Applied Intelligence (2021) 51:2377–2392
https://doi.org/10.1007/s10489-020-01991-y

Robust dialog state tracker with contextual-feature augmentation

Xuejun Zhang[1] · Xuemin Zhao[2] · Tian Tan[3]

Accepted: 28 September 2020 / Published online: 3 November 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
Dialog state tracking (DST), which estimates dialog states given a dialog context, is a core component in task-oriented dialog systems. Existing data-driven methods usually extract features automatically through deep learning. However, most of these models have limitations. First, compared with hand-crafted delexicalization features, such features in deep learning approaches are not universal. However, they are important for tracking unseen slot values. Second, such models do not work well in situations where noisy labels are ubiquitous in datasets. To address these challenges, we propose a robust dialog state tracker with contextual-feature augmentation. Contextual-feature augmentation is used to extract generalized features; hence, it is capable of solving the unseen slot value tracking problem. We apply a simple but effective deep learning paradigm to train our DST model with noisy labels. The experimental results show that our model achieves state-of-the-art scores in terms of joint accuracy on the MultiWOZ 2.0 dataset. In addition, we show its performance in tracking unseen slot values by simulating unseen domain dialog state tracking.

Keywords Human-machine interaction · Task-oriented dialog systems · Dialog state tracking · Contextual self-attention · Learning with noisy labels

1 Introduction

Task-oriented dialog systems, as typical human-machine interaction systems, have attracted intensive research interest in recent years. They allow for natural, personalized interactions with users to help them achieve simple goals such as finding restaurants or booking flights. Dialog state tracking (DST) is a core component of the task-oriented dialog system [8, 42]. It estimates the state of a conversation based on the current utterance and the conversational history. In DST, a state is described by a triplet of the form (domain, slot, value). This triplet represents the values of requested slots given an active domain. A turn state is the specific value of the current utterance. A joint goal is the set of all turn states accumulated across conversational turns. DST aims to track the joint goal of a dialog. Figure 1 shows an example of a dialog with an annotated dialog state, in which the user first books train tickets, and then considers attractions.

For the multi-domain dialog state tracking problem, we assume that there are M domains. $D = \{d_1, d_2, ..., d_M\}$. Taking the MultiWOZ 2.0 dataset as an example, there are seven domains, namely, hotel, taxi, restaurant, attraction, train, police and hospital.Each domain $d \in D$ has N^d slots $S^d = \left\{s_1^d, s_2^d, s_3^d, ...s_{N^d}^d\right\}$, and each slot $s \in S^d$ has K^s possible values $V^s = \{v_1^s, v_2^s, v_3^s, ...v_{K^s}^s\}$. For example, the hotel domain has a slot named area, and the possible values are east, west, south, north and central. However, for some slots, it is difficult to predefined their values V^s in the domain-ontology. Taking the slot named arrive by in the train domain as an example, since the size of V^s can be very large, it is a bad choice to list all possible arrival times that the user may require. In addition, the domain-ontology also varies with time. We denote the dialog context as $X = \{(U_1, R_1), (U_2, R_2), ..., (U_T, R_T)\}$, where U_t is the user utterance and R_t is the agent response in turn t. Formally, we represent the dialog state as $B = \{B_1, B_2, ...B_T\}$, where B_t is the dialog state in turn t.

✉ Xuejun Zhang
zhangxuejun@hccl.ioa.ac.cn

Tian Tan
tiantan@stanford.edu

1 Key Laboratory of Speech Acoustics and Content Understanding, Institute of Acoustics, Chinese Academy of Sciences, Beijing, 100190 China

2 Alibaba Group, Beijing, 100102 China

3 Stanford University, 450 Serra Mall, Stanford, CA 94305, USA

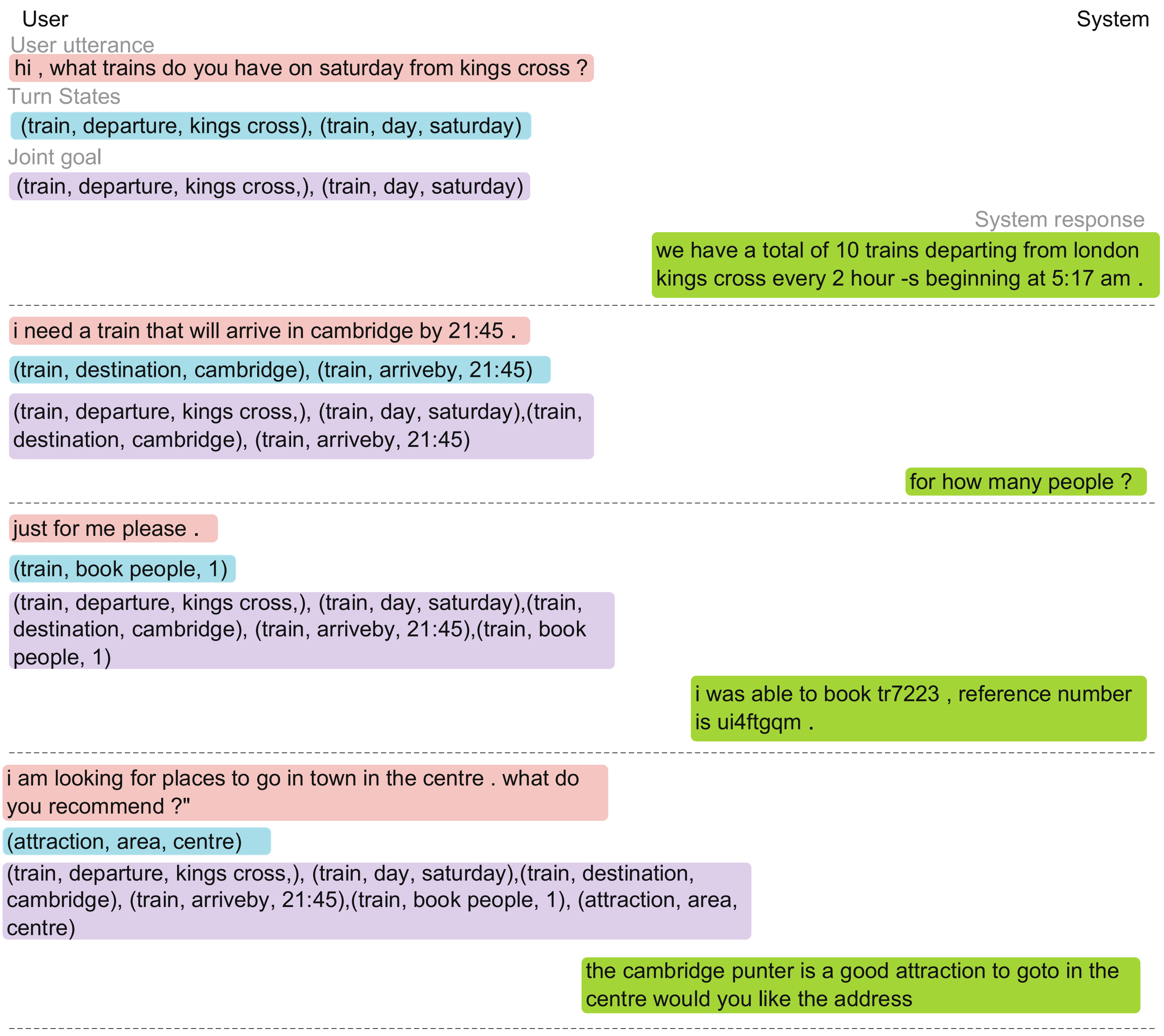
User
System
User utterance
hi , what trains do you have on saturday from kings cross ?
Turn States
(train, departure, kings cross), (train, day, saturday)
Joint goal
(train, departure, kings cross,), (train, day, saturday)
System response
we have a total of 10 trains departing from london kings cross every 2 hour -s beginning at 5:17 am .
i need a train that will arrive in cambridge by 21:45 .
(train, destination, cambridge), (train, arriveby, 21:45)
(train, departure, kings cross,), (train, day, saturday),(train, destination, cambridge), (train, arriveby, 21:45)
for how many people ?
just for me please .
(train, book people, 1)
(train, departure, kings cross,), (train, day, saturday),(train, destination, cambridge), (train, arriveby, 21:45),(train, book people, 1)
i was able to book tr7223 , reference number is ui4ftgqm .
i am looking for places to go in town in the centre . what do you recommend ?"
(attraction, area, centre)
(train, departure, kings cross,), (train, day, saturday),(train, destination, cambridge), (train, arriveby, 21:45),(train, book people, 1), (attraction, area, centre)
the cambridge punter is a good attraction to goto in the centre would you like the address

Fig. 1 Dialog example. The dashed lines separate the turns in the dialog. A turn contains a user utterance (pink), a corresponding turn state(blue), a joint goal (purple) and the system response (green)

The dialog state B_t is a group of (domain, slot, value) tuples. Suppose that there exist J possible (domain, slot) pairs, and Y_j represents the value of the j-th (domain, slot) pair. If the slot value has not been specified by a user or the user indicates that they do not care about the slot value, we set the slot value as "`None`" or "`does not care`". Accordingly, B is the target that the model needs to predict. The input of the DST model is domain D, dialog history X, and slot S, and the output is slot value Y.

Considering that building large-scale task-oriented datasets for new domains is costly and time-consuming, we pay attention to the state tracking ability of the model for execution on an unseen domain [4]. In realistic scenarios, the number of domains is prohibitively large. In unseen domain DST, we assume that we have no training data in a new domain; namely, we train on $M-1$ domains and test on the M-th domain. This verifies the generalization performance of our model on an unseen domain. Formally, we represent the training data as $(X, D_{source}, S_{source}, Y_{source})$ and the test data as $(X, D_{target}, S_{target}, Y_{target})$.

Traditional approaches usually extract delexicalization features to achieve generalization by replacing the slot values that vary lexically or morphologically with generic tags. However, they rely on hand-crafted generic features and complex domain-specific lexicons [17, 17, 23, 37, 47]. Recent data-driven deep learning models have shown promising performance in DST.

The picklist-based approaches treat domain-slot pairs as picklist-based slots, where the values are predicted by performing classification on the candidate value list [22, 28, 45]. They can only work on an ontology with a fixed domain; that is the slots and values in the ontology are predefined and cannot vary dynamically. However, this is not suitable for a real scenario. For example, new attractions or restaurants are often added in the tourism domain, which leads to the dynamic changes in the ontology.

Recent generative approaches generate slot values for DST without relying on fixed vocabularies or domain ontologies [19, 39, 41]. However, such generative methods would generate ill-formatted strings (e.g., repeated words) when generating long strings. For example, with a long hotel address, a small difference makes the whole dialog state tracking result incorrect.

Xu et al.[18], Wu et al.[5], and Le et al. [11] propose span-based approaches, which treat domain-slot pairs as span-based slots, where the values can be converted into substrings of dialog context. However, it is difficult to handle situations where values do not appear in the dialog context. In the example in Fig. 1, the user specifies the value of the "`book people`" slot in the "`train`" domain as "`1`" by saying "`just for me please.`".

To address the aforementioned challenges and take advantage of both span-based slots and picklist-based methods, Zhang et al.[44] and Zhou et al. [46] propose combined approaches that treat domain-slot pairs as span-based slots and picklist-based slots. The values of each span-based slot can be found through span matching with start and end positions in the dialog context, and the values for each picklist-based slot are found in the corresponding candidate value list. It is a manual process to decide whether a slot is a span-based slot or a picklist-based slot.

However, most of these models have some limitations. First, in many approaches the features extracted do not take into account some generic features similar to the delexicalization features in [17, 37]. Thus, they cannot effectively track unseen slot values. Second, existing DST systems based on deep learning generally need large datasets with high-quality labels, which require many passes of expensive human quality assurance [40]. The presence of noise in the dataset hinders the model performance. Regrettably, noisy data are unavoidable in a real scenario. For example, the recently-released MultiWOZ-2.0 [27] in its current form has certain annotation errors [11, 44]. First, the annotations lack consistency. For example, some annotations are "`guesthouse`" but others are "`guest house`". Second, there are erroneous delays in the state updates, which sometimes extend the turns in dialogs [44]. The error of delay has a negative influence on the performance of DST models.

To address these challenges, in this paper, we propose a robust dialog state tracker with contextual-feature augmentation. As mentioned above, dialogue state tracking can benefit from extracting delexicalization features that are irrelevant to specific slot values [17, 17, 23, 37, 47] . Therefore, we use a carefully modified TRADE [39] as the backbone network and extract delexicalization features by contextual-feature augmentation after the encoder stage to enhance the effectiveness of dialog history representation. Contextual-feature augmentation is achieved by a **D**iagonalized **M**asked **S**elf-**A**ttention **(DMSA)** mechanism. The DMSA extracts the contextual features of each word that are irrelevant to itself, such as "`serving` $\langle value \rangle$ `food`", that are similar to the delexicalization features. In previous works, such delexicalization features were obtained using hand-crafted semantic lexicons. To the best of our knowledge, we are the first to propose extracting similar delexicalization features with a novel neural network. In the decoder stage, our tracker uses a copy mechanism to generate slot values from the input user's utterance, where the importance of each word depends on its contextual features. For a slot, the unseen slot values have similar contextual characteristics to the existing slot values. Thus, the tracker can copy words that are relevant to the unseen slot values, which solves the unseen slot value-tracking problem.

In addition, it is well known that noise in the dataset, which is unavoidable in real scenarios, hinders the model performance. We align our motivation to the new training method (co-teaching) [13] and develop a novel and effective deep learning paradigm to train deep networks robustly. As mentioned above, there are certain annotation errors in the commonly used dataset of DST tasks. In our proposed learning paradigm, we view such annotation errors as noisy data and only sample clean data to update the parameters during training. That is, we filter data while training the model, update the model with the clean data, and discard the noisy data. With our proposed deep learning paradigm, we can train the DST model robustly even with certain annotation errors. Although our proposed deep learning paradigm is motivated by co-teaching, there are fundamental differences between them. The original co-teaching method share the loss function across the data filtering and model update steps. This, however, is not reasonable in general. The optimization loss of many tasks is composed of multiple losses, each of which has a different meaning and is not suitable for sample data. Our proposed deep learning paradigm overcomes this by segmenting the loss function into an update loss function and a sample loss function. Furthermore, we refine the algorithm for updating the model and sampling data. To the best of our knowledge, we are the first to propose an improved network training paradigm to deal with annotation errors in DST datasets for stronger performance.

Our contributions are as follows:

- A robust network model with contextual-feature augmentation is proposed for high-performance dialog state tracking.
- We propose a novel contextual-feature augmentation method with a new DMSA mechanism, which can extract the similar delexicalization features that play a crucial role in tracking unseen slot-values. Such delexicalization features are obtained using hand-crafted semantic lexicons in previous works, and we are the first to propose extracting them with a novel neural network.
- We develop a simple but effective deep learning paradigm to train our DST networks robustly with certain annotation errors .
- The experimental results show that the proposed method outperforms state-of-the-art methods on the recently-released MultiWOZ corpus.

The source code will be made available upon publication.

2 Related work

Dialog state tracking Dialog trackers often jointly learn spoken language understanding and dialog state tracking to avoid accumulating errors from spoken language understanding [15, 26, 38]. Subsequent delexicalization trackers utilize hand-crafted semantic dictionaries to hold all the key terms, rephrasing and alternative mentions to delexicalize and achieve generalization [16, 17, 32, 36, 37].

Recently *Mrkšić* et al. proposed using a convolutional neural network to learn utterance representation and achieved better performance than handcrafted feature-based models [22]. However, parameters are not shared across slots. Nouri et al. shared parameters between different slots by introducing a global module [24]. Zhong et al. proposed a global-local architecture using local module to learn slot-specific features [45]. Subsequently, Ren et al. presented a novel model named StateNet, which compares the distances between the dialog history representation and the value vectors in the value-list set [28]. However, they usually require a predefined ontology that is difficult to obtain in advance. The number of slots and values may be large and changeable even though the ontology exits.

To tackle these issues, several methods are proposed to extract the slot values through span matching with start and end positions the in dialog context. For example, Xu et al. [41] utilized an index-based pointer network to copy values from the dialog context. Chao et al., Gao et al., and Perez et al. formulated dialog state tracking as a reading comprehension task and use a neural network to determine the state as a span over tokens within the dialog context [11, 26]. However, tracking states from only the dialog context is not sufficient as many values in DST cannot be found in the context due to annotation errors or diverse descriptions for slot values from users.

Lei et al. [19] and Wu et al. [39] address a drawback of span-based methods; their model generates the values of all slots without relying on fixed vocabularies or spans by using a hierarchical decoder. In such generative methods, however, it is easy to generate ill-formatted strings when the slot value we need to generate is a long string. In contrast, both picklist-based and span-based methods can rely on existing strings rather than generating them.

Zhang et al. [44] proposed separating the slots into span-based slots, whereas a picklist-based slot depends on human heuristics. For example, the requests for "`internet`" are usually "`yes`" or "`no`" with limited choices when users book a hotel; such slots are treated as picklist-based slots. The number given in response to "`book people`" has unlimited values, and they can be found in context; such slots are treated as span-based slots. Zhou et al. [46] modeled multi-domain DST as a question-answering problem. They constructed two types of questions: 1) multiple-choice questions for slots with a limited number of value options and 2) span prediction questions, of which the answers are spans in the context, and are designed for slots that have a large or infinite number of value options. In addition, they developed a dynamically-evolving knowledge graph to improve model performance.

Deep learning with noisy labels A deep network will learn noisy data eventually during training, which makes deep learning with these noisy data challenging. Consequently, there are a number of studies that have put forward solutions to deal with noisy labels [21, 35].

For instance, Veit et al. proposed a multi-task neural network that jointly trains for cleaning noisy labels with a small amount of clean data, and they utilized this neural network to decrease noise in extensively noisy data [34]. Li et al. found that the "`side`" information, which consists of little clean data and label relations, is important for combating noisy data. Based on this motivation, they proposed a distillation algorithm to utilize this "`side`" information to "`hedge the risk`" of noisy labels [20]. Rodrigues et al. introduced a crowdsourcing layer after the output layer to learn with noisy labels [30]. Tanaka et al. designed a joint optimization framework for learning the parameters and estimating the true labels at the same time [31]. Ren et al. presented a novel reweighting algorithm that uses the recently proposed meta-learning method to assign weights to the training data according to their gradient directions [29].

3 Robust dialog state tracker

3.1 Base model

Our DST model is based on the encoder-decoder architecture with attention-based copying (TRADE) [39]. TRADE is composed of three main parts, namely, the utterance encoder, slot gate and state decoder. For each dialog turn, TRADE takes the dialog history of the turn and (domain, slot) pair as its input and then outputs the value of the (domain, slot).

A GRU is used as the encoder to encode the dialog history X_t of the current turn t into an array of fixed-length vectors.

$$H_t, h_t = GRU(X_t) \tag{1}$$

where $X_t \in \mathbb{R}^{|X_t| \times d_{emb}}$ represents the concatenation of all the words in the dialog history. $|X_t|$ is the number of words in the dialog history, and d_{emb} represents the size of the embedding. $H_t \in \mathbb{R}^{|X_t| \times d_{hdd}}$ is the encoded dialog history, and $h_t \in \mathbb{R}^{d_{hdd}}$ is the vector representation of the dialog history, where d_{hdd} is the hidden size of the encoder.

The model uses a GRU as the state decoder to decode multiple output tokens for all (domain, slot) pairs; that is, the decoder decodes J times independently for all (domain, slot) pairs. For the j-th (domain, slot) pair, the slot value is presented as $Y_j = [w_{j0}, w_{j1}, ..., w_{jK}]$, where K is the character number of the slot value. At decoding step $k \in K$, GRU takes $w_{j(k-1)}$ and $h^{dec}_{j(k-1)}$ as its inputs and outputs a hidden state h^{dec}_{jk}. $w_{j(k-1)}$ is the word embedding of the token at step $k-1$, and w_{j0} is the summed embedding of the domain and slot. $h^{dec}_{j(k-1)}$ is the hidden state at step $k-1$, and h^{dec}_{j0} is the output h_t of the encoder.

$$h^{dec}_{jk} = GRU(w_{j(k-1)}, h^{dec}_{j(k-1)}) \tag{2}$$

First, we obtain a probability distribution on the vocabulary P^{vocab}_{jk} by mapping the hidden state h^{dec}_{jk} into the vocabulary space with the vocabulary embedding $E \in \mathbb{R}^{|V| \times d_{hdd}}$, where $|V|$ is the size of the vocabulary. At the same time, a probability distribution on the dialog history $P^{history}_{jk}$ is obtained by calculating the attention score between the hidden state h^{dec}_{jk} and the encoded dialog history H_t:

$$\begin{aligned} P^{vocab}_{jk} &= Softmax(E \cdot (h^{dec}_{jk})\top) \in \mathbb{R}^{|V|} \\ P^{history}_{jk} &= Softmax(H_t \cdot (h^{dec}_{jk})\top) \in \mathbb{R}^{|X_t|} \end{aligned} \tag{3}$$

The final probability distribution P^{final}_{jk} is the weighted-sum of two distributions:

$$P^{final}_{jk} = P^{gen}_{jk} \times P^{vocab}_{jk} + (1 - P^{gen}_{jk}) \times P^{history}_{jk} \tag{4}$$

where P^{gen}_{jk} is a trainable parameter.

We use the context-enhanced slot gate to decide whether any of the candidate (domain, slot) pairs are mentioned. The context vector that is computed using the encoded hidden state H_t is mapped to a probability distribution over the "`pointer`", "`none`", and "`don't care`" categories. Essentially, the slot gate is a three-way classifier. For each (domain, slot) pair, if the gate predicts "`pointer`", we fill its value with the words generated from the state generator. Otherwise, if the predicted value is "`don't care`" or "`none`" , we take "`does not care`" or "`none`" as the value. The slot gate for the j-th (domain,slot) pair G_j is a linear layer parameterized by $W_g \in \mathbb{R}^{3 \times d_{hdd}}$:

$$G_j = Softmax(W_g \cdot (c_{j0})\top) \in \mathbb{R}^3 \tag{5}$$

where c_{j0} is the context vector calculated with the first probability distribution on the dialog history $P^{history}_{j0}$:

$$c_{j0} = P^{history}_{j0} \cdot H_t \in \mathbb{R}^{d_{hdd}} \tag{6}$$

We optimize both the state decoder and the slot gate. The loss of the state decoder $£_{decoder}$ is the cross-entropy loss between the predicted P^{final}_{jk} and the true words $Y_j = [w_{j0}, w_{j1}, ..., w_{jK}]$, where J is the number of the (domain, slot) pair and K is the character number of the value for the j-th (domain, slot) pair:

$$£_{decoder} = \sum_{j=1}^{J} \sum_{k=1}^{K} -log(p^{final}_{jk} \cdot (w_{jk})^\top) \tag{7}$$

Another cross-entropy loss $£_{gate}$ between the predicted gate G_j and the true label y^{gate}_j is used. It is defined as:

$$£_{gate} = \sum_{j=1}^{J} -log(G_j \cdot (y^{gate}_j)^\top) \tag{8}$$

$£$ is the weighted-sum of the losses using hyper-parameters α and β:

$$£ = \alpha £_{gate} + \beta £_{decoder} \tag{9}$$

3.2 Robust dialog state tracker with contextual-feature augmentation

Delexicalization features are considered the key means for efficient unseen slot value tracking. Our model essentially introduces contextual-feature augmentation to the backbone described above, extracting generic features similar to Delexicalization features. Specifically, inputs are processed using the same encoder of the backbone network with learnable parameters, yielding an encoded dialog history

H_t. Then, we employ a contextual-feature augmentation module, as proposed in Section 3.2.1 , to obtain the contextual features of each word C_t that is unrelated to itself in the dialog history. Subsequently, the contextual features are used to decode the value of the (domain, slot) pair with the decoder of the backbone network. Additionally, our model introduces a simple but effective training paradigm to address annotation errors. More details about this are given in Section 3.2.2 .

3.2.1 Contextual-feature augmentation

Recently, attention mechanisms and their variants have brought excellent progress to recent studies and have performed extremely well in many visual tracking tasks [9, 10] owing to their capability of modeling contextual information. Motivated by this, we propose contextual-feature augmentation based on a novel diagonalized masked self-attention (DMSA) mechanism to extract contextual features.

DMSA Token2token self-attention is used in the Transformer [33] to obtain context-aware vector representations at each position. However, a context-aware vector representation is related to the corresponding position word because this word is not excluded from the calculation. To obtain a contextual feature that is irrelevant to the word itself, we should exclude the word itself from the weighted sum, i.e., disable the attention of each token to itself. Based on the above analysis, we propose the DMSA mechanism.

Specifically, we define DMSA as fellows:

$$\mathrm{DMSA}(Q, K, V) = \mathrm{softmax}(E + M^{diag})V \tag{10}$$

where Q, K, and V are the inputs of DMSA. Q is a matrix of queries, K is a a matrix of keys, and V is a matrix of values. E is a similarity matrix between Q and K. The i-th row of Q is denoted as q_i, and the j-th row of K is denoted as k_j, indicating how well q_i and k_j match. The element E_{ij} of matrix E is the similarity score between q_i and k_j:

$$E_{ij} = q_i \cdot k_j^T \tag{11}$$

We propose a diagonalized masked matrix $M^{diag} = \{M_{ij}^{diag}\}$ and add it to the similarity score matrix E.

$$M_{ij}^{diag} = \begin{cases} 0 & i \neq j \\ -\infty & i = j \end{cases} \tag{12}$$

The $-\infty$ value at the diagonal position of M_{ij} leads to an attention score of 0 after the softmax function. This means that we disable the attention of each token to the token at the same position. Thus, the context-aware vector of each word is irrelevant to the word itself. Figure 2 depicts the computation graph of the DMSA mechanism.

Contextual-feature Augmentation We propose a contextual-feature augmentation method, which applies a masked transformer, replacing token2token self-attention with DMSA. The input of the contextual-feature augmentation is the output of the utterance encoder H_t.

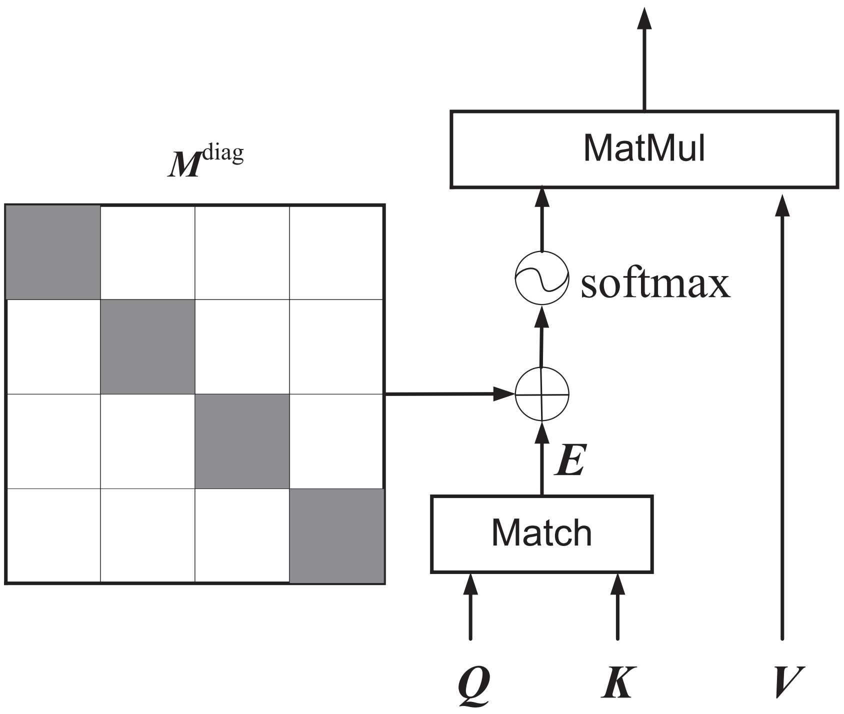

Fig. 2 The architecture of the proposed DMSA. We propose a diagonalized masked matrix M^{diag}, the value at the diagonal position of which is $-\infty$. Then, it is added to the similarity score matrix E. The attention score at the diagonal position is equal to 0 after the softmax function. That is, we disable the attention of each token at the same position. Thus, we exclude the token itself from weighted sum when calculating the context-aware vector representation of each word

First, we map the matrix H_t to n parallel head representations using linear projections with hyper-parameters W^Q, W^K and W^V. Specifically, for the i-th head:

$$Q_i, K_i, V_i = H_t W_i^Q, H_t W_i^K, H_t W_i^V \quad where \quad i \in [1, n] \tag{13}$$

Then, these representations are sent to the DMSA module to obtain the contextual features at the i-th head:

$$G_i = \mathrm{DMSA}(Q_i, K_i, V_i) \quad where \quad i \in [1, n] \tag{14}$$

Multi-head self-attention learns different context representations in multiple semantic spaces:

$$\mathrm{MultiHeadAttention}(H_t) = \mathrm{concat}(G_1 ... G_n) W^O \tag{15}$$

where W^O is a hyper-parameter. Between the input and output of the encoder, residual connections are used. Then, the model applies layer normalization after the residual connection to stabilize the activations of the deep neural network:

$$I = \mathrm{LayerNorm}(H_t + \mathrm{MultiHeadAttention}(H_t)) \tag{16}$$

The output is sent to a feed-forward network parameterized by W_1 and W_2:

$$\mathrm{FFN}(I) = \mathrm{ReLU}(I W_1) W_2 \tag{17}$$

Then the residual connection and layer normalization are exploited to get the final output:

$$C_t = \mathrm{LayerNorm}(\mathrm{FFN}(I) + I) \tag{18}$$

We present the general architecture in Fig. 3. The output $C_t \in \mathbb{R}^{|X_t| \times d_{hdd}}$ is the contextual representation.

With contextual-feature augmentation, we obtain the contextual representations of each token. Consider the utterance ‘`West of Chinese street serving French food`’ as an example. We calculate the contextual features for ‘`Chinese`’ using its context ‘`West of` $\langle value \rangle$ `street serving French food`’ , and the contextual features for ‘`French`’ using its context ‘`West of Chinese street serving` $\langle value \rangle$ `food`’.

The contextual representations that are irrelevant to the word itself, such as the **delexicalization** feature, are replaced or ignored. Previous works extract the delexicalization features depending on predefined semantic dictionaries. In the proposed method, these features can be found automatically in the training process.

In the decoding process, C_t participates in decoding instead of H_t:

$$P_{jk}^{history} = Softmax(C_t \cdot (h_{jk}^{dec})\top) \in \mathbb{R}^{|X_t|} \tag{19}$$

For an unseen slot value that is not involved in the training, such as “`Beijing`” in the utterance “`West of Chinese street serving Beijing food`”, becau-se “`Beijing`” has a similar context as “`French`”, the contextual-features of “`Beijing`” are similar to those of “`French`”. Thus, the history attention score for the token “`Beijing`” is high at the decoder step so that the unseen slot-value can be tracked correctly.

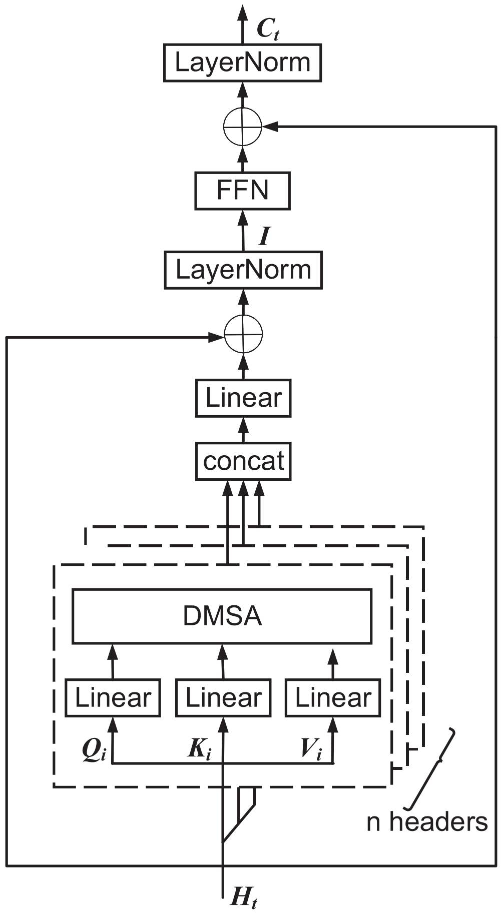

Fig. 3 Contextual feature augmentation

3.2.2 Robust training paradigm

We propose extending the recently introduced co-teaching algorithm [13] for DST to address annotation errors. The main idea is to train two deep neural networks at the same time. As shown in Fig. 4, each DST network chooses small-loss training examples in the batch data as useful examples, and teaches such useful examples to its peer neural network for further training. Therefore, we can train a more robust DST deep learning network than previous methods. Suppose the error comes from the biased choice of the training examples in the first mini-data batch.

In common deep neural networks, the error from the neural network will be passed straight back to itself in the second mini-data batch, and the error flow accumulates progressively. However, in our training paradigm, two deep networks are trained simultaneously, and they have different learning abilities. They can select different kinds of errors

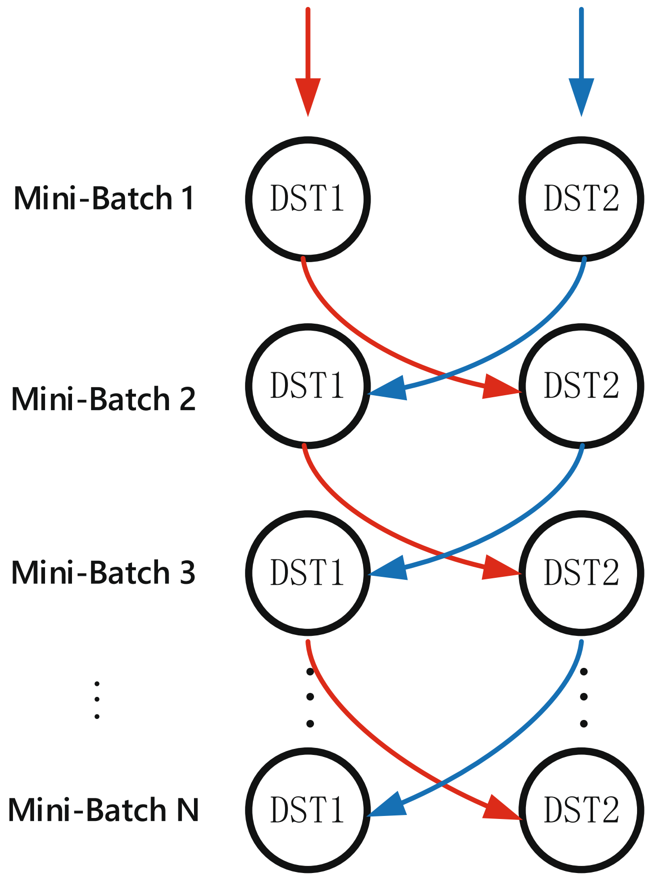

Fig. 4 Robust training paradigm with annotation errors for DST

produced by noisy data. In such an exchange process, the error flow can be mutually reduced by the peer-to-peer network. In addition, nonlinear deep networks using stochastic optimization with momentum can memorize clean data first to become robust [2]. Thus, the peer-to-peer network can resist this error from noisy data owing to its robustness when an error flows into it.

One major challenge when applying co-teaching to DST is that the original approach assumes that the loss function is shared across data filtering and model optimization. This, however, is not reasonable in general because the optimization loss of many tasks is composed of multiple losses, each of which has a different meaning and is not suitable for sample data. For example, the optimized loss of the DST system introduced above is the weighted-sum of the loss function of the slot gate and the loss function of the state decoder. However, the noisy labels come mainly from the state decoder. We address this issue by distinguishing the update loss function $\pounds_u$ from the sample loss function $\pounds_s$. Here, $\pounds_u$ is the optimized loss and $\pounds_s$ is the loss function of the state decoder:

$$\begin{aligned}\pounds_u &= \pounds = \alpha\pounds_{gate} + \beta\pounds_{decoder}\\ \pounds_s &= \pounds_{decoder}\end{aligned} \tag{20}$$

Furthermore, annealing the learning rate, by either using cosine functions or step functions [14], has been proven to be crucial for learning models with higher generalization power. Cosine annealing scheduling has been especially effective in producing state-of-the-art results while removing the need for hyper-parameter searching on the learning rate space[1]. Thus, we propose applying the cosine annealing scheduling in the optimizer. Algorithm 1 below shows the robust training algorithm for DST.

We maintain two neural networks g (with parameter w_g) and f (with parameter w_f). For each epoch T in [0, T_{max}], we first shuffle the training data (step 1), where T_{max} is the maximum epoch number. When a mini-batch $\bar{D}$ is obtained (step 4), we let g choose $R(T)$ percent of the samples in mini-batch $\bar{D}_g$ that have a small training loss (step 5). The process for network f is the same as for g (step 6). Then, the selected samples are fed into the peer neural network as useful knowledge for the parameter update (steps 7 and 8). We repeat this process (steps 4-8) N_{max} times. Then, we update the learning rate η_T and $R(T)$ (step 9), where the hyper-parameters τ and T_k are as specified in Section 4.3.

Algorithm 1 Robust training paradigm with annotation errors.

```
1:  input $w_f, w_g$,parameter $\tau$,epochs $T_k$ and $T_{max}$,
    iterations $N_{max}$
2:  for T in [0,1, ..., $T_{max}$] do
    – 3: shuffle training data $D$
    – for N in [0,1, ..., $N_{max}$] do:
          · 4: get mini-batch $\bar{D}$ from $D$
          · 5: sample data $\bar{D}_g = \underset{D'}{argmin}\pounds_s(w_g, D')$
            //sample $R_T\%$ small-loss data
          · 6: sample data $\bar{D}_f = \underset{D'}{argmin}\pounds_s(w_f, D')$
            /sample $R_T\%$ small-loss data
          · 7: update $w_g = w_g - \eta_T \nabla \pounds_u(w_g, \bar{D}_f)$
            //update $w_g$ with $\bar{D}_f$
          · 8: update $w_f = w_f - \eta_T \nabla \pounds_u(w_f, \bar{D}_g)$
            //update $w_f$ with $\bar{D}_g$
    – end iteration
    – 9:update $\eta_T = 5e^{-4} + 5e^{-4}cos(\pi \frac{T}{T_{max}})$, $R_T =$
      $1 - min\left\{\frac{T}{T_k}\tau, \tau\right\}$
    end epoch
10: output $w_g, w_f$
```

Another important question for designing Algorithm 1 above is that of why sampling small-loss examples according to a dynamic R_T helps us to obtain clean data. The reason is that it helps us to understand the connection between small loss and clean data. Intuitively, small-loss examples have more potential to be the ones with the correct labels. Therefore, our DST network should be more robust and resistant to noisy labels if we train it only using small-loss examples in each mini-data batch. However, the above analysis requires the network to be trustworthy enough that the small-loss examples are truly clean.

Fortunately, we can address this problem exactly by recent studies on the memorization mode of deep neural networks. That is, deep neural networks first memorize clean data and then noisy data when training, even in the presence of noisy data [2, 43]. Thus, they can separate out noisy examples utilizing the loss values at the beginning of training. However, they will overfit noisy data eventually as the number of epochs grows. To address this issue, it is wise to retain more examples in the mini-batch in the initial epochs; that is, R_T is large. Then, we gradually decrease R_T so that our networks can preserve the clean examples and abandon noisy examples before they are memorized.

3.3 Unseen-domain DST

As described in Section 1, unknown-domain DST is trained a dataset $(X, D_{source}, S_{source}, Y_{source})$ and tested on $(X, D_{target}, S_{target}, Y_{target})$, where X is the dialog context; D_{source}, S_{source}, and Y_{source} are the domain, slot, and slot value, respectively, in the source domain; and D_{target}, S_{target}, and Y_{target} are the domain, slot, and slot value respectively, in the target domain. Although D_{source} and D_{target} are different, they may share some of the same slots S. For the slots that are already learned in the source domain, our DST model can directly track them when they are present in the target domain. Contextual-feature augmentation can utilize the contextual features of each token to decode the slot value. For the same slot across D_{source} and D_{target}, even if some slot values in Y_{target} are unseen in Y_{source}, they still can be effectively tracked, as they have a similar context as the slot values in Y_{source}. For example, if our model can track the "`book people`" slot in the "`hotel`" domain, then this tracking can transfer to the "`train`" domain, which shares a similar context. It is challenging to track slots in S_{target} that do not appear in S_{source}, since the model has never been trained to track such slots.

4 Experimental setting

4.1 Data

We use the recently-released MultiWOZ-2.0 dataset [4] and MultiWOZ 2.1 [7], which are composed of multi-domain dialogs. These dialogs come from seven different domains (attraction, hotel, restaurant, train, taxi, police, and hospital) and consist of 37 slots across different domains. MultiWOZ 2.1 is a refined version of MultiWOZ 2.0 in which annotation errors are corrected. Given that two domains (hospital and police) only appear in the training set and have very few dialogs, we conduct experiments on only the other five domains. The DST labels from the training, development and testing datasets are used in our experiments. The datasets we used consist of a total of 30 (domain, slot) pairs and 10438 task-oriented dialogs. Table 1 shows the statistics of the datasets.

4.2 Metrics

We focus on two evaluation metrics described in [22, 39]:

1. Goals ("joint goal accuracy"): the proportion of dialog turns where the dialog state is correctly identified. At each dialog turn t, we compare the predicted dialog state to the label B_t from the original dataset, and the predicted results are considered correct only if all the predicted values exactly match the values in label B_t.
2. Slots ("slot accuracy"): In contrast, we compare each (domain, slot, value) triplet to its truth label.

4.3 Implementation details

In our experiments, we set the number of headers n in multi-head self-attention to 12. The rate of dropout is set to 0.2 at the output of each module to prevent overfitting. We perform mini-batch training with a batch size of 32 using the Adam optimizer (momentum=0.9), and we run 100 epochs. The learning rate annealing is in the range of [0, 0.001]. The model is trained on two NVIDIA Tesla P100 GPUs in parallel. Both α and β in (9) are set to one. We set $R_T = 1 - min\left\{\frac{T}{T_k}\tau, \tau\right\}$ with T_k= 10 and $\tau = 0.35$.

We initialize the word embeddings by concatenating GloVe embeddings [25] and character embeddings [3], where the dimension of each word is 400. Our model employs a greedy search strategy for our state decoder since the slot values are often short in length.

4.4 Models

We compare our results with those of five published baselines and briefly describe these baseline models below:

- **GLAD** [45]: This model proposes a global-local architecture. The global module learns transfer knowledge by sharing the parameters across all the slots, and the local module learns slot-specific features. It computes the semantic similarity between utterances and each predefined slot value. A slot value is selected when the score is higher than a threshold.

Table 1 Statistics of the datasets. The last three rows show the number of dialogs related to each domain

Domain	Hotel	Train	Restaurant	Attraction	Taxi	All Domains
Slots	parking price range book stay internet type book people book day stars name area	book people destination departure day leave at arrive by	book people area price range name book time food book day	name type area	destination arrive by leave at departure	30 slots
Train	3381	3103	3813	2717	1654	14668
Dev	416	484	438	401	207	1946
Test	394	494	437	395	195	1915

- **HYST** [12]: This is a hybrid approach based on hierarchical RNNs and open-vocabulary candidate-generation.
- **TRADE** [39]: This is the current state-of-the-art model based on generation. It bases the DST model on an encoder-decoder architecture with attention-based copying. The source sentences are the dialog history, and the target sentences are the dialog state labels.
- **NADST**[18]: This uses a transformer-based non-autoregressive decoder to decode the dialog state of the current turn.
- **DST-QA**[46]: This method formulates the DST task as a question answering problem and proposes a question answering model for DST. Additionally, it is noted that the (domain, slot) pairs are related, and this method first introduces a knowledge graph to learn the relationships between (domain, slot) pairs.
- **SST**[6]: This method proposes a schema-guided multi-domain dialog state tracker with graph attention networks (SST) that predicts dialog states from dialog utterances and schema graphs that contain slot relations in the edges. We also introduce a graph attention-matching network to fuse information from utterances and graphs and a recurrent graph attention network to control state updating.

Table 2 Joint goal and slot accuracy on the MutiWOZ 2.0 test sets

Model	Goals(%)	Slots(%)
GLAD	35.57	95.44
HYST	38.10	95.63
TRADE	48.62	96.92
DS-DST	51.21	-
DST-QA	51.44	97.24
SST	51.17	-
Our Model	**53.68***	**97.58***
-DMSA	51.88†	97.25†
-robust training	50.98*	96.89†

An asterisk indicates statistically significant improvement over the baseline trackers (paired t-test; $p < 0.01$). A dagger indicates statistically significant improvement over the baseline trackers (paired t-test; $p < 0.05$)

5 Results and analysis

5.1 Results on belief tracking

The proposed model is used in 5 independent runs. The average results are reported. Our model outperforms all the models described in the previous section, indicating that the proposed DST architecture can effectively track the state of the dialog.

As seen in Table 2, our model achieves the highest performance on MutiWOZ 2.0, 53.68% on joint goal accuracy and 97.58% on slot accuracy, a 4.35% relative improvement and a 2.24% absolute improvement over "`DST-QA`" on the joint goal accuracy. Regarding the goals, the gains are always statistically significant (paired t-test, $p < 0.01$).

Table 3 shows the result on the MutiWOZ 2.1 dataset. Compared with TRADE, our model has a 13.82% relative improvement and a 6.27% absolute improvement on the joint goal accuracy. Our performance gain can be attributed to the model capability of learning contextual features that are irrelevant to specific slot values, directly optimizing the evaluation metric of cross-domain joint goal accuracy.

Table 3 Joint goal and slot accuracy on the MutiWOZ 2.1

Model	Goals(%)	Slots(%)
TRADE	45.35	96.55
Our Model	**51.62***	**97.33**†
-DMSA	46.86*	97.30†
-robust training	49.74†	97.32†

The meaning of the asterisk and the dagger are the same as that in Table 2

Table 4 Joint goal accuracy of the unseen domain state tracking

Target domain	Single training		Zero-shot	
	TRADE	Our model	TRADE	Our model
Attraction	71.64	72.51	19.87	20.54
Taxi	76.13	76.15	60.58	64.31
Hotel	55.52	54.98	13.70	14.35
Restaurant	63.35	63.21	11.52	14.40
Train	77.71	77.85	22.37	25.40

5.2 Results on unseen domain tracking

The model performance on unseen domains is shown in Table 4. In our experiments, we choose four domains in the MultiWOZ 2.0 dataset as the source domains and the remaining domain as the target domain. In the "`Zero-Shot`" experiments, models are first trained on the source domains and then tested directly on the target domain. In the "`Single Training`" experiments, the models are trained and tested on the target domain. We compare our model with TRADE in this experiment.

As shown in Table 4, in the "`Single Training`" scenario, our model performs similarly to TRADE, while in the "`Zero-Shot`" scenario, it consistently outperforms TRADE. For example, the "`train`" domain achieves a 3.03% absolute improvement over TRADE on the "`Zero-Shot`" performance, which is much higher than the 0.14% improvement on the "`Single Training`" performance. This is attributed to contextual-feature augmentation, which extracts generalized features that are irrelevant to the slot values. In this way, based on the slots already learned in the source domains, a DST model can directly track the slots that are present in a target domain, even if the slot values in the target domain are different from those in the source domains. The performance on the "`Zero-Shot`" experiments is not especially promising, as it is extremely challenging for the model to track the slots in target domains that do not appear in the source domains.

5.3 Ablation study

Tables 2 and 3 also show the results of the ablation study of our model on the MultiWOZ 2.0 and MultiWOZ 2.1 datasets.

DMSA We experimented with a variant of our model using token2token self-attention instead of DMSA. As shown in Tables 2 and 3, the joint accuracy of the variant without DMSA drops by 1.8% on the MultiWOZ 2.0 dataset and 4.7% on the MultiWOZ 2.1 dataset, demonstrating that contextual-feature augmentation is important for tracking mixed-domain dialog states.

To further illustrate, we take the user utterance "`I would like some Southeast Asian food that is not too expensive`" and the candidate (r-estaurant-food) pair as an example. Figure 5 shows the history attention score between the hidden state h_{jk}^{dec} at decoding step k and the encoded dialog history. For example, k in Fig. 5 is set to 0,1 and 2 (the 0th, 1st and 2nd output tokens of the slot value that we decode step by step). The darker the colour is, the higher the attention score. The figure on the top is the score with the DMSA module, that is, the attention score in (19), and the figure on the bottom is the score without DSMA, that is, the attention score in (3).

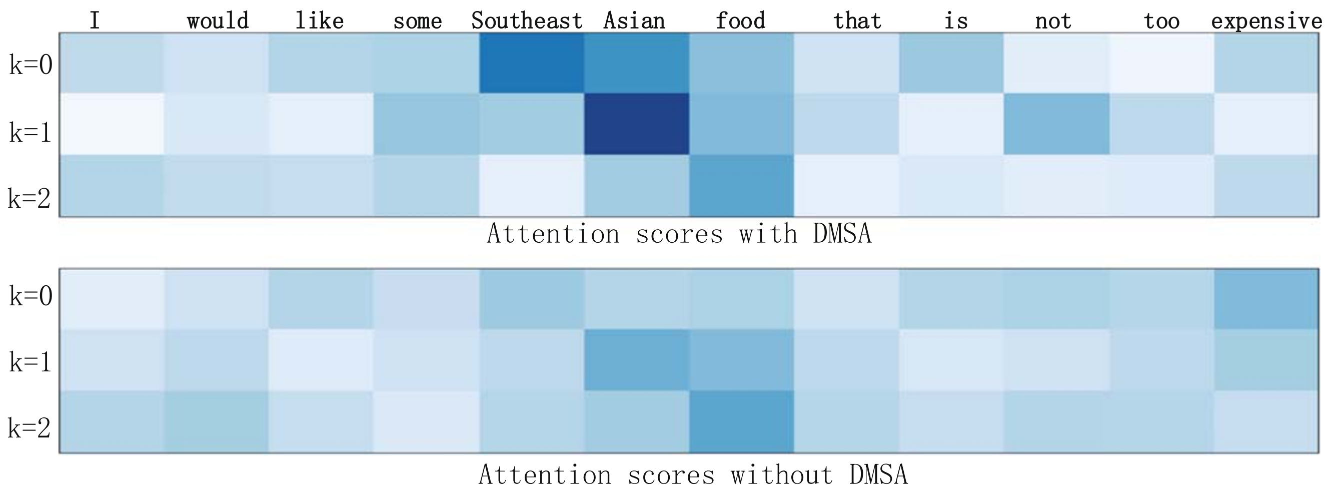

Fig. 5 The historic attention score between the hidden state h_{jk}^{dec} at decoding step k and the encoded dialog history with DMSA (on the top) and without DMSA (on the bottom). The user utterance is "`I would like some Southeast Asian food that is not too expensive`" and the domain slot is a (restaurant-food) pair. The darker the colour is, the higher the attention score. Taking the top figure as an example, at the first decoding step (k=0), the score is high in the token "`Southeast`", and at the second decoding step (k=1), the score is high in the token "`Asian`". Step by step, we can obtain the value "`Southeast Asian food`"

Note that the "`food`" value "`Southeast Asian`" does not appear in the training set; i.e., it is a new slot value. The high score on "`Southeast Asian`" in the top figure shows that our model with DMSA does focus on the keywords relevant to the new slot value pair, even if the word is not involved in training. That is, the contextual features computed based on DMSA capture generic features such as delexicalization features that are unrelated to the word itself.

Robust training This experiment keeps the DMSA component but removes the robust training paradigm, which means we ignore the impact of noisy labels on the performance. The 2.7% drop on the MultiWOZ 2.0 dataset and the 1.8% drop on the MultiWOZ 2.1 dataset demonstrate that noisy labels inevitably degenerate the robustness of the learned models. Our proposed model with a robust training paradigm can train deep models robustly with noisy supervision. Moreover, this method is simple but effective, requiring no extra resources or more complex networks.

5.4 Error analysis

We find that there are different types of prediction errors through error analysis, which is conducive to improving our model. The model prediction errors are divided into 3 categories, as shown in Table 5.

– **Unmentioned slot error**: This can happen when either (1) our prediction is "`None`", while the label is not "`None`"; or (2) our prediction is not "`None`", while the ground truth is "`None`". This kind of prediction error

Table 5 The different types of model prediction errors on the MultiWOZ 2.0 dataset

Category	Predict	Annotation	Context	%
Unmentioned slot error	None	attraction-name-school	I want to visit some architecture; [A] ... there are several magnificent churches. the old schools are quite impressive as well ... [U] Is there a phone number I can get for **the schools** please; [A]the phone number for old schools is ...	14.2
	Restaurant-name-Prezzo	None	...[A]there are several architectural attractions in the centre of town , such as all saints church , holy trinity church , and some others . would you like to know more about them ? [U] can i please have the postcode for the holy trinity church	14.9
Annotation error	restaurant-food-african; restaurant-area centre	None	[U] I would like some south african food that is not too expensive. [A] unfortunately there are no restaurant -s available that fit that criteria ... [U] are there any **african** restaurant -s outside of the city **center**?	22.7
	Hotel-type-guesthouse	Hotel-type-hotel	...[A] i was able to do that and your reservation number is 6kq6nmiq . [U] thanks i am also needing to book at a **guesthouse** for 6 for 5 nights starting on sunday	12.6
Wrong slot value	hotel-internet-UNK	hotel-parking-yes	[U] I am looking for a place in cambridge. **it does not need to include internet** and should be a hotel [A] I was not able to find a hotel type with no internet , **but there are hotel -s with internet**; [U] **okay, that would be fine**, as long as it has free parking...	12.9
	Train-departure-kings lynn	Train-departure-cambridge	...[A]where would you like to go to? [U] i will be going from **canbrige** arriving in **kingls lynn** ...	19.1
	Attraction-name-riverboat	Attraction-name-riverboat georgina	...[A] cotto is located on 183 east road city centre and has the postcode cb11bg. is there anything else i could help you with today? [U] i am also looking for an attraction called **riverboat georgina**...	3.7

is caused by incorrect predictions made by the slot gate model.

- **Annotation error**: This occurs when the ground-truth label in MultiWOZ 2.0 is incorrect. This category contains two types of errors: 1) the annotators fail to obtain the slots mentioned in the dialog history, and 2) the annotators choose the incorrect value of a (domain, slot) pair.
- **Incorrect slot value**: This occurs when the slot value is not recognized by the model. This category contains the following three types of errors: 1) Our model misinterprets the user's intent and fails to predict the slot value based on the dialog history. 2) There are several slot values in the dialog context, and the model chooses the wrong one. 3) Our network refers to a span that is either a subset or a superset of the correct span.

Table 5 shows the error examples and the percentage of each type. An "`unmen- tioned slot error`" accounts for 29.1% of the total errors, which reveals that more attention should be paid to the slot gate module, especially for complex and negative user utterances. In addition, 32% of errors are due to an "`annotation error`". All these errors are caused by the noise data in the MultiWOZ 2.0 dataset. Unfortunately noise is unavoidable, explaining the necessity of combating the noisy labels. This finding also illustrates the necessity of introducing a robust training paradigm and is in alignment with our ablation studies in Table 2, where robust training can boost performance in terms of the joint goal accuracy. An "`wrong slot value`" accounts for 35.7% errors, indicating misunderstanding of the dialog history. It may be helpful to introduce a stronger language module.

6 Conclusion

We introduce a robust dialog state tracker with contextual-feature augmentation, which is a new state-of-the-art model for dialog state tracking. We propose a DMSA mechanism to perform contextual-feature augmentation, extracting the generic contextual features. The contextual features are similar to the delexicalization features. Thus, the state decoder can capture the keywords relevant to the unseen slot values, as they have similar contextual features to the existing values. To combat the noisy labels in the dataset, we introduce a simple but effective training method for our DST model. The experimental results demonstrate that our approach can track the dialog state effectively in an unseen domain and that it significantly outperforms state-of-the-art methods.

In future work, we plan to further improve unseen domain performance, focusing on the new slots that are not seen in the training set. A more powerful language model and a coreference resolution module may be introduced to improve the ability of the model to understand complex utterances.

Appendix

A. Joint accuracy and context length

Figure 6 shows the model performance with different context lengths. The context length refers to the number of pre-

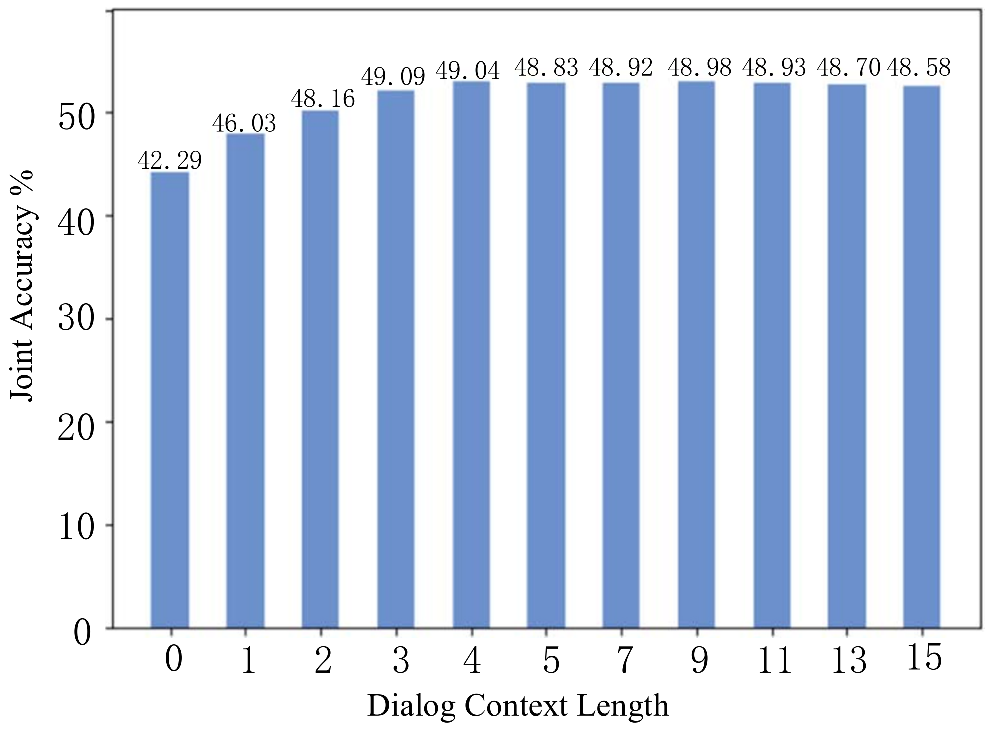

Fig. 6 Joint accuracy. v.s. context length

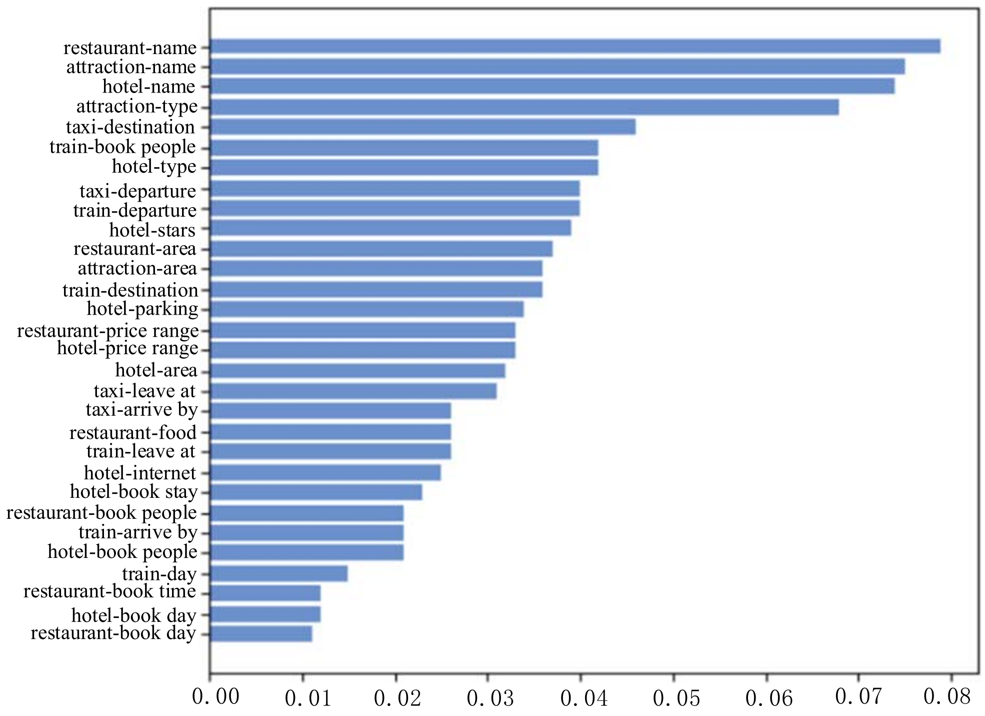

Fig. 7 Error rate of each slot per turn on the MultiWOZ 2.0 dataset

vious turns involved in the dialog history. Our baseline algorithms above utilize all previous turns as the dialog history.

B. Error rate per slot

Figure 7 shows the error rate of each slot. We find that the error rates of name related slots such as "`attraction name`", "`restaurant name`", and "`hotel name`" are high. This may be due to the very large value set of those slots.

Unseen domain error analysis

In Fig. 8, we show the unseen domain analysis of two selected domains, "`hotel`" and "`train`". For the slots that appear in the other four domains, our model can track correctly. For example, the "`area`", "`price range`", "`people`", and "`day`" slots also appeared in the "`restaurant`" domain. However, the "`parking`", "`stars`" and "`internet`" slots, which only appear in the "`hotel`" domain, are difficult for our model to track.

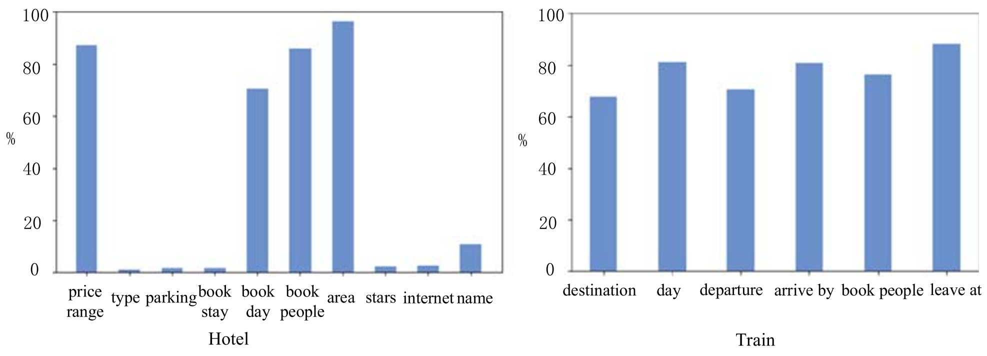

Fig. 8 Error analysis of unseen domains

References

1. Antoniou A, Edwards H, Storkey A (2018) How to train your maml. In: International Conference on Learning Representations (ICLR), pp 770–774
2. Arpit D, Jastrzbski S, Ballas N, Krueger D, Bengio E, Kanwal MS, Maharaj T, Fischer A, Courville A, Bengio Y, et al. (2017) A closer look at memorization in deep networks. In: Proceedings of the 34th International Conference on Machine Learning (ICML), pp 233–242
3. Bowman SR, Vilnis L, Vinyals O, Dai AM, Józefowicz R, Bengio S (2016) Generating sentences from a continuous space. In: CoNLL, pp 10–21
4. Budzianowski P, Wen TH, Tseng BH, 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: Proceedings of the 2018 conference on empirical methods in natural language processing (EMNLP), pp 5016–5026
5. Chao GL, Lane I (2019) Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In: INTERSPEECH, pp 1468–1472
6. 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, pp 7521–7528
7. Eric M, Goel R, Paul S, Kumar A, Sethi A, Goyal AK, Ku P, Agarwal S, Gao S (2020) Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In: LREC
8. Gao J, Galley M, Li L et al (2019) Neural approaches to conversational ai. Foundations and Trends®, in Information Retrieval 13(2-3):127–298
9. Gao P, Yuan R, Wang F, Xiao L, Fujita H, Zhang Y (2020) Siamese attentional keypoint network for high performance visual tracking. Knowl.-Based Syst 193:105448
10. Gao P, Zhang Q, Wang F, Xiao L, Fujita H, Zhang Y (2020) Learning reinforced attentional representation for end-to-end visual tracking. Inform Sci 517:52–67
11. Gao S, Sethi A, Agarwal S, Chung T, Hakkani-Tur D (2019) Dialog state tracking: A neural reading comprehension approach. In: Proceedings of the 20th annual meeting of the special interest group on discourse and dialogue (SIGDIAL), pp 264–273
12. Goel R, Paul S, Hakkani-Tür DZ (2019) Hyst: A hybrid approach for flexible and accurate dialogue state tracking. In: INTERSPEECH, pp 1458–1462
13. Han B, Yao Q, Yu X, Niu G, Xu M, Hu W, Tsang I, Sugiyama M (2018) Co-teaching: Robust training of deep neural networks with extremely noisy labels. In: Advances in neural information processing systems, pp 8527–8537
14. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
15. Henderson M, Gašić M, Thomson B, Tsiakoulis P, Yu K, Young S (2012) Discriminative spoken language understanding using word confusion networks. In: 2012 IEEE Spoken Language Technology Workshop (SLT). IEEE, pp 176–181
16. Henderson M, Thomson B, Young S (2014) Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In: 2014 IEEE Spoken Language Technology Workshop (SLT). IEEE, pp 360–365
17. Henderson M, Thomson B, Young S (2014) Word-based dialog state tracking with recurrent neural networks. In: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pp 292–299
18. Le H, Socher R, Hoi SC (2019) Non-autoregressive dialog state tracking. In: International Conference on Learning Representations (ICLR), pp 146–150
19. Lei W, Jin X, Kan MY, Ren Z, He X, Yin D (2018) Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In: ACL, pp 1437–1447
20. Li Y, Yang J, Song Y, Cao L, Luo J, Li LJ (2017) Learning from noisy labels with distillation. In: Proceedings of the IEEE international conference on computer vision, pp 1910–1918
21. Ma X, Wang Y, Houle ME, Zhou S, Erfani S, Xia S, Wijewickrema S, Bailey J (2018) Dimensionality-driven learning with noisy labels. In: International Conference on Machine Learning (ICML), pp 3355–3364
22. Mrkšić N, Ó Séaghdha D, Wen TH, Thomson B, Young S (2017) Neural belief tracker: Data-driven dialogue state tracking. In: ACL, pp 1777–1788
23. Mrkšić N, Séaghdha DO, Thomson B, Gašić M, Su PH, Vandyke D, Wen TH, Young S (2015) Multi-domain dialog state tracking using recurrent neural networks. In: ACL, pp 794–799
24. Nouri E, Hosseini-Asl E (2018) Toward scalable neural dialogue state tracking model. In: Advances in neural information processing systems (NeurIPS), 2nd Conversational AI workshop
25. Pennington J, Socher R, Manning C (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543
26. Perez J, Liu F (2017) Dialog state tracking, a machine reading approach using memory network. In: the European Chapter of the Association for Computational Linguistics (EACL), pp 305–314
27. Ramadan O, Budzianowski P, Gašić M. (2018) Large-scale multi-domain belief tracking with knowledge sharing. In: ACL, pp 432–437
28. Ren L, Xie K, Chen L, Yu K (2018) Towards universal dialogue state tracking. In: ACL, pp 2780–2786
29. Ren M, Zeng W, Yang B, Urtasun R (2018) Learning to reweight examples for robust deep learning. In: International Conference on Machine Learning (ICML), pp 4334–4343
30. Rodrigues F, Pereira FC (2018) Deep learning from crowds. In: AAAI, pp 552–560
31. Tanaka D, Ikami D, Yamasaki T, Aizawa K (2018) Joint optimization framework for learning with noisy labels. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 5552–5560
32. Thomson B, Young S (2010) Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. Computer Speech & Language 24(4):562–588
33. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: Advances in Neural Information Processing Systems (NIPS), pp 5998–6008
34. Veit A, Alldrin N, Chechik G, Krasin I, Gupta A, Belongie S (2017) Learning from noisy large-scale datasets with minimal supervision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 839–847
35. Wang Y, Liu W, Ma X, Bailey J, Zha H, Song L, Xia ST (2018) Iterative learning with open-set noisy labels. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8688–8696
36. Wang Z, Lemon O (2013) A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In: Proceedings of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pp 423–432

37. Wen TH, Vandyke D, Mrkšić N, Gasic M, Rojas Barahona LM, Su PH, Ultes S, Young S (2017) A network-based end-to-end trainable task-oriented dialogue system. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp 438–449
38. Williams JD (2014) Web-style ranking and slu combination for dialog state tracking. In: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pp 282–291
39. Wu CS, Madotto A, Hosseini-Asl E, Xiong C, Socher R, Fung P (2019) Transferable multi-domain state generator for task-oriented dialogue systems. In: ACL, pp 808–819
40. Xu M, Wang Y, Chi Y, Hua X (2020) Training liver vessel segmentation deep neural networks on noisy labels from contrast ct imaging. In: 2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI). IEEE, pp 1552–1555
41. Xu P, Hu Q (2018) An end-to-end approach for handling unknown slot values in dialogue state tracking. In: ACL, pp 1448–1457
42. Young S, Gašić M, Thomson B, Williams JD (2013) Pomdp-based statistical spoken dialog systems: A review. Proceedings of the IEEE 101(5):1160–1179
43. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O (2017) Understanding deep learning requires rethinking generalization. International Conference on Learning Representations (ICLR), pp 203–207
44. Zhang JG, Hashimoto K, Wu CS, Wan Y, Yu PS, Socher R, Xiong C (2019) Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In: Advances in neural information processing systems (NeurIPS), Conversational AI workshop
45. Zhong V, Xiong C, Socher R (2018) Global-locally self-attentive encoder for dialogue state tracking. In: ACL, pp 1458–1467
46. Zhou L, Small K (2019) Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. In: Advances in neural information processing systems (NeurIPS), Conversational AI workshop
47. Zilka L, Jurcicek F (2015) Incremental lstm-based dialog state tracker. In: 2015 Ieee Workshop on Automatic Speech Recognition and Understanding (ASRU). IEEE, pp 757–762

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Xuejun Zhang received the B.S. degree in electronic communication engineering from North China Institute of Science and Technology, Hebei, China, in 2012 and the M.S. degree in communication engineering from Beijing Institute of Technology, Beijing, China, in 2015. She is currently pursuing the Ph.D. degree in natural language processing from Institute of Acoustics, Chinese Academy of Sciences (CAS) in 2012, Beijing, China. From 2015 to 2018, he was a Research Assistant with the Institute of Acoustics, Chinese Academy of Sciences (CAS), Beijing, China. She used to study cross-lingual semantic representation and conversational text classification. Her current research interest lies in spoken language understanding, dialog state tracking, and dialogue decision.

Xuemin Zhao received the B.S. degree in Communication Engineering from Nankai University, Tianjin, China, in 2007 and the Ph.D. in speech signal processing from Institute of Acoustics, Chinese Academy of Sciences (CAS) in 2012, Beijing, China. From 2012 to 2014, he was a Research Assistant with the Institute of Acoustics, Chinese Academy of Sciences and became an associate researcher since 2014. His research interest includes natural language processing and digital audio watermark techniques. He is in charge of the development of spoken language understanding engines in intelligent agent projects.

Tian Tan received his B.S. degree in telecommunications engineering from Beijing University of Posts and Telecommunications in 2012, and his M.S. degree at Stanford University in 2015. He is currently pursuing a PhD degree in the Civil and Environmental Engineering department with a PhD minor in Computer Science at Stanford University. His current research interests broadly include topics in machine learning and algorithms, such as reinforcement learning, deep learning, deep generative models and statistical learning theory.