveness of our proposed components, we adopt the existing DST model MinTL [5], which is an effective transfer learning framework while showing comparable performance with generative pre-trained language models. First, we introduce a highly effective data augmentation strategy to reduce data discrepancy between written and spoken conversations. Since the training dataset is not provided in the challenge, we augment the existing DST dataset (e.g., MultiWOZ 2.1 [4]) by replacing several names and types of the entities from the given dataset with those of the evaluation dataset. After model training, we then additionally process the predicted value to have a suitable and consistent dialogue state by exploiting Levenshtein post-processing. Lastly, we aggregate the predictions from the differently initialized models by selecting the most predicted value for each slot type, and it is taken as a final prediction. Experimental results show that our model outperforms baselines by about 30% in joint goal accuracy and took 3rd place in the challenge.

II. Related Work

A. Open vocabulary-based DST

Open vocabulary-based DST is one of the main approaches to traditional dialogues state tracking (DST). Unlike the predefined ontology-based approach, open vocabulary-based methods [6], [7], [8] generate slot value at each turn with a generative model such as RNN, LSTM, and GRU [9], [10], [11]. With the advent of pre-trained language models and their remarkable

Manuscript received 16 November 2022; revised 12 April 2023; accepted 23 July 2023. Date of publication 7 August 2023; date of current version 8 December 2023. This work was supported in part by the Ministry of Science and ICT (MSIT), Korea, through the Information Technology Research Center (ITRC) support Program under Grant IITP-2023-2018-0-01405 supervised by the Institute for Information & Communications Technology Planning & Evaluation (IITP), in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korea Government (MSIT) under Grant 2020-0-00368, A Neural-Symbolic Model for Knowledge Acquisition and Inference Techniques, and in part by the Ministry of Science and ICT (MSIT), Korea, through ICT Creative Consilience Program under Grant IITP-2023-2020-0-01819 supervised by the Institute for Information & communications Technology Planning & Evaluation (IITP). The associate editor coordinating the review of this manuscript and approving it for publication was Di Jin. *(Jungwoo Lim and Taesun Whang contributed equally to this work.) (Corresponding author: Heuiseok Lim.)*

Jungwoo Lim is with the Computer Science and Engineering, Korea University, Seongbuk-gu 02841, South Korea (e-mail: wjddn803@korea.ac.kr).

Taesun Whang and Dongyub Lee are with the Naver Corp, Seongnam 463867, South Korea (e-mail: taesun.whang@navercorp.com; dongyub.lee@navercorp.com).

Heuiseok Lim is with the Korea University, Seongbuk-gu 02841, South Korea (e-mail: limhseok@korea.ac.kr).

Digital Object Identifier 10.1109/TASLP.2023.3302232

performance [12], [13], [14], recent studies utilize pre-trained models on DST as well [5], [15]. By exploiting pre-trained language models, the dialogue system does not suffer from task-specific design and extensive human annotations. Moreover, the models get benefits from the pre-trained weights and achieve decent performance with a small fraction of the training data.

B. Handling automatic speech recognition

Dialog state tracking models that receive the output of the automatic speech recognition module inevitably face up to ASR errors. Previous studies of handling such errors can be divided into two approaches. One is to consider these errors within the models directly. The studies of [6], [16] explicitly utilize ASR n-best lists as the additional features with extra encoders to find the correct belief state of the dialogue. Researchers of [17], [18] added the layer of correcting ASR errors in training along with the original NLU tasks. Also, one models ASR sequence as graphs and exploits confusion networks with a neural dialogue state tracker [19]. The other is augmenting data simply to the training data for the robust DST model. Also, the study of [20] leverages an ASR error simulator to inject noise into the error-free text data and subsequently train the dialog models with the augmented data. [21] propose a reinforcement learning (RL) based framework for data augmentation that can generate high-quality data to improve the dialog state tracker. Since we aim to build a robust model without latency in the dialog state tracker, we exploit data augmentation to the given training data directly.

III. Task Description

Multi-domain dialogue state tracking is one of the tasks in DSTC10 track 2; Knowledge-grounded Task-oriented Dialogue Modeling on Spoken Conversations. The main objective of this track is to benchmark the robustness of the conversational models against the gaps between written and spoken conversations [22]. The dataset is transcribed from the human-to-human dialogues about touristic information for San Francisco. Since it is constructed from the transcription, the data suffer from ASR errors. The task is evaluated with joint goal accuracy, and the training set is not limited to any of the datasets.

IV. Approach

A. Problem Formulation

Given a dialogue $\mathcal{D} = \{U_1, R_1, U_2, R_2, \ldots, U_T, R_T\}$, where U and R is user utterance and system response, respectively, we define dialogue state at each turn $B = \{B_1, B_2, \ldots, B_T\}$. The slot value for each domain–slot pair is denoted as $B_t(d_i, s_j) = v$, where d is a domain, s is a slot type, and v is a corresponding slot value. The dialogue state tracking model aims to predict dialogue state at each turn B_t for all domain slot pairs, given previous dialogue state B_{t-1} and dialogue context $C_t = \{U_{t-w}, R_{t-w}, \ldots, R_{t-1}, U_t)\}$, where w is a window size.

B. MinTL

MinTL [5] leverages generative pre-trained language models for multi-domain dialogue state tracking. The main idea of MinTL is to generate dialogue states that need to be changed (Levenshtein Belief Spans) at each turn. Specifically, we concatenate the previous dialogue states B_{t-1} and dialogue context C_t to build source input. In the case of target sequence S_t, they consist of newly updated slots and each slot is updated based on one of the following conditions.

- *Insertion:* $B_{t-1}(d_i, s_j)$ is a empty value and $B_t(d_i, s_j)$ is newly added at turn t.
- *Deletion:* $B_{t-1}(d_i, s_j)$ has a value and $B_t(d_i, s_j)$ is deleted at turn t (i.e., $B_t(d_i, s_j) = \texttt{None}$).
- *Substitution:* $B_{t-1}(d_i, s_j)$ is replaced with the different value $B_t(d_i, s_j)$ at turn t (Both $B_{t-1}(d_i, s_j)$ and $B_t(d_i, s_j)$ are non-empty values).

Each update slot is formed as $s_j \oplus B_t(d_i, s_j)$ for each domain, and the special token for domain $[d_i]$ is added to the beginning of the very first slot. MinTL model is fine-tuned from generative pre-trained language models, such as T5 [13] and BART [14]. The model is trained by minimizing negative log-likelihood of S_t given B_{t-1} and C_t, which is denoted as,

$$\mathcal{L}oss = -\log p(S_t | B_{t-1}, C_t). \tag{1}$$

Especially, BART is significantly effective when fine-tuned on text generation since it performs well on comprehension tasks also. To obtain the aforementioned advantages, we employ a pre-trained generative language model and the design of [5] in this study.

C. Data Augmentation

To build the dataset that covers the entities in the dataset, we augment MultiWOZ dataset by replacing entities of certain slot types in the dialogue. The examples of data augmentation are described in Table I. In order to reduce the distributional bias of entities, we refer to the MultiWOZ labels based on domain and slot types. In other words, we substitute original entities from the MultiWOZ dataset with the corresponding entities from the training set according to the domain and slot types. We randomly choose the entity from the candidates that have the same target domain and slot type to replace. We conduct entity substitution only on the certain slot types as described in Table I. Along with the entity substitution, only 30 % of the values of slot type `day` are replaced with *today* and *tomorrow* since they are limited to the range of *Monday*, *Tuesday*, *Wednesday*, *Thursday*, *Friday*, *Saturday* and *Sunday* in MultiWOZ dataset.

D. Levenshtein Post-Processing

We introduce a method of revising the predicted values depending on the given dataset and database called Lev processing. After obtaining the predicted value from the previous step, we conduct a replacement process utilizing the Levenshtein distance [23]. We first measure the Levenshtein distance between the predicted value and all the values from the corresponding domain and slot types from the database. After choosing the value that has the lowest score, we also apply the word error rate to find the exact matching values to the ground truth. In detail, when the distance between the predicted value and values from the database is longer than the threshold T, we regard this case as a failure. In our qualitative analysis, we conclude that these cases

TABLE I
EXAMPLES OF AUGMENTED DATASET FOR EACH DOMAIN IN THE DSTC10 DATASET

Domain	Augmented Slot Type	Dataset	Dialogues
Restaurant	Name, Area, Day, Food	MultiWOZ 2.1	User: i am looking for a **european** restaurant in **centre** area on **tuesday**. System: **eraina** is a great **european** restaurant in the **centre** of town .
		Augmented Dataset	User: i am looking for a **korean** restaurant in **outer richmond** area on **tomorrow**. System: **um ma son** is a great **korean** restaurant in the **outer richmond** of town .
Hotel	Name, Area, Type, Day	MultiWOZ 2.1	User: I need to book a **hotel** in the **east** that has 4 stars on **monday**. System:There only one **hotel** available in the **east** area. It's called **allenbell**.
		Augmented Dataset	User: I need to book a **hostel** in the **union square** that has 4 stars on **today**. System:There only one **hotel** available in the **union square** area. It's called **adelaide hostel**.
Attraction	Name, Area, Type	MultiWOZ 2.1	User: i am also looking for a **museum** in **west**. System: there are 7 **museum**s in the west . do you have a preference ? User: i have no preference , i just need to know how much the entrance fee is . System: **cambridge and county folk museum** has an entrance fee of 3.50 pounds .
		Augmented Dataset	User: i am also looking for a **amusement park** in **fisherman's wharf**. System: there are 7 **amusement park**s in the **fisherman's wharf**. do you have a preference ? User: i have no preference , i just need to know how much the entrance fee is . System: **7d experience** has an entrance fee of 3.50 pounds .

All augmented slots except day are extracted from the consistent item. Text in bold indicates augmented slot values.

are driven by the spoken conversation which contains erroneous texts. To mitigate this issue, we utilize the word error rate. We extracted the words from the groups of n-gram words which are gathered from the previous dialogue history and computed the word error rate score with the predicted word. Afterward, we choose the lowest value word to measure the Levenshtein distance. The lowest value word is chosen as the final answer and this process made our model more robust to the spoken conversation text regardless of the types of domains.

We also suggest consistent Levenshtein post-processing called Lev_c that matches the slot values according to database values of `name`. Once the predicted value of the name slot is obtained, we additionally find the value of area slots and type slots according to the `name` slots. As we assume that the `name` slot possesses the centralized information of the dialogue, we substitute the previous value with the corresponding value from the database from the matched slot. For example, when the predicted name of the slot for the hotel is "fairmont san francisco", the model predicts the value of the area slot as "embarcadero" even though the ground truth value "nob hill" is indicated in the database already. Therefore, we switch the predicted values of other slot types with the database value according to the predicted name. By exploiting Lev_c, the consistency of the dialogue state increases empirically.

E. Ensemble

To boost performance, we aggregate the slot value prediction results from several randomly initialized models. All slot values are post-processed (either using Lev or Lev_c) first, and then we select the most predicted value for each slot type as a final prediction. When more than half of the models generate none value (empty slot), none value is taken as a final prediction. On the other hand, if the majority of the models generate non-empty values even if the values are slightly different from each other, we choose the most predicted value among the non-empty values.

V. EXPERIMENT

A. Experimental Setup

1) Dataset: To train our model, we adopt a clean version of MultiWOZ [4], [24], which is a commonly used benchmark dataset for multi-domain dialogue state tracking task. This dataset is constructed on the basis of written conversations in 7 domains (e.g., restaurant, hotel, police, and taxi). During the training phase, all training, validation, and test sets are used to train the model. For the evaluation, we use DSTC10[1] dataset provided by the challenge organizers. Unlike MultiWOZ, it is annotated from the spoken conversations and covers only 3 domains (i.e., restaurant, hotel, and attraction). To reduce domain discrepancy, we only use dialogue states belonging to these three domains, and those of the remaining domains are not considered during model training. Corpus statistics for both MultiWOZ and DSTC10 datasets are described in Table II.

TABLE II
CORPUS STATISTICS OF DIALOGUE STATE TRACKING DATASETS

Phase	Training			Evaluation	
Dialogue Type	Written			Spoken	
Dataset	MultiWOZ 2.1			DSTC10	
	Training	Validation	Test	Validation	Test
# dialogues	8434	999	1000	107	783
# turns	56747	7365	7372	936	6588

2) Evaluation metrics: We evaluate our methods using several evaluation metrics. 1) Joint goal accuracy is commonly used as the main metric in dialogue state tracking, and it is used to rank the participants in the challenge. It gets 1 if all predicted slot-value pairs are exactly the same as the ground truth and 0 otherwise at the turn level. 2) Slot accuracy is used to check whether each slot is correctly predicted. 3) Precision, Recall, and F1 score are used for both value and none prediction.

B. Implementation Details

We implemented our model using PyTorch [26] library. We employed BART-base [14] as a pre-trained backbone model as it showed better results than T5 [13] in our experiments. The batch size is set to 32, and the window size for the previous dialogue context is set to 3. For the data augmentation, we replaced slot values with new entities for every epoch so that the model

[1] [Online]. Available: https://github.com/alexa/alexa-with-dstc10-track2-dataset

TABLE III
QUANTITATIVE RESULTS ON THE DSTC10 VALIDATION SET

	Model	Joint Goal	Slot	Value Prediction			None Prediction		
		Accuracy	Accuracy	Precision	Recall	F1	Precision	Recall	F1
Baselines	TripPy [22]	0.53	70.56	56.46	29.93	39.13	74.35	96.10	83.83
	MinTL [5]	0.85	75.04	52.24	49.60	50.89	82.54	95.85	88.69
Ours	MinTL + DA	11.54	85.37	71.10	69.90	70.49	94.44	95.44	94.94
	MinTL + DA + Lev	21.26	88.62	79.71	78.36	79.03	94.44	95.44	94.94
	MinTL + DA + Lev$_c$	26.60	87.92	77.77	76.46	77.11	94.45	95.44	94.94
	MinTL + DA + Lev$_c^\dagger$	**30.24**	**90.02**	**81.74**	**81.61**	**81.67**	**95.56**	**95.67**	**95.61**

† Denotes ensemble model.

can learn the diverse entities that exist in the database of the evaluation set. The model is trained using Adam optimizer [27] with the initial learning rate of 2e-5. During training, the ground truth of the current turn is used as the previous state in the next turn. On the other hand, in evaluation, the prediction result of the current turn is used as the previous state in the next turn so that the result of the front turn is propagated to that of the following turn. For the Levenshtein post-processing, we used fuzzywuzzy[2] library to calculate edit distances between generated slot value and database candidates.

C. Results

Table III reports quantitative results on DSTC10 validation set. For the baseline comparison, we compare our models with the TripPy [25] model, which is the official baseline in the challenge. Also, we report experimental results of the vanilla MinTL [5] to show how effective our proposed method is. We perform ablation analyses based on the MinTL model to explore how each method affects performance improvement. A brief explanation of our proposed methods are as follows.

- **MinTL + DA** aims to reduce data discrepancy between training and evaluation datasets. Slot values related to the information of each item in the database, such as name, area, and type, are replaced with values existing in the evaluation database.
- **MinTL + DA + Lev** finds pre-defined slot values from the database for the slot type in which the value is generated by using Levenshtein distance.
- **MinTL + DA + Lev$_c$** finds an item in the database through Levenshtein distance for name only. In order to ensure that all generated slot values can be *consistent* for the item, we fill the value of each slot type with the item's pre-defined value.
- **MinTL + DA + Lev$_c^\dagger$** is an ensemble model predictions from five MinTL + DA + Lev$_c$ models.

Compared to the other single models, data augmentation and post-processing based on Levenshtein distance significantly improve the joint goal accuracy. Specifically, data augmentation achieves significant improvements in joint goal accuracy from 0.85 to 11.54 and in slot accuracy from 75.04 to 85.37, compared to the MinTL model. Also, Levenshtein post-processing (Lev results in an additional performance improvement of more than 10% in joint goal accuracy. We also report an ensemble of 15 models (MinTL + DA + Lev_c) which are trained with different random initial seeds, and it achieves the highest performance in all evaluation metrics.

TABLE IV
OFFICIAL RESULTS FOR TEST SUBMISSIONS BY DSTC10 PARTICIPANTS

Rank	Team	Entry	Joint Goal Accuracy	Slot Accuracy	Value F1	None F1
1	A11	1	46.16	94.98	91.15	97.82
2	A01	0	36.05	93.67	89.31	96.72
3	**A07**	**1**	**27.73**	**89.48**	**79.50**	**96.16**
4	A10	4	26.79	90.79	83.68	95.71
5	A09	4	18.21	87.59	79.86	92.33
6	A06	3	16.91	85.95	75.29	93.61
7	A05	3	16.15	86.24	76.55	92.31

All rankings are based on the joint goal accuracy metric. Text in bold indicates our model (Team: A07).

In addition, when comparing the model with Lev and Lev_c, we especially observe that Lev_c significantly increases joint goal accuracy and slightly decreases performance in some metrics related to each slot. Even if the model predicts almost all of the slot values, the joint goal accuracy gets 0 if even one slot is predicted incorrectly. Since the dialogue state of each domain contains values of consistent items, Lev_c consistently replaces them with the searched item information found based on the name. One fatal limitation of this method is that if the searched item is not the correct answer, all replaced values can be wrong so that the performance of individual slots is degraded (slot accuracy: -0.7%, value prediction F1: -1.92%).

Table IV lists official results for entry submissions by participants. We only report the teams that achieved above 10% in the joint goal accuracy metric. Each team submitted up to 5 prediction results and the result with the highest joint goal accuracy was used for final ranking. We submitted the predictions of MinTL + DA + Lev$_c$ (ensemble) and took 3rd place in the challenge. Even though the winning team achieves remarkable results, it is notable that we obtain significant performance improvement compared to the baseline without using any other ASR corpora and without additional fine-tuning on the DSTC10 validation set.

D. Qualitative Analysis

Table V shows qualitative results on DSTC10 validation set. For the first example, the value of the name slot is predicted as *secon one* when we train MinTL using MultiWOZ dataset. Also, the value of the area slot is also predicted as *south* since the values are limited to the *north*, *south*, *east*, and *west* in MultiWOZ. Afterward, the model predicts the name correctly and the area value is converted to *san francisco* which is the

[2] [Online]. Available: https://github.com/seatgeek/fuzzywuzzy

TABLE V
QUALITATIVE RESULTS ON THE DSTC10 VALIDATION SET

Dialogues	**Predictions**
User : hi i'm planning a trip to san francisco and i'm looking for recommendation for a u **moderate**ly priced **currying** restaurant in the **outher richmond** area **System**: ok sure let me go and see what i can find. ok so here we do have two options one is called han two kwaan and the second one is **um ma son** which one do you think you might like **User** : the secon one sounds good can you give me their address zip code and phone number please	**MinTL** : [R-name] secon one [R-food] currying [R-area] south [R-pricerange] moderate **MinTL + DA** : [R-name] um ma son [R-food] currying [R-area] san francisco [R-pricerange] moderate **MinTL + DA + Lev** : [R-name] um ma son [R-food] japanese curry [R-area] fisherman's wharf [R-pricerange] moderate **MinTL + DA + Lev$_{c}$** : [R-name] um ma son [R-food] korean [R-area] outer richmond [R-pricerange] moderate
	Ground Truth: [R-name] um ma son [R-food] korean [R-area] outer richmond [R-pricerange] moderate
User : e can you help me find a **public market** in the **embarcadero** **System**: certainly sir uhhh i'm showing one location in embarcadero it's called **ferry building market place** **User** : iawesome uh do you know what the address zip code and phone number **System**: definetely so the address is gonna be one ferry building that's b. l. d. g. and their zip code is nine four one one one and their phone number is listed as four one five nine eight three eighty thirty **User** : awesome gool and vitally do you know if uh there's a place i can park bike bike near there **System**: definitely i am showing this location does have bicycle parking yes **User** : thank you	**MinTL** : [A-name] ferry building market place [A-area] centre [A-type] park **MinTL + DA** : [A-name] ferry building market [A-area] embarcadero [A-type] bike rental **MinTL + DA + Lev** : [A-name] ferry building marketplace [A-area] embarcadero [A-type] bike rental **MinTL + DA + Lev$_{c}$** : [A-name] ferry building marketplace [A-area] embarcadero [A-type] public market
	Ground Truth: [A-name] ferry building marketplace [A-area] embarcadero [A-type] public market

R and A indicate restaurant and attraction, respectively.

value from DSTC10 database on area slots even if it is a wrong answer. When we exploit Lev_c, it is shown that the value gains more consistency than that of models utilizing Lev. Because the restaurant *um ma son* is in the area of *outer richmond* selling *korean* food, conversion of the values has positive impacts on the results. Since there is an ASR error about the food type in the conversation (i.e., currying), the model inevitably predicts the food type based on the dialogue. Even Lev method brings the most similar value from the database, it is difficult to predict the correct value as the raw value is completely unrelated to the ground truth (i.e., korean). Thus, Lev_c is highly effective in that it brings values for the consistent item. Similar consequences can be seen in the second example, especially in the `type` slot. By augmenting training dialogues, MinTL + DA predicts the area as *embarcadero* correctly, not stating *centre*. Moreover, Lev_c also aids in having non-irregular values by switching *bike rental* to *public market*.

VI. Conclusion

In this article, we focused on reducing data discrepancy between training and evaluation data, and that between written and spoken conversations in multi-domain dialogue state tracking. We proposed a highly effective data augmentation strategy and post-processing method based on Levenshtein distance. Experimental results show that our approaches achieve significant improvements in all evaluation metrics. Moreover, we demonstrated how each component affects the outcome through qualitative analysis. For future work, we plan to train the model for predicting consistent dialogue states for each domain in an end-to-end manner rather than post-processing.

References

[1] L. E. Asri et al., "Frames: A corpus for adding memory to goal-oriented dialogue systems," in *Proc. 18th Annu. SIGdial Meeting Discourse Dialogue*, 2017, pp. 207–219.

[2] P. Shah et al., "Building a conversational agent overnight with dialogue self-play," 2018, *arXiv:1801.04871*.

[3] T.-H. Wen et al., "A network-based end-to-end trainable task-oriented dialogue system," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics: Volume*, 2017, pp. 438–449.

[4] M. Eric et al., "MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines," in *Proc. 12th Lang. Resour. Eval. Conf.*, 2020, pp. 422–428.

[5] Z. Lin, A. Madotto, G. I. Winata, and P. Fung, "MINTL: Minimalist transfer learning for task-oriented dialogue systems," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 3391–3405.

[6] M. Henderson, B. Thomson, and S. Young, "Word-based dialog state tracking with recurrent neural networks," in *Proc. 15th Annu. Meeting Special Int. Group Discourse Dialogue*, 2014, pp. 292–299.

[7] K. Yoshino, T. Hiraoka, G. Neubig, and S. Nakamura, "Dialogue state tracking using long short term memory neural networks," in *Proc. 7th Int. Workshop Spoken Dialog Syst.*, 2016, pp. 1–8.

[8] A. Rastogi, R. Gupta, and D. Hakkani-Tur, "Multi-task learning for joint language understanding and dialogue state tracking," in *Proc. 19th Annu. SIGdial Meeting Discourse Dialogue*, 2018, pp. 376–384.

[9] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] M. Schuster and K. K. Paliwal, “Bidirectional recurrent neural networks,” *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, 1997.

[11] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder–decoder approaches,” in *Proc. SSST-8, 8th Workshop Syntax, Semantics Struct. Stat. Transl.*, 2014, pp. 103–111.

[12] J. Devlin, Ming-Wei Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[13] C. Raffel et al., “Exploring the limits of transfer learning with a unified text-to-text transformer,” *J. Mach. Learn. Res.*, vol. 21, pp. 1–67, 2020.

[14] M. Lewis et al., “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 7871–7880.

[15] G.-L. Chao and I. Lane, “BERT-DST: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer,” in *Proc. Interspeech*, 2019, pp. 1468–1472.

[16] M. Vodolán, R. Kadlec, and J. Kleindienst, “Hybrid dialog state tracker with ASR features,” in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, 2017, pp. 205–210.

[17] R. Schumann and P. Angkititrakul, “Incorporating ASR errors with attention-based, jointly trained RNN for intent detection and slot filling,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 6059–6063.

[18] Y. Weng et al., “Joint contextual modeling for ASR correction and language understanding,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 6349–6353.

[19] V. Pal, F. Guillot, M. Shrivastava, J.-M. Renders, and L. Besacier, “Modeling ASR ambiguity for dialogue state tracking using word confusion networks,” 2020, *arXiv:2002.00768*.

[20] L. Wang, M. Fazel-Zarandi, A. Tiwari, S. Matsoukas, and L. Polymenakos, “Data augmentation for training dialog models robust to speech recognition errors,” in *Proc. 2nd Workshop Natural Lang. Process. Conversational AI*, 2020, pp. 63–70.

[21] Y. Yin, L. Shang, X. Jiang, X. Chen, and Q. Liu, “Dialog state tracking with reinforced data augmentation,” in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 9474–9481.

[22] S. Kim et al., “How Robust RU?: Evaluating task-oriented dialogue systems on spoken conversations,” in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 1147–1154.

[23] V. I. Levenshtein, “On perfect codes in deletion and insertion metric,” 1992.

[24] Q. Zhu et al., “ConvLab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems,” in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics: Syst. Demonstrations*, 2020, pp. 142–149.

[25] M. Heck et al., “TripPy: A triple copy strategy for value independent neural dialog state tracking,” in *Proc. 21th Annu. Meeting Special Int. Group Discourse Dialogue*, 2020, pp. 35–44.

[26] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[27] P. D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980*.

Jungwoo Lim received the B.S. degree in library and information science from Sungkyunkwan University, Seoul, South Korea. She is currently working toward the Ph.D. degree with Natural Language Processing and Artificial Intelligence Lab. Her research interests include dialogue systems, question and answering, and relation extraction. She was also a Reviewer for ACL and EMNLP.

Taesun Whang received the M.S. degree in computer science and engineering from Korea University, Seoul, South Korea. He is currently working toward the Ph.D. degree with Natural Language Processing and Artificial Intelligence Lab. His research interests include natural language processing, machine learning, and artificial intelligence. He was also a Reviewer for ACL, NAACL, and EMNLP.

Dongyub Lee received the M.S. degree in computer science and engineering from Korea University, Seoul, South Korea. He is currently working toward the Ph.D. degree with Natural Language Processing and Artificial Intelligence Lab. His research interests include natural language processing, multi-modal retrieval, and XAI. He was also a Reviewer for ACL, NAACL, and EMNLP.

Heuiseok Lim received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Korea University, Seoul, South Korea, in 1992, 1994, and 1997, respectively. He is currently a Professor with the Department of Computer Science and Engineering, Korea University. His research interests include natural language processing, machine learning, and artificial intelligence.