Speech-Aware Multi-Domain Dialogue State Generation with ASR Error Correction Modules

Ridong Jiang[†] **Shi Wei**[†] **Bin Wang**[†] **Chen Zhang**[⋆]
Yan Zhang[⋆] **Chunlei Pan**[⋆] **Jung Jae Kim**[†] **Haizhou Li**[⋆⋆,⋆,‡]
[†]Institute for Infocomm Research (I^{2}R), A*STAR, Singapore
[⋆]National University of Singapore [‡]Kriston AI Lab, China
[⋆⋆]The Chinese University of Hong Kong, Shenzhen, China
jjkim@i2r.a-star.edu.sg

Abstract

Prior research on dialogue state tracking (DST) is mostly based on written dialogue corpora. For spoken dialogues, the DST model trained on the written text should use the results (or hypothesis) of automatic speech recognition (ASR) as input. But ASR hypothesis often includes errors, which leads to significant performance drop for spoken dialogue state tracking. We address the issue by developing the following ASR error correction modules. First, we train a model to convert ASR hypothesis to ground truth user utterance, which can fix frequent patterns of errors. The model takes ASR hypotheses of two ASR models as input and fine-tuned in two stages. The corrected hypothesis is fed into a large scale pre-trained encoder-decoder model (T5) for DST training and inference. Second, if an output slot value from the encoder-decoder model is a name, we compare it with names in a dictionary crawled from Web sites and, if feasible, replace with the crawled name of the shortest edit distance. Third, we fix errors of temporal expressions in ASR hypothesis by using hand-crafted rules. Experiment results on the DSTC 11 speech-aware dataset, which is built on the popular MultiWOZ task (version 2.1), show that our proposed method can effectively mitigate the performance drop when moving from written text to spoken conversations.

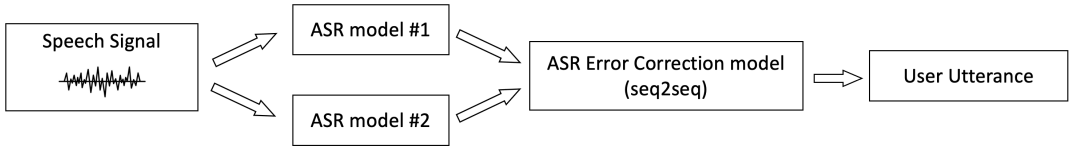
1 Introduction

Conversational agents are evolving rapidly in the last decade and providing solutions for almost every industry including e-commerce, travel, finance, food & beverage, hospitality, gaming and healthcare. The agents that can perform certain tasks or provide transactional services such as restaurant booking and hotel booking need to understand user's intent and track the dialogue states throughout the conversation, known as DST task. The performance of the agents very much depends on the DST results for a fluent and accurate task completion. With the advancement in AI and machine learning, especially the breakthroughs in Natural Language Processing such as Attention, Sequence-to-sequence model and the Transformer Architecture, the research on DST has attracted much attention in the past few years. DST is typically formulated as the problem of estimating user's goal in the form of a list of slot-value pairs when a dialogue progresses with multiple turns. When multiple domains are involved in the conversation (e.g. MultiWOZ datasets (Budzianowski et al., 2018)), the task becomes more challenging because the developed DST model needs to deal with multiple domains and their slot values in every dialogue turn. Different DST models have been developed. The methodologies can be broadly classified into two types. One is classification based method (Wu et al., 2019; Heck et al., 2020; Hosseini-Asl et al., 2020) and another one is generation based method (Zhao et al., 2021, 2022). However, both types of methods are mostly developed for written dialogues. These models do not perform well on spoken dialogues, because the audio form of user utterances in spoken dialogues should be transcribed by an automatic speech recognition (ASR) system, and the transcription involves ASR errors.

Consider a multi-domain spoken dialogue with both ground truths and ASR hypotheses of user utterances below. The ground truth user utterances are associated with, if any, their dialogue states that are affected by ASR errors observed in the corresponding ASR hypotheses. For instance, digits in the ground truth (e.g. "7", "2") are converted into English words in the corresponding ASR hypothesis (e.g. "seven", "two"). A name (e.g. "rosepine", "naturita") is incorrectly split into two words (e.g. "rose pine") and incorrectly transcribed (e.g. "natur") in some ASR hypotheses. The word "pm" is incorrectly transcribed into two words "p m". An example is given below with both ground truth and ASR hypothesis from the user

Proceedings of The Eleventh Dialog System Technology Challenge, pages 105–112

(a) Single-Source ASR Error Correction



(b) Multi-Source ASR Error Correction

Figure 1: An illustration of two types of ASR error correction methods.

side.

USER (GROUND-TRUTH) : i am looking to get some information on the resort at summerlin.

USER (ASR HYPOTHESIS) : i am looking to get some information on the resort at summerlin

SYSTEM : the resort at summerlin is in the centre of town and in the expensive price range.

USER (GROUND-TRUTH) : great can you get me a room for 7 people for 2 nights starting saturday? (hotel-people="7", hotel-stay="2")

USER (ASR HYPOTHESIS) : great can you get me a room for seven people for two nights starting saturday

SYSTEM : yes, i can! you're booked with reference number i34y32y9. is there anything else i can help you with

USER (GROUND-TRUTH) : yes, i also need to find a train going to rosepine. (train-destination="rosepine")

USER (ASR HYPOTHESIS) : yes i also need to find a train going to rose pine

SYSTEM : okay, i have quite a few. can you give me some more information?

USER (GROUND-TRUTH) : i'm departing from naturita on saturday. i want to arrive by 12:58 pm. (train-departure="naturita", train-arriveby="12:58 pm")

USER (ASR HYPOTHESIS) : i'm departing from natur on saturday i want to arrive by 12:58 p m

SYSTEM : i would recommend train id tr5049 which leaves naturita at 11:46 am and arrives in rosepine at 12:36 pm. there are earlier options as well if you wish.

USER (GROUND-TRUTH) : that would be perfect. for 7 people. (train-people="7")

USER (ASR HYPOTHESIS) : that would be perfect for seven people

In this work, we address the issue that errors in the ASR hypotheses of spoken dialogues affect dialogue states, by learning to fix them before passing the ASR hypotheses as input to a DST model and also by fixing them after the DST model generates slot values. Prior work have investigate the possibility to correct errors from ASR engines based on edit alignment (Leng et al., 2021b), non-auto-regressive generation (Leng et al., 2021a) and auto-regressive sequence-to-sequence models (Dutta et al., 2022). Here, we investigate the possibility to merge the multi-source ASR predictions using a pre-trained sequence-to-sequence model as shown in Figure 1. We experiment with DSTC 11 speech-aware dataset. Results and ablation studies show that our proposed approach can effectively mitigate the noise introduced by ASR hypothesis and performance drop when moving from written text to spoken conversations.

2 Methodology

Our model has three parts: pre-processing, encoder-decoder, and post-processing. In the stage of pre-processing, we in particular fix ASR errors in ASR hypothesis that is given as input to the encoder-decoder model, which generates dialogues states as output. In the stage of post-processing, we further try to fix remaining errors in the generated dialog states.

2.1 Pre-processing: Multi-Source ASR Error Correction

The first step of handling speech-based dialogue state tracking tasks is to use the Automatic Speech Recognition (ASR) technique to convert speech

signals into textual format. As mentioned before, it introduces undesired ASR errors which largely impact the performance of existing DST models that take ASR hypothesis (or transcript) as input. It is because most existing DST models are trained and designed to handle the ground truth strings (or "clean text") of both user and agent utterances. Therefore, to bridge the gap between speech-based DST tasks and clean-text-based DST systems, we propose a multi-source ASR error correction module that learns to convert ASR-generated transcripts to clean text. We hypothesize that even though the conversion is not perfect, it still fix some of the ASR errors and thus enhance DST performance.

The input to the ASR error correction is the ASR-generated transcripts and the desired output is the original clean text. First, we use a pre-trained sequence-to-sequence model to handle this task. In experiments, we leverage the pre-trained *T5-small* model. We also experimented with other popular pre-trained models including *T5-base* and *BART-base*, but did not witness significant improvement with increased model sizes.

Second, to better recover the information from speech signals, we propose to leverage ASR transcripts from multiple (two) ASR models. This is because different ASR models may have complementary characteristics which can compensate for each other's weaknesses. As an example, the two utterances and their audio's transcripts from two ASR models (ASR 1 - Our in-house ASR model; ASR 2 - The model used to generate the track dataset) are shown below. Each of the two examples shows an error (highlighted in italic face) by one of the two ASR models, respectively.

USER (GROUND-TRUTH) : i am looking to book a train that is leaving from leaf river to carpentersville on friday.

USER (ASR 1 HYPOTHESIS) : i am looking to book a train that is leaving from **life** river to carpentersville on friday

USER (ASR 2 HYPOTHESIS) : i am looking to book a train that is leaving from leaf river to carpentersville on friday

USER (GROUND-TRUTH) : howdy, i need a train heading into floyd.

USER (ASR 1 HYPOTHESIS) : howdy i need a train heading into floyd

USER (ASR 2 HYPOTHESIS) : **howey** i need a train heading into floyd

The proposed ASR error correction model based on two ASR models is illustrated in Fig. 1. In the Experiments section, we show that the combination of multiple ASR transcripts can lead to better ASR error correction performance than using a single ASR transcript.

Third, besides the multi-source ASR transcription fusion, we also utilize a two-step fine-tuning process for better modeling of data distributions. Instead of directly fine-tuning the pre-trained encoder-decoder model on a multi-source format (Fig. 1b), we first fine-tune the model on single-source data (Fig. 1a). For the first stage of fine-tuning, we use the augmented 100x data which are auto-generated with DST slot replacements as introduced in the Experiments section. During this process, the model learns the mapping between ASR transcripts to clean text with much more amount of data. In the second stage, the model is further fine-tuned in a multi-source ASR error correction format to synergistically incorporate two ASR systems.

2.2 Dialogue state generation using encoder-decoder

Consider a task-oriented dialogue (TOD) which consists of multiple turns of conversation between a user and an agent. We denote the dialogue history at turn T:

$$C_T = \{(u_1, r_1), (u_2, r_2), ...(u_T, r_T)\} \quad (1)$$

, where $u_i, r_i,\ i \in [1, T]$ are user utterance and system response at turn T, respectively. C_T is thus the dialogue context at turn T. This context C_T is used to generate the dialogue state B_T of user utterance at turn T u_T. The dialogue state B_T is represented as slot-value pairs:

$$B_T = \{(S_1^d, V_1^{dT}), (S_2^d, V_2^{dT}), ...(S_j, V_j^{dT})\} \quad (2)$$

, where $S_i^d, V_i^{dT},\ i \in [1, j_T]$ are the i-th slot name and its value at turn T, respectively. The $d \in [1, M]$ denotes the domain, M is the total number of domains, and j_T is the number of slots with non-empty values at turn T. In DSTC11, there are total of 35 slots in 7 domains.

All the utterances and slot names S are concatenated into one sequence as input of the encoder, the decoder generates the belief state with slot values:

$$B_T = seq2seq(C_T, S) \quad (3)$$

We employ T5 pre-trained seq2seq model. The learning objective of this T5 generation model is to

minimize the log-likelihood of B_T given C_T and S:

$$\mathcal{L} = -\sum \log\ p(B_T|C_T, S), \qquad (4)$$

For all user utterances and system responses, we add "user:" and "system:" prefixes to every u_i and r_i, respectively, to differentiate the user's input from system's response.

Following is an example of input to the model:

user: i would like a taxi from saint john's college to pizza hut fen ditton. $\langle sep \rangle$ taxi-destination: $\langle extra\_id\_0 \rangle$ $\langle sep \rangle$ taxi-departure: $\langle extra\_id\_1 \rangle$ $\langle sep \rangle$ hotel-parking: $\langle extra\_id\_2 \rangle$ $\langle sep \rangle$ ...

, and the corresponding output:

$\langle extra\_id\_0 \rangle$ saint john's college $\langle extra\_id\_1 \rangle$ pizza hut fen ditton $\langle extra\_id\_3 \rangle$ none $\langle extra\_id\_4 \rangle$ ...

We used a special token (e.g. $\langle extra\_id\_0 \rangle$) to indicate each slot type and make connection between the slot name in input and the slot value in output. For instance, the slot value "pizza hut fen ditton" in the example output is for the slot whose name is "taxi-departure" in the example input, which are connected via $\langle extra\_id\_1 \rangle$.

2.3 Post-processing

The purpose of the proposed post-processing is to fix prediction errors from the encoder-decoder model due to the ASR errors. As shown in the Fig. 1, the ASR errors can lead to the incorrect prediction on, for instance, time, names of hotel and restaurant, and names of location (e.g. train departure and destination).

2.3.1 Name Error Correction

Name recognition is especially challenging to ASR engines. It is error prone for the recognition of names in spoken dialogue. We thus devise a method of correcting name errors with a dictionary of names that are collected from the training and development data and crawled from Web sites.

For every slot value of name, we apply fuzzy match to find the most similar name from the name dictionary of the slot type. The matching can be one of the following results:

a) Correct match: prediction is correct, matching result is correct (no change to DST accuracy).

b) Good match: prediction is incorrect, matching result is correct (Increasing the DST accuracy).

Slot with name values	Size of dictionary
hotel	32,093
restaurant	1,102
taxi-departure	23,269
taxi-destination	23,314
train-departure	23,086
train-destination	23,089

Table 1: Dictionary size for each slot type whose values are names

Slot Type	**Source URLs**
hotel	wikipedia-1 wikipedia-2 wikipedia-3 wiki-accomodation
restaurant	easyleadz wikipedia opentable leading restaurants bloomberg
train	great american stations wikipedia rail advent
location	britannica-1 britannica-2 wikipedia-1 wikipedia-2 wikipedia-3

Table 2: Web sites from which we crawled potentially relevant names. 'location' can be useful for multiple slot types, including 'taxi-departure/destination' and 'train-departure/destination'.

c) Bad match: prediction is correct, matching result is incorrect (Decreasing the DST accuracy).

d) Wrong match: prediction is incorrect, matching result is still incorrect (no change to DST accuracy).

Obviously, we want to have more good match (Case b) and try to reduce the bad match (Case c). We designed following matching rules to maximize the good match:

i) Get the top 10 similar matching

ii) Reject the shorter items with one word

iii) Reject too long items which are two words longer than the input value

iv) Change to top the rank of the matches which found in "system" utterance

v) Remove items whose score is lower than the preset threshold

After some experiments, we found that the matching rules are beneficial in terms of the evaluation metrics when applied to the categories of hotel, train departure, and train destination. When the slot types are restaurant, taxi-departure and taxi-destination, we keep the original model prediction. We are now using the crawled list of names as shown in Table 2. We believe the correction method can be more generalizable if more accurate candidates can be collected.

2.3.2 Time Error Correction

In this Speech-Aware Dialog Systems Technology Challenge, the ground truth of temporal expressions in dialog states is using "12-hour clock" time convention. The Latin abbreviations "a.m." and "p.m." are normalized to "am" and "pm". We noticed that the temporal expression in ASR hypothesis is different from the ground truth format in both the time convention and the Latin suffixes, as exemplified in the Introduction section. For instance, "3:08 am" in ground truth becomes "3o8 a m" in ASR hypothesis. In addition, ASR output "12o4am" needs to be normalized to "0:04 am". After analyzing ASR outputs, we designed some time normalization rules as a part of the post-processing for further improvement on the DST accuracy.

3 Experiments

3.1 Dataset and Metrics

The challenge dataset is based on MultiWOZ dataset version 2.1 with the following changes:

- Increased the domains from 5 to 7 (new domains: bus and hospital) and valid slots from 30 to 35 (new slots: bus-departure, bus-destination, bus-leaveat, bus-day and hospital-department)
- The original slot values were replaced with new values in the dev and test sets. The main purpose of the changes is to remove the slot values that overlap between the two sets and the training data.
- All time mentions were offset by a constant amount for each dialogue, and the format was standardized to "12-hour clock" with Latin abbreviations "am" or "pm".

TTS Verbatim	**Dev**		**Test**	
	JGA↑	**SER↓**	**JGA↑**	**SER↓**
D3ST-XXL	26.30	27.50	-	-
D3ST-XXL (100x)	40.80	-	-	-
TripPy (4x)	-	-	21.90	32.80
Ours (4x)	44.34	16.79	37.49	20.42
Human Verbatim	**Dev**		**Test**	
D3ST-XXL	22.60	31.60	-	-
D3ST-XXL (100x)	-	-	-	-
TripPy (4x)	-	-	21.20	33.50
Ours (4x)	-	-	29.96	26.94
Human-paraphrased	**Dev**		**Test**	
D3ST-XXL	-	-	-	-
D3ST-XXL (100x)	-	-	-	-
TripPy (4x)	-	-	20.00	33.80
Ours (4x)	-	-	30.66	26.26

Table 3: Performance of our method against baselines. The results w.r.t. D3ST-XXL are provided by the track organizers. The models are evaluated by joint goal accuracy (JGA) and slot error rate (SER). The number in the bracket indicates the amount of augmented data used for model training. For example, "(100x)" means that the amount of augmented data is 100 times of the original MultiWOZ training data. "-" indicates that result for that particular entry is not available.

- Provided 100x training data from data augmentation.

Two evaluation metrics are used to evaluate the performance of the multi-domain DST model. The Joint Goal Accuracy (JGA) and Slot Error Rate (SER) are the primary and secondary metrics, respectively. The metrics can be calculated using the standard MultiWOZ evaluation script[1] after applying the released patch from the organizer[2].

3.2 Implementation

We implement the proposed method with the above described pre-processing and post-processing based on the T5-base model. The model has 220M parameters and 12 encoder-decoder layers. Each layer has 12-headed attention with hidden size 768. We train the model with batch size 32 for 10 epochs. The learning rate is 0.0001 with early stopping patience of 3. The input and output sequence lengths are set to 512 and 128 respectively.

The computational cost is not high. For our training using MultiWOZ 2.1 dataset (4x) on a work-

[1] https://github.com/Tomiinek/MultiWOZ_Evaluation

[2] https://storage.googleapis.com/gresearch/dstc11/patch.2022-11-02.txt

Task	Correction Method	Test set	
		JGA	SER
TTS Verbatim	No correction	29.48	25.55
	ASR correction	34.07	22.28
	Name correction	32.08	23.65
	Time correction	31.97	23.40
	Name & time correction	35.20	21.50
	All corrections	**37.49**	**20.42**
Human Verbatim	No correction	25.49	30.81
	ASR correction	27.40	28.90
	Name correction	27.62	28.95
	Time correction	26.17	30.06
	Name & time correction	28.51	28.20
	All corrections	**29.96**	**26.94**
Human Paraphrased	No correction	24.47	30.63
	ASR correction	27.78	28.14
	Name correction	26.98	28.74
	Time correction	25.20	29.81
	Name & time correction	28.09	27.92
	All corrections	**30.66**	**26.26**

Table 4: Ablation Study of pre-processing and post-processing on test dataset

station with the following specifications: Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz, 256G Memory, using one NVIDIA RTX A6000 GPU, every epoch takes about 48 minutes and 141 minutes for T5-small and T5-base respectively.

3.3 Baselines

We compare our proposed approach against two baselines, D3ST (Zhao et al., 2022) and TripPy (Heck et al., 2020).

- D3ST adopts a seq2seq backbone for dialogue state tracking, and relies on slot and intent descriptions to instruct the model. The input to D3ST is a concatenation of slot descriptions, intents, and the dialogue history. The decoder of D3ST will generate the active slot key-value pairs and the corresponding intents of the current user utterance. We report the results with respect to the best D3ST version, D3ST-XXL (11B parameters), which are released by the track organizers.

- TripPy makes use of three different copy mechanisms for DST, extracting values from the dialog context on-the-fly and combining span-based slot filling methods with memory-based methods. Different from D3ST and our proposed method, the backbone of TripPy is BERT (Devlin et al., 2019), a bidirectional encoder-only model. We hypothesize that it may not generalize as well as seq2seq model, such as T5 (Raffel et al., 2022) to out-of-distribution dialogue with unseen slot key-value pairs.

3.4 Results and Analysis

We used 4x out of the 100x augmented data as additional training data, as we did not have time to process the whole 100x data for training. Table 3 shows the performance evaluation results of our model as well as the baselines w.r.t. the development and test datasets of the challenge. From the results, we can make the following observations: (1) the JGA the D3ST-XXL (100x) is significantly better than D3ST-XXL. The original D3ST-XXL model performs poorly on the modified dev set, which contains a different set of slot key-value pairs than the training data. D3ST-XXL (100x) is less impacted by the modification. This showcases that data augmentation is useful in improving the generalization of the DST models. (2) Our approach performs better than D3ST-XXL (100x) on the development set by 3.54 JGA scores. In addition, it is more data efficient as we only use 4% of the augmented data while D3ST-XXL (100x) utilizes all the augmented data. This can be attributed to the incorporation of ASR correction module. (3) Our approach performs significantly better than TripPy across all the three settings in terms of both JGA and SER. A possible reason is that TripPy is a BERT-based classification model, which may not be robust to distribution shift (in terms of both change in slot key-value pairs and the ASR errors). In addition, the copy mechanisms of TripPy heavily rely on the ontology of the MultiWOZ dataset, which is not present in the data provided by the track organizers.

Table 4 shows the results of ablation study, estimating the impact of the pre-processing of ASR error correction and the post-processing of name error correction and time error correction. The results prove that all the proposed pre-processing and post-processing modules are impactful. In particular, the post-processing (name and time error correction) has the highest performance gain probably due to the high frequency of temporal expressions and the name recognition errors in dialog states. The ASR correction module can help on the time normalization but it is hard to correct and name recognition errors.

Index	ASR Input	2-stage FT	ROUGE-1	ROUGE-2	ROUGE-L
Error Corr. 1	ASR #1	No	96.46	93.57	96.97
Error Corr. 2	ASR #2	No	97.40	95.34	97.78
Error Corr. 3	Multi-Source	No	97.54	95.56	97.89
Error Corr. 4	Multi-Source	Yes	**97.87**	**96.26**	**98.17**

Table 5: Detailed evaluation results of ASR error correction with different hyper-parameters, including using multiple ASR models' hypotheses or not and using the 2-stage fine-tuning or not, on the development set. ASR #1 and #2 refer to our own ASR model and Google ASR model, respectively. The joint model means we use both ASR transcripts.

Error Corr. model	TTS-JGA.	TTS-Slot Acc.	Human-JGA	Human-Slot Acc.
ASR #1	22.76	94.66	17.21	93.44
ASR #2	27.67	95.72	23.69	94.88
ASR #1 + Error Corr. 1	27.71	95.65	21.27	94.44
ASR #2 + Error Corr. 2	29.59	95.86	24.99	95.03
Error Corr. 3	30.58	96.13	25.98	95.36
Error Corr. 4	**32.71**	**96.40**	**26.14**	**95.40**
Ground Truth	44.71	97.21	44.71	97.21

Table 6: Benchmarking results with different ASR error correction methods. The results on dev set are reported. ASR #1 and #2 refer to our own ASR model and Google ASR model, respectively. Error Corr. $1\tilde{4}$ are ASR error correction models listed in Table 5. Ground Truth indicates using ground truth of user utterance instead of ASR hypothesis. "Slot Acc." indicates slot value accuracy.

3.5 Ablation Study on ASR Error Correction

We further conduct ablation study on our pre-processing module: multi-source ASR error correction. We choose the naive T5 model to test the ASR error correction module and filter the possible effect of other components. The results are shown in Table 5 and Table 6. In Table 5, the result is evaluated by ROUGE scores to determine the similarity between clean text with references. We can see that 1) the Google ASR model provides more reliable results than our own ASR model; 2) The performance can be boosted by integrating both models in a multi-source ASR error correction framework; and 3) the two-stage fine-tuning process can further boost the model performance.

Further, we did experiments of integrating the different methods of ASR error correction with the *T5-small* model for the DST task and evaluate the model performance on the development set. The results are shown in Table 6. The results show that each of the two ASR models contributes to significant improvement of the DST model performance and also have synergy when used together for ASR error correction. From the results, we can conclude that the ASR error correction model contributes to the speech-aware DST task.

4 Conclusions

We present our submissions to the Speech Aware Dialog Systems Technology Challenge of DSTC 11. Our works are based on pre-trained encoder-decoder language models. We enhance their performance by integrating ASR error correction, name correction and time correction modules. Our ASR error correction module utilizes two ASR models synergistically. We crawled names from Web sites and use them for the name correction module, instead of relying on the given training data. We manually wrote rules to fix errors of temporal expressions in ASR hypothesis.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Samrat Dutta, Shreyansh Jain, Ayush Maheshwari, Ganesh Ramakrishnan, and Preethi Jyothi. 2022. Error correction in asr using sequence-to-sequence models. *arXiv preprint arXiv:2202.01157*.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Yichong Leng, Xu Tan, Rui Wang, Linchen Zhu, Jin Xu, Wenjie Liu, Linquan Liu, Tao Qin, Xiang-Yang Li, Edward Lin, et al. 2021a. Fastcorrect 2: Fast error correction on multiple candidates for automatic speech recognition. *arXiv preprint arXiv:2109.14420*.

Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiangyang Li, Edward Lin, and Tie-Yan Liu. 2021b. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. *Advances in Neural Information Processing Systems*, 34:21708–21719.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2022. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819, Florence, Italy. Association for Computational Linguistics.

Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, and Yonghui Wu. 2022. Description-driven task-oriented dialog modeling. *arXiv preprint arXiv: Arxiv-2201.08904*.

Jeffrey Zhao, Mahdis Mahdieh, Ye Zhang, Yuan Cao, and Yonghui Wu. 2021. Effective sequence-to-sequence dialogue state tracking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7486–7493, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.