Confident Slot Iterative Learning for Multi-Domain Dialogue State Tracking

Qingyue Wang[♠♣]**, Yanan Cao**[♠1]**, Piji Li**[♢]**, Yanhe Fu**[♠♣]**, Zheng Lin**[♠]**, Cong Cao**[♠]**, Shi Wang**[♡] and **Li Guo**[♠]
♠ Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
♣ School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
♡ Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences,
Institute of Computing Technology, Beijing, China
♢ College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
{wangqingyue,caoyanan,fuyanhe,linzheng,caocong,guoli}@iie.ac.cn
lipiji.pz@gmail.com wangshi@ict.ac.cn

Abstract

Dialogue State Tracking (DST), a key component of task-oriented dialogue systems, tracks user intentions by predicting the values of pre-defined slots in a dialogue. Existing works on DST treat all slots indiscriminately and independently, which ignores the relationships across slots and limits the learning of hard slots (those slots are hard to be predicted correctly), eventually hurting overall performance. In this paper, we propose an iterative learning framework, i.e. iteratively updates the dialogue state with confident slots, to alleviate the aforementioned problem. Specifically, we first employ a scorer to estimate slot confidence. Then, those slots with high confidence are utilized to update the previous state, and the updated state will be fed into the scorer again to recalculate the confidence. In the last iteration, we apply an objective with the confidence penalty to focus on the hard slots. The experiments show that our approach outperforms existing methods on popular datasets.

Keywords: dialogue state tracking; iterative learning; slot confidence

Introduction

Task-oriented dialog systems help users to achieve specific goals using natural languages, such as movie booking and information support. Dialogue state tracking (DST) aims to track the users' requirements at each turn of the dialogue, which consists of a set of slot-value pairs (Young, Gasic, Thomson, & Williams, 2013; Wu et al., 2019). Accurate DST can largely help the downstream tasks of dialogue systems, such as response generation.

Early dialogue state tracking approaches extract value for each slot in a single domain (Williams, Raux, & Henderson, 2016; Henderson, Thomson, & Williams, 2014). Recently, motivated by commercial dialogue systems like Apple Siri and Google Assistant, developing the multi-domain DST becomes an agent amend in real-world applications. Figure 1 shows a multi-domain conversation, involving hotel, restaurant, and taxi domains, and the DST module extracts the slot-value pairs for each domain and each turn. In multi-domain DST, some models have achieved outstanding performances by taking the advantage of deep neural networks and pre-trained language models (Lee, Lee, & Kim, 2019; Sun, Bao, Wu, & He, 2022; Ye, Manotumruksa, Zhang, Li, & Yilmaz, 2021). However, previous works generally treat all slots equally in each turn, which ignores two issues : (1) There exist different types of relationships across slots in

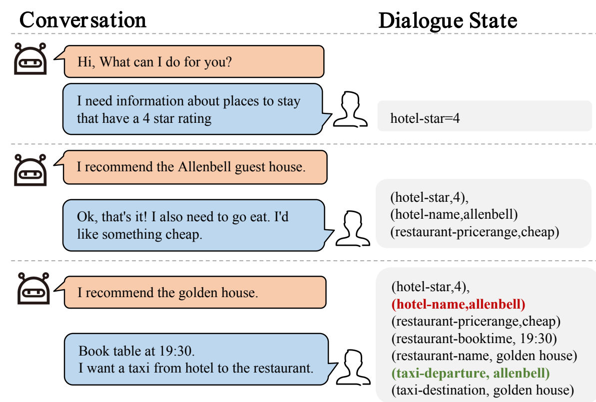


Figure 1: A conversation and its dialogue state for each turn in MultiWOZ dataset (Budzianowski et al., 2018).

multi-domain dialogue, such as mutual exclusive and value sharing. In Figure 1, the slot *taxi-departure* shares the same value with *hotel-name*. But the destination of the taxi must be different from its departure. Therefore, independently predicting each slot-value pair can't consider the useful information from other slots, which hinders the further development of models. (2) These slots have different prediction difficulties, which are mainly caused by the imbalanced distribution of slots and values. Taking the MultiWOZ dataset as an illustration, *(restaurant-food, indian)* occurs 2615 times while *(taxi-destination, acorn guest house)* occurs 79 times. So, previous works perform much differently on different slots. For instance, the TripPy (Heck et al., 2020) achieves 93.03% and 66.98% accuracy on *restaurant-food* and *taxi-destination* respectively (see Figure 2). Previous works are unable to focus on hard slots (those slots are hard to predict correctly) and limit the overall performance.

To tackle the mentioned challenges, we argue that a DST model should be aware of slot confidence: (1) In the case of slot correlation, easy slots (with high confidence) can support the predictions of the hard slots (with low confidence). As shown in Figure 1, compared with slot *hotel-name*, *taxi-departure* is a hard slot because its values have to be inferred across utterances. Due to the "value sharing", the value of *hotel-name* can further help the prediction of *taxi-departure*. (2) Based on estimated confidence, the DST model can give

[1] Corresponding author.

In M. Goldwater, F. K. Anggoro, B. K. Hayes, & D. C. Ong (Eds.), *Proceedings of the 45th Annual Conference of the Cognitive Science Society*.

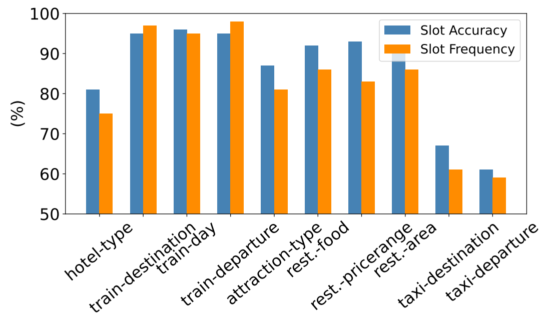


Figure 2: The predicted slot accuracy and slot frequency in MultiWOZ 2.1 dataset. The 'rest.' is the abbreviation for 'restaurant' domain.

more attention to the slots with low confidence, further improving the ability to handle hard slots.

Motivated by the above intuitions, we aim to design a DST model with three principles, estimating slot confidence, utilizing the easy slots to help the prediction of hard slots, and paying more attention to hard slots. To achieve these aims, we propose a confident slot iterative learning framework with three steps: First, given a dialogue context, the scorer module predicts the dialogue state, called the "intermediate state", and utilizes the probability of slot values to estimate the confidence. And the slot whose confidence exceeds a certain threshold is treated as "confident slot" or "easy slot". After that, the updater module modifies the previous state with those confident slots to obtain a new state. The new state will be combined with dialogue context and sent to the scorer again, aiming to re-calculate the confidence at the next iteration. Through several iterations, the confident slots are predicted more and more precisely and can support more available information for hard slots as well. At the last iteration, we apply a designed loss function with the confidence penalty, adjusting the contribution proportion of different slots to focus on those hard slots. Above all, the whole iteration process happens in one conversation turn and the framework is an end-to-end one that there is no misdelivery from the slot confidence.

In summary, the contributions of our work are as follows:

- We introduce slot confidence for DST and propose a confident slot iterative learning framework to solve the slots with different difficulties.
- Experiments show that our approach achieves outstanding performances on multi-domain datasets.

Related Works

The dialogue state can be thought of as the system's belief of the user's goal given the conversation context (Young et al., 2013; McLeay, 2022). All these methods on DST are broadly divided into two categories: classification (Lee et al., 2019; Zhang et al., 2020) and generation (Wu et al., 2019; Q. Wang et al., 2022). The classification method requires that all candidate values of each slot are provided in a pre-defined ontology and the value with the highest probability is the final prediction. Conversely, the generation way generates slot values based on the sequence-to-sequence fashion. In this paper, we focus on the classification method.

Recently, with the recent development of representation learning, most works about DST focus on encoding dialogue context with deep neural networks (Le, Socher, & Hoi, 2020; Peng et al., 2022). SUMBT (Lee et al., 2019) utilizes BERT to encode the dialogue and slot-value pairs and scores each candidate slot-value pair. DST-Picklist (Zhang et al., 2020) performs matchings between candidate values and slot-context encoding by considering all slots as picklist-based slots. DS-DST (Zhang et al., 2020) applies BERT-base to make the contextual word embeddings and extracts the values from the input as a span. LUNA (Y. Wang et al., 2022) explicitly aligns each slot with its most relevant utterance and predicts the value based on this aligned utterance. However, the above methods ignore the correlation among slots and predict each slot separately. SST (Chen et al., 2020) designs the schema graphs which contain slot relations in edges and fuse information from utterances. CSFN-DST (Zhu, Li, Chen, & Yu, 2020) encodes the dialogue context and schema graph by using internal and external attention mechanisms. STAR (Ye et al., 2021) employs word-level attention to obtain slot-specific features and a stacked slot self-attention to learn the correlations among slots. Nevertheless, these state-of-the-art neural belief trackers are overconfident in their decisions and less robust (van Niekerk et al., 2021). In this paper, we consider slot confidence in DST and utilize easy slots to help hard ones through an iteration process.

Problem Formulation

A dialogue with T turns can be represented as $D = \{(R_1,U_1),(R_2,U_2),\ldots,(R_T,U_T)\}$, where R_t and U_t represent system response and user utterance of turn t, respectively. At the turn t, we denote the dialogue context $C_t = \{(R_1,U_1),(R_2,U_2),\ldots,(R_t,U_t)\}$ and the dialogue state as $\mathcal{B}_t = \{(s,v)|s \in \mathcal{S}, v \in \mathcal{V}_s\}$, where s is a slot name from the pre-defined slot set $\mathcal{S}$ and v is the slot value from candidates $\mathcal{V}_s$. Following the convention of existing works (Lee et al., 2019; Kumar, Ku, Goyal, Metallinou, & Hakkani-Tür, 2020), each slot is represented as a special token concatenated by domain and slot, such as *restaurant-food*. Given the dialogue context C_t, the DST is asked to extract the dialogue state $\mathcal{B}_t$ at turn t, i.e. a set of slot-value pairs.

Approach

Overview

The architecture of the proposed model is illustrated in Figure 3. The confident slot iterative learning framework consists of two modules, a confidence scorer which produces an intermediate dialogue state and slot confidence, and an updater

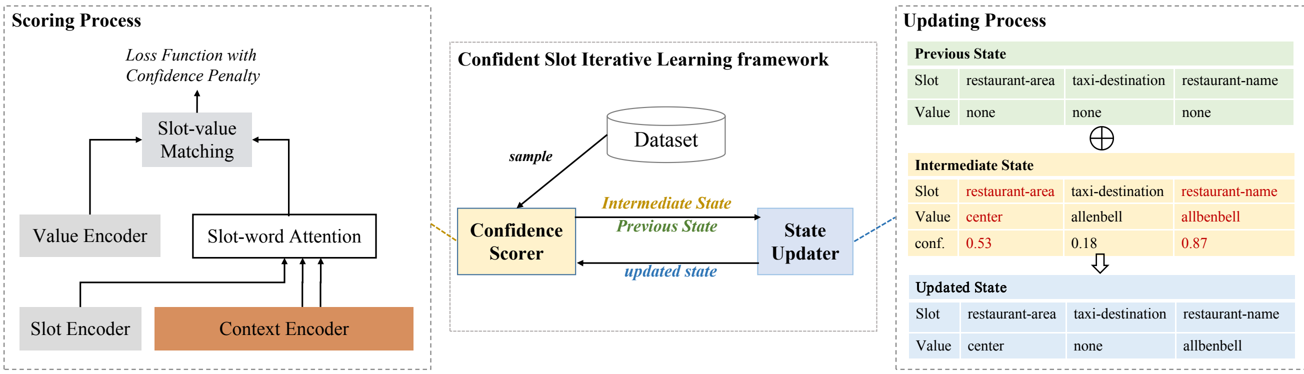


Figure 3: The architecture of confident slot iterative learning framework (best viewed in color). "conf." refers to the slot confidence. Those confident slots (conf.> 0.5) are marked as red. Noted that the whole iteration happens in one dialogue turn.

module modifying the previous dialogue state with confident slots. In the following part, we will describe the detailed architecture and illustrate the process of training and inference.

Confidence Scorer

The scorer mainly contains four components, the context encoder, slot and value encoder, slot-word attention module, and slot-value matching module. The encoders obtain semantic vector representations of dialogue contexts, slots, and candidate values. The slot-value attention module is to retrieve the relevant information corresponding to each slot and the slot-value matching module computes the semantic distance to predict the slot value and estimate the confidence.

Context Encoder Following Ye et al. (2021), we concatenate dialogue context C_t and previous dialogue state $\mathcal{B}_{t-1}$ as the input of context encoder:

$$X_t = [CLS]C_t[SEP]M_{t-1}[SEP] \tag{1}$$

where $M_{t-1} = \bigoplus_{(s,v)\in\mathcal{B}_{t-1}, v\neq none} s \oplus v$ and $\oplus$ is the operation of sequence concatenation. That means that we only include the slots with the value "non-none" in the previous turn dialogue state $\mathcal{B}_{t-1}$. After that, a pre-trained BERT (Devlin, Chang, Lee, & Toutanova, 2019) is applied as the context encoder to represent the input X_t:

$$\boldsymbol{H}_t = \text{BERT}_{finetune}(X_t) \tag{2}$$

where $\boldsymbol{H}_t$ are the representations of all tokens in X_t.

Slot and Value Encoder Previous works (Lee et al., 2019) have shown that fixing the slot-value encoder's weights during training allows the model to maintain the encoded contextual vector of new domains and slot types. Therefore, we leverage another fixed pre-trained BERT to encode slot and candidate values.

$$\begin{aligned} q^s &= \text{BERT}^{[CLS]}_{fixed}([CLS] \oplus s \oplus [SEP]) \\ q^v &= \text{BERT}^{[CLS]}_{fixed}([CLS] \oplus v \oplus [SEP]) \end{aligned} \tag{3}$$

where the vectors corresponding to token "[CLS]" are the final representations.

Slot-word Attention Module The multi-head attention (Vaswani et al., 2017) is applied to obtain slot-related information in context. The query Q vector is slot vector q^s and K matrix and V matrix are the context representation H_t:

$$\begin{aligned} O^s &= \text{MultiHeadAttn}(q^s, H_t, H_t), \\ y^s &= \text{LayerNorm}(W_o O^s_t + b_o) \end{aligned} \tag{4}$$

where y^s is the predicted value representation for slot s.

Slot-Value Matching Module The value with the smallest distance between outputs y^s_t and the representation of target value q^v_t is chosen as the prediction of slot s. The probability of candidate value v at turn t is computed as follows:

$$p(v|X_t, s) = \frac{exp(-d(y^s, q^v))}{\sum_{v\in\mathcal{V}_s} exp(-d(y^s, q^v))} \tag{5}$$

where d is the Euclidean distance.

Loss Function with Confidence Penalty Most existing works on DST (Kim, Yang, Kim, & Lee, 2020; Ye et al., 2021) usually apply common Cross Entropy Loss as the objective, ignoring the difference among slots. In this paper, we introduce slot confidence and focus on low-confident slots by improving their contribution proportions in the loss function:

$$\mathcal{L} = -\sum^{|\mathcal{S}|}(1-c^s)^\gamma \log p(v|X_t, s) \tag{6}$$

where $c^s \in [0,1]$ is the confidence of slot s , and γ is a hyperparameter named confidence penalty factor. As a primary attempt on the DST task, we treat the predicted probability $p(v|X_t, s)$ as the **slot confidence**. The smaller the confidence of the slot is, the harder it is to predict, resulting in a larger proportion of the loss function.

Above all, the confidence scorer module produces an intermediate state and denotes it as $I_t = \{((s,v),c^s))|s \in \mathcal{S}, v \in \mathcal{V}_s\}$, where $v = \arg\max_{v\in\mathcal{V}_s} p(v|C_t, s)$ and $c^s = max_{v\in\mathcal{V}_s}(p(v|C_t, s))$.

State Updater

The key idea of updating process is to utilize the *confident slots* to update the previous state. In this paper, a confident

Table 1: Joint goal accuracy (%) and slot accuracy (%) on MultiWOZ 2.0, 2.1, and 2.2 vs. various baselines.

Model	MultiWOZ 2.0		MultiWOZ 2.1		MultiWOZ 2.2	
	JGA (%)	SA (%)	JGA (%)	SA (%)	JGA (%)	SA (%)
Generation Method						
TRADE (Wu et al., 2019)	48.60	96.92	45.60	-	45.40	-
SOM-DST (Kim et al., 2020)	51.72	-	53.01	-	53.81	-
TripPy (Heck et al., 2020)	53.51	-	55.29	-	53.52	-
Classification Method						
SUMBT (Lee et al., 2019)	42.40	96.44	52.57	97.51	-	-
DST-Picklist (Zhang et al., 2020)	54.39	-	53.30	97.40	-	-
DS-DST (Zhang et al., 2020)	-	-	51.21	97.35	51.70	-
CSFN-DST (Zhu et al., 2020)	51.57	-	52.88	-	-	-
SST (Chen et al., 2020)	51.17	-	55.23	-	-	-
STAR (Ye et al., 2021)	52.93	96.43	55.23	96.52	54.61	97.38
LUNA[1] (Y. Wang et al., 2022)	54.31	97.15	55.29	96.30	54.93	97.42
Our Model	**55.58**	**97.53**	**56.20**	**98.10**	**55.88**	**97.64**

slot refers to the confidence of a slot that exceeds a certain threshold. Specifically, we use those confident slots in I_t and maintain old ones in previous state $\mathcal{B}_{t-1}$ to construct the updated state:

$$\text{Updater}(\mathcal{B}_{t-1}, I_t) = \begin{cases} (s,v) \in I_t, c^s > \alpha \\ (s,v) \in \mathcal{B}_{t-1}, c^s \leq \alpha \end{cases} \tag{7}$$

where the α is a hyper-parameter and refers to the confidence threshold of the slot. At the next iteration, the updated state will be concatenated with dialogue context C_t and fed into the scorer module again to re-compute the slot confidence.

Training and Inference

The whole iteration learning process is illustrated in Algorithm 1. In the training phase, the scorer samples dialogue from the trainset and estimate the slot confidence (line 5). Then, we update the previous dialogue state with confident slots (line 6) to obtain a new state. After several iterations, the ultimate predicted dialogue state is leveraged to train the scorer (line 8,9). As all dialogue samples go through, we then continue to train the scorer for extra epochs until convergence. During inference, the state I_t, obtained at the last iteration, is the final predicted dialogue state at turn t.

Algorithm 1: Confident Slot Iterative Learning

Input: The training dataset $\mathcal{D}$; the Scorer module; the Updater module; the number of training epochs N; the number of iterations L

1 **for** $i \leftarrow 0$ **to** N **do**

2 Take a sample $((C_t, \mathcal{B}_{t-1}), \mathcal{B}_t)$ from dataset $\mathcal{D}$;

3 **for** $k \leftarrow 1$ **to** L **do**

4 $I_t \leftarrow \text{Scorer}(C_t, \mathcal{B}_{t-1})$;

5 $\mathcal{B}_{t-1} \leftarrow \text{Updater}(\mathcal{B}_{t-1}, I_t)$

6 **end**

7 Compute loss using I_t and $\mathcal{B}_t$ as Eq. 6.;

8 Update the parameters of Scorer.

9 **end**

[1]For a fair comparison, we remove the auxiliary task in this method and reproduce the results using the source codes.

Experiments

Datasets and Evaluation Metrics We make the comparison on public popular dialogue datasets. The MultiWOZ dataset (Budzianowski et al., 2018) is the largest publicly available multi-domain task-oriented dialogue dataset, including about 10,000 dialogues within 7 domains and 35 slots. Following previous works (Wu et al., 2019), only five domains (restaurant, hotel, attraction, taxi, and train) are used in the experiments. Previous works (Ye et al., 2021; Qian et al., 2021) point that the MultiWOZ dataset owns obvious slot correlations and data imbalance, thus it is suitable to verify our framework. The WOZ 2.0 (Rojas-Barahona et al., 2017) and DSTC2 (Henderson et al., 2014) are standard single-domain benchmarks for task-oriented dialogue systems, which both contain 3 slots in the restaurant domain. Evaluations on single-domain datasets aim to prove the robustness of our model.

We follow Kim et al. (2020) to use the joint goal accuracy (JGA) and slot accuracy (SA) as evaluation metrics. JGA treats a prediction as correct only if for every turn all slots exactly match the ground truth values. SA is defined as the fraction of slots for which the model predicts the correct value.

Implementation Details We implement the framework in Pytorch (Paszke et al., 2019). We employ bert-base-uncased model as the encoders and only the parameters of the context encoder are fine-tuned. The input sequence length in encoders is set to 512. We set the learning rate and warmup proportion to 5e-5 and 0.1. We use a batch size of 8 and set the dropout rate (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) to 0.1. The threshold of confidence α is 0.6 and the penalty factor is 0.06 in the loss function. The optimal number of iterations L is 2.

Main Results

The results of our model and baselines on MultiWOZ datasets are shown in Table 1. As expected, our proposed model achieves state-of-the-art performance on three version

Table 2: Domain-special accuracy (%) on MultiWOZ 2.1. The SOM-DST (Kim et al., 2020) and TripPy (Heck et al., 2020) are both generation methods.

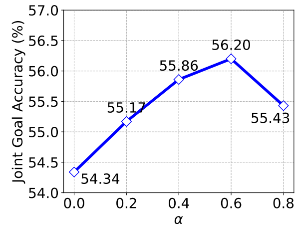
Domain	CSFN-DST	SOM-DST	TripPy	Our Model
Hotel	46.29	49.53	50.21	**52.75**
Train	69.79	70.36	72.51	**76.22**
Taxi	47.35	59.69	37.54	**66.56**
Restaurant	64.64	65.72	**70.47**	69.31
Attraction	64.78	69.83	**73.37**	70.43

Table 3: The joint goal accuracy results on WOZ 2.0 and DSTC2 dataset.

Models	WOZ 2.0	DSTC2
	JGA(%)	JGA(%)
BERT-DST (Chao & Lane, 2019)	87.7	69.3
StateNet (Ren, Xie, Chen, & Yu, 2018)	88.9	75.5
SUMBT (Lee et al., 2019)	90.9	85.6
STAR* (Ye et al., 2021)	87.3	83.7
Our Model	**92.3**$^{\Uparrow +1.4}$	**86.3**$^{\Uparrow +0.7}$

Table 4: Impact of different components of the proposed framework on MultiWOZ 2.1.

Model	Joint Goal Accuracy (%)
Our Model	**56.20**
- Iteration Mechanism	55.04$^{\Downarrow -1.16}$
- Designed Loss	55.87$^{\Downarrow -0.33}$
- Above all	54.89 $^{\Downarrow -1.31}$



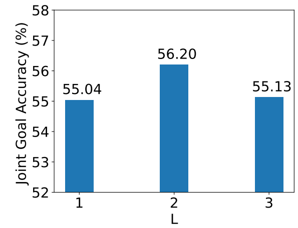


Figure 4: Impact of the different confidence threshold.

Figure 5: Impact of the number of the iteration.

datasets with the joint goal accuracy of 55.58%, 56.20%, and 55.88% respectively. Compared with the other baselines (SST, CSFN-DST, and STAR) that have considered slot correlation, our approach achieves 1.27%∼2.65% absolute performance on MultiWOZ datasets. It's because we obtain confident predictions by the iteration framework, which can build implicit dependency between easy slots and hard slots. Importantly, different from those models which structure the schema graph with prior knowledge (e.g. SST and CSFN-DST), our model doesn't rely on any extra information and is a fully data-driven approach. In Table 2, we present the results for each domain in MultiWOZ 2.1 dataset. The domain-special accuracy is calculated on a subset of the predicted dialogue state. As shown in Table 2, our approach achieves a great improvement in the train and taxi domains. We analyze that these two domains usually share same values with other slots, such as *hotel-name* and *attraction-name*, which easily benefit from the proposed iteration process.

The improvements on WOZ 2.0 and DSTC2 datasets are shown in Table 3. From the results, our model achieves superior performance than the STAR. We analyze that the STAR model leverages a self-attention layer to capture the correlations among slots, which inevitably introduces unreliable information and hurts the prediction seriously. Besides, there are many low-frequency slot values in WOZ 2.0 dataset. Our model with designed loss alleviates the challenge of low-frequency slots to some degree and doesn't hurt the performance even in single-domain.

Discussion

Ablation Study

Impact of Different Components Table 4 illustrates the effectiveness of each part in the proposed framework. (1) We remove the iteration mechanism (set $L = 1$) and only train the scorer module with the designed loss as the DST model. The joint goal accuracy is decreased by 1.16 (%), proving the effectiveness of the available confident slots in the iteration. (2) We train the framework with common CrossEntropy loss (set $\gamma = 0$) to study the impact of the designed loss function. One can observe that the performance deteriorates considerably without the confidence penalty. It illustrates that treating all slots indiscriminately will hurt the overall performance. (3) We remove both the iteration framework and designed loss and find that this severely damages the model performance. Above all, the results indicate that the components of our model are indispensable.

Impact of Threshold of Confidence The threshold value of slot confidence in Equation 7 controls the trade-off between the quality and the number of confident slots. We study the impact of the threshold of confidence and the results are shown in Figure 4. As shown, the threshold value of 0.6 achieves the highest joint goal accuracy on MultiWOZ 2.1, while the accuracy drops by more than 1.0% when using a small threshold (0.2). In fact, a high confidence threshold increases the accuracy of easy slots but decreases the number contributed to the next iteration. The experiment results suggest that the quality of the intermediate dialogue state is more important than the quantity for reaching a better performance.

Impact of the Number of Iteration To study the impact of the number of iterations in the proposed framework, we try the number from one to three to observe the change in the model's performance (see Figure 5). As shown, we find that our model with 2 iterations achieves the best performance on MultiWOZ 2.1 dataset. We explain that too many iterations might produce the wrong prediction for easy slots while too few iterations can't utilize the results of easy slots, which both

Operation Type	Dialogue Context	1st Iteration	2nd Iteration
Adding	User: The area does not matter. Anywhere where I can get a table for 5 at 15:30 on Saturday . System: Ok , I've booked at Yu Garden for 5 at 15:30 on Saturday . User: Great!	(restaurant-name, none) (restaurant-area, do not care) (restaurant-bookpeople, 5) (restaurant-booktime, 15:30)	(restaurant-name, Yu Garden) (restaurant-area, dontcare) (restaurant-bookpeople, 5) (restaurant-booktime, 15:30)
Modification	System: Do you have any particular train stations in mind? User: Yes , the Cambridge station. I would like to leave on Sunday after 10:00 for the Stansted Airport .	(train-departure,the Stansted Airport) (train-destination, the Stansted Airport)	(train-departure,cambridge) (train-destination,the Stansted Airport)
Deletion	System: Hi, what can I do for you? User: Please find a restaurant called stazione restaurant and coffee bar	(restaurant-name, stazione restaurant and coffee bar) (hotel-name, el shaddia guest house)	(restaurant-name, stazione restaurant and coffee bar) (hotel-name, none)

Table 5: Three operation types in the iterative process. The slots with low confidence (below the threshold) are marked in red. The slots predicted correctly the second time are marked in blue.

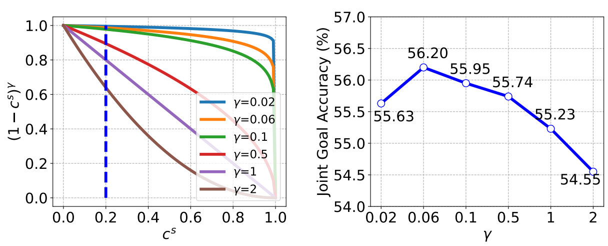


Figure 6: Analysis of confidence penalty factor in loss proportion (left) and JGA (right).

hurt the model's performance.

Analysis of Confidence Penalty

The penalty factor γ in Eq. 6 controls the contribution proportion for each slot in the total loss. When γ is set to 0, the loss function becomes the vanilla Cross-Entropy function. When $\gamma \neq 0$, the function actively adjusts the model to focus on harder slots. We further analyze its influence on the loss proportion and joint goal accuracy in Figure 6. In the left of Figure 6, as the γ decreases, those slots with lower confidence (blue dotted line) take a bigger proportion in the loss. Accordingly, the model achieves a better performance with a smaller γ (0.02$\sim$0.1) in Figure 6 (right). Therefore, we suggest that the appropriate attention to difficult slot learning, i.e. training with a smaller confident penalty factor, can effectively help the DST model to improve the overall performance.

Analysis of Iteration Mechanism

To analyze the working mechanism of the proposed framework, we run an iterative model with two iterations and categorize its operation into three types (see Table 5): (1) **Adding.** In the first example, the model ignores the *restaurant-name* first but discovers its value at the second iteration. We analyze that there exists slot co-occurrence in dialogues (e.g. restaurant-book people and restaurant-name). Those predicted slots at first remind the model aware of the other slots at the next prediction. (2) **Modification**. In the second dialogue example, the model first predicts the same values for slot "train-departure" and "train-destination", which breaks the mutual exclusion principle. However, considering the confident slot "train-destination" and its value "the Stansted Airport", our model corrects the departure of the train with another value. (3) **Deletion**. In the third example, it might that the semantics of the current context is close to the value "el shaddia guest house". Given the additional information, the name of the restaurant, the semantic distance between the context and the wrong value becomes larger. Above all, we conclude that the iterative mechanism effectively enhances the semantics of dialogue context by considering confident slots, which further helps DST be aware of the implicit slot relationships, such as co-occurrence and mutual exclusive.

Analysis of Errors

To check whether the proposed framework has space for improvement, we randomly sample 50 turns on MultiWOZ 2.1 dataset and conclude four types of errors. The first error is the annotation error that the model predicts the right values while the annotation is wrong in the dataset, similar to the finding of Zhou and Small (2019). The second error type comes from the implicit slot expression inside utterances. For instance, the user shortly replies "no" when facing the question "any preference for food?". In this case, the food type "do not care" implicitly expressed in the dialogue context, which is hard to predict. The third error happens when the system informs the value of actively. Another type of error exists in value updating as the dialogue goes on. For example, the user expects the arriving time to be *08:45* while the real-time is *09:29* in the station timetable. In this case, our model fails to update the slot *train-arrive by* timely. Except for annotation errors, the other three error types are mainly caused by lacking explicit modeling of information flow between speakers, which also leads to the research points in the following works.

Conclusion

In this paper, we consider the confidence of slots and propose an iterative learning framework for multi-domain dialogue state tracking. Concretely, a scorer module first estimates the confidence for each slot. After that, we leverage the high confident slots to update the previous dialogue state through several iterations. To focus on the hard slots better, we utilize an objective with confidence penalty to improve the loss proportion of hard slots. Experimental results show that our proposed model outperforms other baselines on both single-domain and multi-domain datasets. In the future, we would like to develop other approaches to evaluate slot confidence and incorporate it into DST effectively.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (NO.2022YFB3102200) and Strategic Priority Research Program of the Chinese Academy of Sciences with No. XDC02030400. We would like to thank the anonymous reviewers for their valuable comments.

References

Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., & Gasic, M. (2018). Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Emnlp.*

Chao, G.-L., & Lane, I. R. (2019). Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *ArXiv*, *abs/1907.03040*.

Chen, L., Lv, B., Wang, C., Zhu, S., Tan, B., & Yu, K. (2020). Schema-guided multi-domain dialogue state tracking with graph attention neural networks. In *Aaai.*

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, *abs/1810.04805*.

Heck, M., van Niekerk, C., Lubis, N., Geishauser, C., Lin, H.-C., Moresi, M., & Gavsi'c, M. (2020). Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Sigdial.*

Henderson, M., Thomson, B., & Williams, J. (2014). The second dialog state tracking challenge. In *Sigdial conference.*

Kim, S., Yang, S., Kim, G., & Lee, S.-W. (2020). Efficient dialogue state tracking by selectively overwriting memory. In *Acl.*

Kumar, A., Ku, P., Goyal, A. K., Metallinou, A., & Hakkani-Tür, D. Z. (2020). Ma-dst: Multi-attention based scalable dialog state tracking. In *Aaai.*

Le, H., Socher, R., & Hoi, S. (2020). Non-autoregressive dialog state tracking. In *Iclr.*

Lee, H., Lee, J., & Kim, T.-Y. (2019). Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *Acl.*

McLeay, A. J. (2022). Joint learning of practical dialogue systems and user simulators..

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Neurips.*

Peng, W., Hu, Y., Xing, L., Xie, Y., Sun, Y., & Li, Y. (2022). Control globally, understand locally: A global-to-local hierarchical graph network for emotional support conversation. *CoRR*, *abs/2204.12749*. Retrieved from `https://doi.org/10.48550/arXiv.2204.12749`

Qian, K., Beirami, A., Lin, Z., De, A., Geramifard, A., Yu, Z., & Sankar, C. (2021). Annotation inconsistency and entity bias in multiwoz. *ArXiv*, *abs/2105.14150*.

Ren, L., Xie, K., Chen, L., & Yu, K. (2018). Towards universal dialogue state tracking. In *Emnlp.*

Rojas-Barahona, L. M., Gai, M., Mrksic, N., hao Su, P., Ultes, S., Wen, T.-H., ... Vandyke, D. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *Eacl.*

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, *15*, 1929-1958.

Sun, H., Bao, J., Wu, Y., & He, X. (2022). Bort: Back and denoising reconstruction for end-to-end task-oriented dialog. In *Naacl-hlt.*

van Niekerk, C., Malinin, A., Geishauser, C., Heck, M., chin Lin, H., Lubis, N., ... Gavsi'c, M. (2021). Uncertainty measures in neural belief tracking and the effects on dialogue policy performance. In *Emnlp.*

Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *ArXiv*, *abs/1706.03762*.

Wang, Q., Cao, Y., Li, P., Fu, Y., Lin, Z., & Guo, L. (2022). Slot dependency modeling for zero-shot cross-domain dialogue state tracking. In *International conference on computational linguistics.*

Wang, Y., Zhao, J., Bao, J., Duan, C., Wu, Y., & He, X. (2022). Luna: Learning slot-turn alignment for dialogue state tracking. In *North american chapter of the association for computational linguistics.*

Williams, J., Raux, A., & Henderson, M. (2016). The dialog state tracking challenge series: A review. *Dialogue Discourse*.

Wu, C.-S., Madotto, A., Hosseini-Asl, E., Xiong, C., Socher, R., & Fung, P. (2019). Transferable multi-domain state generator for task-oriented dialogue systems. In *Acl.*

Ye, F., Manotumruksa, J., Zhang, Q., Li, S., & Yilmaz, E. (2021). Slot self-attentive dialogue state tracking. *WWW*.

Young, S. J., Gasic, M., Thomson, B., & Williams, J. (2013). Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*. Retrieved from `https://ieeexplore.ieee.org/abstract/document/6407655`

Zhang, J., Hashimoto, K., Wu, C.-S., Wan, Y., Yu, P. S., Socher, R., & Xiong, C. (2020). Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Starsem.*

Zhou, L., & Small, K. (2019). Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *ArXiv*, *abs/1911.06192*.

Zhu, S., Li, J., Chen, L., & Yu, K. (2020). Efficient context and schema fusion networks for multi-domain dialogue state tracking. In *The findings of emnlp.*