Task-Oriented Dialogue

Chi Qi, Michael Hu
What is task-oriented dialogue?
**An Example of Task-Oriented Dialogue**

<table>
<thead>
<tr>
<th>GUS:</th>
<th>Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client:</td>
<td>I want to go to San Diego on May 28</td>
</tr>
<tr>
<td>GUS:</td>
<td>What time do you want to leave?</td>
</tr>
<tr>
<td>Client:</td>
<td>I must be in San Diego before 10 am</td>
</tr>
<tr>
<td>GUS:</td>
<td>Would you like P.S.A. flight 102 that arrives at 9.15 am?</td>
</tr>
<tr>
<td>Client:</td>
<td>What is the next flight?</td>
</tr>
<tr>
<td>GUS:</td>
<td>Air California flight 310 that leaves at 8.30am and arrives at 10.00am</td>
</tr>
<tr>
<td>Client:</td>
<td>I’ll take the first one</td>
</tr>
<tr>
<td>GUS:</td>
<td>In what name should I make a reservation, please?</td>
</tr>
<tr>
<td>Client:</td>
<td>Dan Bobrow</td>
</tr>
<tr>
<td>GUS:</td>
<td>I have confirmed the following flight: P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego that leaves at 7.30 am and arrives at 9.15 am.</td>
</tr>
<tr>
<td>GUS:</td>
<td>What date do you want to return on?</td>
</tr>
<tr>
<td>Client:</td>
<td>On Friday in the evening.</td>
</tr>
<tr>
<td>GUS:</td>
<td>Would you like the flight that leaves at 7.45 pm?</td>
</tr>
<tr>
<td>Client:</td>
<td>That’s fine.</td>
</tr>
<tr>
<td>GUS:</td>
<td>I have confirmed the following flight: P.S.A. flight 307 on Friday May 30 from San Diego to San Jose that leaves at 7.45 pm and arrives at 9.30 pm Thank you for calling. Goodbye</td>
</tr>
</tbody>
</table>

---

**Figure 26.8** The travel domain: A transcript of an actual dialogue with the GUS system of Bobrow et al. (1977). P.S.A. and Air California were airlines of that period.
Task-Oriented Dialogue (TOD) Systems

- Help users achieve their specific goals
- Focus on understanding users, tracking states, and generating next actions.
- Minimize the number of turns: fewer turns the better
Key Terms

- **Domain ontology**: a set of knowledge structures representing the kinds of intentions the system can extract from user sentences.
- **Domain**: a domain consists of a collection of slots.
- **Slot**: each of slot can take a set of possible values.

<table>
<thead>
<tr>
<th>Slot</th>
<th>Type</th>
<th>Question Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORIGIN CITY</td>
<td>city</td>
<td>“From what city are you leaving?”</td>
</tr>
<tr>
<td>DESTINATION CITY</td>
<td>city</td>
<td>“Where are you going?”</td>
</tr>
<tr>
<td>DEPARTURE TIME</td>
<td>time</td>
<td>“When would you like to leave?”</td>
</tr>
<tr>
<td>DEPARTURE DATE</td>
<td>date</td>
<td>“What day would you like to leave?”</td>
</tr>
<tr>
<td>ARRIVAL TIME</td>
<td>time</td>
<td>“When do you want to arrive?”</td>
</tr>
<tr>
<td>ARRIVAL DATE</td>
<td>date</td>
<td>“What day would you like to arrive?”</td>
</tr>
</tbody>
</table>
The domain ontology defines the set of actions our model can take.
The ontology file, specific all the values the three informable slots can take.

```json
{
  "informable": {
    "area": ["centre", "north", "west", "south", "east"],
    "pricerange": ["cheap", "moderate", "expensive"]
  }
}
```
Natural language understanding for filling slots

“Show me morning flights from Boston to San Francisco on Tuesday”

- **Task #1: Domain Classification**
  - DOMAIN: AIR-TRAVEL
- **Task #2: Intent Determination**
  - INTENT: SHOW-FLIGHTS
- **Task #3: Slot Filling**
  - ORIGIN-CITY: Boston
  - ORIGIN-DATE: Tuesday
  - ORIGIN-TIME: morning
  - DEST-CITY: San Francisco
How is TOD different from other tasks?

1. Domain specificity.
   ○ A resulting challenge: lack of training data.

2. End goal: helping the user DO something.
   ○ Model must understand user & what they want
   ○ → Requires a deep understanding of dialogue progression

3. A focus on brevity and efficiency
Early Approaches

Approach 1: Rules-based systems.

Approach 2: Dialogue State Architecture
Rule-based systems

Consist of large hand-designed semantic grammars with thousands of rules.

- **SHOW** → show me | i want | can i see | ...
- **DEPART_TIME_RANGE** → (after | around | before) HOUR | morning | afternoon | evening
- **HOUR** → one | two | three | four | eleven (AMPM)
- **FLIGHTS** → (a) flight | flights
- **AMPM** → am | pm
- **ORIGIN** → from CITY
- **DESTINATION** → to CITY
- **CITY** → Boston | San Francisco | Denver | Washington

E.g., Phoenix system (Ward and Issar, 1994)
Rule-based - finite state dialogue manager
The dialogue state architecture

Task-Oriented Dialogue System (Young, 2000)

Model is not end-to-end.
The dialogue state architecture - mostly data-driven

- The Dialogue State Tracker maintains the current state of the dialogue
- A more sophisticated Dialogue Policy compared to rule-based architecture
- A more sophisticated Natural Language Generating component

Data-driven vs Rule-based systems

★ Dialogue manager is more flexible and evolvable.
★ Learn from interaction - need more data, but less hand-craft rules
★ May have surprising/uncontrolled responses in unseen scenarios
A core challenge of task oriented dialogue is getting relevant training data.
Solution: Wizard of Oz (WOZ) Data Collection

**Wizard-of-Oz data collection**: Users think they're talking to a computer, but they're actually talking to a human.

Humans pretending to be computers are called "wizards."

Circa 2016: Wen et al. needed training data relevant to restaurant selection in Cambridge, UK!

Solution: Amazon Mechanical Turk.
Data Collection

The flow: User 1 → Wizard 1 → User 2 → Wizard 2. Each person contributes 1 line to the conversation.

Task 02004: You are looking for and it should serve gastropub food. You don't care about the price range. You want to know the address.

Info Desk: Hello, welcome to the Cambridge restaurant system. You can ask for restaurants by area, price range or food type. How may I help you?
Info Desk: I want a gastropub food
Info Desk: There are 4 restaurants serving gastropub food, what price range do you want?

Customer: (Your response)

I don't care about the price range, just give me the address please.
Info Desk: Hello, welcome to the Cambridge restaurant system. You can ask for restaurants by area, price range or food type. How may I help you?
Customer: I want a gastropub food
Info Desk: There are 4 restaurants serving gastropub food, what price range do you want?
Customer: I don't care.

Please modify the following answers based on the latest customer response:
- What does user want?
  - What is the food type the user wants? gastropub
  - What is the area the user wants? not mentioned
  - What is the price range the user wants? don't care

- What does user ask?
  - Is the user asking for food type of an offered venue? No
  - Is the user asking for price range of an offered venue? No
  - Is the user asking for area of an offered venue? No
  - Is the user asking for postcode of an offered venue? No
  - Is the user asking for phone number of an offered venue? No
  - Is the user asking for address of an offered venue? No
  - Is the user mentioning any restaurant names? No

**Info Desk:** (Your responses)

I would recommend Backstreet Bar, a great gastropub restaurant in the centre. Do you want their phone number? [Submit the HT]

<table>
<thead>
<tr>
<th>Name</th>
<th>Food</th>
<th>Area</th>
<th>Price Range</th>
<th>Phone</th>
<th>Address</th>
<th>Postcode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backstreet Bar</td>
<td>gastropub</td>
<td>centre</td>
<td>expensive</td>
<td>01223 308305</td>
<td>2 Stilton Street City Centre</td>
<td>CB1 2GA</td>
</tr>
<tr>
<td>Royal Standard</td>
<td>gastropub</td>
<td>East</td>
<td>expensive</td>
<td>01223 247877</td>
<td>280 Mil Road City Centre</td>
<td>CB1 3NL</td>
</tr>
<tr>
<td>The One Pizza Kitchen and Bar</td>
<td>gastropub</td>
<td>Centre</td>
<td>moderate</td>
<td>01223 308671</td>
<td>34 - 36 Exchange Street</td>
<td>CB2 3QF</td>
</tr>
<tr>
<td>The Slug and Lettuce</td>
<td>gastropub</td>
<td>Centre</td>
<td>expensive</td>
<td>--</td>
<td>34 - 36 Green Street</td>
<td>CB2 3UJ</td>
</tr>
</tbody>
</table>

Showing 1 to 4 of 4 entries (filtered from 110 total entries)
Data Collection

Resulting training data is very domain specific: both good and bad.

1500 total dialogue turns → 680 total dialogues.

60-20-20 data split. Result: Training set of 408 dialogues.

Cost: $400. About $1 per training example.

Data collection is the main bottleneck of this task. Can we do better?
More data: Multi-domain WOZ (MultiWOZ)

- EMNLP 2018
- Setting: a tourist and a clerk
- Same collection method as Cambridge data set
- Seven domains (Hotel, Train, Attraction, Restaurant, Taxi, Hospital, and Police) and 16 slots (food, leave at, area, etc).
## Dataset comparison

<table>
<thead>
<tr>
<th>Metric</th>
<th>DSTC2</th>
<th>SFX</th>
<th>WOZ2.0</th>
<th>FRAMES</th>
<th>KVRET</th>
<th>M2M</th>
<th>MultiWOZ</th>
</tr>
</thead>
<tbody>
<tr>
<td># Dialogues</td>
<td>1,612</td>
<td>1,006</td>
<td>600</td>
<td>1,369</td>
<td>2,425</td>
<td>1,500</td>
<td>8,438</td>
</tr>
<tr>
<td>Total # turns</td>
<td>23,354</td>
<td>12,396</td>
<td>4,472</td>
<td>19,986</td>
<td>12,732</td>
<td>14,796</td>
<td>113,556</td>
</tr>
<tr>
<td>Total # tokens</td>
<td>199,431</td>
<td>108,975</td>
<td>50,264</td>
<td>251,867</td>
<td>102,077</td>
<td>121,977</td>
<td>1,490,615</td>
</tr>
<tr>
<td>Avg. turns per dialogue</td>
<td>14.49</td>
<td>12.32</td>
<td>7.45</td>
<td>14.60</td>
<td>5.25</td>
<td>9.86</td>
<td>13.46</td>
</tr>
<tr>
<td>Avg. tokens per turn</td>
<td>8.54</td>
<td>8.79</td>
<td>11.24</td>
<td>12.60</td>
<td>8.02</td>
<td>8.24</td>
<td>13.13</td>
</tr>
<tr>
<td>Total unique tokens</td>
<td>986</td>
<td>1,473</td>
<td>2,142</td>
<td>12,043</td>
<td>2,842</td>
<td>1,008</td>
<td>23689</td>
</tr>
<tr>
<td># Slots</td>
<td>8</td>
<td>14</td>
<td>4</td>
<td>61</td>
<td>13</td>
<td>14</td>
<td>24</td>
</tr>
<tr>
<td># Values</td>
<td>212</td>
<td>1847</td>
<td>99</td>
<td>3871</td>
<td>1363</td>
<td>138</td>
<td>4510</td>
</tr>
</tbody>
</table>
Evaluating Task-Oriented Dialogue Systems is also a challenging task.
Human-based evaluation

- **Lab-experiments:** Users were invited to participate in the lab where they interacted with the dialogue system and subsequently filled a questionnaire [Young et al., 2010]. - very controlled, not comparable to real world

- **In-field experiments:** collecting feedback from real users of the dialogue systems - e.g., the Spoken Dialogue Challenge [Black et al., 2011]

- **Crowdsourcing:** using crowdsourcing platforms such as Amazon Mechanical Turk (AMT) - high variability of user behaviour

**Difficult to set-up and to carry out:**
the users need to be properly instructed, the tasks need to be prepared so that the experiment is close to real-world conditions.
Automated Evaluation metrics

- **Dialogue State Tracker performance**
  - End-to-end: Precision, Recall, F-1
  - TRADE: joint and slot accuracy

- **Dialogue Efficiency** [# turns]

- **Corpus Based Evaluation for e2e system**
  - BLEU score
  - Entity matching rate
Checkpoint 1. Questions?

- Introduction to TOD
- Early approaches
- Data collection
- Evaluation
This is a challenging task.
Seq-to-seq for TOD

Now that we’ve introduced TOD and its core challenges, let’s motivate the 1st paper.
A new idea: seq-to-seq for TOD

(Sutskever et al., 2014) <- Machine Translation, Elisabetta & Ben

(Vinyals & Le, 2015) <- Dialogue, Xinyi & Paula

Wen et al: Let's use seq-to-seq for task-oriented dialogue!
A Network-based End-to-End Trainable Task-oriented Dialogue System

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšic, Milica Gašic, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young
I need to find an expensive restaurant that's in the south of the city.

There are several restaurants in the south part of town that serve expensive food. Do you have a cuisine preference?

No I don't care about the type of cuisine.

Chiquito Restaurant Bar is a Mexican restaurant located in the south part of town.

Where is it?

There address is 2G Cambridge Leisure Park Cherry Hinton Road Cherry Hinton, it there anything else I can help you with?

Nope, that's what I was looking for. Thanks a lot!

Thank you very much for using the Cambridge Restaurant System.
Setting

There are 110 restaurants in the DB, each with 9 attributes.

3 Informable slots (constraints)
- food type, price range, area

6 Requestable slots (follow-up questions)
- address, phone number, area code
- food type, price range, area
Delexicalization: replacing specific values with generic tokens. Allows for weight sharing.
"I want tacos."

"I want <v.food>.

"<v.name> has great <v.food>.

"Tacoria has great tacos."
Wen et al.’s model combines SOTA subnetworks into one big model with impressive performance.
1. Intent Network

Pretty straightforward: the encoder of a classic seq-to-seq model.

Role: Natural Language Understanding

Authors tried:

- LSTM
- CNN
2. Belief Tracker
Belief Tracker

Maps input sequence to a distribution over values. Slot-value pairs are things like price → expensive, food type → Tex-mex, etc.

Role: Dialogue State Tracker.

Intent network → sentence level

Belief network → conversation level
Belief Trackers

The model uses a belief tracker per slot.

Model samples representations from all 3 layers.

This doesn’t scale. What if your DB has 100 rows?
You can think of belief trackers as long range parsers.
3. Database Operator

\[ q_t = \bigcup_{s' \in S_I} \{ \arg \max_v p_{s'}^t \} \]

Take the most likely values out of each of 3 informable slots, and write as a SQL query.

Using query results, assign a \{0, 1\} vector over the fields in the database. 1 = relevant.

Finally, point to an entity at random. This entity has an associated phone number, price point, etc.

Role: Dialogue Policy
4. Policy Network

A feed-forward layer. The glue holding all the gradients together.

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

- \( z_t \): intent network output
- \( \hat{p}_t \): belief vector for each slot
- \( \hat{x}_t \): number of DB hits, as a one-hot vector
5. Generation Network

Generation Network

<v.name> serves great <v.food>

Conditioner  DB pointer
Generation Network

Step 1: Generate auto-regressively using an LSTM

\[ P(w_{j+1}^t | w_j^t, h_{j-1}^t, o_t) = \text{LSTM}_j(w_j^t, h_{j-1}^t, o_t) \]

Conditioner

Step 2: Replace delexicalized tokens with DB pointer values.

\(<v\text{name}> \text{ has great } <v\text{food}>. \Rightarrow \text{Tacoria has great tacos.}\)

Role: Natural Language Generation
Optimized: The Attentive Generation Network

$$\alpha_s^{(j)} = \text{softmax}(r^T \tanh(W_r \cdot u_t)) \quad (12)$$

Compute attention weights by looking at literally all the representations that we have.

$$\hat{p}_l^{(j)} = \sum_{s \in G} \alpha_s^{(j)} \tanh(W_{po}^s \cdot \hat{p}_s^t) \quad (11)$$

Use attention weights to recompute probability distribution.

$$o_t^{(j)} = \tanh(W_{zo}z_t + \hat{p}_l^{(j)} + W_{xo}\hat{x}_t) \quad (10)$$

Recompute the conditioning vector.
Model Training

Step 1: Train belief networks using CEL between wizard labels and belief network distributions.
- Train on dialogue state.

Step 2: Train end-to-end using CEL between wizard sentences and machine predictions.
- Train on response.
"usr": {
  "transcript": "I need to find an expensive restaurant that's in the south section of the city.",
  " slu": [ 
    {
      "act": "inform",
      " slots": [ 
        [ 
          "pricerange",
          "expensive"
        ]
      ]
    },
    {
      "act": "inform",
      " slots": [ 
        [ 
          "area",
          "south"
        ]
      ]
    }
  ]
},
" sys": {
  "sent": "There are several restaurants in the south part of town that serve expensive food. Do you have a cuisine preference?"
}
Decoding

Beam search with beam size 10.

\[
    m_t^* = \arg\max_{m_t} \log \frac{p(m_t | \theta, u_t)}{J_t} \\
    m_t^* = \arg\max_{m_t} \left\{ \log p(m_t | \theta, u_t) / J_t \right\}
\]

Highest per-token log probability

“Weighted” decoding

\[
    \lambda \log p(m_t) / J_t + \gamma R_t \}
\]

Reward heuristic:
More reward if the model generates an address when an address is requested

Use a separate language model to predict probability of generating each word
**Evaluation: Belief Trackers**

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

**Conclusion:** Belief trackers learn how to parse commands into a distribution over slot values.

**Precision:** % time requested slot value returned.

**Recall:** % of info returned that was actually requested.

**n-grams go up to trigram.**
Evaluation: Models

Quantitative metrics: BLEU, entity match rate, and success rate.

- **BLEU**: computed on delexicalized forms
- **Entity match rate**: % recommendations of correct type:
  - E.g. You ask for tacos, and the model recommends Tacoria
- **Success rate**: % time entity matches, **and** all follow-up questions are answered.

Qualitative metrics, out of 5: comprehension, naturalness
Evaluation: Models

<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>Baseline</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>Variant</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>Full model w/ different decoding strategy</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</td>
<td>+ weighted</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</td>
<td>+ att.</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</td>
<td>+ att. + weighted</td>
<td>90.88</td>
<td>83.82</td>
<td>0.2304</td>
<td>0.2369</td>
</tr>
</tbody>
</table>

Note: A low BLEU score is okay, as long as success rate is high. We measure success and BLEU using delexicalized forms.
The model learns something!

Figure 3: The action vector embedding $o_t$ generated by the NN model w/o attention. Each cluster is labelled with the first three words the embedding generated.

Clusters generated with t-SNE. t-SNE: a probabilistic cousin of PCA.
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>
</tr>
</thead>
<tbody>
<tr>
<td>Success</td>
<td>98%</td>
</tr>
<tr>
<td>Comprehension</td>
<td>4.11</td>
</tr>
<tr>
<td>Naturalness</td>
<td>4.05</td>
</tr>
<tr>
<td># of dialogues</td>
<td>245</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>NDM</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>
</tbody>
</table>

Comparisons (%)

<table>
<thead>
<tr>
<th>Metric</th>
<th>NDM</th>
<th>HDC</th>
<th>Tie</th>
</tr>
</thead>
<tbody>
<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>
Limitations

The model:

● Cannot handle noisy dialogue.
● Cannot ask user for clarification.
● Gives only 1 recommendation at a time, by construction.
● Cannot generalize. (A limitation our 2nd paper tries to address!)
Takeaways

Wen et al.’s model:

1. Demonstrates that a seq-to-seq approach can work for the task-oriented dialogue task.
2. Composes several SOTA models end-to-end.
3. Defines a procedure for generating data.

A successful proof of concept!
How many different components are there in a task-oriented dialogue system? How are each component and the full system evaluated?
How many different components are there in a task-oriented dialogue system? How are each component and the full system evaluated?

There are 4 components.

1. Natural language understanding, evaluated end-to-end.
2. Dialogue state tracking, evaluated with F1 score.
3. Dialogue policy, evaluated end-to-end.
4. Natural language generation, evaluated end-to-end.

The full model is evaluated on BLEU, entity match rate, success rate, and qualitative metrics such as naturalness.
Checkpoint 2. Questions?
Generalization in TOD

How can we make our model more robust to unseen slot values?

How can we reduce dependence on an ontology?
Transferable Multi-Domain State Generator for Task-Oriented Dialogue Systems

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, Pascale Fung
Dialogue State Tracking (DST)

- A key subtask in dialogue management - Representation of the system's belief of the user's goal(s) at any time during the dialogue given the dialogue history.

Challenges:
- How to define the state space?
- How to tractably maintain the dialogue state?
- Which actions to take for each state?
- Multi-domain, Multi-turn Conversations?
## Dialogue State Tracking (DST)

<table>
<thead>
<tr>
<th>Utterance</th>
<th>Food</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>S</strong> Hello, How may I help you?</td>
<td>Persian</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> I need a Persian restaurant in the south part of town.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> What kind of food would you like?</td>
<td>Persian</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> Persian.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> I’m sorry but there is no restaurant serving persian food.</td>
<td>Portuguese</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> How about Portuguese food?</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> Peking restaurant is a nice place in the south of town.</td>
<td>Portuguese</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> Is that Portuguese?</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> Nandos is a nice place in the south of town serving tasty Portuguese food.</td>
<td>Portuguese</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> Alright. What’s the phone number?</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> The phone number of nandos is 01223 327908.</td>
<td>Portuguese</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> And the address?</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>S</strong> Sure, nandos is on Cambridge Leisure Park Clifton Way.</td>
<td>Portuguese</td>
<td>South</td>
</tr>
<tr>
<td><strong>U</strong> Thank you good bye.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
An example of multi-domain DST

I’m looking for a cheap pizza restaurant in the city center.

Sure. There is D’angelo Pizzeria nearby. How many guests?

Restaurant:
{Price: Cheap,
Type: Pizza,
Area: Center}
An example of multi-domain DST

I’m looking for a cheap pizza restaurant in the city center.

Sure. There is D’angelo Pizzeria nearby. How many guests?

Three people Wednesday at 11am please. Please make sure there’s NO PINEAPPLE on the pizza!

Booked! QWERT is your reservation code.
An example of multi-domain DST

I’m looking for a cheap pizza restaurant in the city center.

Sure. There is D’angelo Pizzeria nearby. How many guests?

Three people Wednesday at 11am please. Please make sure there’s NO PINEAPPLE on the pizza!

Booked! QWERT is your reservation code.

Also looking for some architectural attractions close to the restaurant.

All Saints Church is famous. Would you like to head there?
An example of multi-domain DST

I’m looking for a cheap pizza restaurant in the city center.

Sure. There is D’angelo Pizzeria nearby. How many guests?

Three people Wednesday at 11am please. Please make sure there’s NO PINEAPPLE on the pizza!

Booked! QWERT is your reservation code.

Also looking for some architectural attractions close to the restaurant.

All Saints Church is famous. Would you like to head there?

Yes help me book a taxi between the restaurant and the church.

What time do you need the taxi?

Around 1:30 pm please.
Ontology-based DST

- Given system response and current user utterance, each slot in each domain is predicted to be one of the predefined values in ontology (e.g., the belief tracker in Wen et al. 2016).

Challenges:

- Ontology is hard to obtain in real scenarios
- Need to track lots of slot values
- Cannot track unseen slot values
- Missing domain sharing capacities

DST without ontology?
DST without ontology intuition

Usr: Find me a cheap restaurant at 7 pm.
Sys: What cuisine would you like?
Usr: I’d prefer eating sushi or ramen.
Sys: Where should it be?
Usr: Let’s do in Florence. Also I need a taxi to go there at 6:30 pm.
Sys: ...?

Dialogue History
Encoder
**DST without ontology intuition**

*USR:* Find me a cheap restaurant at 7 pm.
*SYS:* What cuisine would you like?
*USR:* I’d prefer eating **sushi** or **ramen**.
*SYS:* Where should it be?
*USR:* Let’s do in Florence. Also I need a taxi to go there at 6:30 pm.
*SYS:* ...?
DST without ontology intuition

User: Find me a cheap restaurant at 7 pm.
System: What cuisine would you like?
User: I’d prefer eating sushi or ramen.
System: Where should it be?
User: Let’s do in Florence. Also I need a taxi to go there at 6:30 pm.
System: ...?
DST without ontology intuition

**User (Usr):** Find me a cheap restaurant at 7 pm.
**System (Sys):** What cuisine would you like?
**Usr:** I'd prefer eating sushi or ramen.
**Sys:** Where should it be?
**Usr:** Let’s do in Florence. Also I need a taxi to go there at 6:30 pm.
**Sys:** ...?

Diagram:
- **Dialogue History Encoder**
- **State Generator**
- **Taxi & departure**
- **None**
Sequence-to-Sequence (Seq2Seq)

\[ P_{jk}^{\text{vocab}} = \text{Softmax}(E(h_{jk}^{\text{dec}})^T) \in \mathbb{R}^{V_l} \]

Vocabulary Distribution
Seq2Seq with Attention

\[ P_{j_i}^{history} = \text{Softmax}(H_i(h_{j_i}^{dec})^T) \in \mathbb{R}^{|X_i|}. \]
Seq2Seq with Soft Copy Mechanism (See et al. 2017)

\[ P_{j|k}^{\text{final}} = P_{j|k}^{\text{gen}} \times P_{j|k}^{\text{vocab}} + (1 - P_{j|k}^{\text{gen}}) \times P_{j|k}^{\text{history}} \in \mathbb{R}^{|V|} \]

\[ p_{j|k}^{\text{gen}} = \text{Sigmoid}(W_1[h_{j|k}^{\text{dec}}, w_{j|k}, c_{j|k}]) \in \mathbb{R}^1 \]
Transferable Dialogue State Generator (TRADE)

(a) Utterance Encoder

(b) State Generator

(c) Slot Gate $G_j = \text{Softmax}(W_g \cdot (c_{j0})^T) \in \mathbb{R}^3$

Utterances

Bot: Which area are you looking for the hotel?
User: There is one at east town called Ashley Hotel.

Domains
Hotel, Train, Attraction, Restaurant, Taxi

Slots
Price, Area, Day, Departure, name, LeaveAt, food, etc.

......
TRADE Optimization

\[
L_g = \sum_{j=1}^{J} -\log(G_j \cdot (y_j^{\text{gate}})^\top)
\]

(b) Ashley

\[
L_v = \sum_{j=1}^{J} \sum_{k=1}^{|Y_j|} -\log(P_{j_k}^{\text{final}} \cdot (y_{j_k}^{\text{value}})^\top)
\]

State Generator

(c)

Slot Gate \( G_j \)

Context Vector \( C_{j0} \)

Utterance Encoder

Bot: Which area are you looking for the hotel?
User: There is one at east town called Ashley Hotel.

Hotel?

Ex: hotel

\( h_{j0}^{\text{dec}} \)

Ex: name

Domains

Hotel, Train, Attraction, Restaurant, Taxi

Slots

Price, Area, Day, Departure, name, LeaveAt, food, etc.

Ashley

\( P_{j0}^{\text{final}} \)

\( (y_{j0}^{\text{gate}})^\top \)
The dataset info of MultiWOZ - 30 (domain, slot) pairs

<table>
<thead>
<tr>
<th></th>
<th>Hotel</th>
<th>Train</th>
<th>Attraction</th>
<th>Restaurant</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Slots</strong></td>
<td>price, type, parking, stay, day, people, area, stars, internet, name</td>
<td>destination, departure, day, arrive by, leave at, people</td>
<td>area, name, type</td>
<td>food, price, area, name, time, day, people</td>
<td>destination, departure, arrive by, leave by</td>
</tr>
<tr>
<td><strong>Train</strong></td>
<td>3381</td>
<td>3103</td>
<td>2717</td>
<td>3813</td>
<td>1654</td>
</tr>
<tr>
<td><strong>Valid</strong></td>
<td>416</td>
<td>484</td>
<td>401</td>
<td>438</td>
<td>207</td>
</tr>
<tr>
<td><strong>Test</strong></td>
<td>394</td>
<td>494</td>
<td>395</td>
<td>437</td>
<td>195</td>
</tr>
</tbody>
</table>
Multi-domain DST Evaluation metrics

- Joint goal accuracy
  - Compares the predicted dialogue states to the ground truth $B_t$ at each dialogue turn $t$
  - Correct output iff all the predicted values exactly match $B_t$

- Slot accuracy
  - Individually compares each (domain, slot, value) triplet to its ground truth label
Results

TRADE - highest performance on joint goal accuracy

Potential limitations of other models:

- MDBT, GLAD, and GCE all need a predefined domain ontology
- SpanPtr uses index-based copying

<table>
<thead>
<tr>
<th></th>
<th>MultiWOZ</th>
<th>MultiWOZ (Only Restaurant)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Joint</td>
<td>Slot</td>
</tr>
<tr>
<td>MDBT</td>
<td>15.57</td>
<td>89.53</td>
</tr>
<tr>
<td>GLAD</td>
<td>35.57</td>
<td>95.44</td>
</tr>
<tr>
<td>GCE</td>
<td>36.27</td>
<td>98.42</td>
</tr>
<tr>
<td>SpanPtr</td>
<td>30.28</td>
<td>93.85</td>
</tr>
<tr>
<td>TRADE</td>
<td><strong>48.62</strong></td>
<td><strong>96.92</strong></td>
</tr>
</tbody>
</table>
Embeddings cosine similarity visualization

- The rows and columns are all the possible slots in MultiWOZ.
- Slots that share similar values or have correlated values learn similar embeddings.
Unseen Domain DST - zero shot

- Zero-shot setting:
  - No training data in the new domain
  - Generate target values given the context X, target domain D, and target slot S without using any training samples
    - [e.g., train - departure -> taxi -departure].
  - Extremely challenging if the target slot has never been trained.
Zero-shot experiments on an unseen domain

- Trained Single column is the results achieved by training on 100% single-domain data as a reference.
- Taxi domain reaches good performance >60%

<table>
<thead>
<tr>
<th></th>
<th>Trained Single</th>
<th>Zero-Shot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Joint</td>
<td>Slot</td>
</tr>
<tr>
<td>Hotel</td>
<td>55.52</td>
<td>92.66</td>
</tr>
<tr>
<td>Train</td>
<td>77.71</td>
<td>95.30</td>
</tr>
<tr>
<td>Attraction</td>
<td>71.64</td>
<td>88.97</td>
</tr>
<tr>
<td>Restaurant</td>
<td>65.35</td>
<td>93.28</td>
</tr>
<tr>
<td>Taxi</td>
<td>76.13</td>
<td>89.53</td>
</tr>
</tbody>
</table>
Unseen Domain DST - few shot

- Expanding DST for Few-shot setting:
  - 1% of the original training data in the unseen domain is available (around 20 to 30 dialogues)
  - Employ two continual learning techniques - elastic weight consolidation (EWC) and gradient episodic memory (GEM) to fine-tune the model.
  - EWC loss -
    \[ L_{ewc}(\Theta) = L(\Theta) + \sum_i \frac{\lambda}{\alpha} F_i(\Theta - \Theta_{S,i})^2, \]
  - GEM training process -
    Minimize \( L(\Theta) \)
    Subject to \( L(\Theta, K) \leq L(\Theta_S, K) \),
## Domain expansion experiments by excluding one domain and fine-tuning on that domain

<table>
<thead>
<tr>
<th>Evaluation on 4 Domains</th>
<th>Joint Except Hotel</th>
<th>Slot</th>
<th>Joint Except Train</th>
<th>Slot</th>
<th>Joint Except Attraction</th>
<th>Slot</th>
<th>Joint Except Restaurant</th>
<th>Slot</th>
<th>Joint Except Taxi</th>
<th>Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Model (BM) training on 4 domains</td>
<td>58.98</td>
<td>96.75</td>
<td>55.26</td>
<td>96.76</td>
<td>55.02</td>
<td>97.03</td>
<td>54.69</td>
<td>96.64</td>
<td>49.87</td>
<td>96.77</td>
</tr>
<tr>
<td>Fine-tuning BM on 1% new domain</td>
<td><strong>Naïve</strong></td>
<td>36.08</td>
<td>93.48</td>
<td>23.25</td>
<td>90.32</td>
<td>40.05</td>
<td>95.54</td>
<td>32.85</td>
<td>91.69</td>
<td>46.10</td>
</tr>
<tr>
<td></td>
<td><strong>EWC</strong></td>
<td>40.82</td>
<td>94.16</td>
<td>28.02</td>
<td>91.49</td>
<td>45.37</td>
<td>84.94</td>
<td>34.45</td>
<td>92.53</td>
<td><strong>46.88</strong></td>
</tr>
<tr>
<td></td>
<td><strong>GEM</strong></td>
<td><strong>53.54</strong></td>
<td><strong>96.27</strong></td>
<td><strong>50.69</strong></td>
<td><strong>96.42</strong></td>
<td><strong>50.51</strong></td>
<td><strong>96.66</strong></td>
<td><strong>45.91</strong></td>
<td><strong>95.58</strong></td>
<td><strong>46.43</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Evaluation on New Domain</th>
<th>Hotel</th>
<th>Train</th>
<th>Attraction</th>
<th>Restaurant</th>
<th>Taxi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training 1% New Domain</td>
<td>19.53</td>
<td>77.33</td>
<td>44.24</td>
<td>85.66</td>
<td><strong>35.88</strong></td>
</tr>
<tr>
<td>Fine-tuning BM on 1% new domain</td>
<td><strong>Naïve</strong></td>
<td>19.13</td>
<td>75.22</td>
<td><strong>59.83</strong></td>
<td><strong>90.63</strong></td>
</tr>
<tr>
<td></td>
<td><strong>EWC</strong></td>
<td>19.35</td>
<td>76.25</td>
<td>58.10</td>
<td>90.33</td>
</tr>
<tr>
<td></td>
<td><strong>GEM</strong></td>
<td><strong>19.73</strong></td>
<td><strong>77.92</strong></td>
<td>54.31</td>
<td>89.55</td>
</tr>
</tbody>
</table>
Error Analysis - Slots error rate
Zero-shot DST error analysis

(a) Hotel

(b) Restaurant
TRADE Conclusion

- A copy-augmented generative model
- Can conduct multi-domain DST without ontology
- Enables zero-shot, and few-shot DST in an unseen domain with limited performance

Slide reference: Chien-Sheng(Jason) Wu
Further challenges

- The scale of the task-oriented corpora
- The noise and uncertainty in speech recognition
- The ambiguity when understanding human language
- The need to integrate third-party services and dialogue context in the decision-making
- The ability to generate natural and engaging responses
Q2: If we compare the dialogue state tracking models in these two papers, what is the biggest advance of the second paper (Wu et al, 2019)?
Q2: If we compare the dialogue state tracking models in these two papers, what is the biggest advance of the second paper (Wu et al, 2019)?

The biggest advance: predicts slot values directly, without pre-defining an ontology. The model is able to share parameters across different domains for multi-domain tasks.
Bonus paper - Global-Locally Self-Attentive Dialogue State Tracker (GLAD)
System actions in previous turn:
- request(food)
- request(price range)

User utterance:
I just want to eat at a cheap restaurant in the south part of town. What food types are available, can you also provide some phone numbers?

Slot value under consideration:
price range = cheap

Mathematical expressions:

\[ y = \sigma(y^{\text{utt}} + w_y y^{\text{act}}) \in \mathbb{R} \]

\[ y^{\text{utt}} = W^{\text{utt}} q^{\text{utt}} + b^{\text{utt}} \in \mathbb{R} \]

\[ y^{\text{act}} = (q^{\text{act}})^T c^{\text{val}} \in \mathbb{R} \]
References:


