Offline RL for Task-oriented Dialogue Agents

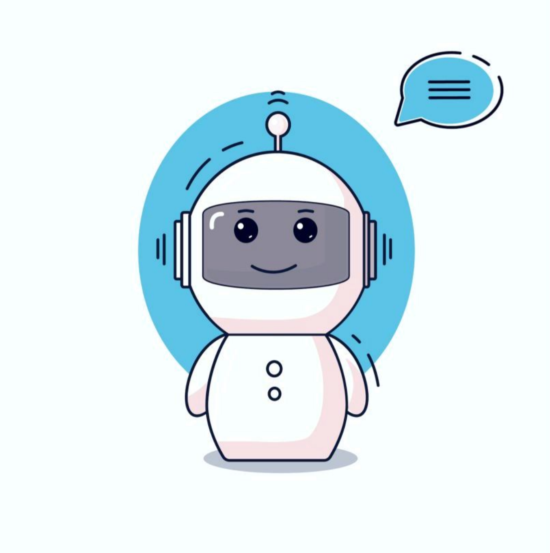
Rashmeet Kaur Nayyar

Mentor: Suchismit Mahapatra

Outline

- Overview of Offline Reinforcement Learning (RL)
- Overview of Task-oriented dialogue agents
- Challenges
- Related work
- Investigation: Approach, Dataset, Experiments
- Insights gained
- WIP: Proposed Problem Formulation and Approach
- Future Applications
- Conclusion

Reinforcement Learning (RL)

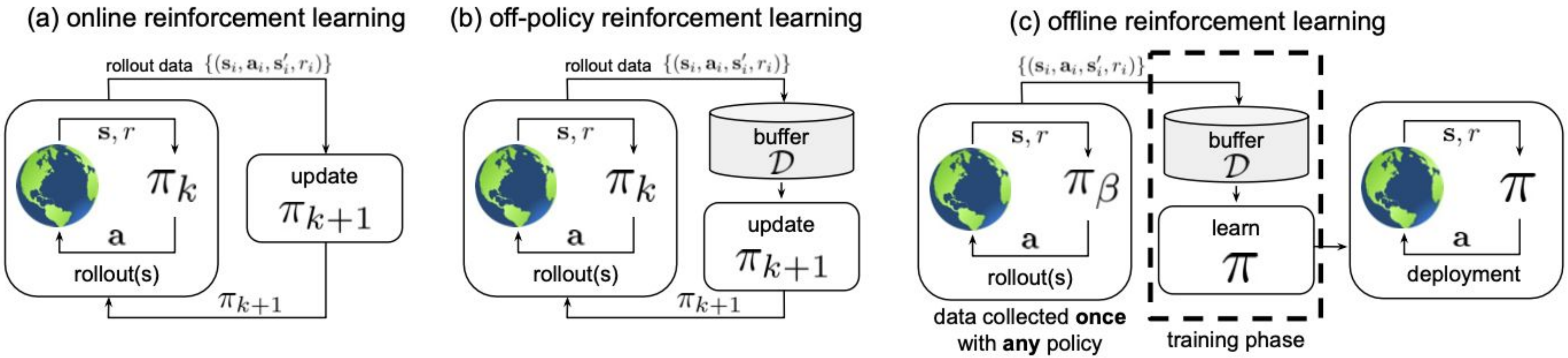


State $s \in S$
Action $a \in A$ $\longrightarrow$

$\longleftarrow$ Next state $s' \in S$
Reward r

Policy π maps S -> A

Online vs Offline RL

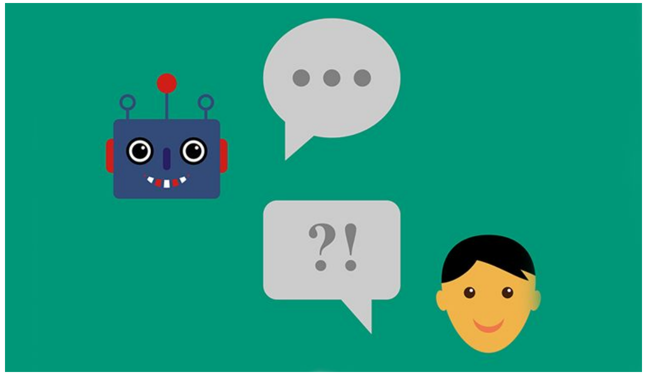


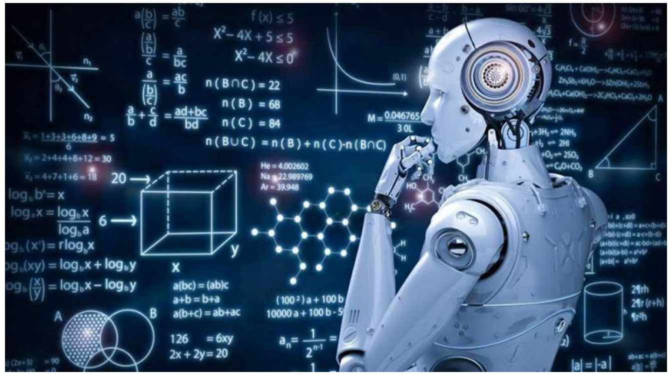
Fully off-policy!

Levine, Sergey, Aviral Kumar, George Tucker, and Justin Fu. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems." *arXiv preprint arXiv:2005.01643* (2020).

Why use Offline/batch RL?

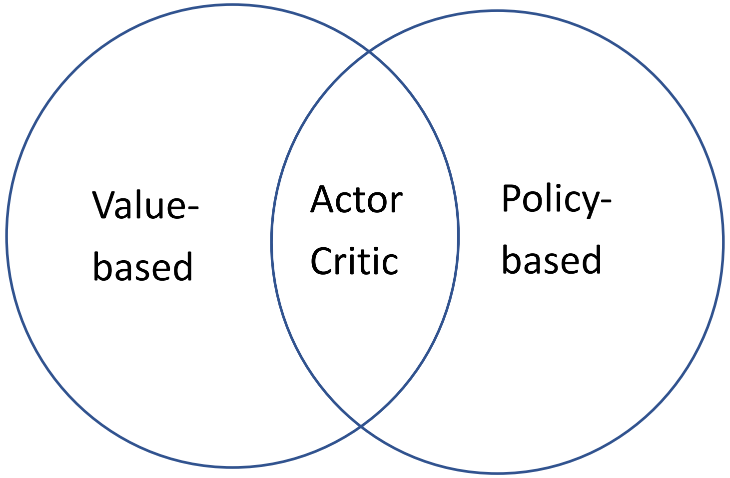
- Relying only on **real-time interaction with environment risky and expensive.**
- Removes the need for generating and training on simulators.
- **Large datasets available** in wide-range of domains.
- Given recent **success in data-driven learning methods**, extraction of near-optimal policies from available data seems promising.





Traditional RL techniques

1. **Approximate Dynamic Programming** (value-based)
   - Compute policy based on learned value function e.g., Q-learning
2. **Policy Gradient** (policy-based)
   - Learn policy directly e.g., Reinforce
3. **Actor-critic** (value and policy-based)
   - Learns both value functions and a policy
4. **Model-based RL**
   - Exploit estimates of dynamics



In principle, any off-policy RL algorithm from each category *could* be used as an offline RL algorithm !

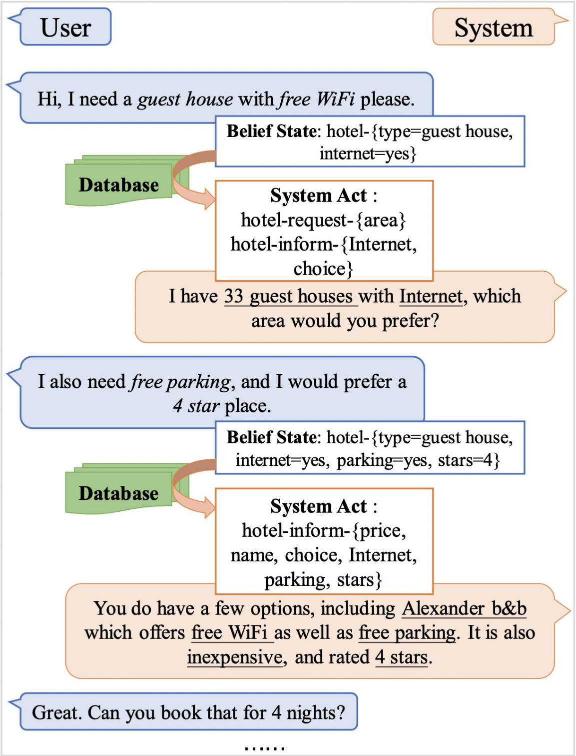
Challenges of offline RL techniques

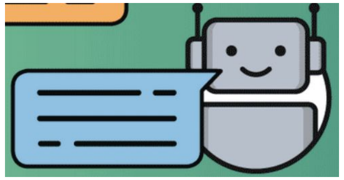
- **No exploration** to discover high-reward regions if not in dataset
- Requires **counterfactual inference** (learn a policy that is better than the dataset policy)
- **Overestimation of values** due to **out-of-distribution actions i.e., distribution shift** due to differences between learned and behavior policies.

Task-oriented vs Open-Domain Dialogue Agents

- **Open-Domain**: Open-ended conversations in fluent human-like natural language

- **Task oriented**: Accomplish a goal described by a user in fluent human-like natural language





Problem Statement

Gain insights to use offline RL to learn dialogue agents that:

- produce human-like language, and
- achieve user goals (are task-oriented).

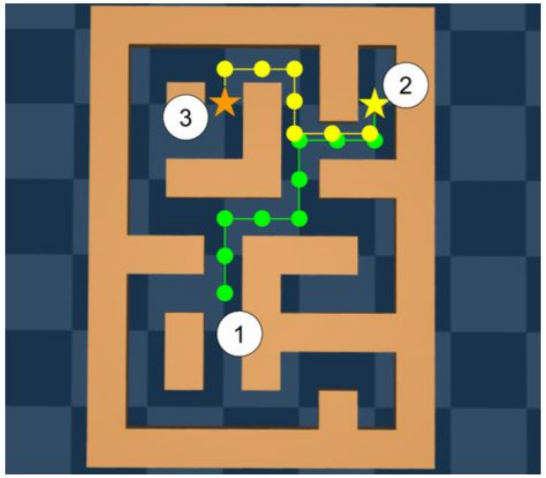
Challenges for task-oriented dialogue agents

Non-trivial to learn effectively from entire offline data due to:

- **Small annotated, sub-optimal task-specific dialogue datasets**.
- **Huge** action spaces.
- **Sparse** feasible actions.
- **Response divergence** from human intelligible language.

Why use offline RL for dialogue agents?

- Lends naturally to a data-driven goal-directed sequential decision-making formulation to **optimize for the task**.
- Allows to learn **a policy better than the best behavior policy in the dataset** (by utilizing inherent compositional structure in temporal process).
- **Large dialogue datasets** & language models readily available to exploit.



General example of exploiting compositional structure in trajectories

Related work in task-oriented dialogue agents

1. Most TOD systems use framework:
   - Natural Language Understanding (NLU) - understand user i.e., track belief-state
   - Dialogue Management (DM) - decide action
   - Natural Language generation (NLU) - generate response

2. SimpleTOD (Hosseini-Asl et. al. 2020)
   - Unified belief, action, and response generation in an end-to-end setting.
   - Limitations: Trained on dialogue turn level i.e., assumes dialogue turns are independent within a session.

3. UBAR (Yang et. al. 2021)
   - Fully end-to-end system trained on dialogue session level.

Related work in task-oriented dialogue agents

4. GPT-Critic (Jang et. al. 2022)

Builds on UBAR, performs iterative on-policy evaluation and improvement via dataset revision.

- Trains a critic network through on-policy evaluation on dataset,

$$\arg\min_{\phi} \underbrace{\mathbb{E}_{(h_t, a_t, r_t, h_{t+1}, a_{t+1}) \sim \mathcal{D}}}_{\text{dataset}} \left[\left(r_t + \gamma \underbrace{Q_{\bar{\phi}}(h_{t+1}, a_{t+1})}_{\text{target network}} - \underbrace{Q_{\phi}(h_t, a_t)}_{\text{critic network}} \right)^2 \right]$$

- Generates response candidates using GPT-2, selects responses using learned critic and generates revised dataset,
- Updates policy using revised dataset.

Related work in task-oriented dialogue agents

5. CHAI (Verma et. al. 2022)

- Steers GPT-2 towards producing task-aware dialogues using critic.
  - Trains critic through off-policy evaluation w.r.t. target policy, generates response candidates using GPT-2, selects responses using it.
  - Limitations: Domain-specific formulation, no partial observability.

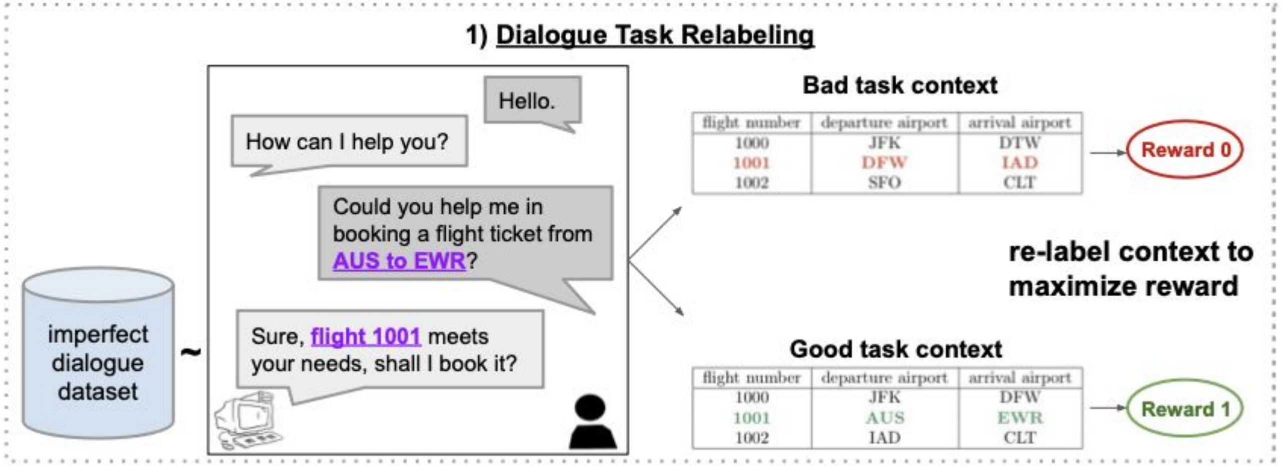
6. CALM (Snell et. al. 2022)

- Directly fine-tunes GPT-2 in a task-aware manner.
  - Reasons about the goal within the language model.
  - Limitations: focuses only on structured databases, more susceptible to internal language model biases.

Investigating CALM

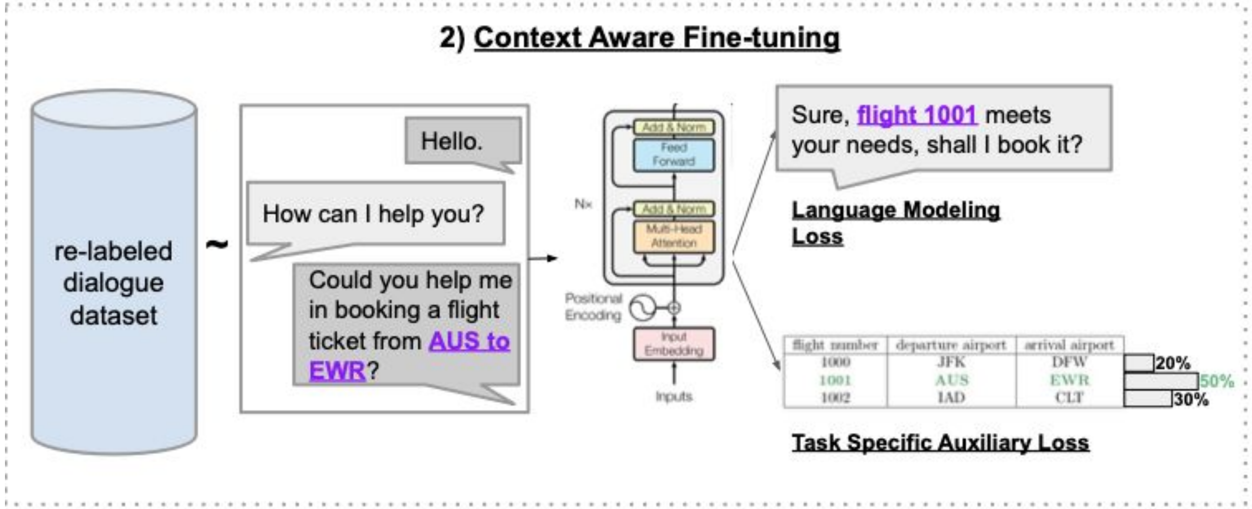
- Conditional imitation strategy + task relabeling (task-aware fine-tuning)

- End to end system – both decision-making and language generation!



Investigating CALM

- Language models – both a dynamics model and a policy! Thus model-free as well as model-based algorithms can be used!



Context-Aware Fine-tuning

- Language modeling objective:

$$\mathcal{L}_{CTX}(\theta) = \max_{\theta} \mathop{\mathbf{E}}_{(\tau, c_o) \sim \mathcal{D}^{\text{off}}} \sum_{t=1}^{T} \Big(\log \pi_\theta(a_t | \tau_{<t}, c_o) + \log \mathcal{T}_\theta(\tau_{<t+1} | \tau_{<t}, a_t, c_o) \Big),$$

Goal: Model Policy and Dynamics
τ = Dialogue trajectory
T = number of trajectories

- Auxiliary objective :

$$\mathcal{C}(\phi) = \max_{\phi} \mathop{\mathbf{E}}_{(c_h, c_o, \tau, \alpha_h) \sim \mathcal{D}^{\text{off}}} \log P_\phi(\alpha_h | \tau, c_o).$$

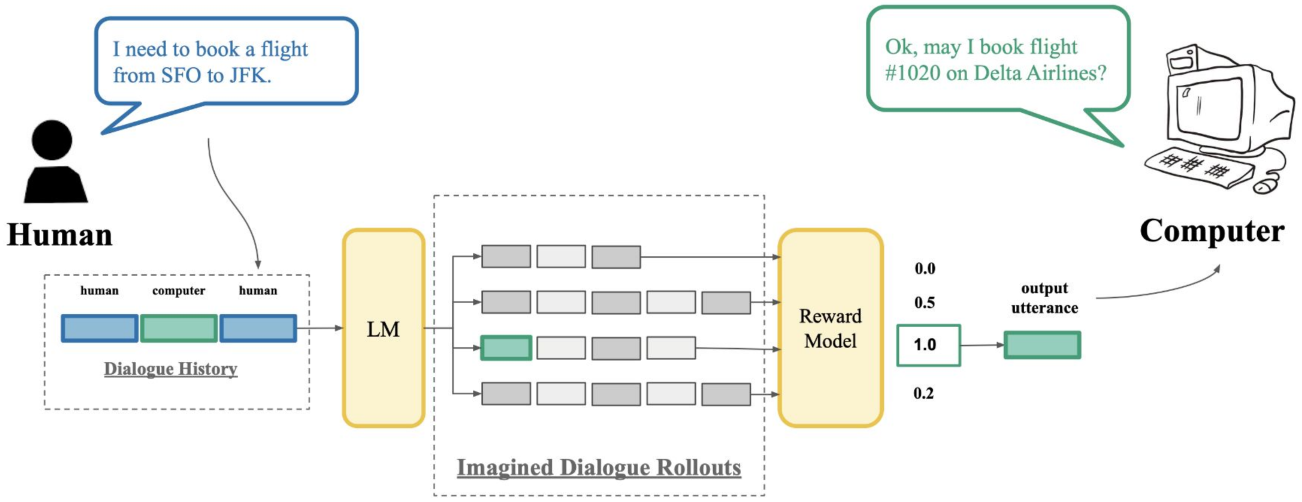
Goal: Given a trajectory and context, predict action to improve context-conditioning
α_h = high-level action that determines success of task
(e.g., flight number)

- Final combined utility function:

$$\max_{\theta, \phi} \mathcal{L}_{CTX}(\theta) + \beta * \mathcal{C}(\phi)$$

Model-based planning

- Utilize dynamics, perform rollouts, rank using estimated reward function, and select a response for the agent.



Dataset and Metrics

- **Dataset** - Air Dialogue (Air reservations)
  - Agent helps user book, cancel, and change flights
- **Metrics**:
  - Success rate – measures how often the system accomplishes the task.
  - BLEU – measures fluency

Experiments

Self-play evaluation:

Reward accuracy: 0.75

Status accuracy: 0.83

Flight accuracy: 0.763

Name accuracy: 0.993

Success rate: 0.84 (baseline 0.38)

Language quality:

BLEU score: 32.86

(baseline 35.75)

Example dialogue generated:

```
==============================
starting convsersation:
conversation data: {'conditioned_reward': 1.0}
==============================
Customer: Hello.
Agent: Hello.
Customer: I am Edward Thompson.
Agent: How can I help you for today?
Customer: Can you help me in booking a ticket from CLT to BOS?
Agent: Yes, I can help you. Could you tell me your dates of travelling?
Customer: My planning dates are from 03/11 to 03/13.
Agent: Give me a moment.
Customer: Sure.
Agent: I have found a flight ticket for you with a price of 100 and a connection of 1. Can I proceed?
Customer: That's great, please proceed.
Agent: Ok, your ticket is booked.
Customer: Thank you for your help.
Agent: Thank you for reaching our agency.
Submit: book , Edward Thompson , 1020
==============================
conversation reward: {'reward': 0.0, 'name': 1.0, 'status': 1.0, 'flight': 0.0}
expected action: {'status': 'book', 'name': 'Edward Thompson', 'flight': [1013]}
==============================
```

Insights for proposed approach

- Use value-based RL instead so that the systems generate better responses for unseen tasks by learning to recombine suboptimal responses [Levine et. al. 2020]. Human evaluation is important to evaluate this!

- It could lead to overestimation issues due to distribution shift. Use conservative Q-learning [Kumar et. al. 2022] to overcome it by learning lower-bounded Q-function (penalize OOD actions).

Importance of “truthfulness”

```
==============================
starting convsersation:
conversation data: {'conditioned_reward': 1.0}
==============================
Customer: Hi. I am Laura Wright.
Agent: Hello, how may I help you?
Customer: Help me in booking a flight from DCA to BOS?
Agent: Sure, kindly mention your travel dates.
Customer: My journey dates are Oct 02 to 04.
Agent: May I know your price limit?
Customer: I have mentioned above.
Agent: Is there any class preference for you?
Customer: No.
Agent: Do you need halt in between your journey?
Customer: I need a direct flight which departs in morning time.
Agent: Ok, please wait for a moment.
Customer: Sure.
Agent: Thanks for waiting, we found a preparation, flight number: 1028 , airline:UA, price:100. Shall I proceed to booking with a direct flight?
Customer: You can reserve it.
Agent: Your flight ticket has been reserved in economy class and your departure time is morning at 7:00AM.
Customer: Thank you.
Agent: Thank you for choosing us.
Submit: book , Laura Wright , 1028
==============================
conversation reward: {'reward': 0.0, 'name': 1.0, 'status': 0.0, 'flight': 0.0}
expected action: {'status': 'no_flight', 'name': 'Laura Wright', 'flight': []}
==============================
```

Insights for proposed approach

- Use structured datasets to verify correctness of dialogues and update reward specifications to make the systems more "truthful". This might help reduce the bias of the language model.

- Inform rate i.e., a measure of how often the system responses are correct would be much more helpful for investigation of "truthfulness" of systems.

Proposed framework (WIP):

1. Fine-tune GPT-2 on task-specific dataset.
2. Perform model-based rollouts to generate candidate dialogues from (or a proposal distribution based on) fine-tuned GPT-2.
3. Train a critic on task-specific offline dialogue dataset (focused on task accomplishment).
4. Rank generated response candidates using the learned critic and select one.

(continued..)

Proposed framework (WIP):

5. Learn a template generator with variables and a module that generates SQL queries from templates. Query from structured task-specific dataset and update the values of the variables to generate prompts (focused on truthfulness).

Example:

User utterance: Could you book a flight for me to the capital of Australia?

Candidate dialogue generated: I found flight flight_0 to the capital of Australia which is Canberra.

Template: I found flight [var] to the capital of Australia which is [var].

SQL queries: SELECT capital FROM table WHERE country = ‘Australia’ (Returns: ‘Canberra’)

SELECT flight FROM table WHERE destination = ‘Canberra’ (Returns: flight_1234)

Prompt: I found flight flight_1234 to the capital of Australia which is Canberra.

Possible Applications at LinkedIn

Building Offline RL Dialogue agents to communicate with users and help them:

- **Search chatbot for recruiters**
- **Search chatbot for users to connect to peers** that they share same professional goals with
- **Search chatbot for users to narrow down interesting opportunities**
- **Search chatbot that advises users for career development**
- **Customer service chatbot on company pages**

Conclusion

Investigated offline RL to build dialogue agents that are more general for wide applicability and that can be fine-tuned for specific problems at LinkedIn in the future.

- Reviewed and critiqued recent relevant literature in depth [Jang et. al. 2022, Verma et. al. 2022, Snell et. al. 2022].
- Reproduced experiments for CALM on Kubernetes using HDFS and evaluated language quality and task accomplishment.
- Proposed formulation and framework (WIP) through insights gained.