Dialogue Datasets

CS294S: Building the Best Virtual Assistant

Ryan Kearns & Lucas Sato
Mentor: Giovanni Campagna

May 14, 2020

Outline

1. Introduction: Why Datasets?

2. MultiWOZ in the Almond/ThingTalk/Genie Context

3. What's In a Dataset

- a. Dialogue Generation
- b. Annotation Generation
- c. Annotation Styles

4. MultiWOZ Revisited

1. Why Datasets?

"Perhaps the most important news of our day is that datasets—not algorithms—might be the key limiting factor to development of human-level artificial intelligence."

- Alexander Wissner-Gross, 2016

Harvard University Institute for Applied Computational Science

Outline

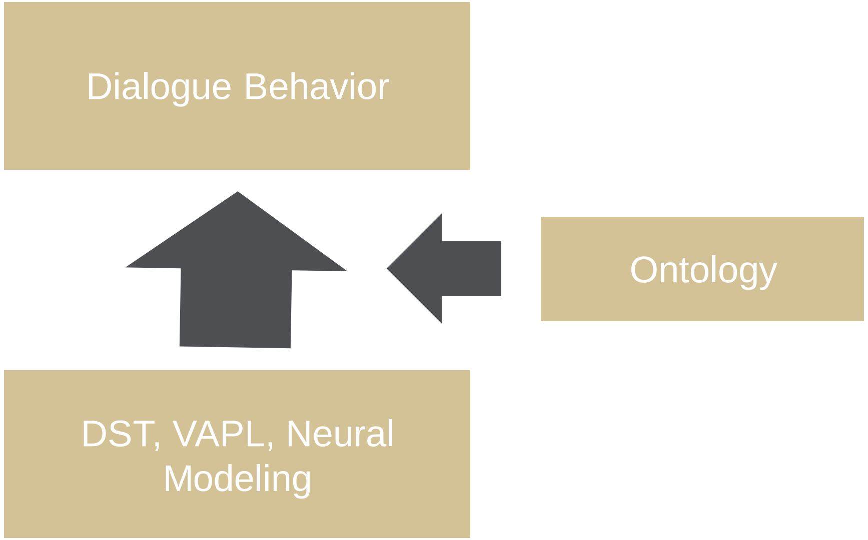
1. Introduction: Why Datasets?
2. **MultiWOZ in the Almond/ThingTalk/Genie Context**
3. What's In a Dataset
   - a. Dialogue Generation
   - b. Annotation Generation
   - c. Annotation Styles
4. MultiWOZ Revisited

2. MultiWOZ in the Almond/ThingTalk/Genie Context

Figure from Kumar et al. 2020

2. MultiWOZ in the Almond/ThingTalk/Genie Context

- MultiWOZ (and most datasets) has a corpus and annotations.
- We personally only use the former. We don't train on MultiWOZ.



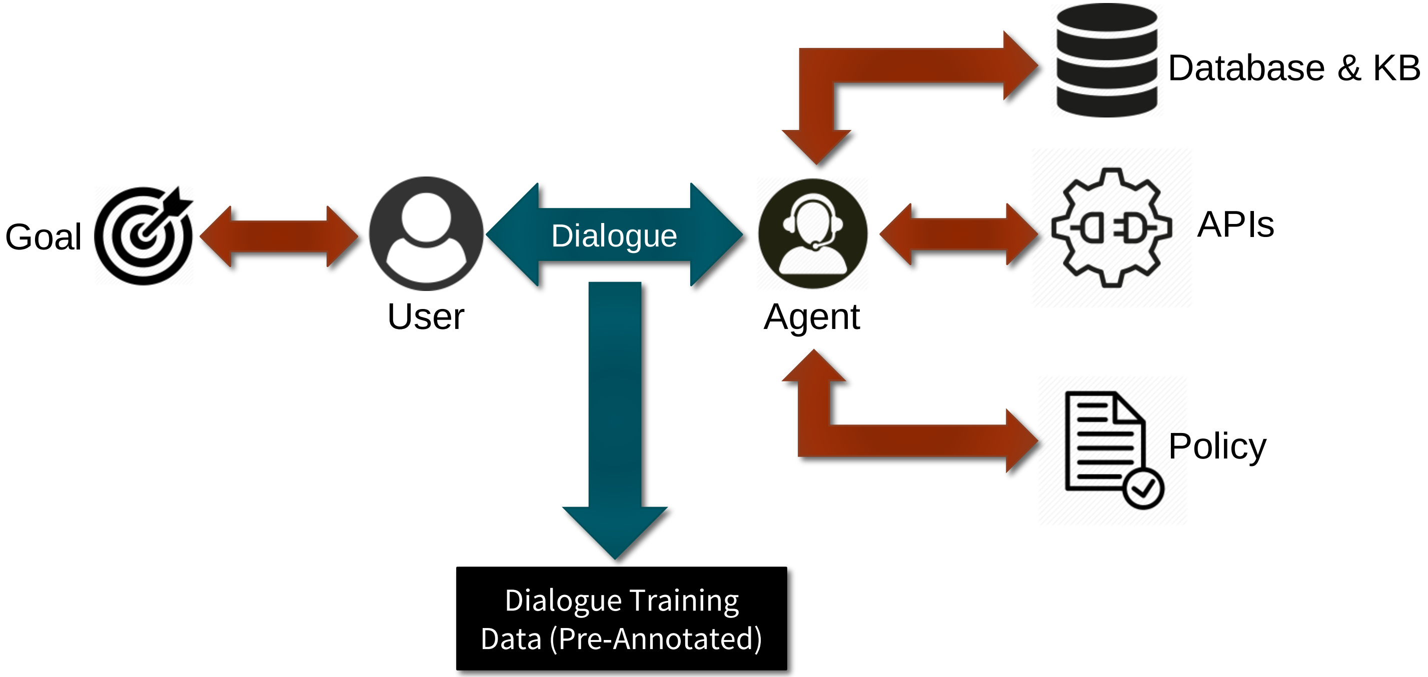
Outline

1. Introduction: Why Datasets?
2. MultiWOZ in the Almond/ThingTalk/Genie Context
3. What's In a Dataset
- **a. Dialogue Generation**
- b. Annotation Generation
- c. Annotation Styles

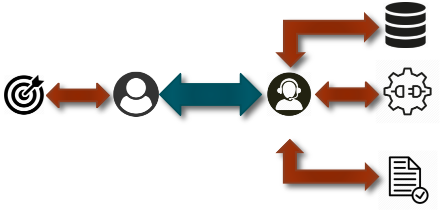
4. MultiWOZ Revisited

3a. Dialogue Generation

Our General Paradigm:



Human-to-Machine



Bootstrap from an existing dialogue system to build a new task-oriented dialogue corpora.

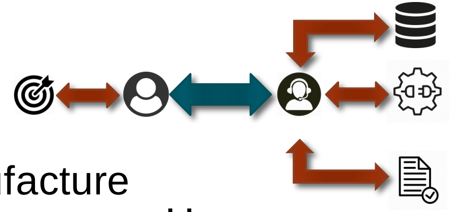
Example: Let’s Go Bus Information System, used for the first Dialogue State Tracking Challenge (DSTC)

User: real humans interacting with the dialogue system
Agent: existing dialogue system, likely following rigid rule-based dialogue policy

Goal: derived from existing dialogue system
Database / KB: derived from existing dialogue system
APIs: derived from existing dialogue system
Policy: derived from existing dialogue system

Great for expanding the capabilities of an existing domain, but can we generalize beyond this domain?

Machine-to-Machine



Engineer a simulated user plus a transaction environment to manufacture dialogue templates en masse, then map those dialogue templates to natural language.

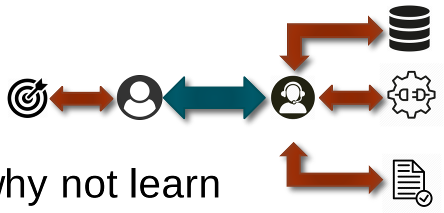
Example: Shah et al., 2018, “a framework combining automation and crowdsourcing to rapidly bootstrap end-to-end dialogue agents for goal-oriented dialogues”

User: engineered, agenda-based simulator
Agent: engineered, likely from a finite-state machine

Goal: derived from scenarios produced by Intent+Slots task schema
Database / KB: domain-specific, wrapped into API client
APIs: provided by developer
Policy: engineered specifically for agent

Great for exhaustively exploring the space of possible dialogues, but will the training data actually match real-world scenarios?

Human-to-Human



If we really want our agents mimicking human dialogue behavior, why not learn from real human conversations?

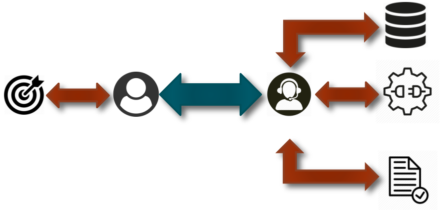
Example: Twitter dataset *(Ritter et al., 2010)*, Reddit conversations *(Schrading et al., 2015)*, Ubuntu technical support corpus *(Lowe et al., 2015)*

User: real humans on the Internet
Agent: real humans on the Internet

Goal: ???
Database / KB: ???
APIs: ???
Policy: real human dialogue policies!

Great for teaching a system real human dialogue patterns, but how will we ground dialogues to the KB + API required by our dialogue agent?

Human-to-Human (WOZ)



Humans produce the best dialogue behavior. Let's use humans to *simulate* a machine dialogue agent, grounding the dialogue in our KB+APIs.

Example: WOZ2.0 *(Wen et al., 2017)*, FRAMES *(El Asri et al., 2017)*, MultiWOZ{1.0, 2.0, 2.1} *(Budzianowski et al., 2018)*

User: crowdworker
Agent: crowdworker, simulating a human-quality dialogue system

Goal: provided by the task description
Database / KB: domain-specific, provided to the agent by experimenters
APIs: domain-specific, provided to the agent by experimenters
Policy: up to the crowdworker – nuanced, but maybe idiosyncratic

Great for combining human dialogue policies with grounding in the specific transaction domain, but annotations will be nontrivial – how do we ensure their correctness?

Dialogue Generation – Summary

Human-to-Machine
Bootstrap from an existing dialogue system to build a new task-oriented dialogue corpora.

Machine-to-Machine
Engineer a simulated user plus a transaction environment to manufacture dialogue templates en masse, then map those dialogue templates to natural language.

Human-to-Human
If we really want our agents mimicking human dialogue behavior, why not learn from real human conversations?

Human-to-Human (WOZ)
Humans produce the best dialogue behavior. Let’s use humans to *simulate* a machine dialogue agent, grounding the dialogue in our KB+APIs.

Dialogue Generation – Pros & Cons

Human-to-Machine

+ Intuitive to use existing dialogue data for dialogue system development
- Only possible to improve existing, working systems. No generalizations to new domains
- Initial system's capacities & biases may encourage behaviors that perform in testing but don't generalize

Machine-to-Machine

+ Full coverage of all dialogue outcomes in domain
- Naturalness of the dialogue mismatches with real interactions
- Hard to simulate noisy conditions typical of real interactions

Human-to-Human

+ Training data will map directly onto real-world interactions
- No grounding in any existing knowledge base or API limits usability

Human-to-Human (WOZ)

+ Ground realistic human dialogue within the capacities of the dialogue system
- High prevalence of misannotation errors

Question

WHICH DIALOGUE GENERATION TECHNIQUE SEEMS MOST SUITED FOR YOUR OWN PROJECT'S DOMAIN?

Outline

1. Introduction: Why Datasets?
2. MultiWOZ in the Almond/ThingTalk/Genie Context
3. What's In a Dataset
- a. Dialogue Generation
- **b. Annotation Generation**
- c. Annotation Styles

4. MultiWOZ Revisited

3b. Annotation generation

"Built-in" annotations (Machine-generated utterances)

- If the utterance is machine-generated, that it probably already has a formal language annotation
- Annotation is not really separate from the dialogue generation
- WikiSQL [Zhong et al. 2017]

\+ Only skill needed is paraphrasing

\- Still less natural and diverse

\- Requires good utterance synthesis

3b. Annotation generation

Manual annotations (Human-generated utterances)

- Annotation as an explicit step in the process
- Usually done on top of provided data, possibly as a separate process
- Spider [Yu et al. 2019]

\+ The dataset and the annotations are probably pretty good

\- Potentially very expensive (experts often required)

\- Sometimes not actually very good

3b. Annotation generation

Machine-assisted annotations (Human-generated utterances)

- Technology used in making the annotation process seamless or easier for humans
- Not necessarily a separate step in the process
- QA-SRL [He et al. 2015]

\+ The dataset and the annotations are probably pretty good

\- Some upfront cost of developing a good system

\- Not always possible

(Implicit) Template → Natural Utterances → Formal Language

Question

HOW DO YOU THINK MACHINE-ASSISTED ANNOTATION COULD WORK IN YOUR PARTICULAR PROJECT?

Outline

1. Introduction: Why Datasets?
2. MultiWOZ in the Almond/ThingTalk/Genie Context
3. **What's In a Dataset**
   - a. Dialogue Generation
   - b. Annotation Generation
   - **c. Annotation Styles**
4. MultiWOZ Revisited

A Fundamental Tradeoff

Expressiveness of your representation vs. **Ease of parsing, annotation, and execution**

3c. Annotation styles

Key Tradeoff: **expressiveness of the representation** vs. **ease of annotation/parsing/execution**

- Logical forms [Zettlemoyer & Collins, 2012; Wang et al. 2015]
- Intent and slot tagging [Goyal et al., 2017; Rastogi et al., 2020; many others…]
- Heirarchical representations [Gupta et al., 2018]
- Executable representations
  - SQL [Zhong et al., 2017; Yu et al., 2019]
  - ThingTalk [Campagna et al., 2019]

Logical forms

Zettlemoyer & Collins, 2012; Wang et al. 2015

Rigid logical formalisms for queries results in a precise, machine-learnable, and brittle representation.

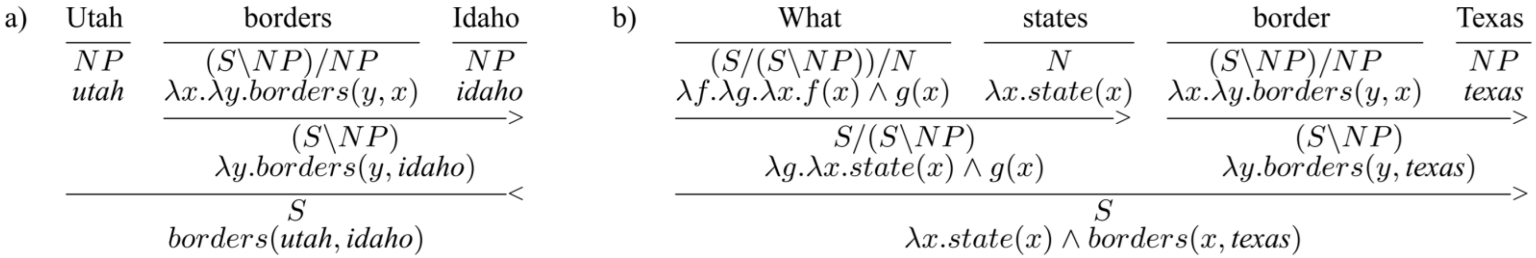


Figure 2: Two examples of CCG parses.

Intent and slot tagging

Goyal et al., 2017; Rastogi et al., 2020; many others…

More ubiquitous, less expert-reliant representation allows coverage of more possible dialogue states.

Table 2: Full ontology for all domains in our data-set. The upper script indicates which domains it belongs to. *: universal, 1: restaurant, 2: hotel, 3: attraction, 4: taxi, 5: train, 6: hospital, 7: police.

act type	inform[*] / request[*] / select[123] / recommend/[123] / not found[123] request booking info[123] / offer booking[1235] / inform booked[1235] / decline booking[1235] welcome[*] /greet[*] / bye[*] / reqmore[*]
slots	address[*] / postcode[*] / phone[*] / name[1234] / no of choices[1235] / area[123] / pricerange[123] / type[123] / internet[2] / parking[2] / stars[2] / open hours[3] / departure[45] destination[45] / leave after[45] / arrive by[45] / no of people[1235] / reference no.[1235] / trainID[5] / ticket price[5] / travel time[5] / department[7] / day[1235] / no of days[123]

Figure from MultiWOZ (Budzianowski et al., 2018)

Hierarchical Annotations

Gupta et al., 2018

Nesting additional intents within slots allows for function composition & nested API calls.

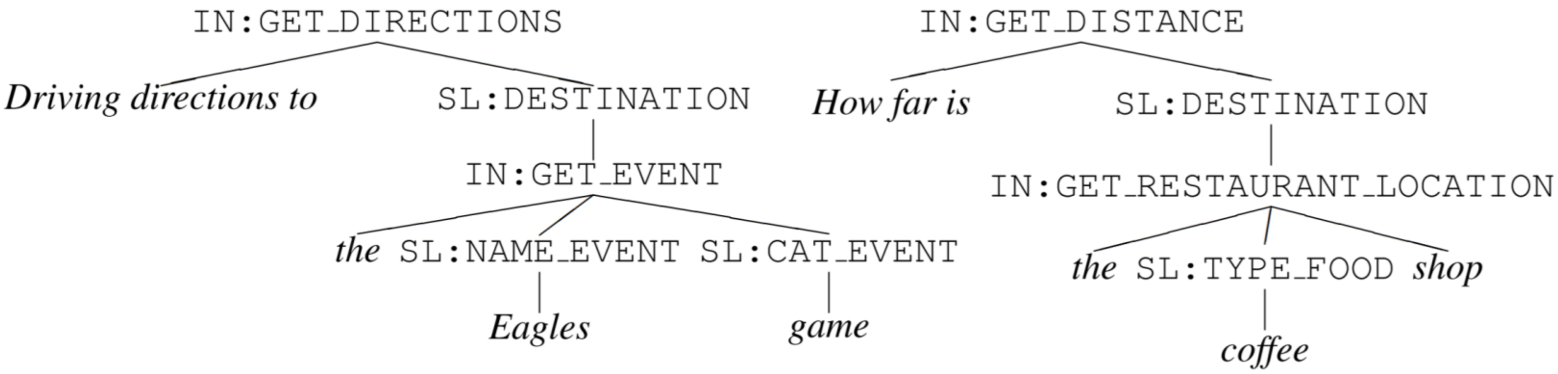


Figure 1: Example TOP annotations of utterances. Intents are prefixed with IN: and slots with SL:. In a traditional intent-slot system, the SL:DESTINATION could not have an intent nested inside it.

Executable Representations: SQL

Zhong et al., 2017; Yu et al., 2019

Structured nature of the SQL representation helps prune the space of possibly generated queries, simplifying the generation problem.

Table: CFLDraft

Pick #	CFL Team	Player	Position	College
27	Hamilton Tiger-Cats	Connor Healy	DB	Wilfrid Laurier
28	Calgary Stampeders	Anthony Forgone	OL	York
29	Ottawa Renegades	L.P. Ladouceur	DT	California
30	Toronto Argonauts	Frank Hoffman	DL	York
...	...	...	...	...

Question:

How many CFL teams are from York College?

SQL:

```
SELECT COUNT CFL Team FROM
CFLDraft WHERE College = "York"
```

Result:

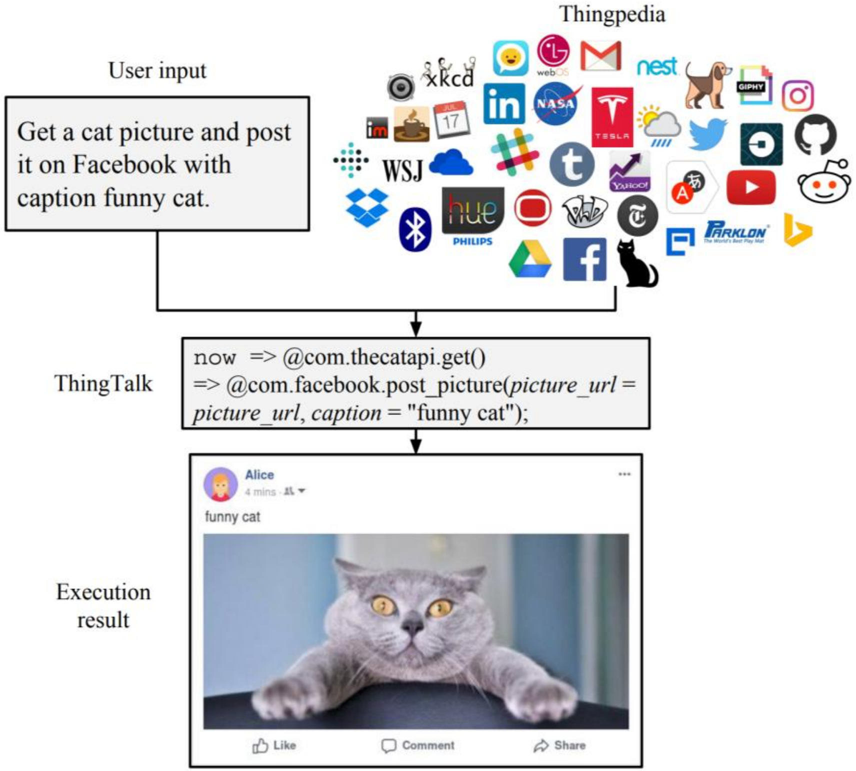
2

Figure 2: An example in WikiSQL. The inputs consist of a table and a question. The outputs consist of a ground truth SQL query and the corresponding result from execution.

Executable Representations: ThingTalk

Campagna et al., 2019

Semantic-preserving transformation rules mean **canonical** examples for training the neural semantic parser.



Outline

1. Introduction: Why Datasets?
2. MultiWOZ in the Almond/ThingTalk/Genie Context
3. What's In a Dataset
   - a. Dialogue Generation
   - b. Annotation Generation
   - c. Annotation Styles
4. **MultiWOZ Revisited**

4. MultiWOZ Revisited

- MultiWOZ is a human-human dataset, mostly annotated, with intent and slot tagging.
    - But we don't use it fully, so that ends up being less important.

- MultiWOZ proposes itself as a benchmark dataset for:
    - Dialogue State Tracking
    - Dialogue Context-to-Text Generation
    - Dialogue Act-to-Text Generation

Question

ARE THERE "BENCHMARKING BLIND SPOTS" OR BIASES THAT YOUR PROJECT MIGHT SUFFER BECAUSE OF THE DATASET CHOICE?

Thank you!