TRACKING OF ENRICHED DIALOG STATES FOR FLEXIBLE CONVERSATIONAL INFORMATION ACCESS

Yinpei Dai, Zhijian Ou, Dawei Ren, Pengfei Yu

Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University, Beijing, China
dyp16@mails.tsinghua.edu.cn, ozj@tsinghua.edu.cn

ABSTRACT

Dialog state tracking (DST) is a crucial component in a task-oriented dialog system for conversational information access. A common practice in current dialog systems is to define the dialog state by a set of slot-value pairs. Such representation of dialog states and the slot-filling based DST have been widely employed, but suffer from three drawbacks. (1) The dialog state can contain only a single value for a slot, and (2) can contain only users' affirmative preference over the values for a slot. (3) Current task-based dialog systems mainly focus on the searching task, while the enquiring task is also very common in practice. The above observations motivate us to enrich current representation of dialog states and collect a brand new dialog dataset about movies, based upon which we build a new DST, called enriched DST (EDST), for flexible movie information access. The EDST supports the searching task, the enquiring task and their mixed task. We show that our new EDST method not only achieves good results on Iqiyi dataset, but also outperforms other state-of-the-art DST methods on the traditional dialog datasets, WOZ2.0 and DSTC2.

***Index Terms*—** Dialog state, dialog state tracking, recurrent neural network, dialog dataset

1. INTRODUCTION

Dialog state tracking (DST) is a crucial component in a task-oriented dialog system for conversational information access. The dialog state summarizes the dialog history, including all previous user utterances and all system actions taken so far. It is passed to the system's dialog policy that decides which action to take. In general, the dialog state consists of elements with human-interpretable meanings from the task ontology, which describes the scope of semantics the system can process. The ontology is often specified by a collection of *slots* and the *values* that each slot can take. So a common practice in most dialog systems is to define the dialog state by a set of slot-value pairs. For example, in a movie information access system, after the user utterance *"I want to see Nolan's thriller"*, the dialog state gets updated to consist of new slot-value pairs *"director=Nolan; genre=thriller"*. In this case, we often say the dialog state contains the value "*Nolan*" for slot "*director*", and the value "*thriller*" for slot "*genre*". To accommodate uncertainty, most modern dialog state tracker maintains a probability distribution over dialog states, which is often called belief state. In practice, the tracker updates a multinomial distribution over all possible values for each slot separately, which is thus often referred to as slot-filling.

Such representation of dialog states and the slot-filling based DST have been widely employed, e.g. in the Dialogue State Tracking Challenge (DSTC) series [1, 2, 3]. However, this standard practice suffers from three drawbacks. First, in the set of slot-value pairs which defines the dialog state, at most one slot-value pair are allowed for an informable slot[1], or say, the dialog state can contain only a single value for a slot. For example, user's sentence *"I want moderately cheap restaurant"* can only be labeled as either *"price range=moderate"* or *"price range=cheap"* but not both in the DSTC2 dataset; in fact, the user wants a cheap or moderate restaurant, and both slot-value pairs should be included in DST. So it is desirable to enrich current dialog state representation to contain multi-values.

Second, when the dialog state contains a value for a slot, the preference is by default to be affirmative, or say, the dialog state can contain only users' affirmative preference over the values for a slot. However, in real-world information access, negative preferences are often expressed by users. Although in DSTC2, users' preference over values is partially supported by extra tags like "*User Actions*", this requires extra efforts to label data and build DST. So it is desirable to enrich current dialog state representation to contain such preference over values.

Third, current popular task-oriented dialog systems and datasets mainly focus on the searching task [4, 5]. The system helps the user to find certain entity by interactively asking for attributes which helps constrain the search (i.e. informable slots). Once found, the

User: Who directs Aliens?
`inform(Film_name=MENTIONED, Aliens=LIKE)`
`request(director)`
System: James Cameron. Do you want to see Aliens?
User: No. I don't want any frightening films. Find me more of his action movies
`inform(Director=MENTIONED, James Cameron=LIKE;`
`Film_name=MENTIONED, Aliens=DISLIKE;`
`Genre=MENTIONED, Action=LIKE, Thriller=DISLIKE)`
System: Sorry no movie reach your requirments.
User: Just find me Cameron's films, no limitation for type.
`inform(Director=MENTIONED, James Cameron=LIKE;`
`Film_name=MENTIONED, Aliens=DISLIKE;`
`Genre=DONT_CARE)`
System: Find many: Avatar, Terminator, Titanic etc.
User: What about Titanic? Is it free?
`inform(Director=MENTIONED, James Cameron=LIKE;`
`Film_name=MENTIONED, Aliens=DISLIKE,Titanic=LIKE;`
`Genre=DONT_CARE)`
`request(payment)`

Fig. 1. Example of the Iqiyi dialog dataset with annotated new dialog states. Slots and values with label `NOT_MENTIONED` are not shown for simplicity. `inform` is related with informable slots, `request` is pertinent to requestable slots.

This work is supported by NSFC grant 61473168.

[1]Informable slots are slots that users can use to constrain the search, such as movie type.

user can retrieve information by asking for requestable slots[2]. In practice, however, the enquiring task in which the user directly asks for the requestable slots of some movies is also very common.

The above observations motivate us to enrich current representation of dialog states and collect a brand new dialog dataset about movies, based upon which we can build a flexible dialog system for accessing movie information. The system supports the searching task, the enquiring task and their mixed task. To overcome the limited representation capability of slot-value pairs, we propose to use two sets of labels for slots and values separately, so that the dialog state can contain multi-values and express preference over values (like/dislike) for a slot. Moreover, a novel DST, called enriched DST (EDST), is developed for tracking the enriched dialog states and supporting more flexible information access. We show that the new EDST method not only achieves good results on Iqiyi dataset, but also outperforms other state-of-the-art DST methods on the traditional dialog datasets, WOZ2.0 [6] and DSTC2 [1].

2. IQIYI DIALOG DATASET

We aim to develop a text-in text-out dialog dataset that contains both searching tasks (e.g. *usr: "I want to see Cameron's recent film". sys: "Find Avatar"*) and enquiring tasks (e.g. *usr: "What's Avatar's director". sys: "Cameron"*). In this dataset, the user is able to change his/her goals at will, and proposes searching constraints freely. Since this is already a non-trival task, we do not take ASR into account. *Movie* is chosen as our dialog domain. The movie database is extracted from Iqiyi movie website [3], which gives the name of our new dataset. A crowdsoucing website, similar to the Wizard-of-Oz (WOZ) website [4], was built to collect the dialog data. All the dialog data are in Chinese. After careful cleaning of the raw data, we collected 800 dialogues in total [4].

Ontology. There are 7 informable slots in total : *Film_name, Director, Actor, Genre, Country, Time, Payment.* All the informable slots can take multiple values, except *Film_name* can take only one value which the user would prefer, since it is more natural for the user to focus on just one favourite movie entity at one turn in the task-oriented system. 11 requestable slots are chosen in total, 7 from informable slots and 4 extra are *Release_date, Critic_rating, Movie_length, Introduction.*

Enriched Dialog States. We enrich the traditional dialog state representation, by employing two sets of labels for slots and values seperately. In this dataset, we use `DONT_CARE, MENTIONED, NOT_MENTIONED` as the labels for each informable slot since they can represent the general status of slots through conversations, and we choose the most common labels `LIKE, DISLIKE, NOT_MENTIONED` for each value. Our new dialog state takes the form as slot-label pairs and value-label pairs. In this tagging scheme, the dialog state could include multiple values and user's polarity preference directly. Figure 1 illustrates an example piece of Iqiyi dialog dataset with annotated new dialog states.

Dialog Tasks. Three types of task descriptions are constructed in the WOZ experiment as the guidence for collecting dialog data: (1) Seaching task: ask the user to find an unknown movie while knowing some attributes of it; (2) Equiring task: given a known movie, the user need to find out some attributes of it; (3) Mixed task: while knowing a movie's name or some attributes, the user needs to search more related unknown movies and their attributes. In the data collection, both users and wizards do not have to follow the task guidance strictly, and are encouraged to provide any kind of conversation as long as our new dialog state can handle.

dataset	WOZ2.0	DSTC2	Iqiyi
domain	restaurant	restaurant	movie
#words	784	1739	1527
#named entity	113	113	147
#slots	3	4	7
#values	212	212	599
#synonyms	≈ 60	$\ll 60$	≈ 200
goal change	11.2%	6.24%	13.1%

Table 1. Comparision among datasets. #slots is the number of informable slots, #values is the number of all the values in ontoloty, #named entity is the number of restaurants or movies in the database. Goal change is the percentage of turns in which the user changes his/her mind.

Comparison. Two typical dialog datasets, WOZ2.0 [6] and DSTC2 [1], are chosen for comparison since they are freely available, and the existing DST methods tested over them become baselines for studying our new DST model. Compared to WOZ2.0 and DSCT2, the Iqiyi dialog dataset not only has more complex dialog states, but also consists of more lexicon variations and goal changes, as shown in Table 1. This makes the Iqiyi dialog dataset a more chanllenging testbed for searching and enquiring dialog tasks.

3. DST MODEL

Enriched Dialog State Tracker (EDST) is a Jordan-type RNN customized for our Iqiyi dialog dataset. Its recurrent component consists of two sub-trackers, value-specific tracker (VST) and slot-specific tracker (SST), which are built for tracking value labels and slot labels separately. Since multiple values are allowed to be chosen, all the values are predicted separately to avoid state space exploding .

3.1. Model Definition

Let s denote the slot entity (e.g. `Genre`), v denote the value entity (e.g. `Thriller`), and V^s denote the vocabulary of all value entities for slot s. S denotes the vocabulary of all slot entities. Let η_v denote the tracking variable corresponding to value v at current turn, so η_v can take `LIKE, DISLIKE, NOT_MENTIONED`. η^s denotes the set of variables $\{\eta_v : v \in V^s\}$, and η represents the set $\{\eta^s : s \in S\}$. We have similar notations for slots. Let ξ_s denote the tracking variable corresponding to slot s at current turn, thus ξ_s can take `DONT_CARE, MENTIONED, NOT_MENTIONED`. ξ is defined as the set $\{\xi_s : s \in S\}$.

Assuming that the dialog is Markovian, we denote the user uterrance at current turn by u, the system act and belief state at the last turn by a and b respectively. The belief state at current turn is defined by $p(\xi, \eta|u, a, b)$, the posterior distribution of ξ, η given u, a, b. The main purpose of our DST is to maintain this belief state at each turn, which is recursively updated as follows: [5]

$$\begin{aligned} p(\xi, \eta) &= \prod_{s \in S} p(\xi_s, \eta^s) = \prod_{s \in S} p(\xi_s|\eta^s) p(\eta^s) \\ &= \prod_{s \in S} \left(p(\xi_s|\eta^s) \prod_{v \in V^s} p(\eta_v) \right) \end{aligned} \tag{1}$$

[2]Requestable slots are slots that users can ask a value for, such as movie length.

[3]http://www.iqiyi.com/dianying/

[4]http://oa.ee.tsinghua.edu.cn/ouzhijian/data/Iqiyi_movie_data.rar

[5]We drop the condition u, a, b and replace label `NOT_MENTIONED` with `NOT_MEN` in the following to simplify the notation.

It can be easily seen that the independence among variables underlying the above factorization is reasonable. Also note that in traditional DST, the belief state for slot s is represented by a multinomial distribution over values $v \in V^s$. In contrast, in EDST, it is represented by $p(\xi_s, \eta^s)$, which is clearly more expressive and flexible.

In the following, we design two trackers, the value-specific tracker (VST) and slot-specific tracker (SST), for tracking $p(\eta_v)$ and $p(\xi_s|\eta^s)$ respectively.

3.2. Value-Specific Tracker

In VST, for each slot s, we iterate over all possible values $v \in V^s$ to update $p(\eta_v)$, and finally we have $p(\eta^s) = \prod_{v \in V^s} p(\eta_v)$. So the basic operation is to update $p(\eta_v)$ for $v \in V^s$, which contains three main steps as described in the following. Its input consists of last system act a, last belief states b, current user utterace u, and the value v over which we need to update the posterior distribution $p(\eta_v)$.

First, we convert the input to value-specific features.

1. From the last belief states b, a 3-dimensional value-specific belief vector is extracted by $f_1(v, \cdot)$ operation, which carries relevant information only to v:

$$f_1(v, b) = \Big(p(\eta_v^{'} = \texttt{LIKE}), p(\eta_v^{'} = \texttt{DISLIKE}), p(\eta_v^{'} = \texttt{NOT\_MEN})\Big)^T \tag{2}$$

where $\eta_v^{'}$ denotes the variable corresponding to v at the last turn. T denotes matrix transpose.

2. From the last system act a, a 6-dimensional value-specific act vector is extracted by $f_2(v, \cdot)$. It consists of 6 indicators about v: (1) whether a requests the slot s. (2) whether a confirms v as liked. (3) whether a confirms v as disliked. (4) whether a confirms other values in slot s. (5) whether a informs v. (6) whether none of above holds.

3. Current user utterance u is converted into a value-specific embedding matrix by $f_3(v, \cdot)$ operation. Suppose current utterance u contains k_u words $u_1, u_2, \ldots, u_{k_u}$. The embedding operation $e(\cdot)$ converts a word into a vector in $\mathbb{R}^d$, where d denotes the word embedding dimension. Let $X \in \mathbb{R}^{k_u \times d}$ denote the word embedding matrix of user utterance u. We convert u into value-specific embedding matrix $f_3(v, u) \in \mathbb{R}^{k_u \times (d+2)}$ by concatenating (denoted by $\oplus$) X with two other vectors as follows:

$$f_3(v, u) = X \oplus x_{dot}(v, u) \oplus x_{str}(v, u) \tag{3}$$

where $x_{dot}(v, u) = \sigma\left(w_1 \left(Xe(v)\right) + b_1\right)$. w_1, b_1 are scalar trainable paramenters. $\sigma(\cdot)$ denotes the sigmoid function. $x_{dot}(v, u) \in \mathbb{R}^{k_u}$ stores the result of dot products between the embedding of v and every word embedding of u followed by a nonlinear transform to $[0, 1]$. Since dot products between similar words' embeddings tend to be large, so this term reflects the extent of v appearing in u.

$x_{str}(v, u) \in \mathbb{R}^{k_u}$ denotes the string-matching binary vector, in which the i-th element taking 1 if the word u_i matches v and 0 otherwise. We use words in the ontology if there's no semantic dictionary. Semantic dictionary is hand-designed task-related synonym lists. By using both x_{dot} and x_{str}, we can combine both the advantages of hand-crafted semantic dictionary and pre-trained word vectors.

Second, based on $f_1(v, b) \in \mathbb{R}^3, f_2(v, a) \in \mathbb{R}^6, f_3(v, u) \in \mathbb{R}^{k_u \times (d+2)}$, we further extract the value-specific features for belief updating of $p(\eta_v)$. We first use a CNN module, similar to [6]. L convolutional filters of window sizes $1, 2, 3$ are applied to value-specific embedding matrix $f_3(v, u)$. The convolutions are followed by the ReLU activation function and max-pooling to produce summary vectors for these n-gram ($n = 1, 2, 3$) like features. The three summary vectors are concatenated into one vector as the output of the CNN. In the following, we use $\phi_{dim}^{sigmoid}(\cdot)$ and $\phi_{dim}^{softmax}(\cdot)$ to denote a fully-connected neural network with one hidden layer having the same size as input layer, where the output layer is of size dim, and is sigmoid and softmax respectively. The value-specific features are then obtained as follows:

$$\begin{aligned} g_f &= \phi_3^{sigmoid}\left(\text{CNN}(f_3(v, u))\right) \\ h_f &= f_1(v, b) \otimes g_f \\ r_{f,i} &= \mathbf{1}(f_2(v, a))[i] = 1) \cdot \text{CNN}(f_3(u, v))\ i = 1, 2, \cdots, 6 \end{aligned} \tag{4}$$

where $\mathbf{1}()$ is indicator function, $f_2(v, a)[i]$ is the i-th element of $f_2(v, a)$, $\otimes$ denotes element-wise product. Note that here g_f and $f_2(v, a)$ act like gates to control the information flow and adapt CNNs for different system acts, which is helpful for dealing with goal changes and expression variations in the conversation.

Finally, we concatenate the above features into one vector, and feed it into a fully-connected network to update $p(\eta_v)$ by:

$$\phi_3^{softmax}\left(h_f \oplus r_{f,1} \oplus \cdots \oplus r_{f,6}\right) \tag{5}$$

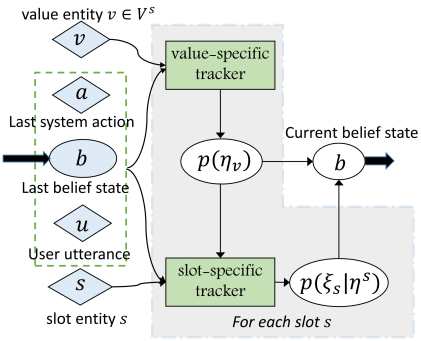

Fig. 2. The architecture of EDST

3.3. Learning and Inference

SST has the same architecture as VST except that value entity v is replaced by slot entity s. Also note that according to our EDST label settings, we have:

$$\begin{aligned} p(\xi_s = \texttt{MENTIONED} | \exists v \in V^s, \eta_v \neq \texttt{NOT\_MEN}) &= 1 \\ p(\xi_s = \texttt{MENTIONED} | \forall v \in V^s, \eta_v = \texttt{NOT\_MEN}) &= 0 \end{aligned} \tag{6}$$

Therefore for updating $p(\xi_s|\eta^s)$, SST in fact only needs to infer about `DONT_CARE, NOT_MEN`. Details are omitted to save space.

During learning, a complete dialog data is decomposed into multiple turn data, each is then transformed into two types of data for training VST and SST respectively. During inference, we first use VST to obtain $p(\eta_v)$, choose the maximum a posterior (MAP) value label for each value $v \in V^s$, and then calculate $p(\xi_s|\eta^s)$ with SST to obtain the complete belief state $p(\xi_s, \eta^s)$.

We separately maintain a VST and a SST for each informable slot, and a VST for each requestable slot. As requestable slots serve to model single-turn user queries, slot tracking across turns and value tracking are not needed. We remove cross-turn input a and b, feed only current user utterace u and the requestable slot label s to SST, which infer whether this requestable slot is mentioned or not.

4. EXPERIMENT

EDST is mainly designed for Iqiyi dialog dataset, but it can also be used for WOZ2.0 and DSTC2. In the latter case, we have only two value labels `MENTIONED`, `NOT_MENTIONED`; during inference, we choose the MAP value label for each value $v \in V^s$, and then choose the most probable value, since a slot can take only a single value.

Glove.300d [7] vectors are used for word embedding. For all experiments, CNN filters $L = 50$, so the output of CNN is a vector of size 150. Dropout with 50% rate is used in intermediate NN layers, and gradient clipping is applied with max global norm 5 to handle exploding gradients. All models are trained under cross-entropy loss with Adam optimizer [8], early-stopping is employed on validation set to prevent over-fitting. Careful design of minibatch sampling[6] is used to solve the label bias problem. The evaluation metrics are Joint Goal and Request, which is the turn-level accuracy of tracking results for all informable slots and all requestable slots respectively.

4.1. Results on WOZ2.0 dataset

In WOZ2.0, since turn-level labels are available, we remove last belief state b from input in EDST. Note that using turn-level based labels usually gives better results than using accumulated belief state labels, since the model do not have to learn the effect of last belief state. A rule-based tracking scheme is used to accumulate the turn-level prediction: substitute the old value label with new one if it is not labeled as `NOT_MEN`. It can been seen from the results in Table 2 that a slight gain in turn-level goal brings a large improvement on the Joint Goal. Our model outperforms all other methods and achieve the best results when combining the advatages of both semantic dictionary and pre-trained word vectors.

Model	Turn-level Goal	Joint Goal	Request
EDST+dict.	**92.8**	**87.5**	**95.3**
EDST	91.6	85.2	95.2
EDST-spec.	85.5	63.0	90.5
RNN [6]	–	70.8	87.1
RNN+dict. [6]	–	83.7	87.6
NBT [6]	–	84.4	91.6

Table 2. Results for delexicalisation-based RNN, NBT and EDST on WOZ2.0. +dict means using the semantic dictionary, -spec. means we feed the original word embedding matrix X instead of value-specific embedding matrix $f_3(v, u)$ to CNNs.

4.2. Results on DSTC2 dataset

In DSTC2, we need to handle ASR problem. We train EDST on the transcripts and evaluate on the testset's ASR N-best lists. Let P_i be the posterior probability for the i-th hypothesis at current turn, hyp_i be the i-th hypothesis utterance, the belief state is represented as follows:

$$p(\xi, \eta|\text{ASR}) = \sum_{i=1}^{N} P_i \, p(\xi, \eta|\text{hyp}_i) \tag{7}$$

Because the language usage in the DSTC2 dataset is less rich, adding semantic dictionary is less useful than in WOZ2.0. Table 3 shows the results of different models trained only with transcripts. Our EDST achieves superior results.

[6]Minibatch size is 256 and 64 for VST and SST respectively. For both VST and SST, the ratio of positive and negative training samples is 1:7 for WOZ2.0, DSTC2 and Iqiyi, except that the ratio is 0.7:0.3:7 for Iqiyi VST of `LIKE`, `DISLIKE`, `NOT_MENTIONED`.

Model	Joint Goal	Request
EDST	73.9	**96.6**
RNN+dict. [6]	72.9	95.7
NBT [6]	73.4	96.5
MemN2N [9]	**74**	–

Table 3. Results for NBT, MemN2N and EDST on DSTC2.

4.3. Results on Iqiyi dataset

For Iqiyi dialog dataset, since there is no off-the-self DST model suitable for the new dialog state labels, we construct a template-based baseline to compare with our EDST. All the template are extracted from the training and validation set[7] by delexicalisation [4], and used in testing with fuzzy string matching. We employ jieba toolbox[8] to segment Chinese words, and learn 25-dimensional semantically specialised word vectors using a method similar to [10]. Final results are shown in Table 4. We find that DST on Iqiyi dataset is more difficult mainly due to lexicon variations and task variety.

Model	Joint Goal	Request
EDST+dict.	**70.1**	**97.4**
EDST	63.6	97.0
template+dict.	46.3	82.5

Table 4. Results on Iqiyi dialog dataset.

5. RELATED WORK

Recent DST studies mainly focus on using deep neural networks. Initially, a word-based RNN with n-gram features is proposed in [11, 12]. It has been shown in [4] that employing CNN features with RNN can yield better results. In [6], given fine-trained semantic word vectors and turn-level labels, rule-based DST with CNNs outperforms traditional RNN models. A hybrid DST is built in [13], which consists of both rule-based part and machine-learning part. Novel structures such as attention-based Seq2Seq [14] and Memory Network [9] are also studied.

Our EDST model is motivated by the Neural Belief Tracker (NBT) [6] and delexicalisation-based RNN belief tracker [4]. NBT utilizes fine-pretrained word vectors to reduce the burden of building semantic dictionary, and delexicalisation-based RNN does not need turn-level labels. Our new EDST model combines the strengths of the two models. Label dependency modeling has also been studied. [15] proposes a CNN-based triangular CRF for sentence-level optimization, [16] combines the Jordan-type and Elman-type RNNs to predict the outputs depending on the last labels. In this work, to deal with the goal change problem, we build a CNN acting as a gate on last belief state to learn how to update the belief state.

6. CONCLUSION AND FUTURE WORK

In this work, we first introduce a new dialog dataset Iqiyi with enriched DST labels which can represent polarity preference and multi-values, then we develop a novel RNN-based EDST and achieve superior performances on WOZ2.0, DSTC2 and our Iqiyi dataset.

There are interesting future works so as to achieve more flexible conversational information access: building the whole dialog system based on EDST; unifying the searching and enquiring tasks in a better way; incorporating semantic parsing.

[7]For Iqiyi dataset, the ratio of training, validation and testing size is 3:1:1.
[8]https://github.com/fxsjy/jieba

7. REFERENCES

[1] Jason Williams, Antoine Raux, and Matthew Henderson, "The dialog state tracking challenge series: A review," *Dialogue & Discourse*, April 2016.

[2] Seokhwan Kim, Luis Fernando DHaro, Rafael E. Banchs, Jason D. Williams, and Matthew Henderson, "The fourth dialog state tracking challenge," *Ai Magazine*, vol. 35, no. 4, pp. 121–124, 2017.

[3] Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason D. Williams, Matthew Henderson, and Koichiro Yoshino, "The fifth dialog state tracking challenge," in *Spoken Language Technology Workshop*, 2017, pp. 324–329.

[4] Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M. Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young, "A network-based end-to-end trainable task-oriented dialogue system," in *EACL*, Valencia, Spain, April 2017, pp. 438–449, Association for Computational Linguistics.

[5] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun Nung Chen, Faisal Ahmed, and Li Deng, "Towards end-to-end reinforcement learning of dialogue agents for information access," in *Meeting of the Association for Computational Linguistics*, 2017, pp. 484–495.

[6] Nikola Mrkšić, Diarmuid O Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young, "Neural belief tracker: Data-driven dialogue state tracking," *ACL*, 2017.

[7] Jeffrey Pennington, Richard Socher, and Christopher Manning, "Glove: Global vectors for word representation," in *Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1532–1543.

[8] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *Computer Science*, 2014.

[9] Fei Liu and Julien Perez, "Dialog state tracking, a machine reading approach using memory network," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, 2017, pp. 305–314.

[10] John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth, "From paraphrase database to compositional paraphrase model and back," *Computer Science*, pp. 98–104, 2015.

[11] Matthew Henderson, Blaise Thomson, and Steve Young, "Word-based dialog state tracking with recurrent neural networks," in *Meeting of the Special Interest Group on Discourse and Dialogue*, 2014, pp. 292–299.

[12] Matthew Henderson, Blaise Thomson, and Steve Young, "Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation," in *Spoken Language Technology Workshop*, 2015, pp. 360–365.

[13] Miroslav Vodoln, Rudolf Kadlec, and Jan Kleindienst, "Hybrid dialog state tracker with asr features," in *Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017, pp. 205–210.

[14] Takaaki Hori, Hai Wang, Chiori Hori, Shinji Watanabe, Bret Harsham, Jonathan Le Roux, John R. Hershey, Yusuke Koji, Yi Jing, and Zhaocheng Zhu, "Dialog state tracking with attention-based sequence-to-sequence learning," in *Spoken Language Technology Workshop*, 2017.

[15] Puyang Xu and Ruhi Sarikaya, "Convolutional neural network based triangular crf for joint intent detection and slot filling," in *Automatic Speech Recognition and Understanding*, 2014, pp. 78–83.

[16] Bing Liu and Ian Lane, "Recurrent neural network structured output prediction for spoken language understanding," in *Proc. NIPS Workshop on Machine Learning for Spoken Language Understanding and Interactions*, 2015.