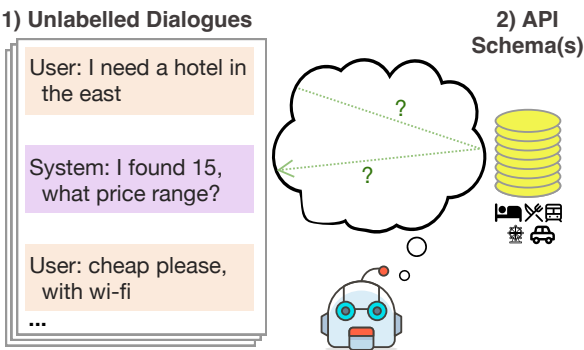


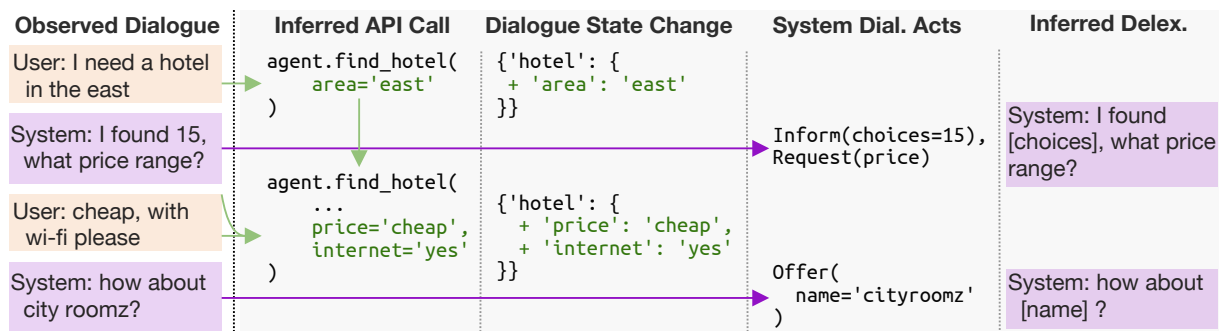
{bking2, jmflanig}@ucsc.edu

Abstract



to complete tasks on behalf of users, have been a longstanding challenge within conversational AI. (DST) (Hu et al., 2022; King and Flanigan, 2023;

icy skeleton' ([Zhang et al., 2023](#)).



1977) (§4.5).

call. Fig. 5a in App. A gives an example of our

Direct DST Prompt

```
response = agent.handle_turn(  
    belief_state=BeliefState(attraction=dict(  
        name='byard art')),  
    last_system_utterance="byard art is at 344 oxford " + \  
        "street, anything else?",  
    user_utterance="Yes, I need a taxi to king station",  
    user_intent=[agent.book_taxi(destination='king station')])
```

Noisy Channel DST Prompt

```
response = agent.handle_turn(  
    belief_state=BeliefState(attraction=dict(  
        name='byard art')),  
    last_system_utterance="byard art is at 344 oxford " + \  
        "street, anything else?",  
    user_intent=[agent.book_taxi(destination='king station')]),  
    user_utterance="Yes, I need a taxi to king station",
```

$Pr(x|y, z, c)$, following [Min et al. \(2022\)](#).²

²https://arxiv.org/pdf/2205.08477v1.pdf

DST For the DST sub-task, we again use both ‘direct’ and ‘channel’ (prompt, completion) pairs.

$$P(f_{\text{prompt}}(H_t))$$

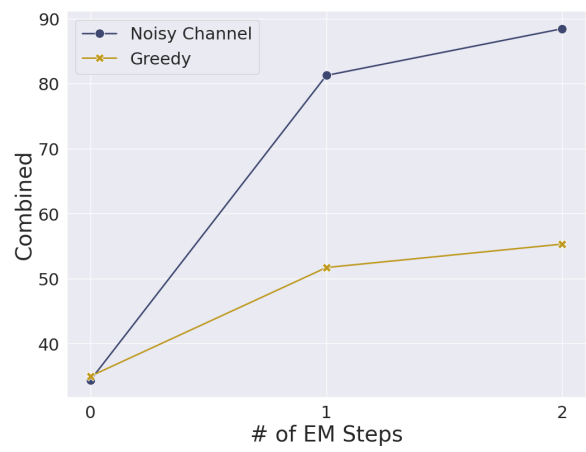
user utterances (r_{t-1}, u_t) and our policy’s act pre-

$$P(f_{\text{prompt}}(r_{t-1}, u_t, A_t))$$

PPTOD (Su et al., 2022)	✓	✓	✓	82.6	72.2	18.2	95.6
DiactTOD (Wu et al., 2023)	✓	✓	✓	89.5	84.2	17.5	104.4
Our (supervised)	✓	✓	✓	67.9	61.7	14.6	79.4
SGP-TOD-GPT3.5 (Zhang et al., 2023)	✓	Few (‡)	✗	82.0	72.5	9.22	86.5
LLaMa [†]	✓	✗	✗	-	4	1.61	-
GPT 3.5 Turbo [†]	✓	✗	✗	44.8	31.2	3.3	41.3
GPT 3.5 Turbo (– gold delex.)	✓	✗	✗	40.7	26.7	3.7	37.4
Ours (StarCoder 15B - no EM)	✓	✗	✗	50.0	19.6	3.2	38
Ours (StarCoder 3B - w/ EM)	✓	✗	✓	78.1	68.3	13.6	86.8

Table 1: Unsupervised end-to-end results in MultiWOZ 2.2. (†) indicates models from Hudeček and Dusek (2023).

IC-DST (StarCoder 15B)	15.66
RefPyDST (StarCoder 15B)	13.88
GPT 3.5 Turbo (Hudeček and Dusek, 2023)	13.05
Ours (StarCoder 15B → 3B)	39.70



⁶<https://huggingface.co/datasets/bigcode/starcoderdata>

Ours (zero-shot)	49.0	15.0	3.0	35.0
Ours (Full EM)	80.5	69.0	13.7	88.5

quires annotation expertise. [Zhang et al. \(2023\)](#)

context examples.⁹

multi-domain setting.

lems using LLMs. [Hu et al. \(2022\)](#) and [\(King](#)

dialogue state is a latent variable. [Liu et al. \(2021a\)](#)

ting (Jin et al., 2018; Liu et al., 2023). However,

References

TODS: Large Language Models for End-to-End Task-Oriented Dialogue Systems. ArXiv:2310.08885 [cs].

[Degeneration](#). ArXiv:1904.09751 [cs].

[Oriented Dialog Systems](#). ArXiv:2109.04314 [cs].

1403–1412. ArXiv:1808.10596 [cs].

34(05):8689–8696.

End-to-End (E2E) Dialogue Metrics We mea-

vided by [Nekvinda and Dušek \(2021\)](#).¹⁰

2020. A Probabilistic End-To-End Task-Oriented Di-

guage Processing (EMNLP), pages 9207–9219, On-

state prediction \hat{y}_t is considered correct only if all

values, such as the name of a restaurant or hotel,

0.95.¹¹

alogue, as detailed in §4.1. Our prompts use python

¹⁰https://github.com/Tomiinek/MultiWOZ_Evaluation

¹¹<https://pypi.org/project/fuzzywuzzy/>

(2019). “x=y” indicates the act can take on arbitrary key-value arguments, and “x=?” indicates the act takes on one

```

1: procedure INITIALOFFLINELABEL( $\mathcal{D}_{train}, \theta_{ret}, \theta$ )
                                     ▷ Initialize example pool

4:   for  $t = 0$  to  $\max_{d \in \mathcal{D}_{train}}$ 
                                     ▷ Loop by increasing turn index
5:     for all  $(d_{id}, u_t, r_{t-1}, r_t)$  in  $\mathcal{D}_{train}$  do
6:        $\hat{b}_{t-1} \leftarrow \mathcal{B}[d_{id}][t-1]$  or  $\emptyset$ 
7:        $\hat{b}_t \leftarrow \text{OFFLINEDST}(\mathcal{P}, \theta_{ret}, \hat{b}_{t-1}, r_{t-1}, u_t)$ 
8:        $\hat{A}_t \leftarrow \text{OFFLINEACTTAG}(\mathcal{P}, \theta_{ret}, u_t, r_t)$ 
9:        $\mathcal{P} \leftarrow \mathcal{P} \cup \{(r_{t-1}, u_t, r_t, \hat{b}_t, \hat{A}_t)\}$ 
                                     ▷ Add in-context example for future labeling
10:    end for
11:  end for
12: end procedure

13: procedure OFFLINEDST( $\mathcal{P}, \theta_{ret}, \hat{b}_{t-1}, r_{t-1}, u_t$ )
14:    $\mathcal{E}_k \leftarrow \theta_{ret}(\hat{b}_t \cdot r_{t-1} \cdot u_t, \mathcal{P})$ 
15:    $\mathcal{C} \leftarrow \Delta b_t \sim P(f_{\text{prompt}}(\mathcal{E}_k, \hat{b}_{t-1}, r_{t-1}, u_t))$ 
                                     ▷ Sample w/ ‘direct’ prompt
16:    $\Delta \hat{b}_t \leftarrow \arg\max_{\Delta b_t \in \mathcal{C}} P(u_t | f_{\text{prompt}}(\mathcal{E}_k, \hat{b}_{t-1}, r_{t-1}, \Delta b_t))$ 
                                     ▷ Re-rank w/ ‘channel’ prompt
17:   return  $\hat{b}_{t-1} + \Delta \hat{b}_t$ 

20:    $\mathcal{E}_k \leftarrow \theta_{ret}(u_t \cdot r_t, \mathcal{P})$ 
21:    $\mathcal{C} \leftarrow A_t \sim (P(f_{\text{prompt}}(\mathcal{E}_k, r_t)))$ 
                                     ▷ Sample w/ ‘direct’ prompt
22:   return  $\arg\max_{A_t \in \mathcal{C}} P(\mathcal{E}_k, A_t, r_t)$ 

```

```

class DialogueAgent:
    <one method per intent in the schema with all informable slots>
    def book_taxi(self, leave_at: str = None, destination: str = None,
                  departure: str = None, arrive_by: str = None) -> Intent:
        """
        book taxis to travel between places

        Parameters:
            leave_at: (str) leaving time of taxi
            destination: (str) destination of taxi
            departure: (str) departure location of taxi
            arrive_by: (str) arrival time of taxi
        """
        pass
    ...

if __name__ == '__main__':
    agent = DialogueAgent()

    # Provide the call matching the user's intent in this context

    <in-context exemplars from self-predictions may go here>

    response = agent.handle_turn(
        belief_state=BeliefState(attraction=dict(
            name='byard art',
            type="museum",
            area="south")),
        last_system_utterance="byard art is at 344 oxford " + \
            "street, anything else?",
        user_utterance="Yes, I need a taxi to king station",
        user_intent=[agent.book_taxi(destination='king station')])

<one Entity per service in schema, with informable + requestable slots>
class Taxi(Entity):
    """
    Parameters:
        leave_at: (str) leaving time of taxi
        destination: (str) destination of taxi
        departure: (str) departure location of taxi
        arrive_by: (str) arrival time of taxi
        type: (str) car type of the taxi
        phone: (str) phone number of the taxi
    """
    ...

< a class for each of the acts supported in our system>
class Inform(Act):
    """Provide information."""
    entity: Entity = None

class Request(Act):
    """Ask for specific information or action."""
    values: List[str] = None
    ...

if __name__ == '__main__':
    agent = DialogueAgent()

    # Provide the dialogue acts matching the observed system response

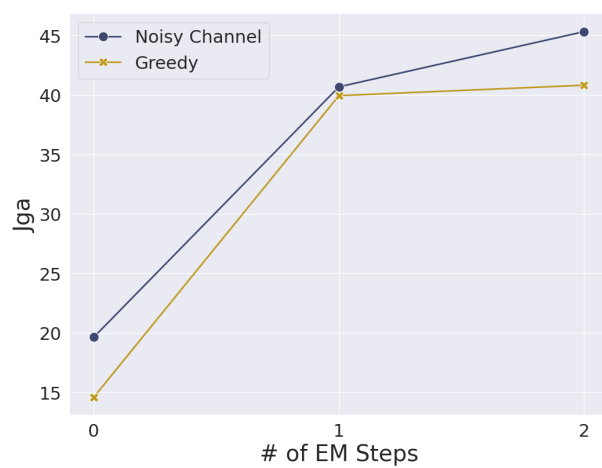
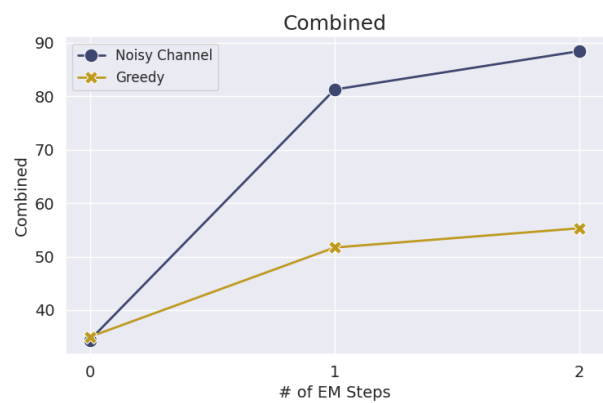
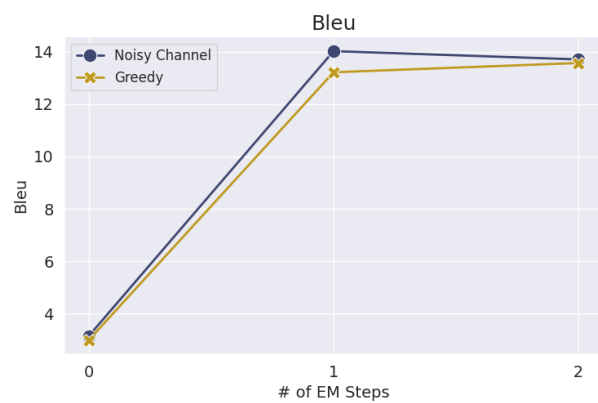
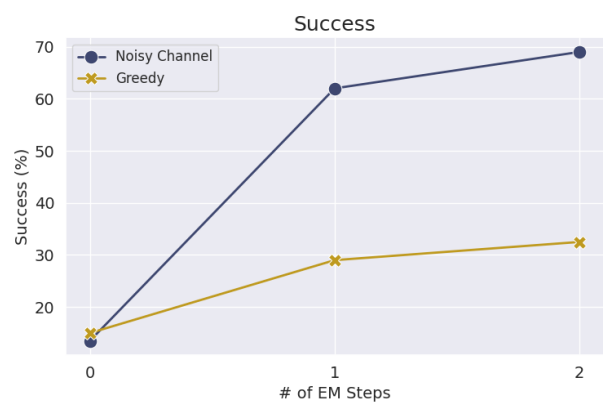
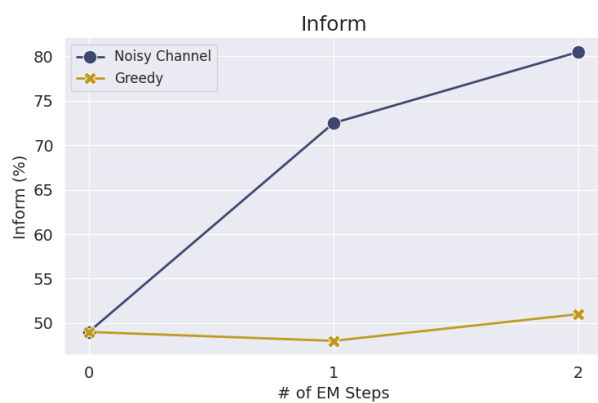
    <in-context exemplars from self-predictions may go here>

    response = agent.handle_turn(
        system_response="Ok, where will you be departing from?",
        system_acts=[Request(values=['departure'])])

```

DST and DAT (Act Tagging), best viewed in color. Key-word arguments are used to include variables from the turn

¹²<https://github.com/bigcode->



	restaurant-		
	[SYS_DA] [SYS_DA] train-inform-ticket-16.50		