A Network-based End-to-End Trainable Task-oriented Dialogue System

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, Steve Young
Cambridge University Engineering Department, Trumpington Street, Cambridge, CB2 1PZ, UK
{thw28,djv27,nm480,mg436,lmr46,phs26,su259,sjy11}@cam.ac.uk

Abstract
Teaching machines to accomplish tasks by conversing naturally with humans is challenging. Currently, developing task-oriented dialogue systems requires creating multiple components and typically this involves either a large amount of handcrafting, or acquiring costly labelled datasets to solve a statistical learning problem for each component. In this work we introduce a neural network-based text-in, text-out end-to-end trainable goal-oriented dialogue system along with a new way of collecting dialogue data based on a novel pipe-lined Wizard-of-Oz framework. This approach allows us to develop dialogue systems easily and without making too many assumptions about the task at hand. The results show that the model can converse with human subjects naturally whilst helping them to accomplish tasks in a restaurant search domain.

1 Introduction
Building a task-oriented dialogue system such as a hotel booking or a technical support service is difficult because it is application-specific and there is usually limited availability of training data. To mitigate this problem, recent machine learning approaches to task-oriented dialogue system design have cast the problem as a partially observable Markov Decision Process (POMDP) (Young et al., 2013) with the aim of using reinforcement learning (RL) to train dialogue policies online through interactions with real users (Gašić et al., 2013). However, the language understanding (Henderson et al., 2014) and language generation (Wen et al., 2015b, Wen et al., 2016) modules still rely on supervised learning and therefore need corpora to train on. Furthermore, to make RL tractable, the state and action space must be carefully designed (Young et al., 2013, Young et al., 2010), which may restrict the expressive power and learnability of the model. Also, the reward functions needed to train such models are difficult to design and hard to measure at run-time (Su et al., 2015, Su et al., 2016).

At the other end of the spectrum, sequence to sequence learning (Sutskever et al., 2014) has inspired several efforts to build end-to-end trainable, non-task-oriented conversational systems (Vinyals and Le, 2015) [Shang et al., 2015; Serban et al., 2015b]. This family of approaches treats dialogue as a source to target sequence transduction problem, applying an encoder network (Cho et al., 2014) to encode a user query into a distributed vector representing its semantics, which then conditions a decoder network to generate each system response. These models typically require a large amount of data to train. They allow the creation of effective chatbot type systems but they lack any capability for supporting domain specific tasks, for example, being able to interact with databases (Sukhbaatar et al., 2015; Yin et al., 2015) and aggregate useful information into their responses.

In this work, we propose a neural network-based model for task-oriented dialogue systems by balancing the strengths and the weaknesses of the two research communities: the model is end-to-end trainable\footnote{We define end-to-end trainable as that each system module is trainable from data except for a database operator.} but still modularly connected; it does not directly model the user goal, but nevertheless, it still learns to accomplish the required task by providing relevant and appropriate responses at each turn; it has an explicit representation of database (DB) attributes (slot-value pairs) which it uses to achieve a high task success rate, but has a distributed representation of user intent (dialogue act)
to allow ambiguous inputs; and it uses delexicalisation and a weight tying strategy (Henderson et al., 2014) to reduce the data required to train the model, but still maintains a high degree of freedom should larger amounts of data become available. We show that the proposed model performs a given task very competitively across several metrics when trained on only a few hundred dialogues.

In order to train the model for the target application, we introduce a novel pipe-lined data collection mechanism inspired by the Wizard-of-Oz paradigm (Kelley, 1984) to collect human-human dialogue corpora via crowd-sourcing. We found that this process is simple and enables fast data collection online with very low development costs.

2 Model

We treat dialogue as a sequence to sequence mapping problem (modelled by a sequence-to-sequence architecture (Sutskever et al., 2014) augmented with the dialogue history (modelled by a set of belief trackers (Henderson et al., 2014)) and the current database search outcome (modelled by a database operator), as shown in Figure 1. At each turn, the system takes a sequence of tokens from the user as input and converts it into two internal representations: a distributed representation generated by an intent network and a probability distribution over slot-value pairs called the belief state (Young et al., 2013) generated by a set of belief trackers. The database operator then selects the most probable values in the belief state to form a query to the DB, and the search result, along with the intent representation and belief state are transformed and combined by a policy network to form a single vector representing the next system action.

This system action vector is then used to condition a response generation network (Wen et al., 2015a; Wen et al., 2015b) which generates the required system output token by token in skeletal form. The final system response is then formed by substituting the actual values of the database entries into the skeletal sentence structure. A more detailed description of each component is given below.

2.1 Intent Network

The intent network can be viewed as the encoder in the sequence-to-sequence learning framework (Sutskever et al., 2014) whose job is to encode a sequence of input tokens \( w_0, w_1, ... w_N \) into a distributed vector representation \( z_t \) at every turn \( t \). Typically, a Long Short-term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997) is used and the last time step hidden layer \( z_N \) is taken as the representation,

\[
   z_t = z_N = \text{LSTM}(w_0, w_1, ... w_N)  \tag{1}
\]

Alternatively, a convolutional neural network (CNN) can be used in place of the LSTM as the encoder (Kalchbrenner et al., 2014; Kim, 2014),

\[
   z_t = \text{CNN}(w_0, w_1, ... w_N)  \tag{2}
\]

and here we investigate both. Since all the slot-value specific information is delexicalised, the encoded vector can be viewed as a distributed intent.
Figure 2: Tied Jordan-type RNN belief tracker with delexicalised CNN feature extractor. The output of the CNN feature extractor is a concatenation of top-level sentence (green) embedding and several levels of intermediate ngram-like embeddings (red and blue). However, if a value cannot be delexicalised in the input, its ngram-like embeddings will all be padded with zeros. We pad zero vectors (in gray) before each convolution operation to make sure the representation at each layer has the same length. The output of each tracker $p_t^v$ is a distribution over values of a particular slot $s$.

representation which replaces the hand-coded dialogue act representation (Traum, 1999) in traditional task-oriented dialogue systems.

2.2 Belief Trackers

Belief tracking (also called Dialogue State tracking) provides the core of a task-oriented spoken dialogue system (SDS) (Henderson, 2015). Current state-of-the-art belief trackers use discriminative models such as recurrent neural networks (RNN) (Mikolov et al., 2010; Wen et al., 2013) to directly map ASR hypotheses to belief states (Henderson et al., 2014; Mrkšić et al., 2016). Although in this work we focus on text-based dialogue systems, we retain belief tracking at the core of our system because: (1) it enables a sequence of free-form natural language sentences to be mapped into a fixed set of slot-value pairs, which can then be used to query a DB. This can be viewed as a simple version of a semantic parser (Berant et al., 2013); (2) by keeping track of the dialogue state, it avoids learning unnecessarily complicated long-term dependencies from raw inputs; (3) it uses a smart weight tying strategy that can greatly reduce the data required to train the model, and (4) it provides an inherent robustness which simplifies future extension to spoken systems.

Using each user input as new evidence, the task of a belief tracker is to maintain a multinomial distribution $p$ over values $v \in V_s$ for each informable slot $s$, and a binary distribution for each requestable slot $3$. Each slot in the ontology $4$ has its own specialised tracker, and each tracker is a Jordan-type (recurrence from output to hidden layer) (Jordan, 1989) RNN with a CNN feature extractor, as shown in Figure (2). Like Mrkšić et al. (2015), we tie the RNN weights together for each value $v$ but vary features $f_v^t$ when updating each pre-softmax activation $g_v^t$. The update equations for a given slot $s$ are,

$$f_v^t = f_v^t_{cnn} \oplus p_{v-1} \oplus p_0^{t-1}$$ (3)  
$$g_v^t = w_s \cdot \text{sigmoid}(W_s f_v^t + b_s) + b'_s$$ (4)  
$$p_v^t = \frac{\exp(g_v^t)}{\exp(g_{\emptyset,s}) + \sum_{v' \in V_s} \exp(g_{v'}^t)}$$ (5)

where vector $w_s$, matrix $W_s$, bias terms $b_s$ and $b'_s$, and scalar $g_{\emptyset,s}$ are parameters, $p_0^t$ is the probability that the user has not mentioned that slot up to turn $t$ and can be calculated by substituting $g_{\emptyset,s}$ for $g_v^t$ in the numerator of Equation 5. In order to model the discourse context at each turn, the feature vector

3Informable slots are slots that users can use to constrain the search, such as food type or price range; Requestable slots are slots that users can ask a value for, such as address.  

A small knowledge graph defining the slot-value pairs the system can talk about for a particular task.

3We don’t use the recurrent connection for requestable slots since they don’t need to be tracked.
where every token in \( u_t \) and \( m_{t-1} \) is represented by an embedding of size \( N \) derived from a 1-hot input vector. In order to make the tracker aware when delexicalisation is applied to a slot or value, the slot-value specialised CNN operator \( \text{CNN}^{(s,v)}(\cdot) \) extracts not only the top level sentence representation but also intermediate n-gram-like embeddings determined by the position of the delexicalised token in each utterance. If multiple matches are observed, the corresponding embeddings are summed. On the other hand, if there is no match for a particular slot or value, the empty n-gram embeddings are padded with zeros. In order to keep track of the position of delexicalised tokens, both sides of the sentence are padded with zeros before each convolution operation. The number of vectors is determined by the filter size at each layer. The overall process of extracting several layers of position-specific features is visualised in Figure 2.

The belief tracker described above is based on Henderson et al. (2014) with some modifications: (1) only probabilities over informative and requestable slots and values are output, (2) the recurrent memory block is removed, since it appears to offer no benefit in this task, and (3) the n-gram feature extractor is replaced by the CNN extractor described above. By introducing slot-based belief trackers, we essentially add a set of intermediate labels into the system as compared to training a pure end-to-end system. Later in the paper we will show that these tracker components are critical for achieving task success. We will also show that the additional annotation requirement that they introduce can be successfully mitigated using a novel pipe-lined Wizard-of-Oz data collection framework.

2.3 Policy Network and Database Operator

Database Operator Based on the output \( p^t_s \) of the belief trackers, the DB query \( q_t \) is formed by,

\[
q_t = \bigcup_{s' \in S_t} \{ \text{argmax}_v p^t_{s'} \} \quad (7)
\]

where \( S_t \) is the set of informative slots. This query is then applied to the DB to create a binary truth value vector \( x_t \) over DB entities where a 1 indicates that the corresponding entity is consistent with the query (and hence it is consistent with the most likely belief state). In addition, if \( x \) is not entirely null, an associated entity pointer is maintained which identifies one of the matching entities selected at random. The entity pointer is updated if the current entity no longer matches the search criteria; otherwise it stays the same. The entity referenced by the entity pointer is used to form the final system response as described in Section 2.4.

Policy network The policy network can be viewed as the glue which binds the system modules together. Its output is a single vector \( o_t \) representing the system action, and its inputs are comprised of \( z_t \) from the intent network, the belief state \( p^t_s \), and the DB truth value vector \( x_t \). Since the generation network only generates appropriate sentence forms, the individual probabilities of the categorical values in the informative belief state are immaterial and are summed together to form a summary belief vector for each slot \( p^t_s \) represented by three components: the summed value probabilities, the probability that the user said they "don’t care" about this slot and the probability that the slot has not been mentioned. Similarly for the truth value vector \( x_t \), the number of matching entities matters but not their identity. This vector is therefore compressed to a 6-bin 1-hot encoding \( \hat{x}_t \), which represents different degrees of matching in the DB (no match, 1 match, ... or more than 5 matches). Finally, the policy network output is generated by a three-way matrix transformation,

\[
o_t = \text{tanh}(W_{zo} z_t + W_{po} \hat{p}_t + W_{xo} \hat{x}_t) \quad (8)
\]

where matrices \( W_{zo}, W_{po}, \) and \( W_{xo} \) are parameters and \( \hat{p}_t = \bigoplus_{e \in E} \hat{p}^t_e \) is a concatenation of all summary belief vectors.

2.4 Generation Network

The generation network uses the action vector \( o_t \) to condition a language generator (Wen et al., 2015b). This generates template-like sentences token by token based on the language model probabilities,

\[
P(w_{j+1} | w^j, h^j_{j-1}, o_t) = \text{LSTM}_j(w^j, h^j_{j-1}, o_t) \quad (9)
\]

where \( \text{LSTM}_j(\cdot) \) is a conditional LSTM operator for one output step \( j \), \( w^j \) is the last output token (i.e. a word, a delexicalised slot name or a delexicalised
slot value), and \( h_{j-1} \) is the hidden layer. Once the output token sequence has been generated, the generic tokens are replaced by their actual values: (1) replacing delexicalised slots by random sampling from a list of surface forms, e.g. `<s.food>` to `food` or `type of food`, and (2) replacing delexicalised values by the actual attribute values of the entity currently selected by the DB pointer. This is similar in spirit to the Latent Predictor Network (Ling et al., 2016) where the token generation process is augmented by a set of pointer networks (Vinyals et al., 2015) to transfer entity specific information into the response.

**Attentive Generation Network** Instead of decoding responses directly from a static action vector \( o_t \), an attention-based mechanism (Bahdanau et al., 2015; Hermann et al., 2015) can be used to dynamically aggregate source embeddings at each output step \( j \). In this work we explore the use of an attention mechanism to combine the tracker belief states, i.e. \( o_t \) is computed at each output step \( j \) by,

\[
\alpha_t^{(j)} = \tanh(W_{xz}z_t + \hat{p}_t^{(j)} + W_{xo}x_t) \tag{10}
\]

where for a given ontology \( G \),

\[
\hat{p}_t^{(j)} = \sum_{s \in G} \alpha_s^{(j)} \tanh(W_{po} \cdot \hat{p}_s) \tag{11}
\]

and where the attention weights \( \alpha_s^{(j)} \) are calculated by a scoring function,

\[
\alpha_s^{(j)} = \text{softmax}(r^T \tanh(W_r \cdot u_t)) \tag{12}
\]

where \( u_t = z_t \oplus \hat{x}_t \oplus \hat{p}_t \oplus w_j \oplus h_{j-1} \), matrix \( W_r \), and vector \( r \) are parameters to learn and \( w_j \) is the embedding of token \( w_j \).

3 Wizard-of-Oz Data Collection

Arguably the greatest bottleneck for statistical approaches to dialogue system development is the collection of appropriate training data, and this is especially true for task-oriented dialogue systems. Serban et al. (Serban et al., 2015a) have catalogued existing corpora for developing conversational agents. Such corpora may be useful for bootstrapping, but, for task-oriented dialogue systems, in-domain data is essential. To mitigate this problem, we propose a novel crowdsourcing version of the Wizard-of-Oz (WOZ) paradigm (Kelley, 1984) for collecting domain-specific corpora.

Based on the given ontology, we designed two webpages on Amazon Mechanical Turk, one for wizards and the other for users (see Figure 4 and 5 for the designs). The users are given a task specifying the characteristics of a particular entity that they must find (e.g. a Chinese restaurant in the north) and asked to type in natural language sentences to fulfill the task. The wizards are given a form to record the information conveyed in the last user turn (e.g. `pricerange=Chinese, area=north`) and a search table showing all the available matching entities in the database. Note these forms contain all the labels needed to train the slot-based belief trackers. The table is automatically updated every time the wizard submits new information. Based on the updated table, the wizard types an appropriate system response and the dialogue continues.

In order to enable large-scale parallel data collection and avoid the distracting latencies inherent in conventional WOZ scenarios (Bohus and Rudnicky, 1999), users and wizards are asked to contribute just a single turn to each dialogue. To ensure coherence and consistency, users and wizards must review all previous turns in that dialogue before they contribute their turns. Thus dialogues progress in a pipe-line. Many dialogues can be active in parallel and no worker ever has to wait for a response from the other party in the dialogue. Despite the fact that multiple workers contribute to each dialogue, we observe that dialogues are generally coherent yet diverse. Furthermore, this turn-level data collection strategy seems to encourage workers to learn and correct each other based on previous turns.

In this paper, the system was designed to assist users to find a restaurant in the Cambridge, UK area. There are three informative slots (`food`, `pricerange`, `area`) that users can use to constrain the search and six requestable slots (`address`, `phone`, `postcode` plus the three informative slots) that the user can ask a value for once a restaurant has been offered. There are 99 restaurants in the DB. Based on this domain, we ran 3000 HITs (Human Intelligence Tasks) in total for roughly 3 days and collected 1500 dialogue turns. After cleaning the data, we have approximately 680 dialogues in total (some of them are unfinished). The total cost for collecting the dataset was \( \sim 400 \text{ USD} \).

4 Empirical Experiments

**Training** Training is divided into two phases. Firstly the belief tracker parameters \( \theta_b \) are
where $\theta$ were only applied after the final convolution layer. The vocabulary size is around 500 for both input and output, in which rare words and words that can never mentioned (e.g. food=none) but its corresponding delexicalised slot (e.g. <v.food>) is generated. For the full model, we have three informative trackers (food, pricerange, area) and seven requestable trackers (address, phone, postcode, name, plus the three informative slots).

Having fixed the tracker parameters, the remaining parts of the model $\theta_b$ are trained using the cross entropy errors from the generation network language model, $L_2(\theta_b) = -\sum_t \sum_j (y^j_t)^\top \log p^j_t$, where $y^j_t$ and $p^j_t$ are output token targets and predictions respectively, at turn $t$ of output step $j$. We treated each dialogue as a batch and used stochastic gradient decent with a small $\ell^2$ regularisation term to train the model. Early stopping was implemented based on the validation set for regularisation and gradient clipping was set to 1. All the hidden layer sizes were set to 50, and all the weights were randomly initialised as a batch and used stochastic gradient decent with a small $\ell^2$ regularisation term to train the model.

**Decoding** In order to decode without length bias, we decoded each system response $m_t$ based on the average log probability of tokens,

\[
    m^*_t = \arg\max_{m_t} \{ \log p(m_t|\theta, u_t)/J_t \} \tag{13}
\]

where $\theta$ are the model parameters, $u_t$ is the user input, and $J_t$ is the length of the machine response. As a contrast, we also investigated the MMI criterion (Li et al., 2016) to increase diversity and put additional scores on delexicalised tokens to encourage task completion. This weighted decoding strategy has the following objective function,

\[
    m^*_t = \arg\max_{m_t} \{ \log p(m_t|\theta, u_t)/J_t - \lambda \log p(m_t)/J_t + \gamma R_t \} \tag{14}
\]

where $\lambda$ and $\gamma$ are weights selected on validation set and $\log p(m_t)$ can be modelled by a standalone LSTM language model. We used a simple heuristic for the scoring function $R_t$ designed to reward giving appropriate information and penalise spuriously providing unsolicited information. We applied beam search with a beamwidth equal to 10, the search stops when an end of sentence token is generated. In order to obtain language variability from the deployed model we ran decoding until we obtained 5 candidates and randomly sampled one as the system response.

**Tracker performance** Table 1 shows the evaluation of the trackers’ performance. Due to delexicalisation, both CNN type trackers and N-gram type trackers (Henderson et al., 2014) achieve high precision, but the N-gram tracker has worse recall. This result suggests that compared to simple N-grams, CNN type trackers can better generalise to sentences with long distance dependencies and more complex syntactic structures.

**Corpus-based evaluation** We evaluated the end-to-end system by first performing a corpus-based evaluation in which the model is used to predict each system response in the held-out test set. Three evaluation metrics were used: BLEU score on top-1 and top-5 candidates (Papineni et al., 2002), entity matching rate and objective task success rate (Su et al., 2015). We calculated the entity matching rate by determining whether the actual selected entity at the end of each dialogue matches the task that was specified to the user. The dialogue is then marked as successful if both (1) the offered entity matches, and (2) the system answered all the associated information requests (e.g. what is the address?) from the user. We computed the BLEU scores on the template-like output sentences before lexicalising with the entity value substitution.

Table 1: Tracker performance in terms of Precision, Recall, and F-1 score.

<table>
<thead>
<tr>
<th>Tracker type</th>
<th>Informable</th>
<th></th>
<th>Requestable</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Prec. Recall F-1</td>
<td>Prec. Recall F-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cnn</td>
<td>99.77% 96.09% 97.89%</td>
<td>98.66% 93.79% 96.16%</td>
<td>99.34% 94.42% 96.82%</td>
<td>98.56% 90.14% 94.16%</td>
<td></td>
</tr>
<tr>
<td>ngram</td>
<td>99.34% 94.42% 96.82%</td>
<td>98.56% 90.14% 94.16%</td>
<td>99.77% 96.09% 97.89%</td>
<td>98.66% 93.79% 96.16%</td>
<td></td>
</tr>
</tbody>
</table>

\[\text{We give an additional reward if a requestable slot (e.g. address) is requested and its corresponding delexicalised slot or value token (e.g. <v.address> and <s.address>) is generated. We give an additional penalty if an informative slot is never mentioned (e.g. food=none) but its corresponding delexicalised value token is generated (e.g. <v.food>). For more details on scoring, please see Table 5.}\]
Table 2: Performance comparison of different model architectures based on a corpus-based evaluation.

<table>
<thead>
<tr>
<th>Encoder</th>
<th>Tracker</th>
<th>Decoder</th>
<th>Match(%)</th>
<th>Success(%)</th>
<th>T5-BLEU</th>
<th>T1-BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baseline</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lstm</td>
<td>-</td>
<td>lstm</td>
<td>-</td>
<td>-</td>
<td>0.1650</td>
<td>0.1718</td>
</tr>
<tr>
<td>lstm</td>
<td>turn recurrence</td>
<td>lstm</td>
<td>-</td>
<td>-</td>
<td>0.1813</td>
<td>0.1861</td>
</tr>
<tr>
<td><strong>Variant</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn, w/o req.</td>
<td>lstm</td>
<td>89.70</td>
<td>30.60</td>
<td>0.1769</td>
<td>0.1799</td>
</tr>
<tr>
<td>cnn</td>
<td>rnn-cnn</td>
<td>lstm</td>
<td>88.82</td>
<td>58.52</td>
<td>0.2354</td>
<td>0.2429</td>
</tr>
<tr>
<td><strong>Full model w/ different decoding strategy</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn</td>
<td>lstm</td>
<td>86.34</td>
<td>75.16</td>
<td>0.2184</td>
<td>0.2313</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn + weighted</td>
<td>lstm</td>
<td>86.04</td>
<td>78.40</td>
<td>0.2222</td>
<td>0.2280</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn + att.</td>
<td>lstm</td>
<td>90.88</td>
<td>80.02</td>
<td>0.2286</td>
<td>0.2388</td>
</tr>
<tr>
<td>lstm</td>
<td>rnn-cnn + att. + weighted</td>
<td>lstm</td>
<td>90.88</td>
<td>83.82</td>
<td>0.2304</td>
<td>0.2369</td>
</tr>
</tbody>
</table>

Table 2 shows the result of the corpus-based evaluation averaging over 5 randomly initialised networks. The Baseline block shows two baseline models: the first is a simple turn-level sequence to sequence model (Sutskever et al., 2014) while the second one introduces an additional recurrence to model the dependency on the dialogue history following Serban et al (Serban et al., 2015b). As can be seen, incorporation of the recurrence improves the BLEU score. However, baseline task success and matching rates cannot be computed since the models do not make any provision for a database.

The Variant block of Table 2 shows two variants of the proposed end-to-end model. For the first one, no requestable trackers were used, only informable trackers. Hence, the burden of modelling user requests falls on the intent network alone. We found that without explicitly modelling user requests, the model performs very poorly on task completion (∼30%), even though it can offer the correct entity most of the time (∼90%). More data may help here; however, we found that the incorporation of an explicit internal semantic representation in the full model (shown below) is more efficient and extremely effective. For the second variant, the LSTM intent network is replaced by a CNN. This achieves a very competitive BLEU score but task success is still quite poor (∼58% success). We think this is because the CNN encodes the intent by capturing several local features but lacks the global view of the sentence, which may easily result in an unexpected overfit.

The Full model block shows the performance of the proposed model with different decoding strategies. The first row shows the result of decoding using the average likelihood term (Equation 13) while the second row uses the weighted decoding strategy (Equation 14). As can be seen, the weighted decoding strategy does not provide a significant improvement in BLEU score but it does greatly improve task success rate (∼3%). The $R_l$ term contributes the most to this improvement because it injects additional task-specific information during decoding. Despite this, the most effective and elegant way to improve the performance is to use the attention-based mechanism (+att.) to dynamically aggregate the tracker beliefs (Section 2.4). It gives a slight improvement in BLEU score (∼0.01) and a big gain on task success (∼5%). Finally, we can improve further by incorporating weighted decoding with the attention models (+ att. + weighted).

As an aside, we used t-SNE (der Maaten and Hinton, 2008) to produce a reduced dimension view of the action embeddings $o_t$, plotted and labelled by the first three generated output words (full model w/o attention). The figure is shown as Figure 3. We can see clear clusters based on the system intent types, even though we did not explicitly model them using dialogue acts.

**Human evaluation** In order to assess operational performance, we tested our model using paid subjects recruited via Amazon Mechanical Turk. Each judge was asked to follow a given task and to rate the model’s performance. We assessed the subjective success rate, and the perceived comprehension ability and naturalness of response on a scale of 1 to 5. The full model with attention and weighted decoding was used and the system was tested on a total of 245 dialogues. As can be seen in Table 3 the average subjective success rate was 98%, which means the system was able to complete the majority of tasks. Moreover, the comprehension ability and naturalness scores both averaged more than 4 out of 5. (See Appendix for some sample dialogues in this trial.)

We also ran comparisons between the NN model
Table 3: Human assessment of the NN system. The rating for comprehension/naturalness are both out of 5.

<table>
<thead>
<tr>
<th>Metric</th>
<th>NN</th>
<th>HDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>98%</td>
<td></td>
</tr>
<tr>
<td>Comprehension</td>
<td>4.11</td>
<td>4.05</td>
</tr>
<tr>
<td>Naturalness</td>
<td>4.05</td>
<td></td>
</tr>
<tr>
<td># of dialogues</td>
<td>245</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: A comparison of the NN system with a rule-based modular system (HDC).

<table>
<thead>
<tr>
<th>Metric</th>
<th>NN</th>
<th>HDC</th>
<th>Tie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subj. Success</td>
<td>96.95%</td>
<td>95.12%</td>
<td></td>
</tr>
<tr>
<td>Avg. # of Turn</td>
<td>3.95</td>
<td>4.54</td>
<td></td>
</tr>
<tr>
<td>Naturalness</td>
<td>46.95*</td>
<td>25.61</td>
<td>27.44</td>
</tr>
<tr>
<td>Comprehension</td>
<td>45.12*</td>
<td>21.95</td>
<td>32.93</td>
</tr>
<tr>
<td>Preference</td>
<td>50.00*</td>
<td>24.39</td>
<td>25.61</td>
</tr>
<tr>
<td>Performance</td>
<td>43.90*</td>
<td>25.61</td>
<td>30.49</td>
</tr>
</tbody>
</table>

* p <0.005, # of comparisons: 164

and a handcrafted, modular baseline system (HDC) consisting of a handcrafted semantic parser, rule-based policy and belief tracker, and a template-based generator. The result can be seen in Table 4.

The HDC system achieved \( \sim 95\% \) task success rate, which suggests that it is a strong baseline even though most of the components were hand-engineered. Over the 164 dialogues tested, the NN system (NN) was considered better than the handcrafted system (HDC) on all the metrics compared. Although both systems achieved similar success rates, the NN system (NN) was more efficient and provided a more engaging conversation (lower turn number and higher preference). Moreover, the comprehension ability and naturalness of the NN system were also rated higher, which suggests that the learned system was perceived as being more natural than the hand-designed system.

5 Conclusions and Future Work

This paper has presented a novel neural network-based framework for task-oriented dialogue systems. The model is end-to-end trainable using two supervision signals and a modest corpus of training data. The paper has also presented a novel crowdsourced data collection framework inspired by the Wizard-of-Oz paradigm. We demonstrated that the pipe-lined parallel organisation of this collection framework enables good quality task-oriented dialogue data to be collected quickly at modest cost.

The experimental assessment of the NN dialogue system showed that the learned model can interact efficiently and naturally with human subjects to complete an application-specific task. To the best of our knowledge, this is the first end-to-end NN-based model that can conduct meaningful dialogues in a task-oriented application.

However, there is still much work left to do. Our current model is a text-based dialogue system, which can not directly handle noisy speech recognition inputs nor can it ask the user for confirmation when it is uncertain. Indeed, the extent to which this type of model can be scaled to much larger and wider domains remains an open question which we hope to pursue in our further work.
Wizard-of-Oz data collection websites

Figure 4: The user webpage. The worker who plays a user is given a task to follow. For each mturk HIT, he/she needs to type in an appropriate sentence to carry on the dialogue by looking at both the task description and the dialogue history.

Figure 5: The wizard page. The wizard’s job is slightly more complex: the worker needs to go through the dialogue history, fill in the form (top green) by interpreting the user input at this turn, and type in an appropriate response based on the history and the DB result (bottom green). The DB search result is updated when the form is submitted. The form can be divided into informable slots (top) and requestable slots (bottom), which contains all the labels we need to train the trackers.

Scoring Table

Table 5: Additional $R_t$ term for delexicalised tokens when using weighted decoding (Equation 14). Not observed means the corresponding tracker has a highest probability on either not mentioned or don’tcare value, while observed mean the highest probability is on one of the categorical values. A positive score encourages the generation of that token while a negative score discourages it.

<table>
<thead>
<tr>
<th>Delexicalised token</th>
<th>Examples</th>
<th>$R_t$ (observed)</th>
<th>$R_t$ (not observed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>informable slot token</td>
<td>&lt;s.food&gt;, &lt;s.area&gt;,...</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>informable value token</td>
<td>&lt;v.food&gt;, &lt;v.area&gt;,...</td>
<td>+0.05</td>
<td>-0.5</td>
</tr>
<tr>
<td>requestable slot token</td>
<td>&lt;s.phone&gt;,&lt;s.address&gt;,...</td>
<td>+0.2</td>
<td>0.0</td>
</tr>
<tr>
<td>requestable value token</td>
<td>&lt;v.phone&gt;,&lt;v.address&gt;,...</td>
<td>+0.2</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Acknowledgements

Tsung-Hsien Wen and David Vandyke are supported by Toshiba Research Europe Ltd, Cambridge. The authors would like to thank Ryan Lowe and Lukáš Žilka for their valuable comments.

References


[Shang et al.2015] Lifeng Shang, Zhendong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In ACL.


Table 6: Some samples of real conversational logs between online judges and the end-to-end system.

<table>
<thead>
<tr>
<th>Human</th>
<th>Machine</th>
<th>Human</th>
<th>Machine</th>
<th>Human</th>
<th>Machine</th>
<th>Human</th>
<th>Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restaurant in any area that serves Vietnamese food</td>
<td>thanh binh serves vietnamese food type . is there anything else i can help you with ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td>i am looking for moderately priced halal food</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
</tr>
<tr>
<td>What is the phone number and postcode</td>
<td>the number is 01223 362456 and the area code is c.b 3 , 0 a.f . is there anything else i can do for you ?</td>
<td>how about european food</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>galleria serves european food type and is in the moderate price range .</td>
<td>address and postcode</td>
<td>the address is 33 bridge street and the area code is c.b 2 , 1 u.w . is there anything else i can help you with today ?</td>
<td>thanks</td>
</tr>
<tr>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>galleria serves european food type and is in the moderate price range .</td>
<td>address and postcode</td>
<td>the address is 33 bridge street and the area code is c.b 2 , 1 u.w .</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
</tr>
<tr>
<td>Hi , i ’ m hungry for some Indonesian food</td>
<td>i am sorry , there are no indonesian restaurants . would you like a different food ?</td>
<td>Oh no . How about Indian ?</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>galleria serves european food type and is in the moderate price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
</tr>
<tr>
<td>What cuisine would you like ?</td>
<td>afghan</td>
<td>Ok . How about Indian ?</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>galleria serves european food type and is in the moderate price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
</tr>
<tr>
<td>afghan</td>
<td>i am sorry , there are no afghan restaurants . would you like something else ?</td>
<td>little seoul serves korean food . is it located at 108 regent street city centre .</td>
<td>i am sorry , there are no halal restaurants in the moderate price range . would you like something else ?</td>
<td>galleria serves european food type and is in the moderate price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
</tr>
<tr>
<td>Korean</td>
<td>little seoul serves korean food and is in the expensive price range .</td>
<td>is it expensive ?</td>
<td>little seoul serves korean food and is in the expensive price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
</tr>
<tr>
<td>Little Seoul serves korean food . it is located at 108 regent street city centre .</td>
<td>is it expensive ?</td>
<td>little seoul serves korean food and is in the expensive price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>the address is 33 bridge street and the area code is c.b 2 , 1 u.w .</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
</tr>
<tr>
<td>is it expensive ?</td>
<td>the address is 33 bridge street and the area code is c.b 2 , 1 u.w .</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is there anything else i can help you with today ?</td>
<td>the address is 33 bridge street and the area code is c.b 2 , 1 u.w .</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
</tr>
<tr>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Galleria serves european food type and is in the moderate price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
</tr>
<tr>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Galleria serves european food type and is in the moderate price range .</td>
<td>is there anything else i can help you with today ?</td>
<td>thanks</td>
<td>galleria is a european restaurant in the moderate price range . would you like their phone ?</td>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Good bye</td>
<td>than you for using the cambridge restaurant system .</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Appendix: Sample dialogues