DIALOGUE STATE TRACKING WITH CONVOLUTIONAL SEMANTIC TAGGERS

Mandy Korpusik, James Glass

MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA 02139, USA
korpusik@mit.edu, glass@mit.edu

ABSTRACT

In this paper, we present our novel approach to the 6th Dialogue State Tracking Challenge (DSTC6) track for end-to-end goal-oriented dialogue, in which the goal is to select the best system response from among a list of candidates in a restaurant booking conversation. Our model uses a convolutional neural network (CNN) for semantic tagging of each utterance in the dialogue history to update the dialogue state, and another CNN for predicting the best system action template. Our model is competitive with the top two submissions to the challenge, achieving 100% precision on subtasks 1 and 2 with a CNN rather than an LSTM for action selection, and a CNN for slot-value tagging, instead of an LSTM or CRF.

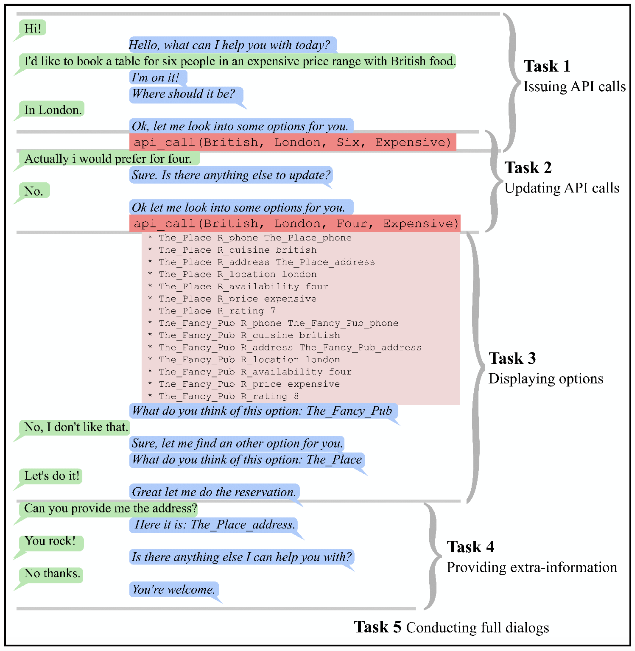
Index Terms— Dialogue State Tracking, Semantic Tagging, Convolutional Neural Network, Recurrent Neural Network, Conditional Random Field

1. INTRODUCTION

A critical component of spoken dialogue systems (SDS) is tracking the user's goal over the course of the conversation, also known as dialogue state tracking (DST). The system must update the state of the dialogue after each user query by determining their intent (e.g., making a restaurant reservation), and selecting the corresponding value for each slot specified by the user, such as `Chinese` for the `food` slot, and `city_center` for the `area` slot, in the restaurant domain. Given the current dialogue state, the agent can then decide how best to respond to the user to accomplish the desired task for him or her.

In this work, we specifically examine the 6th Dialogue State Tracking Challenge (DSTC6) [1] track with a corpus collected by Facebook AI Research for end-to-end goal-oriented dialogue. In this task, the goal is making a restaurant reservation for the user, given all their constraints on the location, cuisine, price range, atmosphere, and party size. This overall task is broken down into five subtasks: 1) issuing API calls, 2) updating API calls, 3) displaying options, 4) providing extra information, and 5) conducting full dialogues (see Fig. 1). This requires not only dialogue management, but also querying knowledge bases (KB) of restaurants to respond with the correct information to the user. The motivation for this challenge is to build end-to-end models that have the potential to scale up and generalize to new domains better than existing SDS methods, where the dialogue state is designed to be domain-dependent.

In particular, we focus on the first two tasks in this paper, as in [3], since tasks three and four can be simply handled by KB lookup. We compare our convolutional neural network (CNN) approach to the top-2 DSTC6 participants, since they achieve 100% top-1 precision (P@1) on the test set for all subtasks. We demonstrate that our CNN technique is competitive, reaching 100% P@1 on subtasks 1 and 2, without requiring any LSTMs, as were used in both the top submissions. In addition, our approach uses a CNN semantic tagger that we developed in previous work for tagging natural language meal descriptions [4], which we establish is *generalizable* to other tasks and domains, such as restaurant booking in DSTC6, *without requiring any task-specific hyperparameter fine-tuning*.



Hi!

Hello, what can I help you with today?

I'd like to book a table for six people in an expensive price range with British food.

I'm on it!

Where should it be?

In London.

Ok, let me look into some options for you.

`api_call(British, London, Six, Expensive)`

Actually i would prefer for four.

Sure. Is there anything else to update?

No.

Ok let me look into some options for you.

`api_call(British, London, Four, Expensive)`

```
* The_Place R_phone The_Place_phone
* The_Place R_cuisine british
* The_Place R_address The_Place_address
* The_Place R_location london
* The_Place R_availability four
* The_Place R_price expensive
* The_Place R_rating 7
* The_Fancy_Pub R_phone The_Fancy_Pub_phone
* The_Fancy_Pub R_cuisine british
* The_Fancy_Pub R_address The_Fancy_Pub_address
* The_Fancy_Pub R_location london
* The_Fancy_Pub R_availability four
* The_Fancy_Pub R_price expensive
* The_Fancy_Pub R_rating 8
```

What do you think of this option: The_Fancy_Pub

No, I don't like that.

Sure, let me find an other option for you.

What do you think of this option: The_Place

Let's do it!

Great let me do the reservation.

Can you provide me the address?

Here it is: The_Place_address.

You rock!

Is there anything else I can help you with?

No thanks.

You're welcome.

Task 1 Issuing API calls
Task 2 Updating API calls
Task 3 Displaying options
Task 4 Providing extra-information
Task 5 Conducting full dialogs

Fig. 1. An illustration of the five subtasks in the DSTC6 track for end-to-end goal-oriented dialogue [2].

2. RELATED WORK

Semantic Tagging A substantial body of work has been devoted to spoken language understanding (SLU) and, in particular, semantic tagging of user utterances in restaurant and food domains. Early work in SLU explored generative and discriminative models for the well-known Air Travel and Information System (ATIS) corpus [5], including finite-state transducers (FSTs), support vector machines (SVMs), and conditional random field models (CRFs) [6]. Other work boosted the performance of statistical models by extracting keywords based on a dependency parse of the user utterance [7]. More recently, deep learning models have been used, such as recur-

This research was sponsored by a grant from Quanta Computing, Inc., and by the Department of Defense (DoD) through the National Defense Science Engineering Graduate Fellowship (NDSEG) Program.

rent neural networks (RNNs) [8] and bidirectional RNNs for joint slot filling with domain and intent classification [9].

For the restaurant domain, prior state-of-the-art involved complicated CRFs with carefully hand-crafted features, such as semantic dependency features from query dependency parses [10], and word vector and distributional prototype similarity features [11] were used in CRFs for semantic tagging of meal descriptions. More recent work on the food domain has applied CNNs, outperforming the CRFs on the spoken test set without manual feature engineering [4].

Dialogue State Tracking Traditionally, spoken dialogue systems relied on separately trained components for spoken language understanding (SLU) and dialogue state tracking. The SLU component would identify slot-value pairs from the speech recognition output, which would be passed to the state tracking module to update the belief state [12, 13]. However, this pipeline of steps would accumulate errors, as the SLU component often would not have the necessary context to accurately predict the slot values. Thus, belief tracking research shifted to jointly predicting slot-value pairs and updating the dialogue state [14, 15].

Prior work by *Henderson et al.* fed delexicalized user utterances into an RNN, which output a distribution over slot values [16]. However, delexicalizing the input requires a manually defined semantic dictionary that maps from slot-value pairs to all possible text forms. To avoid this reliance on hand-crafted semantic dictionaries, *Mrksic et al.* recently demonstrated the ability of their Neural Belief Trackers (NBT) [17] to match the performance of delexicalization-based models, without requiring any hand-crafted semantic dictionaries, as well as the ability to significantly outperform such models when the semantic resources are not available. In addition, *Zhong et al.'s* state-of-the-art work has explored deep learning methods for dialogue state tracking, but with recurrent (instead of convolutional) self-attentive encoders [18]. *Rastogi et al.* also use RNNs in their multi-domain deep learning model for state tracking [19], and *Korpusik et al.* use CNNs without any pre-trained word vectors [20].

For the DSTC6 challenge, the top-2 performing methods each used separately trained slot-value trackers to refine the initial ranking of system response candidates generated by an action selector [21, 3]. We take a similar approach, but use a CNN instead of a long short-term memory (LSTM) network to select system actions. We also use a CNN for semantic tagging, rather than the CRF used in *Bai et al.* [3] or the LSTM in *Ham et al.* [21] for slot-value tracking, and we use less feature engineering than these systems. *Bai et al.* use a heuristic strategy for the final scoring, with two branches based on the preliminary scoring module (e.g., if `api_call` is selected, then update a candidate response's score based on the relative index of the word that last occurred in the dialogue history, using weights according to the output of a separately trained "Uncertainty CRF").

3. CNN DIALOGUE STATE TRACKER

3.1. Baselines

We compare against the baselines used by *Bai et al.* [3]: ranking candidate system responses randomly, according to tf-idf values, with a support vector machine (SVM), with a vanilla LSTM, and with a hierarchical LSTM. We also implement a binary CNN model with a sigmoid output layer (see Fig. 2) that predicts whether each candidate is a good system response, given the input user utterance and dialogue history. To get the final ranking, we use the sigmoid output probability for each candidate.

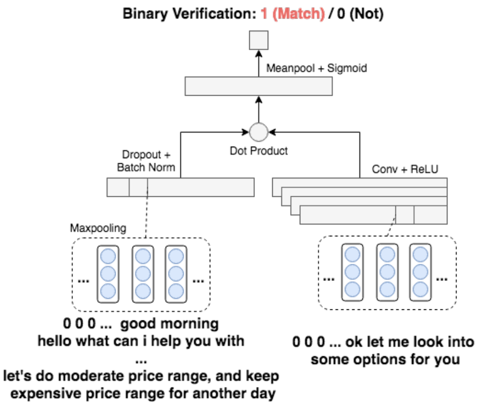

Binary Verification: 1 (Match) / 0 (Not)
Meanpool + Sigmoid
Dot Product
Dropout + Batch Norm
Conv + ReLU
Maxpooling
...
...
...
...
0 0 0 ... good morning hello what can i help you with ... let's do moderate price range, and keep expensive price range for another day
0 0 0 ... ok let me look into some options for you

Fig. 2. The binary CNN baseline that predicts whether each candidate system response is the correct answer. There are two inputs to the network: the full dialogue history concatenated with the last user utterance on the left, and a candidate system response on the right.[2]

3.2. CNN Architecture

Our best model pushes us closer to *generating* the next system response, rather than simply selecting the best response from among the list of candidates. Our full system architecture, shown in Fig. 3, consists of one CNN for semantic tagging and updating the dialogue state (see Fig. 4 for an example utterance with its tags), another CNN for action selection (see Fig. 5), and a final response generation step filling in the template with the slot values in the final dialogue state.

3.2.1. Semantic Tagging

As in our prior work, where we used CNNs for semantic tagging on meal descriptions for a diet tracking application [22, 23, 24, 25], we applied a CNN to semantic tagging of the user utterances in the restaurant booking conversations. We select a CNN here rather than a CRF because it has demonstrated comparable performance in our previous semantic tagging work [4], without requiring any feature engineering. We choose it over the more typical LSTM due to its faster training time.[3] We also illustrate the interpretability aspect of the learned convolutional filters in Table 5, by identifying patterns among the tokens that have the highest activation from the filters.

The CNN tagger is composed of a word embedding layer followed by three stacked 1D convolutional layers, with kernel windows spanning lengths of five, five, and three tokens, respectively. It learned 150-dimension embeddings without using pre-trained word vectors, used 64 filters per convolutional layer, applied ReLU activation, and trained with the Adam optimizer on cross-entropy loss for up to 15 epochs with early stopping determined by no performance gain on the validation set (20% split). We trained a separate tagger for each of the two subtasks we evaluated in the DSTC6 challenge, since that performed better than jointly training a tagger on both.

To convert the DSTC6 data to training data for semantic tagging, we searched for `api_call` utterances (see Table 1) within the list of utterances for each dialogue, since that provided us with the gold standard value for each slot. Each `api_call` has the following order: `cuisine`, `location`, `number`, `price`, `atmosphere`. For example, a possible utterance might be `api_call italian paris four cheap romantic`. We then found exact string

[2] We pad input tokens with zero to the maximum utterance length seen in training, apply 0.1 dropout, and perform batch norm after maxpooling.

[3] Training the LSTM takes $>$ 5x longer than training the CNN on Task 2.

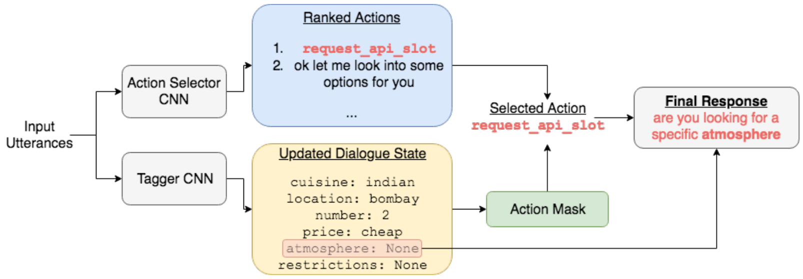


Fig. 3. The full system architecture, where the action selector CNN ranks the 15 possible system action templates; the tagger CNN updates the dialogue state by tagging each utterance in the dialogue history; and the final response is generated by selecting the top-ranked action, applying an action mask, and populating slots with dialogue state values. Here, since there are missing slot values, the `api_call` action is masked out, and the next slot missing a value is `atmosphere`, so it is requested by the system as the final response.

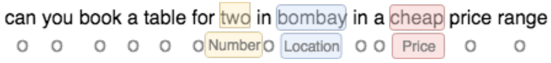

can you book a table for two in bombay in a cheap price range
O O O O O O Number O Location O O Price O O

Fig. 4. A sample user utterance with its color-coded semantic tags (i.e., number in the party, restaurant location, and price range).

matches for each of the slot values among the previous utterances, labeling each utterance with that slot type (e.g., `food` for each token `paris`) and the remaining tokens with the other semantic tag `O`. At test time, we ran each user utterance in the dialogue history for the DSTC6 challenge through the trained CNN tagger, updating the dialogue state each time a new slot was identified, using the tagged token as the value for that slot (e.g., `two` is the value specified for the `number` slot in the example utterance shown in Fig. 4).

3.2.2. Action Selection

The second CNN in our system is trained to predict the best candidate system response template from among the 15 possible options in Table 1. As shown in Fig. 5, this CNN takes only one input (the full dialogue history of utterances concatenated together) and feeds it to a learned word embedding layer of 64 dimensions. The embeddings for each token are fed through a 1D convolutional layer with a window size of three tokens, ReLU activation, and 64 filters. The output of this layer is then maxpooled before passing through a final feed-forward layer with a softmax activation to output a probability for each of the 15 possible candidate system response templates.

3.2.3. Final Response Generation

After the dialogue state is updated by feeding each utterance through the CNN tagger, and the possible system actions are ranked by the CNN action selector, an action mask is applied (as in *Ham et al.* [21]), and the action templates are populated with slot values from the dialogue state. The action mask is formed based on the dialogue state—if any slot values are not yet specified, the `api_call` action is masked out, and if all slot values are specified, the `request_api_slot` action is masked out. In the final step, if the `api_call` action is selected, the values are populated using the current dialogue state; likewise, if the `request_api_slot` action is selected, the system response for the next slot that is still missing its value (again according to the current dialogue state) is chosen (see Table 2 for the responses generated for each requested slot).[4]

Action Template
ok let me look into some options for you
`api_call`
i'm on it
hello what can i help you with today
sure is there anything else to update
you're welcome
what do you think of this option:
great let me do the reservation
sure let me find another option for you
here it is
whenever you're ready
the option was
i am sorry i don't have an answer to that question
is there anything i can help you with
`request_api_slot`

Table 1. The 15 possible system action templates. The `api_call` action is populated with the values for each of the slots (i.e., cuisine, location, number, price range, atmosphere) in the current dialogue state, while the `request_api_slot` template is mapped to a response for the next missing slot the system needs to call the API.

4. DSTC6 EXPERIMENTS

4.1. Data

The DSTC6 dialogue data we used is an extension of the bAbI dialogue data in *Bordes and Weston* [2], where the dialogues for restaurant reservation are generated through simulation based on a knowledge base (KB) of restaurants.[5] Each restaurant is specified by a cuisine (e.g., French), location (e.g., Tokyo), price range (e.g., expensive), atmosphere, and dietary restrictions. Each restaurant has a party size option of 2, 4, 6, or 8, and a phone number and address. There are 10,000 example dialogues in the training set for each subtask, for which we train tagger and action selector CNNs *separately*.

We report precision @ $\{1, 2, 5\}$ as our evaluation metric (i.e., the number of times in the test set that the correct system response candidate is ranked first, in the top-2, and among the top-5 responses, respectively). In our preliminary experiments, we evaluate on the first of four test sets, in which the knowledge base is the same between training and test, and where dietary restrictions are omitted.

[4] We use the deterministic order for requesting missing slots.

[5] See our prior work [20] for our CNN approach applied to the DSTC2 restaurant data from the University of Cambridge and Microsoft Research.

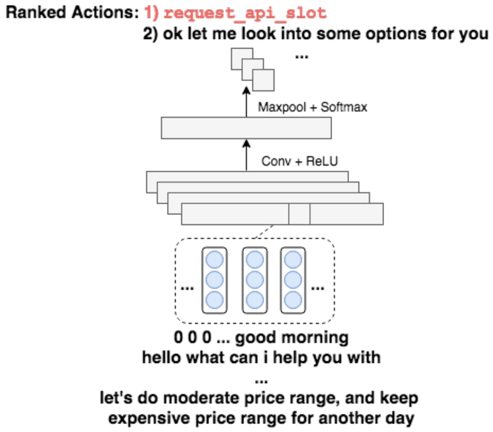

Ranked Actions: 1) request_api_slot
2) ok let me look into some options for you
...
Maxpool + Softmax
Conv + ReLU
...
...
0 0 0 ... good morning
hello what can i help you with
...
let's do moderate price range, and keep
expensive price range for another day

Fig. 5. The CNN for action selection, which takes as input the full dialogue history of utterances concatenated together, and outputs the probabilities for each possible system response with a softmax.

Slot	System Response
Cuisine	any preference on a type of cuisine
Location	where should it be
Number	how many people would be in your party
Price	which price range are you looking for
Atmosphere	are you looking for a specific atmosphere

Table 2. The system response returned for each of the six possible slots when the `request_api_slot` action template is chosen.

4.2. Results

As shown in Table 3, our model is competitive with the top participant [21] in the DSTC6 challenge, achieving 100% precision on the first test set for subtasks 1 and 2. Our binary CNN model is outperformed by the SVM, vanilla LSTM, and hierarchical LSTM in P@1, though it does better than the SVM in P@2 and outperforms both the SVM and LSTM in P@5. Note that the action mask is a critical piece of our system (due to predicting `api_call` with missing slots, and `request_api_slot` with all slots filled)—without it, P@1 drops from 100% to 91.1% on Task 1 (and remains 100% on Task 2). In Table 4, we show the results of the tagger on our automatically generated tagging data for subtask 1. We see examples of tricky user utterances in Fig. 6, where the model makes a mistake when there are two possible tokens for the same tag. In Table 5, we show the tokens that have the highest activations for the tagger's CNN filters.

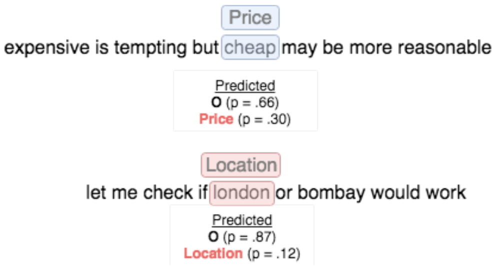

Price
expensive is tempting but cheap may be more reasonable
Predicted
O (p = .66)
Price (p = .30)
Location
let me check if london or bombay would work
Predicted
O (p = .87)
Location (p = .12)

Fig. 6. Two examples of semantic tagging errors, where the model incorrectly labels both `cheap` and `london` as `O`, rather than the correct tags `Price` and `Location`, respectively.

	Task 1			Task 2		
Model	P@1	P@2	P@5	P@1	P@2	P@5
Random	10.2	20.4	50.9	0.95	19.5	46.7
TFIDF	21.0	29.9	52.2	36.7	47.4	66.9
SVM	81.3	81.6	83.0	74.5	76.4	78.9
LSTM	84.3	90.6	98.5	77.8	84.0	97.8
Hier. LSTM	88.6	94.1	99.9	81.7	92.6	100
Bai et al.	99.8	100	100	99.7	100	100
Ham et al.	100	100	100	100	100	100
Binary CNN	78.9	88.9	99.7	69.0	79.3	99.6
Our Model	100	100	100	100	100	100

Table 3. We report the precision for each of the baseline methods, the top-2 submissions to the DSTC6 challenge [21, 3], our baseline binary CNN, and our final softmax CNN model.

Semantic Tag	Precision	Recall	F-score
Cuisine	100	96.9	98.4
Location	100	95.9	97.9
Number	100	100	100
Price	96.9	96.5	96.7
Atmosphere	100	100	100
All	99.8	99.8	99.8

Table 4. Precision, recall, and F-scores of our CNN semantic tagger on each of the semantic tags in the automatically generated tagging test set for the first subtask of DSTC6.

Filter	Top-3 Highest Activation Tokens
19	french, spanish, italian
52	two, six, four
63	bombay, london, paris

Table 5. Top-3 tokens with high activation for the learned filters in the semantic tagger's third CNN layer—filter 19 picks out cuisines, filter 52 isolates party numbers, and filter 63 identifies locations.

5. CONCLUSION

In this work, we have demonstrated that the CNN tagger we designed in previous work for semantic tagging of natural language meal descriptions [4] is general enough to be directly applied to the 6th Dialogue State Tracking Challenge (DSTC6) without requiring task-specific hyperparameter fine-tuning. Our model, which combines the CNN tagger with a CNN action selector, achieves 100% precision on subtasks 1 and 2 of the end-to-end goal-oriented dialogue track, and is competitive with the top challenge participants.

In future work, we will experiment on the remaining three subtasks (displaying options, providing extra information, and conducting full dialogues), as well as the other three test sets for each subtask. We could also add a feature to our CNN that indicates whether all the slots have been filled or not when predicting the action template, which should allow the network to automatically learn the action mask. Finally, we aim to jointly train the tagger and action selector CNNs as one fully end-to-end model.

6. REFERENCES

[1] Y. Boureau, A. Bordes, and J. Perez, "Dialog state tracking challenge 6 end-to-end goal-oriented dialog track," Tech. Rep., Tech. Rep, 2017.

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

[3] Z. Bai, B. Yu, G. Chen, B. Wang, and Z. Wang, "Modeling conversations to learn responding policies of e2e task-oriented dialog system," *Dial. Syst. Technol. Challenges*, vol. 6, 2017.

[4] M. Korpusik and J. Glass, "Spoken language understanding for a nutrition dialogue system," *IEEE Transactions on Audio, Speech, and Language Processing*, 2017.

[5] C. Hemphill, J. Godfrey, G. Doddington, et al., "The ATIS spoken language systems pilot corpus," in *Proceedings of the DARPA speech and natural language workshop*, 1990, pp. 96–101.

[6] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *Proceedings of the Eighth International Conference on Spoken Language Processing (Interspeech)*, 2007, pp. 1605–1608.

[7] G. Tur, D. Hakkani-Tür, L. Heck, and S. Parthasarathy, "Sentence simplification for spoken language understanding," in *Proceedings of the 2011 IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 5628–5631.

[8] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, et al., "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2015.

[9] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y. Chen, J. Gao, L. Deng, and Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM," in *Interspeech*, 2016, pp. 715–719.

[10] J. Liu, P. Pasupat, Y. Wang, S. Cyphers, and J. Glass, "Query understanding enhanced by hierarchical parsing structures," in *Proceedings of 2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2013, pp. 72–77.

[11] M. Korpusik, C. Huang, M. Price, and J. Glass, "Distributional semantics for understanding spoken meal descriptions," *Proceedings of 2016 IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[12] B. Thomson and S. Young, "Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems," *Computer Speech & Language*, vol. 24, no. 4, pp. 562–588, 2010.

[13] Z. Wang and O. Lemon, "A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information," in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 423–432.

[14] 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.

[15] K. Sun, L. Chen, S. Zhu, and K. Yu, "The SJTU system for dialog state tracking challenge 2," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014, pp. 318–326.

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

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

[18] V. Zhong, C. Xiong, and R. Socher, "Global-locally self-attentive dialogue state tracker," *arXiv preprint arXiv:1805.09655*, 2018.

[19] A. Rastogi, D. Hakkani-Tür, and L.y Heck, "Scalable multi-domain dialogue state tracking," in *Proceedings of 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 561–568.

[20] M. Korpusik and J. Glass, "Convolutional neural networks for dialogue state tracking without pre-trained word vectors or semantic dictionaries," in *Proceedings of 2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018.

[21] J. Ham, S. Lim, and K. Kim, "Extended hybrid code networks for dstc6 fair dialog dataset," *Dial. Syst. Technol. Challenges*, vol. 6, 2017.

[22] M. Korpusik, N. Schmidt, J. Drexler, S. Cyphers, and J. Glass, "Data collection and language understanding of food descriptions," *Proceedings of 2014 IEEE Spoken Language Technology Workshop (SLT)*, 2014.

[23] M. Korpusik, Z. Collins, and J. Glass, "Semantic mapping of natural language input to database entries via convolutional neural networks," *Proceedings of IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[24] M. Korpusik, Z. Collins, and J. Glass, "Character-based embedding models and reranking strategies for understanding natural language meal descriptions," *Proceedings of Interspeech*, 2017.

[25] M. Korpusik and J. Glass, "Convolutional neural networks and multitask strategies for semantic mapping of natural language input to a structured database," in *Proceedings of IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6174–6178.