Towards End-to-End Learning for Efficient Dialogue Agent by Modeling Looking-ahead Ability

Zhuoxuan Jiang[1], Xian-Ling Mao[2]*, Ziming Huang[3], Jie Ma[3], Shaochun Li[3]
[1]IBM Research / Shanghai, China
[2]Beijing Institute of Technology / Beijing, China
[3]IBM Research / Beijing, China
{jzxjiang,hzmzi,bjmajie,lishaoc}@cn.ibm.com, maoxl@bit.edu.cn

Abstract

Learning an efficient manager of dialogue agent from data with little manual intervention is important, especially for goal-oriented dialogues. However, existing methods either take too many manual efforts (e.g. reinforcement learning methods) or cannot guarantee the dialogue efficiency (e.g. sequence-to-sequence methods). In this paper, we address this problem by proposing a novel end-to-end learning model to train a dialogue agent that can look ahead for several future turns and generate an optimal response to make the dialogue efficient. Our method is data-driven and does not require too much manual work for intervention during system design. We evaluate our method on two datasets of different scenarios and the experimental results demonstrate the efficiency of our model.

1 Introduction

Research for dialogue system attracts a lot of attentions recently due to its potential huge value of reducing human cost in many commercial domains, such as restaurant reservation (Henderson et al., 2014b), travel planning (Peng et al., 2017) and retail service (Zhang et al., 2019). The majority of dialogue agents with goals are expected to be efficient to complete tasks with as few as possible dialogue turns, in contrast with those chit-chat counterparts (Ritter et al., 2011). The following two examples show the comparison of efficient and inefficient cases respectively. The senarios is restaurant reservation and the agent's goal is to reserve a table at noon.

Efficient example:
"*Human*: We don't have empty tables at 11 o'clock tomorrow. All are reserved."
"*Agent*: How about 12 o'clock? We are also okay then."

Inefficient example:
"*Human*: We don't have empty tables at 11 o'clock tomorrow. All are reserved."
"*Agent*: What else time is available?"
"*Human*: 12 o'clock is ok."
"*Agent*: All right. We want that time."

For expressing the same opinion, the inefficient example consumes four turns while the efficient example only needs two. As it can be seen, the efficiency is important for goal-oriented dialogue systems to time-savingly achieve goals.

Usually, a dialogue system consists of a pipeline of natural language understanding (NLU), dialogue management (DM) and natural language generation (NLG), where the DM is treat as two separate components: dialogue state tracking (DST) and dialogue control (DC, i.e. dialogue policy selection). The DM is widely considered to be relevant to the dialogue's efficiency, and it is challenging to make it efficient. Recently, methods based on reinforcement learning are proposed for the policy selection component to build efficient dialogue systems. However, there are some drawbacks of reinforcement learning based methods. For example, they requires lots of human work to design the learning strategy. Also a real-world environment which is essential for the agent to learn from is expensive, such as from domain experts. Moreover, training the dialogue manager as a two separate components could lead to error propagation issue (Rastogi et al., 2018).

In addition to reinforcement learning based methods, sequence-to-sequence based methods are also popular recently, because they can learn a dialogue agent purely from data and almost without too many human efforts. The error propagation issue can also be reduced because they are end-to-end, and they have better scalability for different scenarios. However, it is difficult to build efficient dialogue agents by those methods since their objective functions for training mod-

*Xian-Ling Mao is the corresponding author.

els are usually inclined to general responses, such as *I don't know*, *yes* and *OK*, or often generate the same response for totally different contexts because the contextual information is not well-modeled by those methods (Dodge et al., 2015).

In this paper, we address the problem of learning an efficient dialogue manager from the perspective of reducing manual intervention and error propagation, and propose a new sequence-to-sequence based approach. The proposed end-to-end model contains a novel looking-ahead module for dialogue manager to learn the looking-ahead ability. Our intuition is that by predicting the future several dialogue turns, the agent could make a better decision of what to say for current turn, and therefore goals could be sooner achieved in a long run.

More specifically, our model includes three modules: (1) encoding module, (2) looking-ahead module, and (3) decoding module. At each dialogue turn, three kinds of information, the goals, historical utterances and the current user utterance, are utilized. First they are encoded by three separate Bidirectional Gated Recurrent Units (BiGRU) models. Then the three encoded embeddings are concatenated to one vector, which is then sent to a new bidirectional neural network that can look ahead for several turns. The decoding module will generate utterances for each turn through a learned language model. At last, by considering all the predicted future utterances, a new real system utterance for the next turn is re-generated by using an attention model through the same language model.

Our proposed approach has several advantages. First, it is an end-to-end model and does not take too many human efforts for system design. Although the goals should be handcrafted for specific scenario, the number of goals is small and it is a relatively easy work. Moreover, compared with naive sequence-to-sequence based models, our agent can make the dialogue more efficient by modeling the looking-ahead ability. Experimental results show that our model performs better than baselines on two datasets from different domains, which could suggest that our model is also scalable to various domains.

The contributions in this paper include:

- We identify the problem that how to make dialogues efficient by exploiting as little as possible manual intervention during system design from the perspective of end-to-end deep learning.
- We propose a novel end-to-end and data-driven model that enables the dialgoue agent to learn to look ahead and make efficient decisions of what to say for the next turn.
- Experiments conducted on two datasets demonstrate that our model performs better over baselines and can be applied to different domains.

2 Related Work

In most situations, the dialogue systems require handcrafted definition of dialogue states and dialogue policies (Williams and Young, 2007; Henderson et al., 2014a; Asher et al., 2012; Chen et al., 2017). Those methods make the pipeline of dialogue systems clear to design and easy to maintain, but suffer from the massive expensive human efforts and the error propagation issue (Henderson et al., 2014c; Liu and Lane, 2017).

Reinforcement learning based methods for dialogue policy selection are widely studied recently (Lipton et al., 2018; Dhingra et al., 2017; Zhao and Eskenazi, 2016; Su et al., 2016). These methods only need human to design the learning strategies and do not require massive training data. However, the expensive domain knowledge and human expert efforts for agents to learn from are necessary (Liu et al., 2018; Shah et al., 2018). Therefore, hybrid methods that integrate supervised learning and reinforcement learning are proposed recently (Williams et al., 2017; Williams and Zweig, 2016). Thus, collecting massive training data becomes another manual work.

More recently, end-to-end dialogue systems attract much attention because almost no human efforts are required and they are scalable for different domains (Wen et al., 2017; Li et al., 2017; Lewis et al., 2017; Luo et al., 2019), especially with sequence-to-sequence based models (Sutskever et al., 2014). Although those models have been proved to be effective on chit-chat conversations (Ritter et al., 2011; Li et al., 2016a; Zhang et al., 2018), how to build agents that are goal-oriented with efficient dialogue managers through end-to-end approaches still remains questionable (Bordes et al., 2017; Joshi et al., 2017), and we investigate the question in this paper.

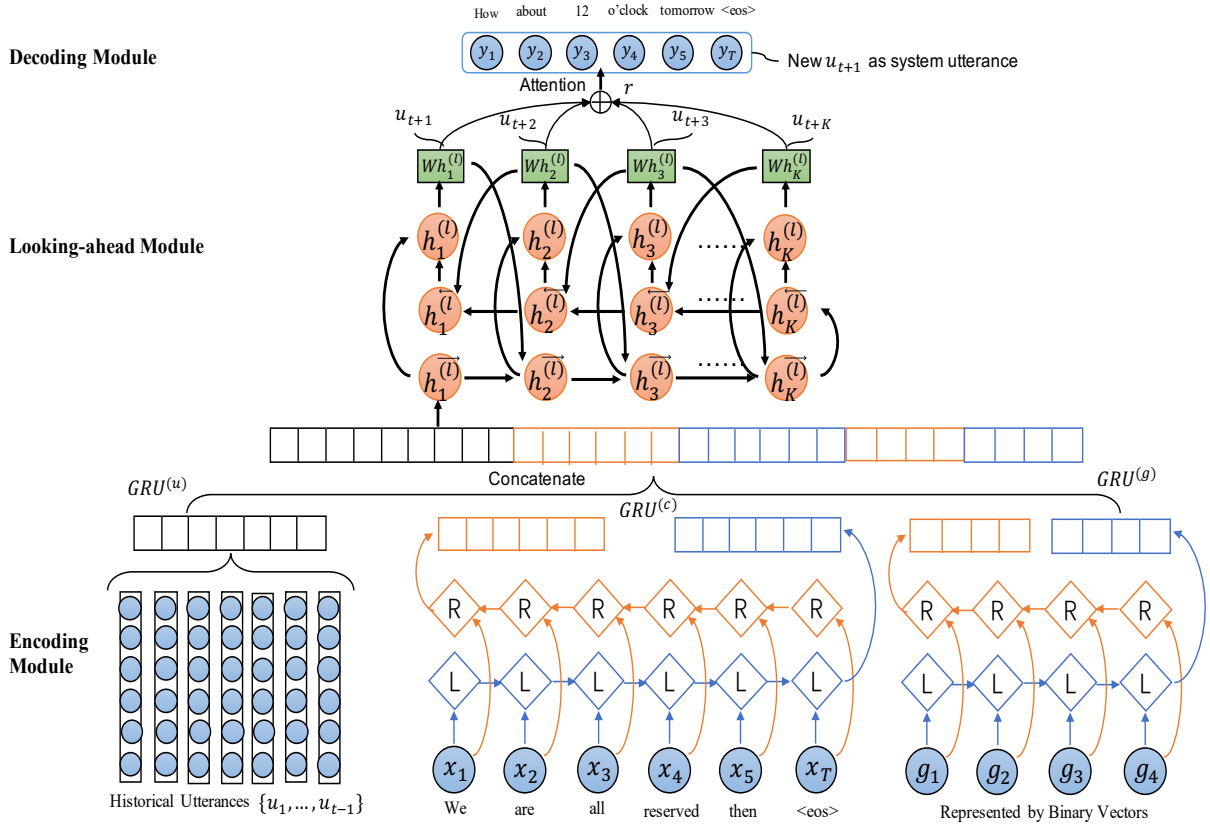

Figure 1: End-to-end model for learning looking-ahead ability.

Our idea of enabling the agent to be efficient by modeling looking-ahead ability is inspired by the AI Planning concept, which is a traditional searching technology in the field of AI, and is suitable for goal-based tasks, such as robotics control (Norvig and Russell, 1995). Recently, the concept is borrowed to dialogue system communities and integrated into deep learning models. For example, a trade-off method for training the agents neither with real human nor with user simulators is proposed, in order to obtain better policy learning results (Peng et al., 2018). In addition, at earlier time, the planning idea has been utilized for improving the dialogue generation task (Stent et al., 2004; Walker et al., 2007).

3 End-to-end Dialogue Model

We propose an end-to-end model that contains three modules: (1) encoding module, (2) looking-ahead module, and (3) decoding module. Figure 1 shows the model architecture. We leverage Bidirectional GRU models (Bahdanau et al., 2014) to encode agent goals, historical and current utterances. Then the obtained representations by encoding goals and utterances are regarded as inputs of the looking-ahead module, and they are used to predict several future turns. At last the predicted future turns are merged by an attention model and the new real system utterance is generated for the next turn.

Suppose for each dialogue session we have T turns, and we do not distinguish whether it is user's turn or system's turn. If the agent has S goals that are denoted as $g = \{g_1, g_2, ..., g_S\}$, each goal is formalized as a binary vector. For example in the restaurant reservation scenario, we can define that each variate in the vector $[1, 0]$ corresponds to a yes-no condition, such as the 1 means agent accepts bar table and the 0 means agent does not want to change time. As to the utterance information, imagine at turn $t \in \{1, ..., T\}$, we denote utterances $\{u_1, ..., u_{t-1}\} \in \mathbb{U}$ for historical ones and $u_t \in \mathbb{U}$ for current user utterance. Our model predicts the system and user utterances $\{u_{t+1}, u_{t+2}, ..., u_{t+K}\}$ for the next K turns and then a new u_{t+1} is generated as the system utterance after considering all the predicted turns. The model separates the current user utterance from historical ones in order to highlight the user's current states. In general, the model is end-to-end and

needs little human intervention or domain knowledge.

3.1 Encoding Module

In this module, the agent goals, historical utterances within the dialogue session, and the current user utterance are encoded by using three GRU models which is expected to learn long-range temporal dependencies (Cho et al., 2014). $GRU^{(g)}$ is defined to encode agent's goals g and the final hidden state $h^{(g)}$ is taken as the representation of goals. The input of $GRU^{(g)}$ is a one-hot binary vector with length S. $GRU^{(u)}$ is used to encode the historical utterances, and $GRU^{(c)}$ is used to encode the current user utterance. $h^{(u)}$ and $h^{(c)}$ are denoted as the final encoded representations of $GRU^{(u)}$ and $GRU^{(c)}$ respectively.

To get the i-th hidden state for the three GRUs, respective inputs include the previous hidden state $h^{(g)}_{i-1}$, $h^{(u)}_{i-1}$ or $h^{(c)}_{i-1}$, and the embeddings of current observations, $E(g_i)$, $E(u_i)$ or $E(x_i)$, where g_i is a goal, u_i is an utterance and x_i is a token. For the textual tokens, we use the Word2vec embeddings as their representations (Mikolov et al., 2013). Then the token embeddings are averaged to represent utterances. The formal denotation of the hidden states for the three GRU models is:

$$h^{(g)}_i = GRU^{(g)}(h^{(g)}_{i-1}, E(g_i)), \tag{1}$$

$$h^{(u)}_i = GRU^{(u)}(h^{(u)}_{i-1}, E(u_i)), \tag{2}$$

$$h^{(c)}_i = GRU^{(c)}(h^{(c)}_{i-1}, E(x_i)), \tag{3}$$

where $E(\cdot)$ represents the embeddings.

The final output of the encoding module is a concatenation of $h^{(g)}$, $h^{(u)}$ and $h^{(c)}$, which is denoted as $h^{\overrightarrow{l}}_1 = [h^{(g)}, h^{(u)}, h^{(c)}]$. $h^{\overrightarrow{l}}_1$ serves as the input of the following looking-ahead module. The right arrow means the initial direction to train the looking-ahead module is from the current to the future.

3.2 Looking-ahead Module

With the input of $h^{\overrightarrow{l}}_1$, this module predicts several future dialogue turns. Since the process is sequential, we propose a recurrent neural network to model the process. In order to exploit the predicted information for later generating a real system utterance, another recurrent neural network is used to backtrack the information from future to current. To reduce the computing cost, the two neural networks share the same parameters, and the whole looking-ahead module looks similar to a bidirectional GRU as shown in Figure 1.

We denote the module as $GRU^{(l)}$. $\{h^{(l)}_k | k > 0\}$ represent the predicted hidden states for future turns. To get $h^{(l)}_k$, the hidden states from two directions, $h^{\overrightarrow{l}}_k$ and $h^{\overleftarrow{l}}_k$, are concatenated. To calculate each $h^{\overrightarrow{l}}_k$ or $h^{\overleftarrow{l}}_k$, their inputs include the previous hidden state and the previously-predicted hidden state. Formally, suppose we look ahead for K turns, the hidden state of $h^{(l)}_k$ is calculated as following:

$$h^{\overrightarrow{l}}_k = GRU^{\overrightarrow{l}}(h^{\overrightarrow{l}}_{k-1}, Wh^{(l)}_{k-1}), \tag{4}$$

$$h^{\overleftarrow{l}}_k = GRU^{\overleftarrow{l}}(h^{\overleftarrow{l}}_{k+1}, Wh^{(l)}_{k+1}), \tag{5}$$

$$h^{(l)}_k = [h^{\overrightarrow{l}}_k, h^{\overleftarrow{l}}_k], \tag{6}$$

where W is a weight parameter and $Wh^{(l)}_k$ is the hidden state for predicting future turns. If $K = 1$, it means our model has no looking-ahead ability and it degrades to a naive goal-based sequence-to-sequence model.

3.3 Decoding Module

For generating the real system utterance, as seen in Figure 1, the green hidden states $\{Wh^{(l)}_k | k > 0\}$ are combined through an attention based model (Wang et al., 2016). The formal denotation is:

$$e_k = tanh(W^{(a)} Wh^{(l)}_k)], \tag{7}$$

$$v_k = \frac{\exp(e_k)}{\sum_{k=1}^{K} \exp(e_k)}, \tag{8}$$

$$r = \sum_{k=1}^{K} v_k h^{(l)}_k, \tag{9}$$

where $W^{(a)}$ is the attention weight parameter and r is the input representation for generating a new u_{t+1} that is regarded as the real system utterance.

Given the hidden state $Wh^{(l)}_k$, the decoding module can also generate the corresponding utterance for learning the looking-ahead ability. We share the parameters of decoding with those in the

encoding module, in order to reduce the computing cost (Vinyals and Le, 2015). The token sequence in u_{t+k} is generated from left to right by selecting the tokens with the maximum probability distribution through a language model learned by the following equation:

$$p_\theta(y_j^{(t+k)}|y_{1,2,\ldots,j-1}^{(t+k)}) \propto \exp(E^T W h_k^{(l)}). \quad (10)$$

3.4 Model Training

To train the proposed model, we define a loss function to maximize three terms: (1) a language model for predicting tokens in language generation, (2) the probability distribution of predicting utterances of future dialogue turns, and (3) a binary classifier to predict if the dialogue will be complete or not. The final joint loss function is formally denoted as:

$$\begin{aligned} L(\theta) = & \underbrace{-\sum_u \sum_i \log p_\theta(x_i|x_{1,\ldots,i-1})}_{language\ model\ loss} \\ & \underbrace{-\alpha \sum_{u,g} \sum_k \sum_i \log p_\theta(y_i^{(t+k)}|y_{1,\ldots,i-1}^{(t+k)}, u, g)}_{looking\ ahead\ prediction\ loss} \\ & \underbrace{-\beta \sum_c \log p(z_c|c, u_{t+1})}_{dialog\ state\ prediction\ loss}, \end{aligned} \quad (11)$$

where

$$u_{t+1} = \arg max_y p_\theta(y|r), \quad (12)$$

$$\begin{aligned} \log p(z_c|c, u_{t+1}) = & z_c \log(g(c, u_{t+1})) \\ & + (1 - z_c) \log(1 - g(c, u_{t+1})). \end{aligned} \quad (13)$$

$g(\cdot)$ is a sigmoid function and z_c is the label of the dialogue that current user utterance c belongs to, where 1 means the dialogue ends up with goals achieved while 0 means the goals are not achieved. The three terms are weighted with two hyper-parameters α and β. We adopt stochastic gradient descent method to minimize $L(\theta)$.

In the looking-ahead module, the hidden state $Wh_k^{(l)}$ is used to generate an utterance $y^{(t+k)}$, and is also used to calculate $h_{k+1}^{\overrightarrow{l}}$ and $h_{k-1}^{\overleftarrow{l}}$. We design an EM-like algorithm to optimize the loss function, as described in Algorithm 1. Line 3-4 optimize the language model, i.e. the first term of $L(\theta)$. Line 5-16 optimize the looking-ahead module, i.e. the second term, among which Line 7-14 are for E-step and Line 15-16 are for M-step. In E-step the language model is fixed for updating all the hidden states $h_k^{(l)}$ in looking-ahead module, and in M-step all the hidden states are fixed for updating the language model. Line 17-18 optimize the third term of $L(\theta)$, which is a binary classifier.

Algorithm 1: Learning algorithm for $L(\theta)$

input : Dialogue utterances U, Agent goals g, Looking-ahead turns K
output: Agent model θ

1 Randomly initializing parameters;
2 **for** $c \in U$, *g and historical utterances* $\{u\}$ **do**
3 **for** $x_i \in c$ **do**
4 Optimizing $p_\theta(x_i|x_{1,\ldots,i-1})$;
5 $h_1^{\overrightarrow{l}} = [h^{(g)}, h^{(u)}, h^{(c)}]$;
6 $u_{t+1} = \arg max_y p_\theta(y|r)$;
  E-Step: Update $h_k^{(l)}$ with fixed language model;
7 **for** $k = 1 : K$ **do**
8 $u_{t+k} = \arg max_y p_\theta(y|h_k^{(l)})$;
9 $h_k^{\overrightarrow{l}} = [h_{k-1}^{\overrightarrow{l}}, Wh_{k-1}^{(l)}]$;
10 $h_K^{\overleftarrow{l}} = h_K^{\overrightarrow{l}}$;
11 **for** $k = K - 1 : 1$ **do**
12 $h_k^{\overleftarrow{l}} = [h_{k+1}^{\overleftarrow{l}}, Wh_{k+1}^{(l)}]$;
13 **for** $k = 1 : K$ **do**
14 $h_k^{(l)} = [h_k^{\overrightarrow{l}}, h_k^{\overleftarrow{l}}]$;
  M-Step: Update language model with fixed $h_k^{(l)}$;
15 **for** $k = 1 : K$ **do**
16 Optimizing $p_\theta(y_i^{(t+k)}|y_{1,\ldots,i-1}^{(t+k)})$;
17 $u_{t+1} = \arg max_y p_\theta(y|r)$;
18 Optimizing $p(z_c|c, u_{t+1})$;
19 **return** θ;

4 Experiments

4.1 Data Collection

We use two datasets for two different scenarios to evaluate our model. Table 1 shows the statistics of two datasets.

4.1.1 Dataset 1 - Object Division

Dataset 1 contains crowd-sourced dialogues between humans collected from Amazon Mechanical Turk platform (Lewis et al., 2017). The dataset is for *object division task* and both sides have separate goals of each object's value. We use the textual data and transform their goals to yes-no questions as our binary vectors. The information of each dialogue session's final state, agree or disagree, is used for training the agent.

4.1.2 Dataset 2 - Restaurant Reservation

We construct the Dataset 2 to testify that if our model can be applied to various scenarios. We

Metric	Dataset 1	Dataset 2
Number of Dialogues	5,808	1,613
Average Turns per Dialogue	6.6	6.3
Average Words per Turn	7.6	8.9
Number of Words	566,779	98,726
% Goal Achieved	80.1%	71.5%

Table 1: Statistic on the two datasets.

choose the scenario of restaurant table reservation for demonstration.

In this scenario, the two agents are expected to dialogue with each other and they have their own different goals. We denote Agent A as the role of a customer and Agent B as the restaurant side. At the beginning of each dialogue session, Agent A is given a time slot and the number of people, together with several other goals (e.g. can share table or not), and all the constraints are translated into the agent's goals as binary vectors. Agent A then talks with Agent B to reserve a table. Agent B also has its constraints (e.g. whether bar tables are available during a certain time period), which are also taken as its goals. During dataset generation, two goal pools are predefined for each side and the goals in each dialogue session are randomly sampled from the pools. Agent A and Agent B cannot see each other's goals. They dialogue through natural language until a final decision (agreement or disagreement) is reached. In summary, the target of this experiment is to see if our model can find an intersection of two agents' goals in a more efficient way.

To generate dialogues of Dataset 2, we resort to Watson AI platform [1] for natural language understanding by defining intents and entities with examples. We adopt the AI planning method (Ghallab et al., 2016) as the dialogue manager. We design the dialogue states and actions. The goals are represented as binary vectors, and the STRIPS algorithm is used to search the shortest path to goals at each turn and return the first action for generating the next response. Table 2 shows a sample dialogue.

4.2 Training Sample Preparation

For each dialogue session with T turns, we reorganize the utterances into T samples. For each turn $t = \{1, 2, ..., T\}$, we can get the current user utterance c, and a training sample is created with a historical utterance sequence $\{u_1, u_2, ..., u_{t-1}\}$,

[1] https://www.ibm.com/watson/ai-assistant/

Alice: May I reserve a table for 6 people at 17 tomorrow?
Bob: Sorry, we don't have a table at this point.
Alice: Can we sit at the bar then?
Bob: We don't have a bar in the restaurant.
Alice: Can I have more expensive tables then?
Bob: My apologies, we are required not to do that.
Alice: In this case, can I reserve a bigger table?
Bob: Yes, we have VIP rooms but more expensive.
Alice: I want that.
Bob: OK.
Alice: Bye.

Table 2: Sample of Dataset 2.

and the goals g are consistent with the same dialogue session. The future K turns of utterances $\{u_{t+1}, u_{t+2}, ..., u_{t+K}\}$ are used as the supervised information. In total, we get 38,333 and 10,162 samples including training set and test set for the two datasets respectively.

4.3 Baselines

Since our model is based on purely data-driven learning, we compare our model with the supervised counterparts. Our baselines include:

- Seq2Seq(goal): This is a naive baseline by adapting the sequence-to-sequence model (Sutskever et al., 2014) and encoding goals, which removes the looking-ahead module and the supervised information of final state prediction from our model.
- Seq2Seq(goal+state): This is a baseline model by removing the looking-ahead module from our proposed model. The parameter α is set to zero.
- Seq2Seq(goal+look): This is a baseline model by removing the supervised information of final state prediction from our model. The parameter β is set to zero.
- Seq2Seq(goal+look+state): This is our proposed model that includes all the modules and supervised information.

4.4 Evaluation Criteria

In a dialogue system, it could be treat as efficient if it obtains more final goal achievement with as few as possible dialogue turns. Thus we set two criteria for evaluating and comparing models adopted in our experiments: (1) the *goal achievement ratio* that means the ratio of the number of goal achieved dialogue over the number of attempted dialogues), and (2) the *average dialogue turns*.

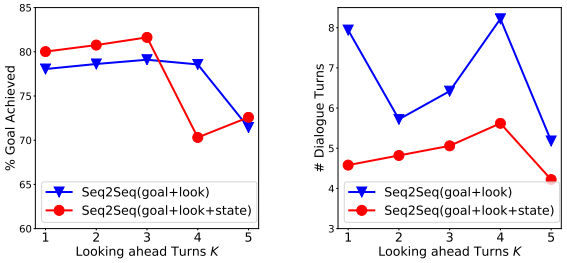

Figure 2: vs. looking-ahead turns on Dataset 1

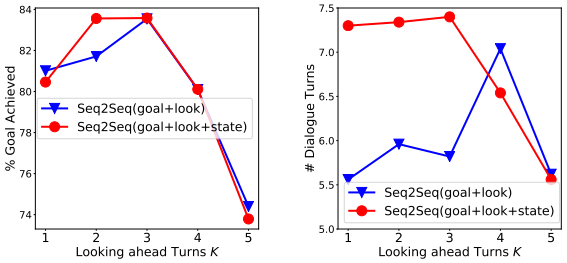

Figure 3: vs. looking-ahead turns on Dataset 2

4.5 Evaluator

Our experiments are to achieve goals through conversations, and it is difficult to directly adopt existing simulators (Asri et al., 2016). We refer to the work (Li et al., 2016b) and fine-tune it to our task. For each dataset, a naive sequence-to-sequence model that encodes goals is regarded as the user simulator. We run 1000 times of dialogue sessions using the simulator.

Apart from using the simulator, we also invite humans to dialogue with the agents for 100 times each person for each dataset and we report the average results.

4.6 Training Settings

All the baselines are implemented by PyTorch. One-hot input tokens are embedded into a 64-dimensional space. The goals are encoded by $GRU^{(g)}$ with a hidden layer of size 64. The sizes of hidden states in input utterance encoder $GRU^{(u)}$, $GRU^{(c)}$ and looking-ahead module $GRU^{(l)}$, $h_k^{(l)}$, are all set to 256. A stochastic gradient descent method is employed to optimize the model with a mini-batch size of 32 for supervised learning, an initial learning rate of 1.0, momentum with $\mu = 0.1$, and clipping gradients 0.5 in L^2 norm. The best model is chosen from the processing of training the model for 400 epochs. After that, the learning rate decays by a factor of 2 for every epoch. The initial hyper-parameters setting in the loss function (Equation (11)) is $\alpha = 0.05$ and $\beta = 1.0$. Words that appear in the training dataset for less than 5 times are replaced with the 'unknown' ($\langle unk \rangle$) token. A validation dataset is employed to choose the optimal hyper-parameters.

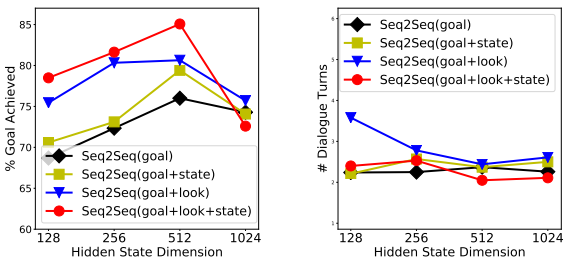

Figure 4: vs. hidden state dimension on Dataset 1

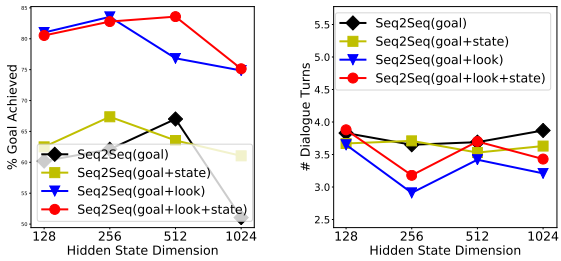

Figure 5: vs. hidden state dimension on Dataset 2

4.7 Results and Analysis

Table 3 shows the performance of baselines against user simulator and human on the two datasets. Both reveal that models that learn looking-ahead ability can achieve better performance and deliver more efficient dialogues in terms of both goal achievement ratio and dialogue turns. However, in the table, the dialogue turns of Seq2Seq(goal+look+state) are larger than those of Seq2Seq(goal+look), which may suggest that more dialogue turns lead to more achievement. In spite of this, the looking-ahead ability learned by our model is demonstrated to be effective on the two different scenarios. Moreover, the supervised information of final states (the third term of Equation (11)) is also proven effective in delivering more achievement, which can be seen from the second and last rows of Table 3. Compared to the human evaluation, the results with the simulator generally are better. It is probable that human eval-

Model	Dataset 1				Dataset 2			
	vs. Simulator		*vs. Human*		*vs. Simulator*		*vs. Human*	
	% Achieved	# Turns	% Achieved	# Turns	% Achieved	# Turns	% Achieved	# Turns
Seq2Seq(goal)	76.00	4.74	67.74	7.87	67.10	7.38	54.1	7.56
Seq2Seq(goal+state)	79.41	4.74	70.97	6.35	67.37	7.42	58.1	8.04
Seq2Seq(goal+look)	80.64	6.54	74.19	5.41	83.54	**5.82**	60.3	**6.94**
Seq2Seq(goal+look+state)	**85.07**	**4.10**	**77.42**	**5.02**	**83.58**	6.36	**61.2**	7.30

Table 3: Performance on two datasets against the user simulator and human.

Seq2Seq(goal) Model:
Alice: i just want the book Bob: no way i have the book and you can take others Alice: bye <Conversation end>
Seq2Seq(goal+look+state) Model:
Alice: i keep the book you can have the hat and two balls Bob: i will take the book and you can have others Alice: can we make a deal? Bob: ok deal Alice: okay Bob: bye <Conversation end>

Table 4: Example that tends to achieve goals on Dataset 1, vs. simulator.

Seq2Seq(goal) Model:
Alice: can you help me book a table for 2 people at 6pm? Bob: sorry, we dont have a table at this point Alice: can i reserve the seats at the bar instead? Bob: yes you can Alice: can i book a table for us Bob: sure i have written down your reservation Alice: ok Bob: bye <Conversation end>
Seq2Seq(goal+look+state) Model:
Alice: can you help me book a table for 2 people at 6pm? Bob: sorry we dont have a table at this point Alice: can i reserve the seats at the bar instead? Bob: sure i have written down your reservation Alice: bye <Conversation end>

Table 5: Example that costs less dialogue turns under the same goals on Dataset 2, vs. simulator.

uators tend to be more rigorous and more turns are necessary to achieve goals.

We also evaluate the models' performance with different parameters. Figure 2 and Figure 3 show the results of using different looking-ahead turns K in the looking-ahead module on Dataset 1 and Dataset 2 respectively. In most cases, Seq2Seq(goal+look+state) is better than Seq2Seq(goal+look). Setting the number to 3 is the recommended value, since our scenarios are relatively simple and it is enough for the agents to look ahead for 3 dialogue turns. The results are consistent with those in Table 3.

The performance on different dimension sizes of hidden states used in all the baselines is evaluated and the results are shown in Figure 4 and Figure 5. We find that setting the size to 512 yields the optimal results by balancing the performance and the computing cost in most cases.

In addition, we take some case studies for dialogue examples between the agents trained by Seq2Seq(goal+look+state) (proposed model) and the simulator (Seq2Seq(goal)) for the two datasets. *Alice* denotes the agent and *Bob* denotes the simulator. Table 4 is a case where our model with the looking-ahead ability (Seq2Seq(goal+look+state)) tends to achieve goals while consume more dialogue turns. Table 5 gives an example where our model not only tends to achieve goals but also takes fewer dialogue turns. Based on our observation, the language models for natural language generation is learned well in spite of that confused utterances are generated sometimes.

5 Conclusion

In this paper, we propose an end-to-end model towards the problem of how to learn an efficient dialogue manager without taking too many manual efforts. We model the looking-ahead ability for foreseeing several turns and then the agent can make a decision of what to say that leads the conversation to achieve goals with as few as possible dialogue turns. Experiments on two datasets from different domains demonstrate that our model is efficient in terms of goal achievement ratio and average dialogue turns. Our method is also scalable and can reduce error propagation due to the nature of end-to-end learning.

For the future work, we expect to investigate whether other kinds of abilities, such as reasoning ability, can be modeled for agent towards the problem. In addition to the efficiency issue, the quality of natural language generation should also be paid attention in order to guarantee the quality of overall dialogue system.

References

Nicholas Asher, Alex Lascarides, Oliver Lemon, Markus Guhe, Verena Rieser, Philippe Muller, Stergos Afantenos, Farah Benamara, Laure Vieu, Pascal Denis, S. Paul, S. Keizer, and C. Degrémont. 2012. Modelling strategic conversation: The stac project. In *SemDial*, page 27.

Layla El Asri, Jing He, and Kaheer Suleman. 2016. A sequence-to-sequence model for user simulation in spoken dialogue systems. In *INTERSPEECH*, pages 1151–1155.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. ArXiv preprint arXiv:1409.0473.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *ICLR*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems- recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *SSST-8*, pages 103–114.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *ACL*, pages 484–495.

Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. 2015. Evaluating prerequisite qualities for learning end-to-end dialog systems. ArXiv preprint arXiv:1511.06931.

Malik Ghallab, Dana Nau, and Paolo Traverso. 2016. *Automated Planning and Acting*. Cambridge University Press.

Matthew Henderson, Blaise Thomson, and Jason Williams. 2014a. The second dialog state tracking challenge. In *SIGDIAL*, pages 263–272.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014b. The third dialog state tracking challenge. In *SLT*, pages 324–329.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014c. Word-based dialog state tracking with recurrent neural networks. In *SIGDIAL*, pages 292–299.

Chaitanya K. Joshi, Fei Mi, and Boi Faltings. 2017. Personalization in goal-oriented dialog. In *NIPS*.

Mike Lewis, Denis Yarats, Yann N. Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning for negotiation dialogues. In *EMNLP*, pages 2443–2453.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *NAACL*, pages 110–119.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. In *IJCNLP*, pages 733–743.

Xiujun Li, Zachary C Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016b. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. 2018. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *AAAI*, pages 5237–5244.

Bing Liu and Ian Lane. 2017. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. In *INTERSPEECH*, pages 2506–2510.

Bing Liu, Gokhan Tür, Dilek Hakkani-Tür, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *NAACL*, pages 2060–2069.

Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. 2019. Learning personalized end-to-end goal-oriented dialog. In *AAAI*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Peter Norvig and Stuart J. Russell. 1995. *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *ACL*, pages 2182–2192.

Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *EMNLP*, pages 2231–2240.

Abhinav Rastogi, Raghav Gupta, and Dilek Hakkani-Tur. 2018. Multi-task learning for joint language understanding and dialogue state tracking. In *SIGDIAL*, pages 376–384.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-driven response generation in social media. In *EMNLP*, pages 583–593.

Pararth Shah, Dilek Hakkani-Tür, Bing Liu, and Gokhan Tür. 2018. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In *NAACL*, pages 41–51.

Amanda Stent, Rashmi Prasad, and Marilyn Walker. 2004. Trainable sentence planning for complex information presentation in spoken dialog systems. In *ACL*, page 79.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. ArXiv preprint arXiv:1606.02689.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. ArXiv preprint arXiv:1506.05869.

Marilyn Walker, Amanda Stent, François Mairesse, and Rashmi Prasad. 2007. Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research*, 30.

Yequan Wang, Minlie Huang, Li Zhao, and Xiaoyan Zhu. 2016. Attention-based lstm for aspect-level sentiment classification. In *EMNLP*, pages 606–615.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, pages 438–449.

Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *ACL*, pages 665–677.

Jason D. Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialogue systems. *Computer Speech & Language*, 21(2):393–422.

Jason D. Williams and Geoffrey Zweig. 2016. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. ArXiv preprint arXiv:1606.01269.

Wei-Nan Zhang, Yiming Cui, Yifa Wang, Qingfu Zhu, Lingzhi Li, Lianqiang Zhou, and Ting Liu. 2018. Context-sensitive generation of open-domain conversational responses. In *COLING*, pages 2437–2447.

Zheng Zhang, Lizi Liao, Minlie Huang, Xiaoyan Zhu, and Tat-Seng Chua. 2019. Neural multimodal belief tracker with adaptive attention for dialogue systems. In *WWW*, pages 2401–2412.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *SIGDIAL*, pages 1–10.