Lecture 14

Multimodal Virtual Assistants

1. Web Support Tasks
2. Do-it-here

By Giovanni Campagna and Jackie Yang
Genie: Open Pretrained Assistant

Agents:
- Customer Support
- Restaurants
- Play Songs
- Turn on Lights

Generic Dialogue Models:
- Transaction Dialogue State Machine
  - Read Form-filling Instructions
  - DB Access Dialogues
  - Abstract API Dialogues

Grounding Primitives:
- Fill Web Forms
- DB Schemas
- APIs

Pretrained Language Models
From Web Instructions in NL

Customer Service Web Page

Help instructions

Help & Customer Service

Gifts, Gift Cards, and Registries ➔ Gift Cards ➔

Re redeem a Gift Card

Agent instructions

1. Ask the user for the claim code
2. Go to Redeem a Gift Card.
3. Enter the claim code and select Apply to Your Balance.

Grounding Open-Domain Instructions to Automate Web Support Tasks
Nancy Xu, Sam Masling, Michael Du, Giovanni Campagna,
Larry Heck, James Landay, Monica S Lam
RUSS (Rapid universal support system): Natural Language instructions are parsed to an intermediate DSL that can be executed interactively in a DOM.
6 actions for most web instructions

<table>
<thead>
<tr>
<th>Agent Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>@goto(url)</td>
<td>Navigate to the given URL</td>
</tr>
<tr>
<td>@enter(element_id, dict_key)</td>
<td>Find the closest match to the given dictionary key and enter its value in the given input element</td>
</tr>
<tr>
<td>@click(element_id)</td>
<td>Click on the given element</td>
</tr>
<tr>
<td>@read(element_id)</td>
<td>Read the content of the given element to the user</td>
</tr>
<tr>
<td>@say(message)</td>
<td>Read the given message to the user</td>
</tr>
<tr>
<td>@ask(dict_key)</td>
<td>Ask the user for the value of a dictionary key</td>
</tr>
</tbody>
</table>
Mapping Instructions to Actions

**Instruction**: “Enter the user’s order number in the text field that says order number”

**DOM**:

- element_id: 1, type = "body"
- element_id: 2, type = "h1", text = "Your Orders"
- element_id: 3, type = "form"
- ...
- element_id: 48, type = "label", text = "order number"
- element_id: 49, type = "input"
- ...

**Action**: `@enter(text = order_number, element = 49)`
**ThingTalk: Intermediate Representation**

**Instruction:** “Enter the user’s order number in the text field that says order number”

**DOM:**
```
element_id: 1, type = "body"
  element_id: 2, type = "h1", text = "Your Orders"
  element_id: 3, type = "form"
    ...
  element_id: 48, type = "label", text = "order number"
  element_id: 49, type = "input"
```

**ThingTalk:**
```
@retrieve(description = “order number”, type = input)
  ⇒ @enter(text = order_number, element = id)
```

**Action:** @enter(text = order_number, element = 49)
@retrieve: Grounding ThingTalk to the DOM

**Instruction**: “Enter the user’s order number in the text field that says order number”

**DOM**:
```
element_id: 1, type = "body"
element_id: 2, type = "h1", text = "Your Orders"
element_id: 3, type = "form"
  ...
element_id: 48, type = "label", text = "order number"
element_id: 49, type = "input"
  ...
```

**ThingTalk**:
```
@retrieve(description = "order number", type = input)
  ⇒ @enter(text = order_number, element = id)
```

**Action**: @enter(text = order_number, element = 49)
Semantic Parser: Language -> ThingTalk

{
"@click": 0.03,  
"@enter": 0.08,  
"description": 0.06,  
"color": 0.01, ...
}
Training

- Template-based synthesis method to generate sample instruction -> ThingTalk pairs.

- Total: 1.5M training samples from roughly 840 distinct templates.
@retrieve(
  description="order number",
  type=input
)

Features
  type=input
  description="order number"

DOM:
  element_id: 1, type = "body"
  element_id: 2, type = "h1", text = "Your Orders"
  element_id: 3, type = "form"
  ...
  element_id: 48, type = "label", text = "order number"
  element_id: 49, type = "input"
  ...

filter by type and location

encode and score text with SentenceBERT

element id = 49
Evaluation: The RUSS Evaluation Dataset

**Instruction:** “Enter the user’s order number in the text field that says order number”

**DOM:**
- element_id: 1, type = "body"
- element_id: 2, type = "h1", text = "Your Orders"
- element_id: 3, type = "form"
  ...
- element_id: 48, type = "label", text = "order number"
- element_id: 49, type = "input"

**ThingTalk:**
- @retrieve(description = "order number", type = input)
  ⇒ @enter(text = order_number, element = id)

**Action:** @enter(text = order_number, element = 49)
# Accuracy on Semantic Parsing

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUSS (1.5M training parses)</td>
<td>87.0%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ablations</th>
<th>Accuracy (dev)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUSS (1.5M training parses)</td>
<td>88.2%</td>
</tr>
<tr>
<td>- entity extraction</td>
<td>77.6%</td>
</tr>
<tr>
<td>- 1M training parses, entity extraction</td>
<td>70.0%</td>
</tr>
</tbody>
</table>
# Accuracy on Grounded Instructions

<table>
<thead>
<tr>
<th>Model</th>
<th>Grounding Acc (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUSS</td>
<td>63.6%</td>
</tr>
<tr>
<td>End-to-End Baseline</td>
<td>51.1%</td>
</tr>
<tr>
<td>PhraseNode</td>
<td>46.5%</td>
</tr>
</tbody>
</table>
End-To-End Evaluation: User Study

# 1 Redeem Amazon Gift Card
# 2 Get Pinterest Ad Account Number
# 3 Log out of all Spotify accounts
# 4 Create new Walmart account
# 5 Send Google feedback
Conclusion

• Trained once, run on any web service instructions
• Read instructions → answer calls immediately

• 76.6% end-to-end accuracy
• 69% of users prefer RUSS over following web instructions
DoThisHere

How to use multi-modal interaction to solve the problem of cross-app tasks on mobile devices

Jackie (Junrui) Yang
Monica S. Lam
James A. Landay

UIST 2020
Users do many cross-app tasks on their smartphones

- 31.8% of smartphone usage sessions involved two or more apps [1]
- Our survey on MTurk (N=75) shows eight common categories of cross-app tasks

Users do many cross-app tasks on their smartphones
Current interfaces offer limited support for information transfer.

For simple information, users have to read info and remember the result.

For complex information, users have to go back and forth between apps.

Hard to get it right with voice.
Cross-app tasks on laptops and desktops is easy

- Multiple windows
- Clipboard (⌘+C)
But it doesn’t work as well on a mobile phone

• Small screen size
• Limited windowing support
Multimodal to the rescue

- Prior work [1] on multimodal interaction showed that:
  - User prefer to reply on touch(pen) input for specification of location information
  - User prefer to use speech for issuing commands or out-of-view objects

How would you solve the information transfer problem?

• Prior work [1] on multimodal interaction showed that:
  • User prefer to reply on touch(pen) input for specification of location information
  • User prefer to use speech for issuing commands or out-of-view objects

• Information transfer on mobile device:
  • For simple information, user have to read info and remember the result
  • For complex information, users have to go back and forth between apps
DoThisHere introduces: Query, Do, and Keep

For **simple information**, user can use **Query** and **Do** to transfer information.

For **complex information**, user can keep part of the interface for reference.

---

**Insert my duo code here.**

**Show me the review on IMDB**

**Keep this.**
DoThisHere can be portable to other platforms

DoThisHere is based on existing mobile infrastructure.
It’s hard to gain support from third-party apps initially. Leveraging existing app integration in virtual assistants mitigates this “cold start problem”.
DoThisHere can support majority of cross-apps tasks

• Among the cross-app tasks we collected, DoThisHere can help on 95% of the unsupported tasks
DoThisHere may help reduce users’ cognitive load

- We asked users to complete different cross-app tasks using DoThisHere and Baseline Android multitasking.
DoThisHere may help reduce users’ cognitive load

- Our user study (N=12) indicated that DoThisHere may help on reducing user’s cognitive load (NASA-TLX)

Task: 1: Keep 2, 3: Query 4: Do

*: p< 0.05, **: p < 0.01
We believe every smartphone in the future should be equipped with DoThisHere to help us interact with our smartphone naturally and productively.

Question?: jackiey@stanford.edu