# CONVOLUTIONAL NEURAL NETWORKS FOR DIALOGUE STATE TRACKING WITHOUT PRE-TRAINED WORD VECTORS OR SEMANTIC DICTIONARIES

*Mandy Korpusik, James Glass*

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

## ABSTRACT

A crucial step in task-oriented dialogue systems is tracking the user's goal over the course of the conversation. This involves maintaining a probability distribution over possible values for each slot (e.g., the `food` slot might map to the value `Turkish`), which gets updated at each turn of the dialogue. Previously, rule-based methods were applied to dialogue systems, or models that required hand-crafted semantic dictionaries mapping phrases to those that are similar in meaning (e.g., `area` might map to `part of town`). However, these are expensive to design for each domain, limiting the generalizability. In addition, often a spoken language understanding (SLU) component precedes the dialogue state update mechanism; however, this leads to compounded errors as the output from one module is passed to the next. Instead, more recent work has explored deep learning models for directly updating dialogue state, bypassing the need for SLU or expert-engineered rules. We demonstrate that a novel convolutional neural architecture without any pre-trained word vectors or semantic dictionaries achieves 86.9% joint goal accuracy and 95.4% requested slot accuracy on WOZ 2.0.

***Index Terms***— Convolutional Neural Networks, Dialogue State Tracking, Word Vectors, Semantic Dictionaries

## 1. INTRODUCTION

Spoken dialogue systems have gained popularity recently with the rise of personal intelligent assistants such as Siri and Cortana. These systems can be divided into two categories: chatbots that simply entertain the user through fun conversation, and task-oriented dialogue systems that accomplish a goal for the user, such as making a restaurant reservation or booking a flight. The standard approach for task-oriented dialogue typically follows a pipeline of steps, starting with intent detection (i.e., determining the user's goal), followed by spoken language understanding (SLU) of the user utterance to determine precisely what the user is requesting. For example, the user's intent may be to book a restaurant, for which the relevant semantic tag values would be `Chinese` for the `food` slot, and `centre` for the `area` slot. Subsequently, the user's goal is updated based on the output of the SLU component, the next system action is selected via the predicted user goal, and finally the system responds according to the chosen action (see Fig. 1 for the standard system pipeline).

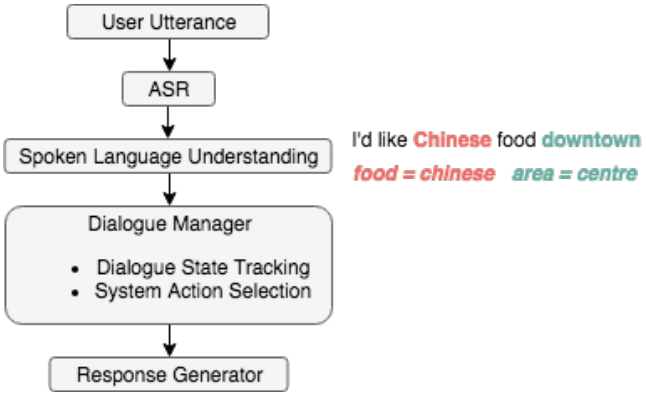


**Fig. 1**. The typical flow of a dialogue system, with spoken language understanding followed by dialogue state tracking.

In such a framework, however, there are several drawbacks. First, the SLU component is an intermediate step between the user query and updating the state of the dialogue (i.e., revising the current prediction for the user's goal) [1, 2, 3], which may cause errors to accumulate further down the pipeline. Recent approaches avoid this by eliminating the intermediate SLU and directly tracking the state of the dialogue, given the user utterance [4, 5]. In addition, prior approaches often relied on hand-crafted features, such as semantic dictionaries that map words to synonyms of each other (e.g., `area` may map to `part of town`), or even entirely rule-based dialogue systems. Current approaches explore end-to-end neural models for dialogue management instead [6, 7].

In this work, we examine the WOZ 2.0 written dataset for restaurant booking dialogues [8, 9], where the task is to predict the state of the dialogue at each turn. In particular, the dialogue state consists of the user's goal at that turn, which is composed of a set of slots that the user is either request-

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.

| |
|---|
| **User:** Is there any place here in the centre that serves corsica food?<br>`food = corsica; area = centre` |
| **System:** What price range are you looking for?<br>**User:** Any price range will do.<br>`food = corsica; area = centre; price = dontcare` |
| **System:** There are no restaurants available matching your criteria. Would you like to try a different area, price range, or food type?<br>**User:** Are there any restaurants in the centre that serves North American type of food?<br>`food = north_american; area = centre; price = dontcare` |
| **System:** Yes. The gourmet burger kitchen serves north american food and is located in the centre part of the city. Would you like their location?<br>**User:** Can I get the phone number?<br>`food = north_american; area = centre; price = dontcare; requested = phone` |
| **System:** The phone number for gourmet burger kitchen is 01223 312598.<br>**User:** Thank you. Good bye.<br>`food = north_american; area = centre; price = dontcare` |

**Table 1**. Example WOZ 2.0 dialogue snippet, with the corresponding slots specified by the user at each turn. Note that midway through the dialogue, the user changes the goal `food` slot value from `corsica` to `north_american`.

ing (e.g., "What is the phone number?" would indicate the requested slot `phone`) or informing the system (e.g., "I want Turkish food," which maps the informable slot `food` to the value `Turkish`). This requires keeping track of the history of the conversation, as well as the context from the previous system response, as illustrated in Table 1.

The approach we take in this work is a deep learning model, specifically a convolutional neural network (CNN) architecture for tracking the user's goal at each turn in a restaurant booking dialogue. We show that without relying on any semantic dictionaries or pre-trained word vectors, our model is competitive with state-of-the-art, achieving 86.9% joint goal accuracy and 95.4% requestable slot accuracy.

## 2. RELATED WORK

**CNNs for NLP** Many researchers in the natural language processing (NLP) community are currently exploring convolutional neural networks (CNNs) for processing text. In question answering, recent work showed improvements using deep CNN models for text classification [10, 11, 12], following the success of deep CNNs for computer vision [13, 14]. In other work, parallel CNNs predict the similarity of two input sentences by computing a word similarity matrix between the two sentences as input to a CNN [15, 16, 17]. Attention-based CNNs were applied to sentence matching [18] and machine comprehension [19], as well as for mapping natural language meal descriptions to their corresponding entries in a food database [20, 21, 22, 23, 24, 25].

| Slot-Value | Synonyms |
|---|---|
| Food=Cheap | [affordable, budget, low-cost, low-priced, ...] |
| Area=Centre | [center, downtown, central, city centre, ...] |
| Rating=High | [best, high-rated, highly rated, top-rated, ...] |

**Table 2**. Example rephrasings for three slot-value pairs in a semantic dictionary for restaurant booking.

**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 recogition output, which would be passed to the state tracking module to update the belief state [26, 2]. 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 [27, 28].

Typically, these jointly trained SLU and dialogue state updating models rely on a delexicalization-based strategy, which translates various instantiations of slot and value mentions in the user utterance into generic labels; this approach requires hand-crafted semantic dictionaries in order to perform the mapping from specific wordings to generic slot-value labels. Prior work by *Henderson et al.* fed delexicalized user utterances into a recurrent neural network, which output a distribution over slot values [29]. However, delexicalizing the input requires a manually defined semantic dictionary that maps from slot-value pairs to all possible text forms, or synonyms (see Table 2 for examples of slot-value pair synonyms).

To avoid this reliance on hand-crafted semantic dictionaries, *Mrksic et al.* recently demonstrated the ability of their Neural Belief Trackers (NBT) [6] 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. However, these Neural Belief Trackers still require pre-trained word vectors tailored to retain semantic relationships. While our work is similar to theirs in that we both leverage CNNs for dialogue state tracking, our work, on the other hand, does not rely on pre-trained word vectors, and directly predicts matching slot values instead of doing binary classification for each slot-value pair; in contrast, the NBT is trained to learn representations of user utterances and slot-value pairs that are used for binary classification (i.e., whether or not a given slot-value pair is mentioned in the user utterance).

In addition, *Zhong et al.'s* state-of-the-art work has ex-

plored deep learning methods for dialogue state tracking, but with recurrent (instead of convolutional) self-attentive encoders [7], and again considers each slot-value pair one at a time, while we predict the matching slot value from among all options simultaneously. Their self-attentive RNN model encodes user utterances, system actions, and each slot-value pair under consideration, but again relies on pre-trained Glove word embeddings [30], and character embeddings, while our model does not require any pre-trained embeddings. Finally, *Rastogi et al.* also feed delexicalized utterances into their multi-domain deep learning model for state tracking [31].

## 3. CNN DIALOGUE STATE TRACKER

The goal of our work is to accurately update the current belief state of the dialogue by predicting, at each turn, the correct slot values specified by the user, specifically for a restaurant booking task. There are two types of slots: informable (i.e., the user is providing information about the type of restaurant they want, such as the cuisine), and requestable (i.e., the user is asking for information about the restaurant, such as the telephone number),[1] as shown in Table 3 which enumerates all possible informable and requestable slots, as well as the number of values available for each slot. Since the informable slots are also requestable, the system must differentiate whether the user is providing or requesting information. In addition, there is an imbalance of data, since the `Food` informable slot has many possible values, whereas `Area` and `Pricerange` have fewer than 10, and some slot-value pairs appear more often than others in the training data.[2]

| Slot | Type | Num Values |
|---|---|---|
| `Food` | Informable, Requestable | 75 |
| `Area` | Informable, Requestable | 7 |
| `Pricerange` | Informable, Requestable | 4 |
| `Name` | Requestable | N/A |
| `Address` | Requestable | N/A |
| `Phone` | Requestable | N/A |
| `Postcode` | Requestable | N/A |
| `Signature` | Requestable | N/A |

**Table 3**. All possible informable and requestable slots.

As discussed in Section 2, prior work has either used hand-crafted features and semantic dictionaries for dialogue state tracking with delexicalization, or neural models relying on pre-trained semantic word vectors, while ours does not.

Below, we describe in detail our novel convolutional neural dialogue state tracker, with two variants: one model with a binary sigmoid output layer indicating the presence of each requestable slot (Fig. 3), and another separately trained model for each informable slot (Fig. 2) with a softmax output layer to predict the slot value. We also discuss two post-processing techniques for boosting performance to illustrate the importance of error analysis for gaining insight into why a system is underperforming and finding a solution based on human intuition about the task. Thus, we combine deep learning models with expert knowledge into a hybrid approach in order to overcome the limitations of purely neural methods.

### 3.1. Informable Slot Models

As shown in Fig. 2, we separately trained a model for each of the informable slots (i.e., `Food`, `Area`, and `Pricerange`).[3] Each model is composed of an embedding layer (which is not pre-trained, and is learned during training), into which we fed the user utterance concatenated with the previous system response as the input $\boldsymbol{x}$, where $\boldsymbol{x}$ is composed of the sequence of learned word vectors for the input tokens $\boldsymbol{w_1}, \boldsymbol{w_1}, ..., \boldsymbol{w_n}$. The input had two options, chosen via the development set:[4]

1. The user utterance concatenated with the full system response, omitting the system response if the user utterance starts with "no" (i.e., correcting the system), and using only the final question asked by the system.

2. The user utterance concatenated with all the slots requested by the system (i.e., in its dialogue act).

This is followed by a single convolutional layer with max-pooling to get a representation $\boldsymbol{r}$ of the embedded input $\boldsymbol{x}$:

$$\boldsymbol{r} = \texttt{maxpool}(\texttt{ReLU}(\texttt{Conv1D}(\boldsymbol{x}))) \quad (1)$$

Finally, a feed-forward layer with a softmax on top is used to directly predict the probability of all possible slot values:

$$\boldsymbol{o} = \texttt{softmax}(W\boldsymbol{r} + b) \quad (2)$$

where $W$ is a learned weight matrix and $b$ is a bias term in the final feed-forward layer.

### 3.2. Requestable Slot Model

The requestable slot model is also a CNN (shown in Fig. 3), but with a separate binary sigmoid output layer for each possible requestable slot (see Table 3), instead of one softmax layer on top, as in the informable slot models. The input to this model is the first option we tried for the informable slot models (i.e., the user utterance concatenated with the full system response, omitting the system response if the user says "no," and using only the final question asked by the system).

[1] http://camdial.org/ mh521/dstc/downloads/handbook.pdf

[2] Note that requestable-only slots (e.g., address) do not have values that can be specified by the user, since the user is requesting the value.

[3] We also tried jointly training all the slots, but found that separately training the models boosts performance over a single jointly trained model.

[4] Separately processing the user utterance and system response was worse.

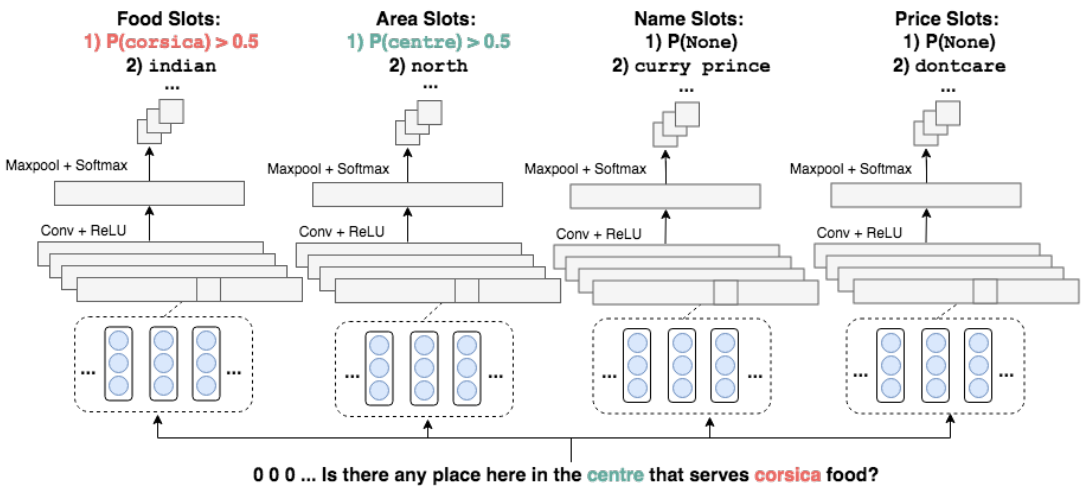


**Fig. 2**. The CNN architecture for separately trained models for each of the informable slot types.

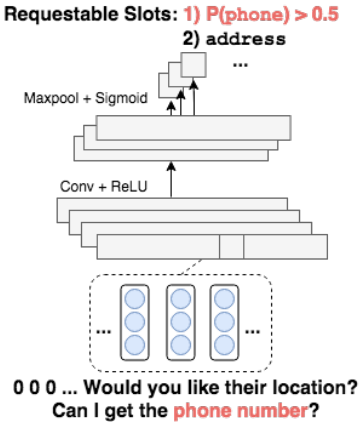


**Fig. 3**. The CNN architecture for the requestable slot model.

### 3.3. Hybrid Deep Learning and Knowledge-Based Method

While neural network models are incredibly powerful and have demonstrated success over prior state-of-the-art approaches in many fields, including computer vision, speech recognition, and natural language processing, there are still limitations to using these models that are often referred to as "black boxes." In our work, we simply feed the raw user utterance and system response into the model, which then outputs predicted slot-value pairs, requiring no manual feature engineering, pre-trained word vectors, or semantic dictionaries. However, this can make it difficult to interpret why the model behaves the way it does, and may limit performance since the model does not inherently have common-sense knowledge about the real world, or in this case, the restaurant booking task. Thus, by manually investigating test examples where the system made prediction errors, we are able to boost the model's performance by guiding it in the right direction based on expert knowledge of the task and dataset. Such an approach is illustrative of a hybrid between deep learning without any manual feature engineering, and expert knowledge-based systems, which we use to address the limitations of the purely neural model.

#### *3.3.1. Post-Processing Techniques to Boost Performance*

1. Delexicalization of the *input* to the model is common practice on dialogue state tracking tasks. In our work, we take a different approach, and perform string matching of slot values as a post-processing step to correct for any omitted slots (e.g., if some slot values were not seen in training), *after* the model makes its predictions.

2. We also check whether any slots that were requested by the system in a given turn (as specified by the system's dialogue act) were not predicted by that slot's model (i.e., the top value was `None`). If so, we add the next-highest predicted value for that slot to the goal state.

### 3.4. Implementation Details

We pad the input to 51 tokens (i.e., the maximum length of the concatenated user utterances and system responses seen during training). The input 64-dimension embedding layer is followed by a 1D convolution with 64 filters spanning windows of three tokens, with a rectified linear unit (ReLU) activation and dropout of probability 0.2. Each network is trained to predict the matching one-hot label array given the input user utterance; that is, the softmax is trained to predict 1 for each slot-value pair that is specified by the user, and 0 for all others. For the separately trained models, we add a `None` value for each slot type, and assign this value a 1 for each example that does not contain the specified slot type. The model is trained with the Adam optimizer [32] on binary cross-entropy loss.

The setup is the same for the requestable slot model, with a threshold of 0.5 at test time for each requestable slot.

We tune several threshold hyperparameters on the development set: at the start of a dialogue, we use a threshold of 0.5 for a predicted slot value when adding new slots to the goal state, while a higher threshold of 0.9 is best for adding new slots during the dialogue, and an even higher threshold of 0.99 for updating the value of slots already in the state. In addition, we set a threshold of 0.2 that must be exceeded in order to add slots requested by the system's dialogue act. Finally, the best input for the `Area` slot (and for all slots in the Sim-GEN movie booking task in Section 4.1) is the full system response concatenated with the user input, while the best input for the WOZ `Pricerange` and `Food` slots is the user utterance concatenated with the system's requested slots.

## 4. EXPERIMENTS

### 4.1. Datasets

For our experiments, we report results on the WOZ 2.0 dataset,[5] in which Turkers assumed the role of the system or user in dialogues similar to those used in the 2nd Dialogue State Tracking Challenge (DSTC2),[6] so we can compare our performance to that of state-of-the-art approaches on a standard dialogue system benchmark. This task involves restaurant booking, where the user specifies his or her goal as a set of informable and requestable slots, as described in Section 3. The WOZ data is written, not spoken, requiring semantic understanding rather than robustness to speech recognition errors. Our final model is trained on the full training and development set, with hyperparameters tuned on the development set, and is evaluated on the test utterances. To demonstrate our model's generalization capability, we also evaluate on the Sim-GEN dataset of conversations between an agent and a simulated user for buying movie tickets [33].

### 4.2. Metrics

As is commonly used in dialogue state tracking experiments, we report results on two slot tracking metrics:

- **Goals**: the proportion of dialogue turns where all the user's *informable* slots (i.e., search goal constraints) were correctly identified.
- **Requests**: the proportion of dialogue turns where all the user's *requestable* slots were correctly identified.

### 4.3. Results

As seen in Fig. 4, our CNN model without semantic dictionaries or pre-trained word vectors, achieved 86.9% goal accuracy and 95.4% requests accuracy on the held-out WOZ 2.0 dataset. In Fig. 5, our CNN outperforms the state-of-the-art hierarchical LSTM for jointly tracking dialogue state and predicting system actions [34] on the Sim-GEN movies data.

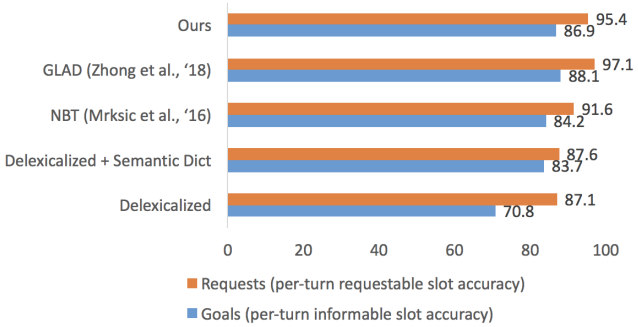


**Fig. 4**. Goal and request accuracy of our model compared to a strong delexicalization baseline and two state-of-the-art neural methods: NBT (Neural Belief Tracker) and GLAD (Global-Locally Self-Attentive Dialogue State Tracker).

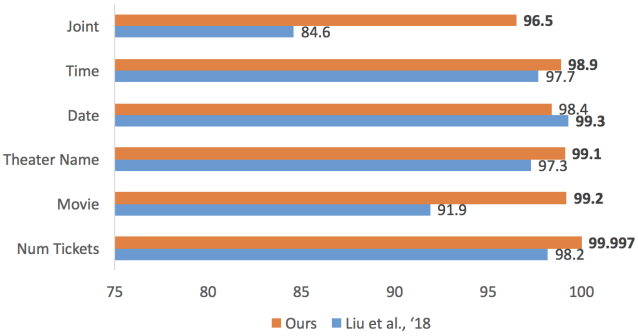


**Fig. 5**. Accuracy of our model, as compared to the state-of-the-art [34] on slots in the Sim-GEN movies dataset.

### 4.4. Ablation Study

Here we show the importance of the two post-processing techniques discussed in Section 3.3 that combine the deep learning model with expert knowledge based on error analysis. As shown in Table 4, the biggest gain in performance on the WOZ dataset is from exact string matching of slot values in the user utterance, and the best overall model is achieved by using both techniques. We also note that there is no gain from applying these techniques to the Sim-GEN movies dataset.

We see examples of errors made by the model on WOZ in Table 5, where due to synonyms such as "expensively" and "upscale" for "expensive," the model is unable to recognize out-of-vocabulary words that have similar meaning. In addition, the user correction for the `area` slot in the third example, where the user specifies "anywhere," is challenging since the system asks about the "centre" of town. Finally, understanding that the user wants a "cheap" restaurant if they are "close to broke" requires advanced commonsense reasoning.

[5] http://mi.eng.cam.ac.uk/ nm480/woz_2.0.zip
[6] http://camdial.org/ mh521/dstc/

| Model | WOZ Goals | Sim-GEN Goals |
|---|---|---|
| Best | **86.9** | **96.5** |
| w/o technique 1 | 76.6 | 96.5 |
| w/o technique 2 | 82.6 | 96.5 |
| w/o technique 1 or 2 | 72.7 | 96.5 |

**Table 4**. Goal development set accuracy of our model, on WOZ and Sim-GEN, without the two post-processing techniques in Section 3.3: 1) exact string matching of slot values in the user utterance, and 2) adding the slot value with highest predicted probability for slots requested by the system.

| |
|---|
| **User:** Hello, I'm looking for a **nice** restaurant with vegetarian food.<br>**True**: `food = vegetarian`<br>**Pred**: `food = vegetarian; price =` **`expensive`** |
| **User:** Hi, I want a Tuscan restaurant that's **expensively** priced.<br>**True**: `food = tuscan; price =` **`expensive`**<br>**Pred**: `food = vegetarian; price =` **`cheap`** |
| **System:** No such results found. Would you like me to search for any Mediterranean restaurants in the **centre**?<br>**User:** Is there a Lebanese place **anywhere** around?<br>**True**: `food = lebanese; area =` **`dontcare`**; `price = dontcare`<br>**Pred**: `food = lebanese; area =` **`centre`**; `price = dontcare` |
| **User:** I like Persian but I'm close to **broke**.<br>**True**: `food = persian; price =` **`cheap`**<br>**Pred**: `food = persian` |
| **System:** I will search for the most nearby English restaurant.<br>**User:** It should be an **upscale** English restaurant.<br>**True**: `food = english; price =` **`expensive`**<br>**Pred**: `food = english` |

**Table 5**. Examples of incorrect slot-value predictions made by the system due to the lexical variation used by Turkers in the WOZ 2.0 dataset, which requires semantic understanding.

### 4.5. Qualitative Analysis

To investigate whether our model is learning semantically meaningful embeddings after passing the input user utterance through the convolutional layer followed by maxpooling, we used Euclidean distance to identify the top-$n$ nearest neighbor embedded utterances to several utterances selected from the held-out test set (note that we are focusing on the model trained to predict requestable slots for the purposes of this analysis). In Table 6, we see that the nearest neighbor embeddings are indeed similar in meaning to the query user utterance (e.g., "any" is most similar to the learned vectors for "uh any" and "ah any"), and "chinese food" is most similar to the learned embedding for "um chinese food"), as expected.

To further understand the behavior of our neural network model and illustrate that is interpretable, rather than simply a black box, we also extracted the top-10 tokens that had the highest activations when passed through the learned CNN filters in the requestable slots model. As shown in Table 7, some filters appear to be identifying requestable slots (e.g., postcode, post, center), whereas others are focused specifically on finding different types of food (e.g., caribbean, indian, etc.).

| User Utterance | Top-3 Nearest Neighbors |
|---|---|
| *phone number* | and phone number<br>whats phone number<br>phone number please |
| *any* | uh any<br>ah any<br>any range |
| *chinese food* | um chinese food<br>what about thai food<br>romanian food |

**Table 6**. Top-3 nearest neighbors for three test user utterances, using Euclidean distance on the model's learned embeddings (i.e., after convolving and maxpooling the input).

| CNN Filter | Top-10 Tokens |
|---|---|
| 11 | caribbean, indian, type, food, bistro, serve, something, thai, singaporean, romanian |
| 13 | european, canapes, indian, bistro, japanese, caribbean, world, persian, italian, british |
| 16 | postcode, post, center, thank, restaurant, then, i, need, could, uh |
| 19 | phone, telephone, does, their, the, is, south, east, i, in |
| 50 | code, expensive, type, moderate, serving, kind, any, my, anything, cheap |

**Table 7**. Top-10 highest activation tokens for several learned CNN filters, where filters 11 and 13 isolate cuisines, and filters 16, 19, and 50 focus on three types of requestable slots: postcode, phone, and pricerange, respectively.

## 5. CONCLUSION

We have demonstrated that our novel convolutional architecture that directly predicts a user's goal slots during a task-oriented dialogue in the restaurant booking domain, given the user utterance and system response, achieves 86.9% joint goal accuracy and 95.4% requested slots on the WOZ 2.0 test set, without any semantic dictionaries or pre-trained word vectors.

In future work, we plan to extend our approach to other domains, such as student-advisor and Ubuntu user datasets in the recently launched DSTC7 challenge,[7] and to predict not only the user's goal, but the next system response. In addition, we plan to modify our model so as to handle the noisy ASR test set of DSTC2—this may require tricks such as summing the scores from each ASR hypothesis, applying word dropout, and learning character n-gram embeddings, as in [7].

[7] http://workshop.colips.org/dstc7/index.html

## 6. REFERENCES


[1] S. Young, M. Gašić, S. Keizer, F. Mairesse, J. Schatzmann, B. Thomson, and K. Yu, "The hidden information state model: A practical framework for POMDP-based spoken dialogue management," *Computer Speech & Language*, vol. 24, no. 2, pp. 150–174, 2010.

[2] 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.

[3] J. Williams, "Web-style ranking and SLU combination for dialog state tracking," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014, pp. 282–291.

[4] L. Zilka and F. Jurcicek, "Incremental LSTM-based dialog state tracker," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 757–762.

[5] N. Mrkšić, D. Séaghdha, B. Thomson, M. Gašić, P. Su, D. Vandyke, T. Wen, and S. Young, "Multi-domain dialog state tracking using recurrent neural networks," *arXiv preprint arXiv:1506.07190*, 2015.

[6] 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.

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

[8] J. Williams, A. Raux, D. Ramachandran, and A. Black, "The dialog state tracking challenge," in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 404–413.

[9] M. Henderson, B. Thomson, and J. Williams, "The second dialog state tracking challenge," in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2014, pp. 263–272.

[10] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[11] A. Conneau, H. Schwenk, Y. Lecun, and L. Barrault, "Very deep convolutional networks for text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, p. 11071116.

[12] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," in *Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference*, 2017, pp. 353–358.

[13] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] P. Sermanet, S. Chintala, and Y. LeCun, "Convolutional neural networks applied to house numbers digit classification," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 3288–3291.

[15] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, and X. Cheng, "Text matching as image recognition," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 2793–2799.

[16] Z. Wang, H. Mi, and A. Ittycheriah, "Sentence similarity learning by lexical decomposition and composition," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 1340–1349.

[17] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Proceedings of Advances in neural information processing systems (NIPS)*, 2014, pp. 2042–2050.

[18] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "ABCNN: Attention-based convolutional neural network for modeling sentence pairs," in *Transactions of the Association for Computational Linguistics*, 2016, vol. 4, pp. 259–272.

[19] W. Yin, S. Ebert, and H. Schütze, "Attention-based convolutional neural network for machine comprehension," in *Proceedings of 2016 NAACL Human-Computer Question Answering Workshop*, 2016, pp. 15–21.

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

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

[22] 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.

[23] 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.

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

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," *Proceedings of 2014 Conference on Empirical Methods on Natural Language (EMNLP)*, vol. 12, 2014.

[31] 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.

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

[33] P. Shah, D. Hakkani-Tür, G. Tür, A. Rastogi, A. Bapna, N. Nayak, and L. Heck, "Building a conversational agent overnight with dialogue self-play," *arXiv preprint arXiv:1801.04871*, 2018.

[34] B. Liu, G. Tur, D. Hakkani-Tur, P. Shah, and L. Heck, "Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems," *arXiv preprint arXiv:1804.06512*, 2018.