An End-to-End Trainable Neural Network Model with Belief Tracking for Task-Oriented Dialog

Bing Liu[1], *Ian Lane*[1,2]

[1]Electrical and Computer Engineering, Carnegie Mellon University
[2]Language Technologies Institute, Carnegie Mellon University

liubing@cmu.edu, lane@cmu.edu

Abstract

We present a novel end-to-end trainable neural network model for task-oriented dialog systems. The model is able to track dialog state, issue API calls to knowledge base (KB), and incorporate structured KB query results into system responses to successfully complete task-oriented dialogs. The proposed model produces well-structured system responses by jointly learning belief tracking and KB result processing conditioning on the dialog history. We evaluate the model in a restaurant search domain using a dataset that is converted from the second Dialog State Tracking Challenge (DSTC2) corpus. Experiment results show that the proposed model can robustly track dialog state given the dialog history. Moreover, our model demonstrates promising results in producing appropriate system responses, outperforming prior end-to-end trainable neural network models using per-response accuracy evaluation metrics.

Index Terms: spoken dialog systems, end-to-end model, task-oriented, dialog state tracking, language understanding

1. Introduction

Task-oriented spoken dialog system is a prominent component in today's virtual personal assistants, which enable people to perform everyday tasks by interacting with devices via voice input. Traditional task-oriented dialog systems have complex pipelines, with a number of independently developed and modularly connected components. There are usually separated modules in a pipeline for natural language understanding (NLU), dialog state tracking (DST), dialog management (DM), and natural language generation (NLG) [1, 2, 3, 4]. One limitation with such pipeline approach is that it is inherently hard to adapt a system to new domains, as all these modules are trained and fine-tuned independently. Moreover, errors made in upper stream modules may propagate to downstream components, making it tedious to identify and track the source of error [5].

To address these limitations, efforts have been made recently in designing end-to-end frameworks for task-oriented dialogs. Wen et al. [6] proposed an end-to-end trainable neural network model with modularly connected neural networks for each system component. Zhao and Eskenazi [5] introduced an end-to-end reinforcement learning framework that jointly performs dialog state tracking and policy learning. Li et al. [7] proposed an end-to-end learning framework that leverages both supervised and reinforcement learning signals and showed promising dialog modeling performance. Such end-to-end trainable neural network models can be optimized directly towards the final system objective functions (e.g. task success rate) and thus ameliorate the challenges of credit assignment and online adaptation [5].

In this work, we present an end-to-end trainable neural network model for task-oriented dialog that applies a unified network for belief tracking, knowledge base (KB) operation, and response creation. The model is able to track dialog state, interface with a KB, and incorporate structured KB query results into system responses to successfully complete task-oriented dialogs. We show that our proposed model can robustly track belief state given the dialog history. Our model also demonstrates promising performance in providing appropriate system responses and conducting task-oriented dialogs compared to prior end-to-end trainable neural network models.

2. Related Work

2.1. Dialog State Tracking

In spoken dialog systems, dialog state tracking, or belief tracking, refers to the task of maintaining a distribution over possible dialog states which directly determine the systems actions. Dialog state tracker is a core component in many state-of-the-art task-oriented spoken dialog systems [6, 8]. Conventional approaches for DST include using rule-based systems and generative methods that model the dialog as a dynamic Bayesian network [9]. Discriminative approaches using sequence models such as CRF [10] or RNN [11, 12] address the limitation of generative models with the flexibility in exploring arbitrary features [13] and achieve state-of-the-art DST performance.

2.2. End-to-End Task-Oriented Dialog Models

Conventional task-oriented dialog systems typically require a large number of domain-specific rules and handcrafted features, which make it hard to extend a good performing model to new application domains. Recent approaches to task-oriented dialog cast the task as a partially observable Markov Decision Process (POMDP) [4] and use reinforcement learning for online policy optimization by interacting with users [14]. The dialog state and system action spaces have to be carefully designed in order to make the reinforcement policy learning tractable [4].

With the success of end-to-end trainable neural network models in non-task-oriented chit-chat dialog settings [15, 16], efforts have been made in carrying over the good performance of end-to-end trainable models to task-oriented dialogs. Wen et al. [6] proposed a neural network based model that is end-to-end trainable yet still modularly connected. The model has separated modules for intent estimation, belief tracking, policy learning, and response generation. Our model, on the other hand, use a unified network for belief tracking, KB operation, and response generation, to fully explore knowledge that can be shared among different tasks. Bordes and Weston [17] recently proposed modeling the dialog with a reasoning approach using end-to-end memory network. Their model selects a best system response directly from a list of response candidates without explicitly tracking dialog state. Comparing to this approach, our

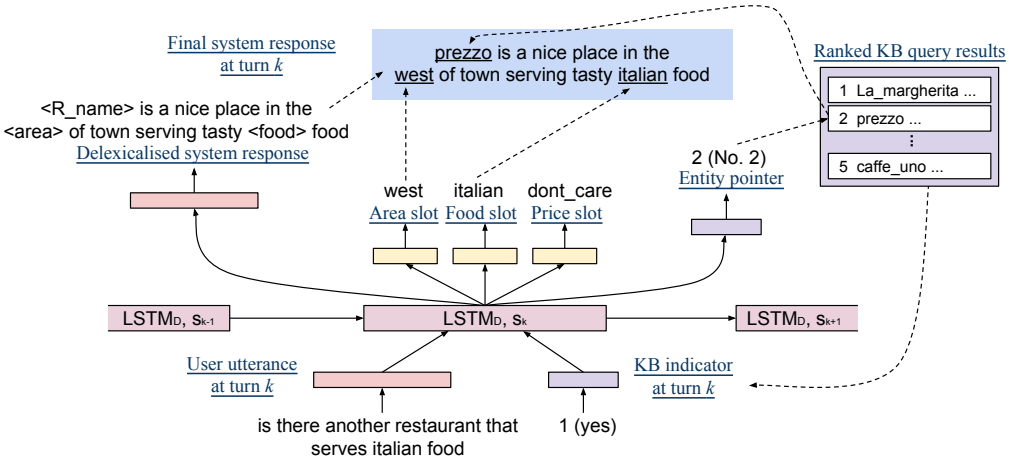

Figure 1: *System architecture of the proposed end-to-end trainable neural network model for task-oriented dialog.*

model tracks dialog state over the sequence of turns explicitly, as it is shown in [18] that robust dialog state tracking is likely to boost success rate in task completion. Moreover, when generating final system response, instead of letting the model to select a final response directly from a large pool of candidate responses, we let our model to select skeletal sentence structure from a short list of candidates and then replace the delexicalised tokens with the state tracking outputs. This method will help to reduce the number of training samples required [11] and make the model more robust to noises in dialog state.

3. Proposed Method

We model task-oriented dialog as a multi-task sequence learning problem, with components for encoding user input, tracking belief state, issuing API calls, processing KB results, and generating system responses. The model architecture is as shown in Figure 1. Sequence of turns in a dialog is encoded using LSTM [19] recurrent neural networks. Conditioning on the dialog history, state of the conversation is maintained in the LSTM state. The LSTM state vector is used to generate: (1) a skeletal sentence structure by selecting from a list of delexicalised system response candidates, (2) a probability distribution of values for each slot in belief tracker, and (3) a pointer to an entity in the retrieved KB results that matches the user's query. The final system response is generated by replacing the delexicalised tokens with the predicted slot values and entity attribute values. Each model component is described in detail in below sections.

3.1. Utterance Encoding

Utterance encoding here refers to encode a sequence of words into a continuous dense vector. Popular methods include using bag-of-means on word embeddings and RNNs [20, 21]. We use bidirectional LSTM to encode the user input to an utterance vector. Let $\mathbf{U}_k = (w_1, w_2, ..., w_{T_k})$ be the user input at the kth turn with T_k words. The user utterance vector U_k is represented by: $U_k = [\overrightarrow{h_{T_k}^{U_k}}, \overleftarrow{h_1^{U_k}}]$, where $\overrightarrow{h_{T_k}^{U_k}}$ and $\overleftarrow{h_1^{U_k}}$ are the last forward and backward utterance-level LSTM states at kth turn.

3.2. Belief Tracking

Belief tracking, or dialog state tracking, maintains and adjusts the state of a conversation, such as user's goals, by accumulate evidence along the sequence of a dialog. After collecting new evidence from a user's input at turn k, the neural dialog model updates the probability distribution $P(S_k^m)$ over candidate values for each slot type $m \in M$. For example, in restaurant search domain, the model maintains a multinomial probability distribution over each of user's goals on restaurant area, food type, and price range. At turn k, the dialog-level LSTM (LSTM_D) updates its hidden state s_k and use it to infer any updates on user's goals after taking in the user input encoding U_k and KB indicator I_k (to be described in section below).

$$s_k = \text{LSTM}_\text{D}(s_{k-1},\ [U_k,\ I_k]) \quad (1)$$

$$P(S_k^m \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}) = \text{SlotDist}_m(s_k) \quad (2)$$

where SlotDist_m is a multilayer perceptron (MLP) with softmax activation function over the slot type $m \in M$.

3.3. Issuing API Calls

Conditioning on the state of the conversation, the model may issue an API call to query the KB based on belief tracking outputs. A simple API call command template is firstly generated by the model. The final API call command is produced by replacing the slot type tokens in the command template with the best hypothesis for each of the goal slot from the belief tracker.

In restaurant search domain, a simple API call command template can be "$api\_call\ \langle area \rangle\ \langle food \rangle\ \langle pricerange \rangle$", and the slot type tokens are to be replaced with the belief tracker outputs to form the final API call command "$api\_call\ west\ italian\ dontcare$".

3.4. KB Results Processing

Once the neural dialog model receives the KB query results, it suggests options to users by selecting entities from the returned list. Instead of treating KB results as unstructured text (more specifically, as user utterances as in [17, 22, 23]) and processing them with machine reading comprehension approach, we treat KB results as a list of structured entities and let the model to select appropriate entity pointers. Outputs from KB search or database query typically have well defined structures, with entity attributes associated with entity index. Other than letting the model to learn such entity-attribute association purely from the training dialog corpus as in [17, 22, 23], we keep such

structural information in our system and let the model to learn to select proper entity pointers from a ranked list.

At turn k of a dialog, a binary KB indicator $\mathbf{I}_k$ is passed to the neural dialog model. This indicator is decided by the number of retrieved entities from the last API call and the current entity pointer. When the system is in a state to suggest an entity to user, if a zero value $\mathbf{I}_k$ is received, the model is likely to inform user the unavailability of entity matching the current query. Otherwise if $\mathbf{I}_k$ has a value of one, the model will likely pick an entity from the retrieved results based on the updated probability distribution of the entity pointer $P(E_k)$:

$$P(E_k \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}) = \mathrm{EntityPointerDist}(s_k) \quad (3)$$

where $\mathrm{EntityPointerDist}$ is an MLP with $\mathrm{softmax}$ activation.

3.5. System Response Generation

At kth turn of a dialog, a skeletal sentence structure R_k is selected from a list of delexicalised response candidates. The final system response is produced by replacing the delexicalised tokens with the predicted slot values and entity attribute values. For example, replacing $\langle food \rangle$ to *italian*, and replacing $\langle R\_name \rangle$ to *prezzo* as in Figure 1.

$$P(R_k \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}) = \mathrm{ResponseDist}(s_k) \quad (4)$$

where $\mathrm{ResponseDist}$ is an MLP with $\mathrm{softmax}$ activation.

3.6. Model Training

We train the neural dialog model by finding the parameter set θ that minimize the cross-entropy of the predicted and true distributions for goal slot labels, entity pointer, and delexicalised system response jointly:

$$\begin{aligned} \min_{\theta} \sum_{k=1}^{K} -\Big[& \sum_{m=1}^{M} \lambda_{S^m} \log P(S_k^{m*} \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}; \theta) \\ & + \lambda_E \log P(E_k^* \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}; \theta) \\ & + \lambda_R \log P(R_k^* \mid \mathbf{U}_{\leq k},\ \mathbf{I}_{\leq k}; \theta) \Big] \end{aligned} \quad (5)$$

where λs are the linear interpolation weights for the cost of each system output. S_k^{m*}, E_k^*, and R_k^* are the ground truth labels for each task at the kth turn.

3.7. Alternative Model Designs

The model architecture (Figure 1) described above assumes that the hidden state of the dialog-level LSTM implicitly captures the complete state of the conversation, i.e. the user goal estimation and the previous system actions. Intuitively, the model is likely to provide a better response if it is informed about the goal slot value estimations explicitly and is aware of its previous responses made to the user. Thus, we design and evaluate a few alternative model architectures to verify such assumption:

(1) Model with previously emitted delexicalised system response connected back to dialog-level LSTM state:

$$s_k = \mathrm{LSTM_D}(s_{k-1},\ [U_k,\ I_k,\ R_{k-1}]) \quad (6)$$

(2) Model with previously emitted slot labels connected back to dialog-level LSTM state:

$$s_k = \mathrm{LSTM_D}(s_{k-1},\ [U_k,\ I_k,\ S_{k-1}^1, ..., S_{k-1}^M]) \quad (7)$$

(3) Model with both previously emitted response and slot labels connected back to dialog-level LSTM state:

$$s_k = \mathrm{LSTM_D}(s_{k-1},\ [U_k,\ I_k,\ R_{k-1},\ S_{k-1}^1, ..., S_{k-1}^M]) \quad (8)$$

4. Experiments

4.1. Dataset

We use data from DSTC2 [24] for our model evaluation. This challenge is designed in the restaurant search domain. Bordes and Weston [17] transformed the original DSTC2 corpus by adding system commands and removing the dialog state annotations. This transformed corpus contains additional API calls that the system would make to the KB and the corresponding KB query results. In this study, we combine the original DSTC2 corpus and this transformed version by keeping the dialog state annotations and adding the system commands (API calls). We can thus perform more complete evaluation of our model's capability in tracking the dialog state, processing KB query results, and conducting complete dialog. Statistics of this dataset is summarized in the Table 1.

Table 1: *Statistics of the converted DSTC2 dataset.*

Num of train & dev / test dialogs	2118 / 1117
Num of turns per dialog in average (including API call commands)	7.9
Num of area / food / pricerange options	5 / 91/ 3
Num of delexicalised response candidates	78

4.2. Model Configuration and Training

We perform mini-batch model training with batch size of 32 using Adam optimization method [25]. Regularization with dropout is applied to the non-recurrent connections [26] during model training with dropout rate of 0.5. We set the maximum norm for gradient clipping to 5 to prevent exploding gradients.

Hidden layer sizes of the dialog-level LSTM and the utterance-level LSTM are set as 200 and 150 respectively. Word embeddings of size 300 are randomly initialized. We also explore using pre-trained word vectors [27] that are trained on Google News dataset to initialize the word embeddings.

4.3. Results and Analysis

Similar to the evaluation methods used in [17, 22, 23], we evaluate the task-oriented dialog model in a ranking setting. We report the prediction accuracy for goal slot values, entity pointer, delexicalised system response, and final system response which has the delexicalised tokens replaced by predicted values.

We first experiment with different text encoding methods and recurrent model architectures to find best performing model. Table 2 shows the evaluation results of models using different user utterance encoding methods and different word embedding initialization. Bidirectional LSTM (Bi-LSTM) shows clear advantage in encoding user utterance comparing to bag-of-means on word embedding (BoW Emb) method, improving the joint goal prediction accuracy by 4.6% and the final system response accuracy by 1.4%. Using pre-trained word vectors (word2vec) boosts the model performance further. These results show that the semantic similarities of words captured in the pre-trained word vectors are helpful in generating a better representation of user input, especially when the utterance contains words or entities that are rarely observed during training.

Table 3 shows the evaluation results of different recurrent model architectures. The Hierarchical LSTM model in Table 3 refers to the last model in Table 2. Models in row 2 to 4 of Table 3 refer to the three recurrent model architectures dis-

Table 2: *Prediction accuracy for entity pointer, joint user goal, delexicalised system response, and final system response of the transformed DSTC2 test set using different encoding methods and word vector initialization.*

Model	Entity Pointer	Joint Goal	De-lex Res	Final Res
BoW Emb Encoder	93.5	72.6	55.4	51.2
+ word2vec	93.6	74.3	55.9	51.5
Bi-LSTM Encoder	93.8	**77.2**	55.8	52.6
+ word2vec	**94.4**	76.6	**56.6**	**52.8**

cussed in section 3.7. As illustrated in these results, models that conditioned on previous emitted labels in generating system response achieve lower prediction accuracy across all of the four evaluation metrics. These observations are contrary to our intuition and analysis made in previous section. We believe the degraded performance is mainly due to the data sparsity issue of the dataset that used in the experiment. Given a certain dialog context, there might be multiple system response candidates that can be used to generate a suitable final response. With limited number of training samples, the model is likely to overfit the training set and not to generalize the strong modeling capacity well during inference. The overfitting issue might be less of a problem in the word-by-word response generation setting, and this is to be studied further in our future work.

Table 3: *Prediction accuracy with different recurrent model architectures.*

Model	Entity Pointer	Joint Goal	De-lex Res	Final Res
Hierarchical LSTM	**94.4**	**76.6**	**56.6**	**52.8**
+ feed de-lex res (1)	93.6	74.8	55.4	51.8
+ feed goal slots (2)	94.1	75.3	55.3	51.8
+ feed both (3)	93.7	72.7	55.3	51.6

As observed in Table 3, the joint goal tracking performance is directly related to final response prediction accuracy. To further investigate our model's capability on belief tracking, we conduct error breakdown analysis for each goal slot. We compare our model to two other recently proposed belief tracking models, an RNN based model [11] and the Neural Belief Tracker [28], in the setting of only using live ASR hypothesis as model input. As the results show in Table 4, our system achieves promising belief tracking performance comparable to the state-of-the-art systems.

Table 4: *Dialog state tracking performance on DSTC2 test set, comparing to previous approaches.*

Model	Area Goal	Food Goal	Price Goal	Joint Goal
RNN	92	86	86	69
RNN + sem. dict	91	86	93	73
NBT-DNN [28]	90	84	94	72
NBT-CNN [28]	90	83	93	72
Hierarchical LSTM	90	84	93	73

Finally, we report the model performance in producing final system responses and compare it to other published results following the per-response accuracy metric used in prior work. Even though using the same evaluation measurement, our model is designed with slightly different settings comparing to other published models in Table 5. Instead of using additional matched type features (i.e. KB entity type feature for each word, e.g. whether a word is a food type or area type, etc.) as in [17, 23], we use user's goal slots at each turn that are mapped from the original DSTC2 dataset as additional supervised signals in our model. Moreover, instead of treating KB query results as unstructured text, we treat them as structured entities and let our model to pick the right entity by selecting the most appropriate entity pointer. Our proposed model successfully predicts 52.8% of the true system responses, outperforming prior end-to-end trainable neural dialog systems.

Table 5: *Performance of the proposed model in per-response accuracy comparing to previous approaches.*

Model	Per-res Accuracy
Memory Networks [17]	41.1
Gated Memory Networks [29]	48.7
Sequence-to-Sequence [22]	48.0
Query-Reduction Networks [23]	51.1
Hierarchical LSTM	**52.8**

To further understand the prediction errors made by our model, we conduct human evaluation by inviting 10 users to evaluate the appropriateness of the responses generated by our system. While some of the errors are made on generating proper API calls due to the errors in dialog state tracking results, we also find quite a number of responses that are considered appropriate by our judges but do not match to the reference responses in the test set. For example, there are cases where our system directly issues the correct API call (e.g. "$api\_call\ south\ italian\ expensive$") based on user's inputs, instead of asking user for confirmation of a goal type (e.g. "*Did you say you are looking for a restaurant in the south of town?*") as in the reference corpus. By taking such factors into consideration, our system was able to generate appropriate responses in 73.6% of the time based on feedback from the judges. These results show that the per-response accuracy evaluation metric may not well correlate with human judgments [30], and better dialogs evaluation measurements should to be further explored.

5. Conclusions

In this work, we propose a novel end-to-end trainable neural network model for task-oriented dialog systems. The model is able to track dialog belief state, interface with knowledge bases by issuing API calls, and incorporate structured query results into system responses to successfully complete task-oriented dialogs. In the evaluation in a restaurant search domain using a converted dataset from the second Dialog State Tracking Challenge corpus, our proposed model shows robust performance in tracking dialog state over the sequence of dialog turns. The model also demonstrates promising performance in generating appropriate system responses, outperforming prior end-to-end trainable neural network models.

6. References

[1] A. I. Rudnicky, E. H. Thayer, P. C. Constantinides, C. Tchou, R. Shern, K. A. Lenzo, W. Xu, and A. Oh, "Creating natural dialogs in the carnegie mellon communicator system." in *Eurospeech*, 1999.

[2] S. Young, "Using pomdps for dialog management," in *Spoken Language Technology Workshop, 2006. IEEE*. IEEE, 2006, pp. 8–13.

[3] A. Raux, B. Langner, D. Bohus, A. W. Black, and M. Eskenazi, "Lets go public! taking a spoken dialog system to the real world," in *in Proc. of Interspeech 2005*. Citeseer, 2005.

[4] S. Young, M. Gašić, B. Thomson, and J. D. Williams, "Pomdp-based statistical spoken dialog systems: A review," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1160–1179, 2013.

[5] T. Zhao and M. Eskenazi, "Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning," in *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Los Angeles: Association for Computational Linguistics, September 2016, pp. 1–10. [Online]. Available: http://www.aclweb.org/anthology/W16-3601

[6] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gašić, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young, "A network-based end-to-end trainable task-oriented dialogue system," *arXiv preprint: 1604.04562*, April 2016.

[7] X. Li, Y.-N. Chen, L. Li, and J. Gao, "End-to-end task-completion neural dialogue systems," *arXiv preprint arXiv:1703.01008*, 2017.

[8] J. Williams, A. Raux, and M. Henderson, "The dialog state tracking challenge series: A review," *Dialogue & Discourse*, vol. 7, no. 3, pp. 4–33, 2016.

[9] J. D. Williams and S. Young, "Partially observable markov decision processes for spoken dialog systems," *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.

[10] S. Lee, "Structured discriminative model for dialog state tracking," in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 442–451.

[11] M. Henderson, B. Thomson, and S. Young, "Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised gate," in *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 2014, pp. 360–365.

[12] M. Henderson, B. Thomson, and S. Young, "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)*, 2014, pp. 292–299.

[13] M. Henderson, "Machine learning for dialog state tracking: A review," in *Proc. of The First International Workshop on Machine Learning in Spoken Language Processing*, 2015.

[14] M. Gašić, C. Breslin, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis, and S. Young, "On-line policy optimisation of bayesian spoken dialogue systems via human interaction," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 8367–8371.

[15] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364*, 2015.

[16] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," *arXiv preprint arXiv:1507.04808*, 2015.

[17] A. Bordes and J. Weston, "Learning end-to-end goal-oriented dialog," *arXiv preprint arXiv:1605.07683*, 2016.

[18] F. Jurčíček, B. Thomson, and S. Young, "Reinforcement learning for parameter estimation in statistical spoken dialogue systems," *Computer Speech & Language*, vol. 26, no. 3, pp. 168–192, 2012.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," in *Proceedings of The 17th Annual Meeting of the International Speech Communication Association*, 2016.

[21] D. Hakkani-Tür, G. Tur, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, and Y.-Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm," in *Proceedings of The 17th Annual Meeting of the International Speech Communication Association*, 2016.

[22] M. Eric and C. D. Manning, "A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue," in *EACL*, 2017.

[23] M. Seo, S. Min, A. Farhadi, and H. Hajishirzi, "Query-reduction networks for question answering," in *International Conference on Learning Representations*, 2017.

[24] M. Henderson, B. Thomson, and J. Williams, "The second dialog state tracking challenge," in *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, vol. 263, 2014.

[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[27] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[28] N. Mrkšić, D. O. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," *arXiv preprint arXiv:1606.03777*, 2016.

[29] J. Perez and F. Liu, "Gated end-to-end memory networks," *arXiv preprint arXiv:1610.04211*, 2016.

[30] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation," *arXiv preprint arXiv:1603.08023*, 2016.