Demonstration selection for in-context learning

Alexian Plancke
Stefano Galligani

December 2025

1 Introduction

Dialogue State Tracking (DST) is a core component in task-oriented dialogue systems (like chatbots or virtual assistants). Its goal is to maintain and update a representation of the conversation's current state as the dialogue progresses. The "state" typically includes domains (e.g. restaurant, taxi, hotel) slots for each domain (e.g., restaurant name, time, number of people) and their values (e.g., "Le Petit Bistro," "8 PM," "4"). These slots are chosen because they are the key points in the conversation.
The role of DST is to continuously updates these slots based on user inputs and system actions. This is used by the system to understand what information has been provided, what is missing, and what actions to take next (to pursue with our example, asking for special needs or directly making a reservation). DST is challenging in many aspects, like handling of ambiguity, user corrections, and incomplete information.

In-Context Learning (ICL) is a way to improve capability of large language models (LLMs) to perform tasks without the need of explicit fine-tuning, only by adding examples in the input. This way, the model "learns" patterns from the input context, but it doesn't update its weights. Because of this, ICL allows LLMs to adapt to new tasks quickly, such as translation, summarization, or code generation, just by enhancing the prompt. ICL may not match fine-tuned models for specialized tasks, but it reduces the need for task-specific resource draining training by leveraging the versatility of modern LLMs.

The goal of this project is to test different techniques of ICL applied to DST and to measure their effectiveness.

2 Methodology

2.1 Description of the experiments

As DST consists in predicting, at each dialogue turn, the set of domains and slot key–value pairs corresponding to the user's intent, the experiments consist in providing different kinds of examples to a model prompted to output a JSON object containing the predicted state, evaluate the accuracy of the results, and compare them to find the most efficient.
We chose, in our experiments, to include the domains in the prompt and onl focus on predicting slot keys and values.
The experimental pipeline follows these main steps:

1. Dataset preparation
   A dialogue dataset is loaded and cleaned: audio information is discarded, and only textual components (user utterances, agent responses, domains, and ground-truth slots) are retained.

2. Prompt construction
   For each test sample, a prompt is dynamically built. It contains:

   - A general instruction describing the DST task.
   - k demonstration examples (dialogues with their correct slot annotations) similar to the target, that establish a context;
   - The target dialogue turn, for which the model must predict the slots.

   Different strategies are considered for constructing the prompts:

   - Show only the user dialogue or the full conversation;
   - Change the number of demonstrations;
   - Change the embedder used to evaluate similarity between dialogue turns.

3. Model inference
   The prompt is passed to a pretrained instruction-following LLM, which generates a textual response intended to represent the slots in JSON format.

4. Post-processing
   Since LLM outputs may contain syntactic errors, a JSON repair mechanism is applied to ensure the output can be parsed and evaluated.

5. Evaluation
   The predicted dialogue states are compared against the ground truth using standard DST metrics.

2.2 Models used

For dialogue state prediction, the model LLaMA-3.2-3B-Instruct was chosen, this model provides a good balance between the resources needed to run it and its performance.
Most importantly, this model is an instruction-tuned causal language model designed to follow natural language prompts effectively. It is especially well-suited for in-context learning scenarios, where task instructions and examples are provided directly in the prompt without any parameter fine-tuning. Additionally, one of two sentence embedding models is used to retrieve relevant demonstrations, selected by computing cosine similarity between the embedding of a test dialogue and embeddings of training dialogues. These models are:

- Dialog2Flow (D2F), a dialogue-specialized model trained on task-oriented dialogue data;
- LaBSE, a general-purpose multilingual sentence embedding model.

2.3 Metrics

Selecting adequate metrics is essential to compare the various experiments over criteria that are measurable and relevant to the goal. Fortunately, the standard Dialogue State Tracking (DST) metrics using the MultiWOZ evaluation framework is a de-facto standard for a rational evaluation in this field.
The main metrics used for this project are:

- Joint Goal Accuracy (JA) is the ratio correctly predicted turns, or, in other words, the frequency of turns where each slot is correctly identified with no mistakes.
- Slot Error Rate (SER) is the proportion of slot errors (missing, incorrect, or spurious slots) divided by the total number of ground-truth slots. Lower values indicate better performance.
- Slot Precision is the Correctly predicted slots divided by the Total predicted slots. It represents the proportion correct predictions made by the model.
- Slot Recall is the Correctly predicted slots divided by the Total ground-truth slots. It represents the proportion of ground-truth slot–value pairs that are successfully predicted.
- Slot F1-score is the harmonic mean of precision and recall, providing a balanced measure of slot prediction quality.

3 Implementation details

3.1 Dataset of dialogues: SpokenWOZ-Whisper

The dialogue dataset used in this project is SpokenWOZ-Whisper[2], a processed version of the SpokenWOZ dataset. SpokenWOZ provides transcribed spoken dialogues with structured dialogue state annotations, making it suitable for DST tasks. SpokenWOZ-Whisper is a custom version with transcripts generated by Whisper, which the authors claim reduces word error rate.
The occurrences in the dataset contain a full history of the dialogue, a list of active domains a nested dictionary mapping domains to sequences of slot–value pairs. Recordings that are also present in the dataset aren't used for this project. That's the reason why they are dropped to release precious resources.

3.2 Dataset embedding

Dataset embedding consists in encoding each dialogue into a fixed-size vector representation. Therefore, it becomes possible to efficiently select training examples that are semantically close to a given test dialogue with a similarity-based retrieval of demonstration examples. Those examples can then be consumed by the in-context learning part of the pipeline.
Only the conversation between the user and the agent is used to generate the embedding, since at test time we generall don't have the dialogue state available.

3.3 Prompt creation

This module is used to transform a sample from the dataset into the structured natural language prompt that will be passed to the LLM to infer the dialogue state. This prompt includes the task instruction, some optional examples, and the target dialogue, together with its domains, for which the dialogue state must be predicted.

3.3.1 Dialogue representation strategies

The prompt can be generated in two different configurations: user-only and user–agent. The user-only configuration includes only user utterances, while the user–agent configuration includes both user and agent turns.

3.3.2 Demonstration selection strategies

Demonstrations are usually selected using cosine-similarity calculated with one of the two sentence embedders used, but they can also be selected at random, to establish a baseline against which the embedders are tested.

3.4 Token-length control

We put a hard limit on the maximum number of generated tokens to avoid a waste of resources, set to the length of the ground-truth dialogue state plus 20. This approach balances completeness of the answer with computational efficiency.

3.5 Output repairing

LLM tend to generate invalid JSON or unwanted data after the JSON. That is because LLMs are not using a formal grammar, but select the next token between the one that are statistically the most likely to be fit for this place. To fix this problem, the JSON generated is "repaired" into a valid JSON structured that can then be parsed.

3.6 Evaluation

For evaluation we use the MultiWOZ and SpokenWOZ Evaluation repository[3], which allows for evaluation of DST on SpokenWOZ with fuzzy matching. We modified the code to allow for the evaluation of turns independent from each other.

4 Experiments and results

4.1 User-only vs user and agent

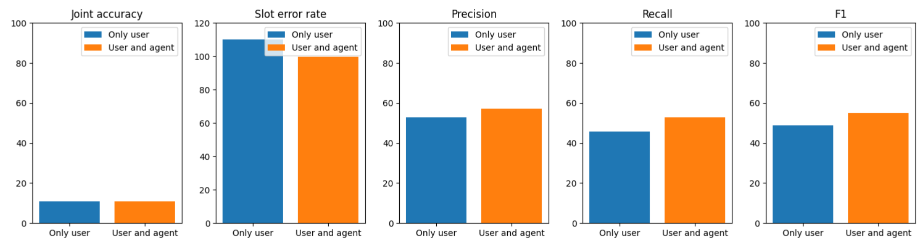

Figure 1: Metrics for dialogue turns variations

	JA	SER	Precision	Recall	F1
U	11%	110.02%	52.73%	45.72%	48.98%
UA	11%	99.82%	57.20%	52.82%	54.92%

Table 1: Metrics for dialogue turns variations

This experiment is comparing user-only utterances and full conversations. All other parameters are kept identical: 3 examples are selected using D2F for

similarity.
When including the agent's utterances in the prompt, both precision and recall are noticeably higher, and SER is lower. This is expected, as the agent's turns can include relevant information which is otherwise lost: for example, in some circumstances the agents asks the user to choose between two options, and the user simply replies that the prefer the ”first” or ”second”. Without knowing what the agent said, in a situation like this it is impossible to reconstruct the correct state.
These results confirm that agent turns provide the necessary additional context to resolve ambiguous user intents.

4.2 Demonstration selection

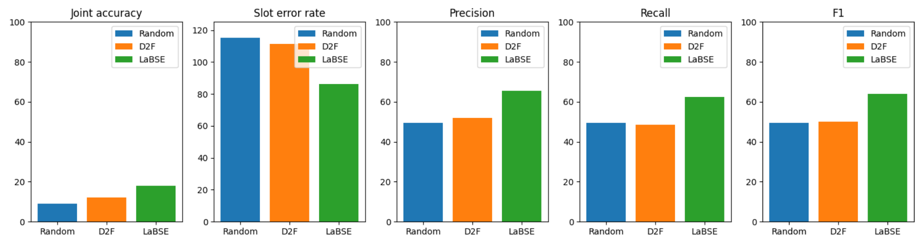

Figure 2: Metrics for demonstration selection strategies

	JA	SER	Precision	Recall	F1
Random	9%	115.12%	49.36%	49.36%	49.36%
D2F	12%	111.48%	52.05%	48.45%	50.19%
LaBSE	18%	86.34%	65.64%	62.30%	63.92%

Table 2: Metrics for demonstration selection strategies

This experiment evaluates the impact of different demonstration selection strategies.
Three modes of selection are compared:

- Randomly selected demonstrations;
- Similarity-based using D2F embeddings;
- Similarity-based using LaBSE embeddings.

The prompts are always constructed using 3 examples and only user utterances. The random selection strategy serves as a baseline to verify that using examples that are similar to the dialogue we are considering is actually useful. The results show that this is the case, with both embedders performing generally better.
Interestingly, LaBSE can outperform D2F in all metrics, although the latter is

specifically trained on dialogues. We don't know exactly why this happens, but a possibility is that D2F may be ”too good”, as many of the similar samples that it finds are from different turns of the same dialogue, and are therefore very similar to each other. LaBSE also finds some samples that come from the same dialogue, but not all of them.
This suggests that variety among the chosen examples could be as important as their similarity with the test sample.

4.3 K demonstrations

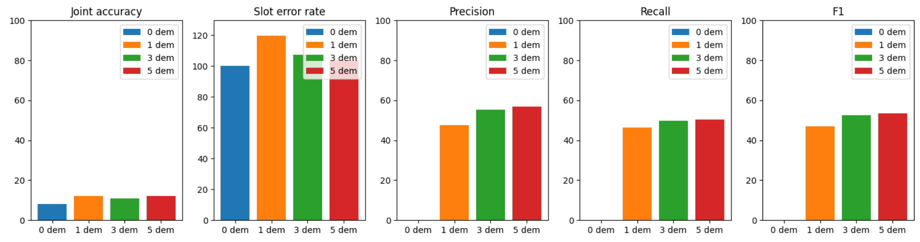

Figure 3: Metrics for varying numbers of demonstrations

	JA	SER	Precision	Recall	F1
0	8%	100%	0%	0%	0%
1	12%	119.49%	47.74%	46.27%	47.00%
3	11%	107.29%	55.24%	49.91%	52.44%
5	12%	103.46%	57.00%	50.46%	53.53%

Table 3: Metrics for varying numbers of demonstrations

In this experiments we test a varying number of demonstrations in the prompts. The embedder used is D2F and only user utterances are included.
With 0 demonstrations the output of the model is always invalid, because there is no reference for how the syntax should be, therefore 0 keys are predicted and the results reflect this (the SER being exactly 100% is an artifact of this edge case).
We believe that the model could have performed better just by having information on the structure of the reply we expected, without necessarily attending to full examples. In this spirit, future work could explore the impact of grammar-constrained decoding to see how a more robust zero-shot baseline would perform.
With the exception of the JA remaining mostly stable, the increase of the number of demonstrations improves all the other metrics.
The improvement between 3 and 5 demonstrations is lower than the one between 1 and 3, suggesting that increasing more could give diminishing returns and that we are close to the best performance we can get to by changing only this parameter.

5 Conclusions

While ICL remains below the performance of fine-tuned DST models reported in the literature, our results show that it achieves reasonable slot-level accuracy with minimal task-specific training. Those results are placing ICL as a resource-efficient alternative to fine-tuning for DST, provided that there is enough relevant context for the LLM.
On top of this, our findings suggest three important takeaways:

- In the context of task-oriented dialogues, including agent utterances substantially improves performance. We can deduce that it adds relevant data to the context, and we suspect this could be true for other kind of additional information about the context of the dialogue.
- While specialized embedders like D2F appear to be a logical choice, a general purpose models like LaBSE was more efficient. This suggests that for ICL, a 'coverage-based' retrieval strategy that shows different slot combinations may be more effective than a 'pure similarity' strategy. Future work should further explore this direction by explicitly enforcing intra-demonstration diversity and penalizing same-dialogue or same-slot duplicates, for example through Maximal Marginal Relevance or slot-coverage-based retrieval strategies.
- While increasing the number of demonstrations improves local slot prediction (Precision and Recall), it wasn't the case for JA. This suggests that adding more examples helps the model refine individual values but does not help on addressing the overall complexity of the dialogue state. The persistent gap in JA highlights the difficulty for LLMs to maintain globally consistent dialogue states, where a single slot error invalidates the entire turn.

We believe research in this area can be essential in understanding how to get the best out of LLMs while reducing the resources needed for training them. However, these conclusions are limited to a single model and dataset, and more experiments involving different combination of models and dataset are necessary to assess whether these findings can be generalized.

References

[1] Hegde et al. *Factors affecting the in-context learning abilities of LLMs for dialogue state tracking*, Interspeech 2025.

[2] Šimon Sedláček *SpokenWOZ-whisper*, `https://huggingface.co/datasets/pirxus/spokenwoz-whisper`

[3] Tomáš Nekvinda, BUT Speech@FIT *MultiWOZ Evaluation*, `https://github.com/BUTSpeechFIT/MultiWOZ_Evaluation`