A User Simulator

User Goal In the task-completion dialogue setting, the first step of user simulator is to generate a feasible user goal. Generally, a user goal is defined with two types of slots: request slots that user does not know the value and expects the agent to provide it through the conversation; inform slots are slot-value pairs that user know in the mind, serving as soft/hard constraints in the dialog; slots that have multiple values are termed as soft constraints, which means user has preference, and user might change its value when there is no result returned from the agent based on the current values; otherwise, slots that have with only one value serve as hard constraint. Table 3 shows an example of a user goal in the composite task-completion dialogue.

<table>
<thead>
<tr>
<th>inform</th>
<th>request</th>
</tr>
</thead>
<tbody>
<tr>
<td>dst_city=LA</td>
<td>price=?</td>
</tr>
<tr>
<td>numberofpeople=2</td>
<td>return_time_dep=?</td>
</tr>
<tr>
<td>depart_date_dep=09-04</td>
<td>return_date_dep=?</td>
</tr>
<tr>
<td>or_city=Toronto</td>
<td>depart_time_dep=?</td>
</tr>
<tr>
<td>seat=economy</td>
<td>hotel_prices=?</td>
</tr>
<tr>
<td>hotel_city=LA</td>
<td>hotel_date_checkout=?</td>
</tr>
<tr>
<td>hotel_numberofpeople=2</td>
<td>hotel_name=?</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: An example of user goal

First User Act This work focuses on user-initiated dialogues, so we randomly generate a user action as the first turn (a user turn). To make the first user-act more reasonable, we add some constraints in the generation process. For example, the first user turn can be inform or request turn; it has at least two informable slots, if the user knows the original and destination cities, or_city and dst_city will appear in the first user turn etc.; If the intent of first turn is request, it will contain one requestable slot.

During the course of a dialogue, the user simulator maintains a compact stack-like representation named as user agenda (Schatzmann and Young, 2009), where the user state $s_u$ is factored into an agenda $A$ and a goal $G$, which consists of constraints $C$ and request $R$. At each time-step $t$, the user simulator will generate the next user action $a_{u,t}$ based on the its current status $s_{u,t}$ and the last agent action $a_{m,t-1}$, and then update the current status $s_{u,t}$. Here, when training or testing a policy without natural language understanding (NLU) module, an error model (Li et al., 2017b) is introduced to simulate the noise from the NLU component, and noisy communication between the user and agent.

B Algorithms

Algorithm 1 outlines the full procedure for training hierarchical dialogue policies in this composite task-completion dialogue system.
Algorithm 1 Learning algorithm for HRL agent in composite task-completion dialogue

1: Initialize experience replay buffer $D_1$ for meta-controller and $D_2$ for controller.
2: Initialize $Q_1$ and $Q_2$ network with random weights.
3: Initialize dialogue simulator and load knowledge base.
4: for episode = 1:N do
5:   Restart dialogue simulator and get state description $s$
6:   while $s$ is not terminal do
7:     extrinsic_reward := 0
8:     $s_0 := s$
9:     select a subtask $g$ based on probability distribution $\pi(g|s)$ and exploration probability $\epsilon_g$
10:    while $s$ is not terminal and subtask $g$ is not achieved do
11:       select an action $a$ based on the distribution $\pi(a|s, g)$ and exploration probability $\epsilon_c$
12:          Execute action $a$, obtain next state description $s'$, perceive extrinsic reward $r^e$ from environment
13:          Obtain intrinsic reward $r^i$ from internal critic
14:     Sample random minibatch of transitions from $D_1$
15:     $y = \begin{cases} r^i & \text{if } s' \text{ is terminal} \\ r^i + \gamma \max_{a'} Q_1(s', g, a'; \theta_1) & \text{otherwise} \end{cases}$
16:    Perform gradient descent on loss $\mathcal{L}(\theta_1)$ according to equation 2
17:    Store transition $(s, g, a, r^i, \{s', g\})$ in $D_1$
18:     Sample random minibatch of transitions from $D_2$
19:     $y = \begin{cases} r^e & \text{if } s' \text{ is terminal} \\ r^e + \gamma \max_{a'} Q_2(s', g', a'; \theta_2) & \text{otherwise} \end{cases}$
20:    Perform gradient descent on loss $\mathcal{L}(\theta_2)$ according to equation 3
21:     extrinsic_reward += $r^e$
22:     $s = s'$
23:   end while
24:   Store transition $(s_0, g, extrinsic\_reward, s')$ in $D_2$
25:   end while
26: end for