Neural Processing Letters (2019) 50:1611–1625
https://doi.org/10.1007/s11063-018-9950-1

Hierarchical Dialog State Tracking with Unknown Slot Values

Guohua Yang[1] · **Xiaojie Wang**[1] · **Caixia Yuan**[1]

Published online: 1 November 2018

Abstract
Dialog state tracking (DST) is the key component of goal-driven Spoken Dialog Systems. Almost all existing dialog state trackers are unable to handle unknown slot values. The continuous emergence of unknown slot values in dialogs is inevitable. If unknown slot values cannot be accurately detected and distinguished, the dialog states cannot be correctly updated in real time. This paper proposes a hierarchical dialog state tracking framework to model the dialog state tracking with unknown slot values. Three levels are included in the framework. Unknown slot values are identified at the first-level by a two-layer cascaded neural network as well as known slot values. Distributions for unknown and known slot values are updated separately in the second level and integrated in the third level. Experimental results on DSTC and WOZ2.0 datasets show that the proposed framework achieves good performance. Especially, the detection and distinction of unknown slot values greatly improve the final performance of dialog state tracking, illustrating the effectiveness of our proposed framework for addressing the problem of unknown slot values.

Keywords Dialog state tracking · Unknown slot values · Negative examples · Joint model

1 Introduction

Goal-driven Spoken Dialog Systems (SDSs) provide a natural language interface for human machine interaction to help users achieve their goals, such as finding restaurants or booking flights. SDSs have been widely applied in daily lives and industry for various purposes, and have attracted a large number of researchers into the study of it.

Dialog state tracking (DST) is the key component of goal-driven SDS. It is responsible for maintaining and updating dialog state at each time step as the dialog progresses, and lays the basis for the system to decide how to respond to the users [1]. A predefined set of slots needs

✉ Guohua Yang
yangguohua@bupt.edu.cn

Xiaojie Wang
xjwang@bupt.edu.cn

Caixia Yuan
yuancx@bupt.edu.cn

1 Center for Intelligence of Science and Technology (CIST), Beijing University of Posts and Telecommunications, Beijing 100000, China

to be filled as the goal-driven conversation progresses. The dialog states are distributions of all possible values of these slots.

The dialog state tracking Challenge (DSTC) [2,3] provides a common testbed and evaluation standards for DST. It strongly promotes the research of DST. So far, six sessions of DSTC have been successfully held, and each has its own features and tasks. The first DSTC uses human–computer dialogs in the bus timetable domain without changing the user's goals and is intended to explore different types of mis-match between the training and test data. DSTC2 releases a large number of dialogs related to restaurant searching and introduces the problem of changing user goals. DSTC3 is released to address the problem of adaptation to a new domain, involving the task of recognizing unknown slot values, at the same time distinguishing which specific unknown slot values is expressed in the user utterance. The fourth challenge focuses on human–human dialogs, which are much more unstructured and noisy than human–machine dialogs. The fifth challenge introduces a cross-language dialog state tracking task to address the problem of adaptation to a new language. The sixth challenge introduces three tasks, namely, end-to-end goal oriented dialog learning, end-to-end conversation modeling, and dialogue breakdown detection.

Many different belief tracking approaches have been proposed in the literatures. Early DST used hand-crafted rules. Generative models of DST employed Bayesian Network, the distribution over possible dialog states is updated by Bayesian inference [4,5]. Bohus and Rudnicky [6] proposed the first discriminative state tracking. Subsequent works had explored numerous variations of discriminative approach [7–10]. Henderson et al. [11] proposed a first word-based DST model. Different from traditional DST models which used outputs of Natural Language Understanding (NLU) as inputs, the word-based DST mapped directly from the utterances to an updated belief state without an explicit NLU. Subsequently, many NLU and DST joint models were proposed and most achieved good results [12–16]. There had also been some works on combining the generative model or rule-based method with the discriminative model to update the dialog state respectively, and these ideas achieved good performance [17,18].

Most of the above models generally rely on either classification over a fixed slot value set or scoring each candidate slot value pairs separately, in which the slot values are predefined, the value might be unseen but not unknown. An unseen value is a different expression of a predefined slot value, and it is not observed during the training, while an unknown value is out of the predefined slot values. For example, a predefined value set of the slot *food type* might be $\{Spanish, French, Chinese\}$ for a restaurant dialogue system, with the joining of new people from different places, new slot values like $\{Italian, Vietnamese\}$ will occur in dialogs. They are not different expressions of existing slot values and cannot be classified to any of them. They are unknown values to the system. In practical applications, however, the continuous emergence of unknown slot values in dialogs is an inevitable phenomenon. Firstly, a slot may find more and more values with the increasing of users. Secondly, it is impossible to collect training data for all slot values, the training corpus is limited, especially the initial corpus for new fields. If unknown slot values cannot be accurately recognized and distinguished, the dialog states cannot be correctly updated in real time, solving the problem of unknown slot values plays an important role in improving the performance of DST.

Henderson et al. [11,19] firstly introduces a de-lexicalization module in dialog state tracking model to identify mentions of unseen slot values. The de-lexicalization strategy is replacing slot values mentioned in the dialog text with generic tags. Such a conversion allows the models to generalize much better to infrequent or unseen values. And then Mrkšić et al. [12] learns vector representation for user utterance and candidate slot-value pairs to make final decisions on whether the user expresses the current candidate slot-value pair, using the pre-trained word embeddings to handle unseen values in the utterance. These models are only designed to handle unseen values.

Kadlec et al. [23] derives several heuristics based on the existing NLU results to track the dialog state, they make changes to the original NLU hypotheses through identifying several obvious shortcomings of it, and finally achieves the optimal results on DSTC3 dataset which suffers from the problem of unknown slot values. In fact, they don't do any work on unknown slot values.

Relevant works include zero-shot or few-shot learning problem and novel class detection or anomaly detection. The general idea behind zero-shot or few-shot learning is to map the input and class labels to a semantic space in which similar classes are represented by closer points in the space [20,21]. However, these methods are only tried on simple datasets at the moment. Complex datasets need to design complex map functions to ensure that the input and class labels are mapped to the same semantic space. Traditional novel class or anomaly detection puts the detected novel or abnormal samples in cache to form its training corpus after finding enough samples, not only needs to retrain, but also introduces time constraints for distinguishing between novel or abnormal classes and known classes [22].

Unknown slot values can be considered as new or anomaly categories of dialog state, the detection of unknown slot values can be treated as the process of novel class or anomaly detection. But after an unknown slot value being detected, it needs to be included in the process of state updating to facilitate the action generation.

Therefore, to deal with unknown slot values, we consider dividing DST into several steps. Firstly, detecting whether an unknown slot value is present or not. Then updating the distribution for the unknown and known slot values. Finally integrating distributions for unknown slot values with that of known slot values. On the one hand, the updating of unknown and known slot values is at the same time, and all parameters of the three-level are learned simultaneously to minimize the loss functions. On the other hand, the subtasks of detection and update are related tightly. Joint modeling the three-step process makes them help each other.

Based on this, we propose a hierarchical dialog state tracking framework to model the dialog state tracking problem with unknown slot values. The whole framework consists of three levels of model. The first-level model of a cascaded neural network is used to detect whether an unknown slot value occurs. The second-level model has two parts, one is the update scheme for known slot value and the other is the update scheme for unknown slot value. The third-level model updates the dialog state on the basis of the first-level and second-level models and finally obtains the state of the dialog. The experimental results on DSTC and WOZ2.0 datasets show that the proposed hierarchical dialog state tracking framework achieves better results than those of [19,23], and achieves comparable performance with the model in Mrkšić et al. [12]. Especially, the detection and distinction of unknown slot values greatly improve the final performance of dialog state tracking.

The remainder of this paper is organized as follows. Section 2 describes our hierarchical dialog state tracking framework. The framework is introduced firstly, then each level of the framework is presented respectively. Section 3 introduces settings of the experiment. Section 4 details the experiment and the analysis of the results. Section 5 draws conclusions.

2 Method

Following Henderson et al. [19] and Mrkšić et al. [12], our model works on slot-by-slot. We therefore introduce our model on slot S for example. A vocabulary of values for slot S is $\{V_1, V_2, \ldots, V_m, V_{m+1}, \ldots, V_N\}$, where one part of the values is $V^{known} =$

$\{V_1, V_2, \ldots, V_m\}$, it refers to the set of known slot values. Any value might have different expressions, for example V_j has n_j possible expressions, i.e. $\left\{V_{j1}, V_{j2}, \ldots, V_{jn_j}\right\}$, a slot value being known means that there is at least one expression of that value occurring in training. Another part of values is $V^{unknown} = \{V_{m+1}, V_{m+2}, \ldots, V_N\}$, which refers to the set of unknown slot values. An unknown slot value is the value any expression of which does not occur in training.

Since there is no training data for unknown slot values, classification based models are unable to detect and distinguish them. We design a hierarchical dialog state tracking model (HDSTM) to be able to detect whether an unknown slot value occurs in the user utterance and distinguish which specific unknown slot value is expressed in the user utterance at the time of updating the state of the dialog.

The basic framework of the HDSTM is shown in Fig. 1. There are three levels. The first-level including a detection model D is used to detect whether an unknown slot value occurs in a user utterance. The second-level consists of two parts, one is the update scheme O for known slot values and the other is the update scheme N for unknown slot values. On the basis of the results in lower levels, the third-level model R updates the dialog state and finally gets the state label $STATE_t$ of the *t-th* dialog turn I_t. In Fig. 1, the abbreviation KB stands for knowledge base, and it will be detailed in Sect. 2.2.2.

Before we detail the models in different levels one-by-one, we first describe the inputs of the model. A belief tracker needs user utterance as inputs, and the dialog acts of the system leading up to the user utterance are also helpful. Examples are given in following.

(1) System Request: System asks the user about the value of a specific slot. For example, if the user responds to the system action request (slot = area) with "any", then the user utterance "any" refers to the slot "area", not to other slots such as "pricerange" and "food".
(2) System Confirm: System asks the user to confirm whether a specific slot-value pair is part of their desired constraints. For instance, if the system action is confirm (food = indian), the user answers with "yes", "yes" refers to the slot-value pair "food:indian", not to others.

Let $u_t = u_{t1}, u_{t2}, \ldots, u_{tk}$ be the user utterance at turn t, $m_{t-1} = m_{(t-1)1}, m_{(t-1)2}, \ldots, m_{(t-1)z}$ be the preceding system acts, k is the number of words in the user utterance and z is the number of words in the system acts. For simplification, we concatenate them into $f_t = w_{t1}, \ldots, w_{tk}, w_{t(k+1)}, \ldots, w_{tL} = u_{t1}, \ldots, u_{tk}\#m_{(t-1)1}, \ldots, m_{(t-1)z}$, the user

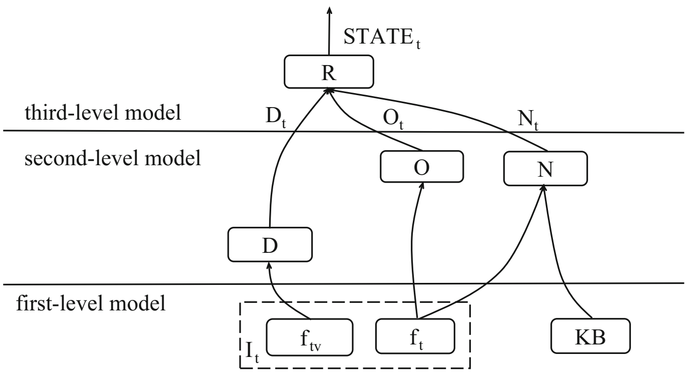

Fig. 1 Basic framework of HDSTM

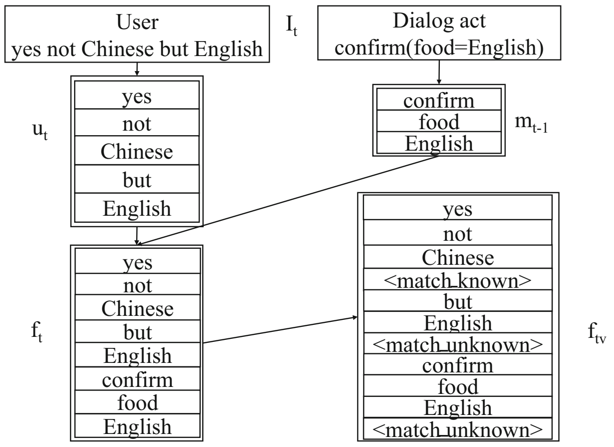

Fig. 2 Example of feature extraction for one turn, giving f_t and f_{tv}

utterance is in front, a symbol "#" denotes the concatenation operation, w_{ti} is the *i-th* word in f_t and w_{ti} denotes u_t for $1 \leq i \leq k$, w_{ti} denotes m_{t-1} for $k+1 \leq i \leq L$. The maximum number of words in f_t is L. Let $f_{tv} = d_{t1}, d_{t2}, \ldots, d_{tk}, d_{t(k+1)}, \ldots, d_{tM}$ be tagged features of f_t, f_{tv} is created from features f_t through appending occurrence of specific values by common tags, the maximum number of words in f_{tv} is M. There are two generic symbols here, one is $\langle match\_known \rangle$, the other is $\langle match\_unknown \rangle$. All occurrence known slot values in f_t are appended by the symbol $\langle match\_known \rangle$, all unknown slot values are appended by the symbol $\langle match\_unknown \rangle$ through querying the knowledge base. For example, if "Chinese" is a known slot value and "English" is an unknown slot value of slot *food type* (The ontology of the dataset provides the candidate values for each specific slot, the slot values occurring in training data are known slot values, and the rest are unknown slot values). The user utterance might be "yes not Chinese but English", and its proceeding dialog act might be confirm (food = English). Feature extraction for this dialog turn t is outlined in Fig. 2. The following details the model of each level.

2.1 Detection Model

The detection model is used to detect whether an unknown slot value occurs or not, it is a binary classifier. For slot S, the classifier has two values, namely $\{V_{unk}, V_{notunk}\}$. If an unknown slot value occurs at *t-th* turn, then the classifier should output value V_{unk}, otherwise V_{notunk}. In the detection model, the belief state b_t at the *t-th* turn for slot S is the distribution over the two values $\{V_{unk}, V_{notunk}\}$.

The belief state b_t at the *t-th* turn is given by previous belief state b_{t-1}, the last system actions and a new observation [5]. Direct using of word sequence T_t, which is the concatenation of the corresponding user utterance and its preceding dialog acts, results in serious data sparseness. In this paper, T_t is first mapped to $\varphi(T_t)$, the representations of T_t, and then states are tracked based on $\varphi(T_t)$ and the previous belief state b_{t-1}. Then tracking the dialog state of slot S at turn t is to construct maps in (1) and (2).

$$\varphi : T_i \rightarrow \varphi(T_i) \tag{1}$$

$$p : \varphi(T_1), b_0, \ldots, \varphi(T_t), b_{t-1} \rightarrow S = \hat{a}_t \tag{2}$$

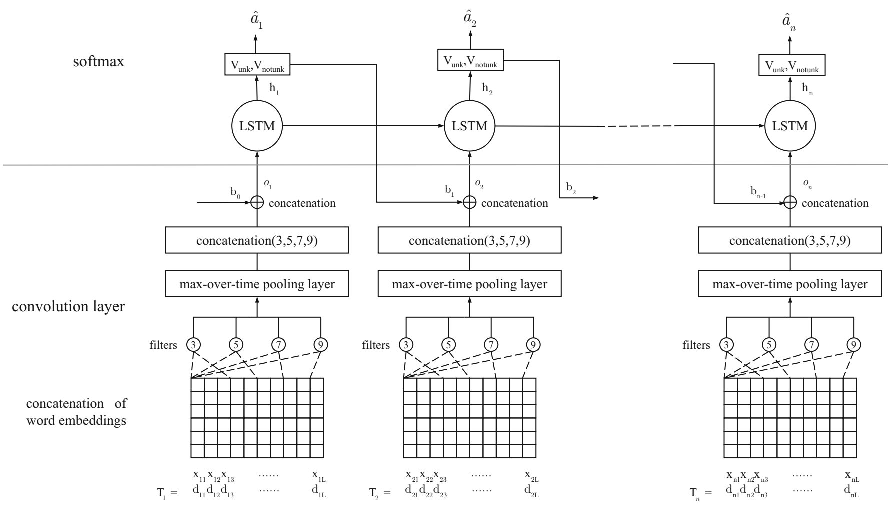
softmax
convolution layer
concatenation of word embeddings
LSTM
V_{unk}, V_{notunk}
$\hat{a}_1$
$\hat{a}_2$
$\hat{a}_n$
h_1
h_2
h_n
b_0
b_1
b_2
b_{n-1}
o_1
o_2
o_n
concatenation
concatenation(3,5,7,9)
max-over-time pooling layer
filters
3
5
7
9
$x_{11}x_{12}x_{13}$ ...... x_{1L}
$T_1 =$ $d_{11}d_{12}d_{13}$ ...... d_{1L}
$x_{21}x_{22}x_{23}$ ...... x_{2L}
$T_2 =$ $d_{21}d_{22}d_{23}$ ...... d_{2L}
$x_{n1}x_{n2}x_{n3}$ ...... x_{nL}
$T_n =$ $d_{n1}d_{n2}d_{n3}$ ...... d_{nL}

Fig. 3 Structure of the first-level detection model

where b_0 is the initial belief state distribution, $\hat{a}_t \in \{V_{unk}, V_{notunk}\}$, and for the detection model $T_t = f_{tv}$.

Different from previous pipeline models which model two maps in (1) and (2) separately, a cascaded structure is proposed in this paper to jointly model the two maps, which are trained jointly. NLU and DST are therefore integrated in a single model.

There are two layers in the joint model. The upper layer employs a Long Short Term Memory (LSTM) [24], which receives vector representations of word sequence, belief state of previous turns, and encodes them into a latest hidden state. The bottom layer adopts Convolutional Neural Networks (CNN) [25,26] to receive tagged features f_{tv} to construct turn representation. The detailed structure of the detection model is shown in Fig. 3.

Here, x_{ti} represents the m-dimensional word vector of d_{ti}, initialized randomly, and x_{ti} is input to the underlying CNN in order. The CNN involves a filter w, which is applied to a window of h words to produce a new feature. For example, conducting convolution operation on $x_{ti:t(i+h-1)}$ produces a feature $c_{ti} = f\left(w \cdot x_{ti:t(i+h-1)} + b\right)$, where b is a bias term and f is a non-linear function. The filter is applied to each possible window of words $\left\{x_{t1:th}, x_{t2:t(h+1)}, \ldots, x_{t(L-h+1):tL}\right\}$ to generate a feature map $c_t = \left[c_{t1}, c_{t2}, \ldots, c_{t(L-h+1)}\right]$. Then, the maximum value of the feature map is selected as the feature corresponding to this particular filter by applying a max-pooling layer. Multiple filters and multiple feature maps are adopted in this paper. Therefore, the fixed dimensional vector representation $\varphi(f_{tv})$ of turn t is the concatenation of the features selected from different feature maps generated by different filters.

$\varphi(f_{tv})$ is passed to the upper LSTM as the representation of the tagged features f_{tv}. The current belief state b_t is updated through the upper LSTM as follows:

$$o_t = \varphi(f_{tv}) \oplus b_{t-1} \tag{3}$$

$$h_t = LSTM(h_{t-1}, o_t) \tag{4}$$

$$b_t = softmax(linear(h_t)) \tag{5}$$

The input of the upper LSTM o_t is the concatenation of the current utterance representation $\varphi(f_{tv})$ and the previous belief state b_{t-1}. The hidden state h_t of the upper LSTM is used to output the belief state b_t by a softmax operation on a liner layer.

For slot *S*, the value with the maximum confidence score is the state label of the *t-th* turn.

$$\hat{a}_t = argmax\,(b_t) \tag{6}$$

Let θ be the network parameters which are randomly initialized as θ_0. The detection model is trained as supervised learning. We use cross-entropy as the loss function. For a dialog segment, the ground-truth slot value is $A = a_1, a_2, \ldots, a_{DL}$, the predicted confidence score for A is $\hat{A} = \hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{DL}$. Here, *DL* is the maximum dialog length. The state tracking loss for this dialog segment is computed as follows:

$$J_D(\theta) = -\sum_{i=1}^{DL} log\hat{a}_i \tag{7}$$

The main problem of unknown slot values recognition is the absence of training data for them. The most straightforward idea is to construct training data for them. It is much easier to construct negative examples for the detection model for known slot values than to construct training data for unknown slot values. Without changing the dataset, known slot values with relatively lower frequency in the training set are used as negative examples, that is are simulated as unknown slot values to enable the ability of recognizing unknown slot values. We select simulated negative samples this way to investigate the effect of the proposition of it on experimental results. We conduct this in following procedures. Firstly, the number of training data for known slot values is sorted in descending order, then the last several known slot values are simulated as unknown slot values, and the number of the simulated negative samples is increased gradually to investigate the effect of the proposition of it on experimental results.

2.2 Second-Level Model

2.2.1 Update Scheme for Known Slot Values

When the first-level detection model outputs value V_{notunk}, unknown slot values are not detected at *t-th* turn, the dialog state is updated through the update scheme for known slot values. The update scheme for known slot values is multiple classifiers, the belief state b'_t at the *t-th* turn for slot *S* is the distribution over the known slot values $V^{known} = \{V_1, V_2, \ldots, V_m\}$.

The update scheme for known slot values adopts the same architecture with the first-level detection model, that is, f_t is first mapped to $\varphi(f_t)$, the representation of f_t, through the bottom CNN layer, and then belief states of the current turn b'_t are tracked based on the corresponding utterance representation $\varphi(f_t)$ and the previous belief state b'_{t-1} through the upper LSTM layer, finally the value with the maximum confidence score $\hat{a}'_t$ is the state label of the *t-th* turn. In the update scheme for known slot values, $\hat{a}'_t \in \{V_1, V_2, \ldots, V_m\}$. Detailed network computing for the update scheme for known slot values is as follows:

$$\varphi : f_t \rightarrow \varphi(f_t) \tag{8}$$

$$o'_t = \varphi(f_t) \oplus b'_{t-1} \tag{9}$$

$$h'_t = LSTM\left(h'_{t-1}, o'_t\right) \tag{10}$$

$$b'_t = softmax\left(linear\left(h'_t\right)\right) \tag{11}$$

$$\hat{a}'_t = argmax\left(b'_t\right) \tag{12}$$

As with the first-level detection model, the update scheme for known slot values O is also trained as supervised learning. With ground-truth state labels $A' = a'_1, a'_2, \ldots, a'_{DL}$ for a given dialog segment and its predicted confidence score $\hat{A}' = \hat{a}'_1, \hat{a}'_2, \ldots, \hat{a}'_{DL}$, the state tracking cross-entropy loss is computed as follows:

$$J_O(\theta) = -\sum_{i=1}^{DL} log\hat{a}'_i \tag{13}$$

2.2.2 Update Scheme for Unknown Slot Values

When the first-level detection model outputs value V_{unk}, unknown slot values are detected at *t-th* turn, the dialog state is updated through the update scheme for unknown slot values N. Model fails in distinguishing between different unknown slot values for no training data for them. Although the unknown slot values do not appear in the training corpus, it can be assumed that the unknown slot values appear in external corpus, which is reasonable in most cases. Therefore, a knowledge base can be constructed to distinguish between different unknown slot values. When the first-level detection model outputs value V_{unk}, the knowledge base will be queried to determine which specific unknown slot value $\hat{a}''_t \in V^{unknown} = \{V_{m+1}, V_{m+2}, \ldots, V_N\}$ is expressed in the utterance.

We constructed the knowledge base by fuzzy string matching. Fuzzy string matching is the string match applied between each possible window of words in the utterance with a slot value, the window size is the number of words in the slot value, when the match ratio is greater than a certain threshold, the segment in the utterance is selected to put into a raw semantic dictionary, then we check if the resulting item is a possible expression of the slot value manually. At the same time, we integrate the contents of the semantic dictionaries produced by Henderson et al. [19] and Mrkšić et al. [12] to form the knowledge base. The knowledge base used in this paper is a slot value dictionary extended from the de-lexicalization slot value dictionaries produced by Henderson et al. [19] and Mrkšić et al. [12]. Items in such a knowledge base for two slot-value pairs are *FOOD=CHEAP:[cheaper, budget, affordable, inexpensive, economic,...]* and *PRICERANGE =MODERATE: [moderately, medium, reasonably priced,...]*.

2.3 Third-Level Model

The third-level model R updates the dialog state on the basis of the first-level and second-level models, the specific update process of dialog state for the *t-th* turn for S is shown in formula (14).

$$STATE_t = \begin{cases} \hat{a}''_t, & \text{if } D_t = V_{unk} \\ \hat{a}'_t, & \text{if } D_t = V_{notunk} \end{cases} \tag{14}$$

When the output of the detection model at *t-th* turn D_t is V_{unk}, an unknown slot value is detected, the dialog state of the *t-th* turn is the result of the update scheme for unknown slot values $\hat{a}''_t$, otherwise is the result of the update scheme for known slot values $\hat{a}'_t$. Finally the state label $STATE_t$ of the *t-th* turn is obtained.

3 Experimental Settings

3.1 Dataset

While there is a large body of empirical studies in DST, the evaluation protocols of DST on unknown slot values present a degree of variations. We present comparisons to several state-of-the-art studies in the same data settings. Three datasets are used to verify the performance of the proposed model and these corpora share almost the same domain ontology.

(1) DSTC3 is a human machine dialog data in the field of tourist information searching and was collected using Amazon Mechanical Turk. A corpus of 2264 dialogs was collected for evaluation. A total of 3235 dialogs are divided by a ratio of about 8:2 to get a training set and a verification set. Thus, the training set, validation set and test set of DSTC3 contain 2597, 638, and 2264 dialogs respectively. And each turn in the dialog may contain one or more slots. The ability of trackers to discover and distinguish unknown slot values is studied in this paper, thus the tracker is evaluated on a total of three slots, i.e. “pricerange”, “food”, “area”. Slot “name” will not be analyzed here, for almost all the values of slot “name” is *Null* in the dataset. The known and unknown value types for each slot in DSTC3 and the proportion of unknown slot values in test set are shown in Table 1.

(2) WOZ2.0 was collected with users assuming the role of the system or the user of a task-oriented dialogue system. The users were asked to type in natural language sentences, each contributed just a single turn to each dialogue. Before contributing their turns, users must review all previous turns in that dialogue to ensure coherence and consistency, and were encouraged to learn and correct each other based on previous turns. This turn-level data collection strategy yields a total of 1200 dialogues, containing 600 training, 200 validation and 400 test sets respectively. The known and unknown value types for each slot in WOZ2.0 and the proportion of unknown slot value types in test set are shown in Table 2.

(3) DSTC2 is a human machine dialog data in the field of restaurant searching, and also was collected using Amazon Mechanical Turk. According to the official division, the training set, the validation set and the test set of DSTC2 contain 1612, 506, and 1117 dialogs, respectively. Although the DSTC2 dataset does not suffer from the problem of unknown slot values, we also implement our model on DSTC2 dataset for comparison with the model in Mrkšić et al. [12].

Table 1 Statistics of DSTC3 dataset

Slot	Pricerange	Food	Area	Whole
Type				
Known	5	76	7	88
Unknown	1	12	14	27
Number				
Known	18,068	15,481	11,981	9216
Unknown	647	3234	6734	9499
Unknown proportion (%)	3.46	17.28	35.98	50.76

Table 2 Statistics of WOZ2.0 dataset

Slot	Pricerange	Food	Area	Whole
Type				
Known	5	75	7	87
Unknown	0	2	1	3
Number				
Known	1646	1643	1645	1642
Unknown	0	3	1	4
Unknown proportion(%)	0.0	0.18	0.06	0.24

3.2 Evaluation

Accuracy is used as the criterion for performance evaluation, it is the ratio of the dialogs correctly predicted states to the total number of dialogs.

$$Accuracy = \frac{dialogs\ correctly\ predicted\ states}{total\ number\ of\ dialogs} \tag{15}$$

3.3 Parameters

In the first-level detection model, the dimension of word vector, initialized randomly, is 50. The model is trained using stochastic gradient descent algorithm [27], whose momentum and learning rate are set to 0.9 and 0.01 respectively, and the batch size is set to 5. Filters 3, 5, 7, and 9 are adopted in the bottom CNN layer, each filter has 100 feature maps, and each 100 features obtained by filters 3, 5, 7, and 9 are spliced as the representation of the corresponding turn. The number of hidden nodes in the upper LSTM layer is set to 64. Except for the initial method and the dimension of word vectors, the update scheme for known slot values uses the same parameters as the first-level detection model. The dimension of word vector is 300 and the word vectors are initialized with semantically specialized Paragram-SL999 vectors in the update scheme for known slot value [28].

4 Experiment and Analysis

4.1 Experimental Results

Experimental results of the hierarchical dialog state tracking model and other models trained and evaluated on DSTC and WOZ2.0 datasets are shown in Table 3. “Pricerange”, “food”, “area” represent the three slots of the datasets, “all” represents the average of the accuracy, and “joint” is the percentage of dialogs with all three slots correctly predicted. The optimal performance on each dataset is shown in bold.

As can be seen from Table 3, the accuracy of the HDSTM proposed in this paper is higher than that of the model in Henderson et al. [19] which focuses on the problem of unseen slot values and Kadlec et al. [23] which derives several heuristics based on the existing NLU results, doing nothing on the problem of unknown slot values in fact. To the best of our knowledge, the HDSTM sets a new state-of-the-art result on DSTC3 dataset. The model in Mrkšić et al. [12] achieves the state-of-the-art results on WOZ2.0 and DSTC2 datasets so

Table 3 DSTC and WOZ2.0 test set accuracies of DST models

Dataset	Model	Pricerange (%)	Food (%)	Area (%)	All (%)	Joint (%)
DSTC3	Henderson et al. [19]	–	–	–	90.0	64.6
	Kadlec et al. [23]	–	–	–	–	66.6
	HDSTM	96.41	89.88	91.93	**92.74**	**72.71**
WOZ2.0	Mrkšić et al. [12]	–	–	–	–	84.4
	HDSTM	92.52	94.24	96.61	94.46	**84.51**
DSTC2	Mrkšić et al. [12]	–	–	–	–	**73.4**
	HDSTM	91.25	80.93	90.49	87.56	68.40

Table 4 Identification results for unknown slot value "cuban"

Role	Input	State		
		True	HDSTM	Mrkšić et al. [12]
User	"... Restaurant serving cuban food..."	Cuban	**Cuban**	Null
...	...	...	...	...
User	"... Looking for a cuban restaurant..."	Cuban	**Cuban**	Null
...	...	...	...	...

far. We also evaluated the performance of our model on WOZ2.0 and DSTC2 datasets, and found that the HDSTM achieved comparable performance with Mrkšić et al. [12]. Since the WOZ2.0 and DSTC2 datasets are not designed to focus on the task of handling unknown slot values. In WOZ2.0 dataset, there is little unknown slot values in slot "food", "area", and no unknown slot values in slot "pricerange" and the DSTC2 dataset does not suffer from the problem of unknown slot values. The experimental results illustrate that our model applies not only to datasets with unknown slot values, but also to regular datasets without unknown slot values. Statistical significance tests are implemented by five-fold cross validation on WOZ2.0 and DSTC2 datasets, the same conclusions are drawn.

4.2 Dialog State Tracking V.S. Unknown Slot Values

The ability of the model to identify unknown slot values plays a significant role in DST as unknown slot values appear. Table 4 gives an example, "cuban" is an unknown value for slot "food" in WOZ2.0 dataset, the predicted output of the model in Mrkšić et al. [12] is the known slot value *Null* when "cuban" occurs, the model in Mrkšić et al. [12] makes a mistake, because it cannot process unknown slot values. While our model gives correct result, and we highlight the correct result in bold in Table 4.

As can be seen from Table 4, classification based models, such as Mrkšić et al. [12], are inherently not capable of handling unknown values, while our model can solve the problem of unknown slot values effectively. We also implement our model and the model in Mrkšić et al. [12] on the datasets with increasing number of unknown slot values, and find that as the number of unknown slot values increases, the performance of the model in Mrkšić et al. [12] decreases rapidly, while the performance of our model decreases gradually. Both the DSTC and WOZ2.0 don't suffer from the problem of having different numbers of unknown slot values, we pick the "food" slot of WOZ2.0 to simulate different numbers of unknown

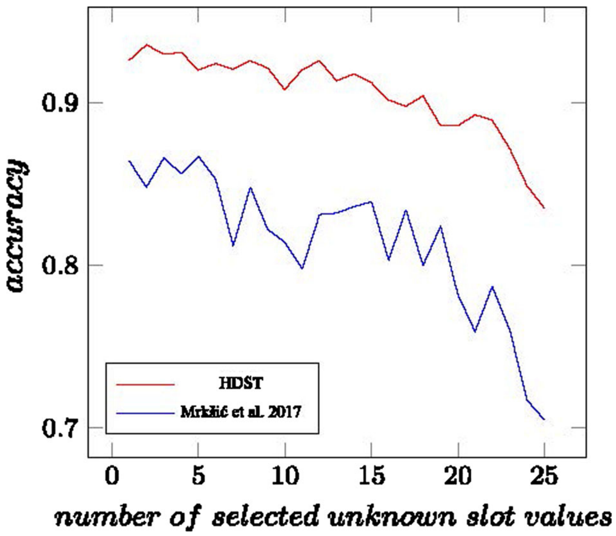

Fig. 4 Accuracy of models on the datasets with increasing numbers of unknown slot values

slot values to conduct the investigation. Specifically, we gradually select increasing numbers of food types in the training set as unknown and discard all the training instances where the correct food type is the selected unknown types. The accuracy of our model and the model in Mrkšić et al. [12] on the datasets with increasing numbers of unknown slot values is shown in Fig. 4, the experimental results prove our point of view.

The benefits of our model are evident when a large number of unknown slot values emerges. The experimental result on DSTC3 dataset is a good example. As can be seen from Table 1, if the problem of unknown slot values can not be solved, at least 50.76% dialog state of the test set in DSTC3 will be wrongly judged. The joint dialog state tracking accuracy is 72.71%, among which 32.49% is from the unknown slot values. The overall state tracking performance is greatly improved because of the accurate detection and distinction of unknown slot values in DSTC3 dataset.

4.3 Influence of Negative Examples

Does the method of constructing negative examples adopted in this paper have an effect on the performance of known slot values and the experimental results will be influenced by choosing different numbers of negative examples? To solve these doubts, firstly we analyze the impact of negative examples on the performance of known slot values, then experiments are designed to check whether selecting different numbers of negative examples affects the experimental results on DSTC and WOZ2.0 datasets.

4.3.1 Influence on the Performance of Known Slot Value

The experimental results for models at each level in the hierarchical dialog state tracking framework on DSTC and WOZ2.0 datasets are shown in Table 5.

The performance of each level model is very good. Experimental results of N show that the unknown slot values can be well distinguished by querying knowledge base, whether it is

Table 5 Accuracy of each level model in HDSTM

Dataset	Slot	D (%)	O (%)	N (%)
DSTC3	Pricerange	98.84	97.47	99.51
	Food	98.56	90.51	92.12
	Area	93.80	94.77	98.65
WOZ2.0	Pricerange	95.82	95.40	98.0
	Food	99.02	94.03	96.71
	Area	98.78	97.60	99.77
DSTC2	Pricerange	96.15	94.15	77.08
	Food	89.60	88.31	65.59
	Area	97.74	91.37	80.30

a simulated unknown slot value or a real unknown slot value. The occurring of unknown slot values can be effectively detected by the detection model D and the simulated unknown slot values can be well distinguished by querying the knowledge base, thus the adopted method of constructing negative examples has a little influence on the performance of known slot values.

Experimental results on WOZ2.0 and DSTC2 in Table 3 also illustrate this point. It can be seen from Table 2 that there is almost no unknown slot values in WOZ2.0 and DSTC2 datasets, experimental results on WOZ2.0 and DSTC2 can be seen as the results of only processing known slot values. It can be concluded that the adopted method of constructing negative examples has a little influence on the performance of known slot values.

4.3.2 Influence on the Overall Performance

We analyze whether the experimental results are sensitive to the number of negative examples. Take slot "food" of WOZ2.0 as an example, firstly the number of training data for known slot values is sorted in descending order. Then, the last 23, 24, 25, up to 70 known slot values are simulated as unknown slot values. The last 23 known slot values account for 2.64% of the training data and the last 70 account for 49.57% of the training data. The accuracies of the hierarchical dialog state tracking model that simulates different numbers of unknown slot values for slot "food" on WOZ2.0 dataset are shown in Fig. 5.

It can be observed from Fig. 5 that the curve of slot "food" is smoothing, and the amplitude of variation is less than 0.06. It shows that experimental results are not sensitive to the numbers of negative examples, it keeps stable in a wide range. The curves for other slots on its corresponding dataset show the same trends within a certain percentage of training data.

5 Conclusions

This paper proposes a hierarchical dialog state tracking framework to model the dialog state tracking problem with unknown slot values. The experimental results on DSTC and WOZ2.0 datasets show that the proposed framework achieves good performance. In particular, the discovery and distinction of unknown slot values greatly improve the final performance of dialog state tracking, illustrating the effectiveness of our proposed framework for addressing the problem of unknown slot values. Besides, the model applies not only to datasets with unknown slot values, but also to regular datasets without unknown slot values.

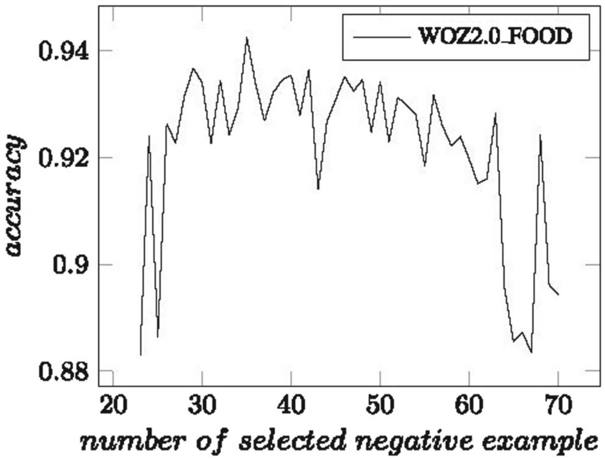

Fig. 5 Influence of selecting different numbers of negative examples

Unknown slot values can be effectively detected by constructing negative examples. Analysis of experimental results illustrates the adopted method of constructing negative examples has little influence on the performance of known slot values. Besides, under the condition of the selected negative examples accounting for a certain percentage of training data, the influence of selecting different numbers of negative examples on the experimental results is not obvious. It proves the generality and applicability of the HDSTM.

However each level model in hierarchical dialogue state tracking framework is separated now. The integrated training of the whole framework and its influence on experimental results need to be further explored.

Acknowledgements This paper is supported by NSFC (No. 61273365), NSSFC (2016ZDA055), 111Project (No. B08004), Beijing Advanced Innovation Center for Imaging Technology, Engineering Research Center of Information Networks of MOE, China.

References

1. Williams J, Raux A, Henderson M (2016) The dialog state tracking challenge series: a review. Dialogue Discourse 7(3):4–33
2. Henderson M, Thomson B, Williams J (2014) The second dialog state tracking challenge. In: SIGDIAL, pp 263–272
3. Henderson M, Thomson B, Williams JD (2014) The third dialog state tracking challenge. In: SLT workshop, pp 324–329
4. Williams JD, Poupart P, Young S (2005) Factored partially observable markov decision processes for dialogue management. In: Proceedings of the workshop on knowledge and reasoning in practical dialog systems, IJCAI, pp 76-82
5. Young S, Gašić M, Thomson B, Williams JD (2013) Pomdp-based statistical spoken dialog systems: a review. Proc IEEE 101(5):1160–1179
6. Bohus D, Rudnicky A (2006) A 'k hypotheses + other belief updating model. In: Proceedings of the AAAI workshop on statistical and empirical approaches for spoken dialogue systems
7. Metallinou A, Bohus D, Williams J (2013) Discriminative state tracking for spoken dialog systems. In: ACL, pp 466–475
8. Williams JD (2014) Web-style ranking and slu combination for dialog state tracking. In: SIGDIAL, pp 282–291

9. Ren H, Xu WQ, Yan YH (2014) Markovian discriminative modeling for cross-domain dialog state tracking. In: SLT workshop, pp 342–347
10. Kim S, Banchs RE (2014) Sequential labeling for tracking dynamic dialog states. In: SIGDIAL, pp 332–336
11. Henderson M, Thomson B, Young S (2014) Word-based dialog state tracking with recurrent neural networks. In: SIGDIAL, pp 292–299
12. Mrkšić N, Séaghdha DÓ, Wen TH, Thomson B, Young S (2017) Neural belief tracker: data-driven dialogue state tracking. In: ACL, pp 1777–1788
13. Yang XH, Liu J (2015) Dialog state tracking using long short-term memory neural networks. INTERSPEECH 2015:1800–1804
14. Shi HJ, Ushio T, Endo M, Yamagami K, Horii N (2017) Convolutional neural networks for multi-topic dialog state tracking. Springer, Berlin
15. Jang Y, Ham J, Lee BJ, Chang Y, Kim KE (2017) Neural dialog state tracker for large ontologies by attention mechanism. In: SLT workshop, pp 531–537
16. Mrkšić N, Séaghdha DÓ, Thomson B, Gašić M, Su PH, Vandyke D, Wen TH, Young S (2015) Multidomain dialog state tracking using recurrent neural networks. In: ACL, pp 794–799
17. Lee BJ, Kim KE (2016) Dialog history construction with long-short term memory for robust generative dialog state tracking. Dialogue Discourse 7(3):47–64
18. Sun K, Chen L, Zhu S, Yu K (2014) The sjtu system for dialog state tracking challenge 2. In: SIGDIAL, pp 318–326
19. Henderson M, Thomson B, Young S (2015) Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In: SLT workshop, pp 360–365
20. Yazdani M, Henderson J (2015) A model of zero-shot learning of spoken language understanding. In: EMNLP, pp 244-249
21. Ferreira E, Jabaian B, Lefevr F (2015) Online adaptative zero-shot learning spoken language understanding using word-embedding. In: ICASSP, pp 5321–5325
22. Mu X, Zhu F, Du J, Lim EP, Zhou ZH (2017) Streaming classification with emerging new class by class matrix sketching. In: AAAI
23. Kadlec R, Vodolán M, Libovický J, Macek J, Kleindienst J (2014) Knowledge-based dialog state tracking. In: SLT workshop, pp 348–353
24. Gers F, Schmidhuber J, Cummins F (2000) Learning to forget: continual prediction with lstm. Neural Comput 12(10):2451–2471
25. Kalchbrenner N, Grefenstette E, Blunsom P (2014) A convolutional neural network for modelling sentences. In: ACL
26. Kim S, Banchs RE, Li HZ (2016) Exploring convolutional and recurrent neural networks in sequential labelling for dialogue topic tracking. In: ACL, pp 963–973
27. Zhang S, Choromanska A, LeCun Y (2015) Deep learning with elastic averaging sgd. In: NIPS, pp 685–693
28. Mrkšić N, Séaghdha DÓ, Thomson B, Gašić M, Rojas-Barahona L, Su PH, Vandyke D, Wen TH, Young S (2016) Counter-fitting word vectors to linguistic constraints. In: Conference of the North American Chapter of the association for computational linguistics: human language technologies, pp 142–148