**INTRODUCTION**

**If-Then Recipes/Programs**
A conditional statement of "If This, Then That": whenever the trigger condition ("This") is satisfied, the action ("That") will be performed.

**4 Components:** Trigger ("tc"), Trigger Function ("tf"), Action Condition ("ac"), Action Function ("af").

**Semantic Parsing for If-Then Recipes**
Parsing a natural language (NL) description to a corresponding If-Then recipe.

**Example:**
Create a link note on Evernote for my liked tweets.

**Application:**
- Widely adopted for Task/Routine Automation and Smart-Home: “Text me if the door is unlocked”, “Send me the weather report every day at 7AM”, etc.

**MOTIVATION**

**Description Ambiguity**
An NL description can be ambiguous or contain incomplete information.

**User “need to create”**
- General description: ac Twitter, tf New liked tweet by you, tc Evernote, af Create a link note

**Based on ~4K recipes collected from real users [1], 80% of recipe descriptions are ambiguous!**

**May fail a well-trained semantic parser.**

**INTERACTIVE SEMANTIC PARSING**

**Our Solution: Ask Human Questions**
An intelligent agent can ask clarifying questions to resolve description ambiguity.

**Our Aim:**
- Improve parsing accuracy with minimal questions, without supervision on when/what to ask.

**Hierarchical RL (HRL)**

**Hierarchical Policy [2]**

- **High-level policy** $x_{(0)}(s_{(0)})$
- **Low-level policy** $x_{(1)}(s_{(1)})$

**Rules:**
- High-level policy selects a subtask $g_1$ (i.e., completing one of the 4 components) to work on.
- Low level policy completes each subtask by taking actions to either make a prediction or ask user.
- Semantic parsing = a sequence of high/low-level decisions.

**Low-Level Policy Function:**

$\hat{x}_{(1)}^*(s_{(1)})$: the low-level state representation for subtask $s_{(1)}$

$\hat{x}_{(1)}^*(s_{(1)}) = \text{tanh}(w_{h}(s_{(1)};...;s_{(1)});v_{h};s_{(1)};...;s_{(1)}))$

We define one policy for each subtask.

**High-Level Policy Function:**

$b_i$: whether the subtask $s_{i}$ has been predicted

**Training by Rewarding**

**Low-level reward** (when taking action $a_i$ for subtask $g_i$):

$r_{tb}(s_i, a_i) = \begin{cases} 1 & \text{if } a_i = \ell_g \\ 0 & \text{if } a_i = \text{AskUser} \\ -\beta & \text{otherwise} \end{cases}$

$\ell_g$: true label of subtask $g_i$. $-\beta$ is the penalty for asking questions.

**High-level reward for $g_1$**: accumulative low-level reward for completing $g_1$.

**Optimization:**
Maximize accumulative high/low-level reward via REINFORCE [3].

**EXPERIMENTS**

**Experiment Setup**

**Dataset:**
- Training: 291,285 <NL description, Recipe> pairs from [4].
- Testing set collected & annotated by [1]:

<table>
<thead>
<tr>
<th>Size</th>
<th>CI (%)</th>
<th>VI-1/2</th>
<th>VI-3/4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>1.97</td>
<td>1.92</td>
<td>3.850</td>
<td></td>
</tr>
</tbody>
</table>

- **CI:** recipes with descriptions clear in all 4 subtasks for annotators.
- **VI-1/2:** recipes containing 1 or 2 yague subtasks for annotators.
- **VI-3/4:** recipes containing 3 or 4 yague subtasks for annotators.

**Methods to Compare:**

- **LAM [5]:** state-of-the-art, non-interactive.
- **LAM-rule:** rule-based agent, ask user when prob of prediction is lower than 0.85
- **LAM-sup:** agent with "AskUser" action, trained via SL on pseudo labels.
- **HRL:** our proposed agent trained via RL.
- **HRL-fixedOrder:** HRL with a fixed high-level order to predict to – if – ac – af, following previous work (e.g., [6]).

**Metrics:**
- C+F Acc: accuracy when all 4 components are correct.
- #Ask: number of clarifying questions.

**Simulation Evaluation**

**Model**
- Model $\text{C+F Acc}$ $\#\text{Asks}$ $\text{Accuracy}$

<table>
<thead>
<tr>
<th>LAM</th>
<th>$\text{C+F Acc}$</th>
<th>$#\text{Asks}$</th>
<th>$\text{Accuracy}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.975</td>
<td>0.006</td>
<td>0.665</td>
</tr>
<tr>
<td>LAM-rule</td>
<td>0.894</td>
<td>0.006</td>
<td>0.765</td>
</tr>
<tr>
<td>LAM-sup</td>
<td>0.939</td>
<td>0.006</td>
<td>0.815</td>
</tr>
</tbody>
</table>

**Conclusions**
- Interactive > Non-interactive.
- Rule-based agent tends to ask redundant questions.
- HRL vs. HRL-fixedOrder: HRL achieves significantly better performance with fewer questions.

**Acknowledgement**

[NSF] [Fujitsu]

**REFERENCES**

[5] Lu et al., 2016. Latest attention for if-then program synthesis. In NIPS.