

# Emotionally Intelligent Task-oriented Dialogue Systems: Architecture, Representation, and Optimisation

Shutong Feng, Hsien-chin Lin, Nurul Lubis, Carel van Niekerk, Michael Heck,  
Benjamin Ruppik, Renato Vukovic, Milica Gašić

Heinrich Heine University Düsseldorf, Germany

{fengs, linh, lubis, niekerk, heckmi, ruppik, revuk100, gasic}@hhu.de

## Abstract

Task-oriented dialogue (ToD) systems are designed to help users achieve specific goals through natural language interaction. While recent advances in large language models (LLMs) have significantly improved linguistic fluency and contextual understanding, building effective and emotionally intelligent ToD systems remains a complex challenge. Effective ToD systems must optimise for task success, emotional understanding and responsiveness, and precise information conveyance, all within inherently noisy and ambiguous conversational environments. In this work, we investigate architectural, representational, optimisational as well as emotional considerations of ToD systems. We set up systems covering these design considerations with a challenging evaluation environment composed of a natural-language user simulator coupled with an imperfect natural language understanding module. We propose **LUSTER**, an LLM-based Unified System for Task-oriented dialogue with End-to-end Reinforcement learning with both short-term (user sentiment) and long-term (task success) rewards. Our findings demonstrate that combining LLM capability with structured reward modelling leads to more resilient and emotionally responsive ToD systems, offering a practical path forward for next-generation conversational agents.

## 1 Introduction

Conversational artificial intelligence (AI) enables machines to engage in natural, human-like interactions using spoken or written language. The rapid advancement of large language models (LLMs) has significantly enhanced the capabilities of conversational AI (Touvron et al., 2023; Abdin et al., 2024). Trained on vast amounts of text data, LLMs can generate fluent, context-aware responses, and

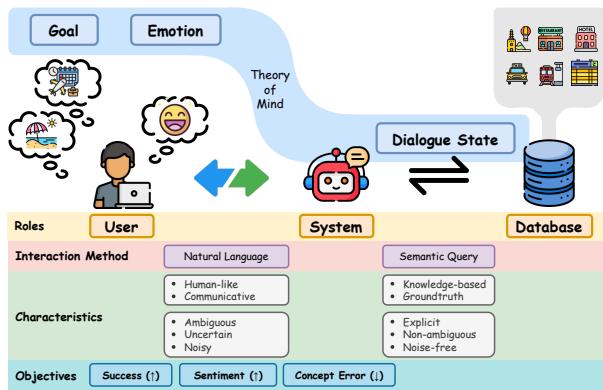


Figure 1: Goal-driven Human-machine Interaction

follow complex instructions. This significantly improves chat applications in both linguistic fluency and handling of nuanced or open-ended interactions (OpenAI, 2023; DeepSeek-AI, 2025).

Among the diverse conversational AI applications, task-oriented dialogue (ToD) systems stand out as a specialised class to help users accomplish a goal through several turns of natural language interactions (Jurafsky and Martin, 2025) (see Figure 1). Beyond task completion, effective communication also hinges on the user’s emotional experience (Picard, 1997). The user expresses both goal-driven and emotional cues in inherently ambiguous, uncertain, and noisy natural language (Feng et al., 2022). The system interprets this input and estimates a dialogue state, an internal representation of the user’s goal, functionally akin to a theory of mind (Premack and Woodruff, 1978). This state, typically represented symbolically (e.g., slot-value pairs), supports precise, unambiguous queries to a database, which likewise stores information in a structured, symbolic format. Thus, high-performing ToD systems must (1) **maximise task success** by correctly fulfilling the user’s goal, (2) **enhance sentiment** by maintaining a positive emotional interaction, and (3) **minimise error** in conveying database information. Related errors are termed hallucina-

tions in natural language generation (NLG) (Jurafsky and Martin, 2025), word errors in automatic speech recognition (Young et al., 2006), and reasoning errors in neural-symbolic systems (Garcez et al., 2002). In ToDs, explicitly represented semantics enable detection of both missing and hallucinated values, and we collectively term them as *concept errors*.

Designing a ToD system involves critical design and training choices that shape its ability to understand and assist users. A key **architectural** choice lies in whether to adopt a modular, end-to-end, or agentic approach. Modular systems separate the dialogue pipeline into components such as natural language understanding (NLU), dialogue state tracking (DST), policy, and NLG, each trained individually (Young et al., 2013). In contrast, end-to-end systems model the entire pipeline as a single unit for reduced error propagation and joint optimisation (Wen et al., 2017). Recently, agentic designs have emerged, leveraging LLMs’ reasoning capability for task-oriented decision-making (Yao et al., 2023).

These architectures align with different **optimisation strategies**. The traditional divide-and-conquer method trains each module in isolation. While this approach offers a certain level of explainability of intermediate processing steps, there could be information loss between components, leading to overall suboptimal behaviour. Alternatively, optimising the dialogue policy in the full interactive loop can better adapt decision-making to natural-language interaction outcomes (Feng et al., 2024a). End-to-end trainable systems, if paired with reinforcement learning (RL), can allow holistic user adaptation but face challenges like data inefficiency and the need for sophisticated reward design.

**Emotion modeling** is another essential yet often overlooked aspect of ToDs (Feng et al., 2022). Systems may handle emotion implicitly via hidden language representations, or explicitly through dedicated emotion detection and response planning stages (Stricker and Paroubek, 2024). In RL, emotion can also be integrated into training by using user sentiment as a short-term reward to guide policy updates (Feng et al., 2024a).

The representational considerations vary across systems. For the **dialogue state**, most systems use explicit tracking to provide clear intermediate representations for downstream tasks and database

query, though latent states are also possible (Lin et al., 2024). **Dialogue actions** can likewise be explicit (Geishauser et al., 2022; Hosseini-Asl et al., 2020), typically delexicalised (domain, intent, slot types but no values), or implicit, encoded as latent vectors learned by the model (Lubis et al., 2020). While modular systems use NLG to produce lexicalised responses, end-to-end systems usually output delexicalised **responses** to facilitate learning and evaluation. However, the strong lexical capabilities of modern LLMs are reducing the necessity of delexicalisation in both action planning and response generation.

Finally, **training methodology** critically impacts system performance. Supervised learning (SL) on annotated data offers a strong foundation but struggles with diverse user behaviours. RL enables adaptive optimisation using short-term (e.g., user sentiment) and long-term (e.g., task success) rewards from simulated or real-world interactions. A hybrid training paradigm that combines SL with RL often best balances stability and adaptability (Williams et al., 2017).

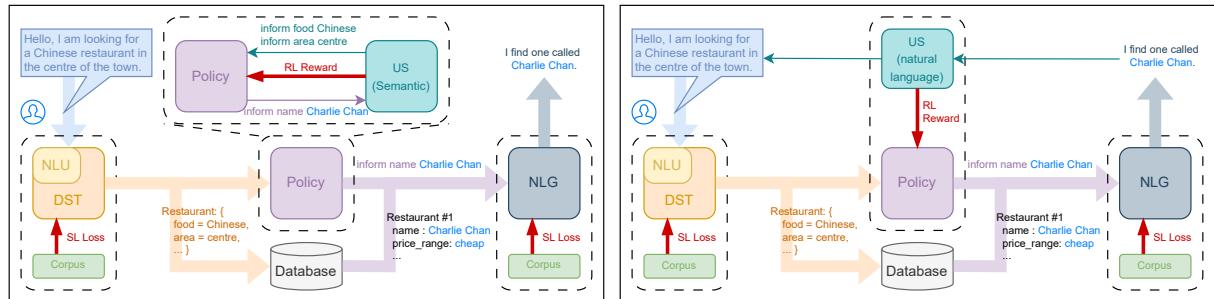
Together, these design axes form a rich space of possibilities and trade-offs involving scalability, flexibility, interpretability, and robustness. Our contributions are threefold:

- We construct a diverse suite of ToD systems representative of each design and training paradigm (Figure 2), and systematically analyse them.
- We harness the advantages of different design considerations in a principled manner and address a previously unexplored combination of design choices. Specifically, we build the first end-to-end ToD system that integrates fully lexicalised representations, an LLM backbone, and RL with both short-term affective and long-term task success rewards via online user simulation.
- With our proposed method, we build **LUSTER**, an LLM-based **Unified System for Task-oriented dialogues with End-to-end Reinforcement learning**. LUSTER substantially improves task success and reduces concept errors compared to alternative approaches.

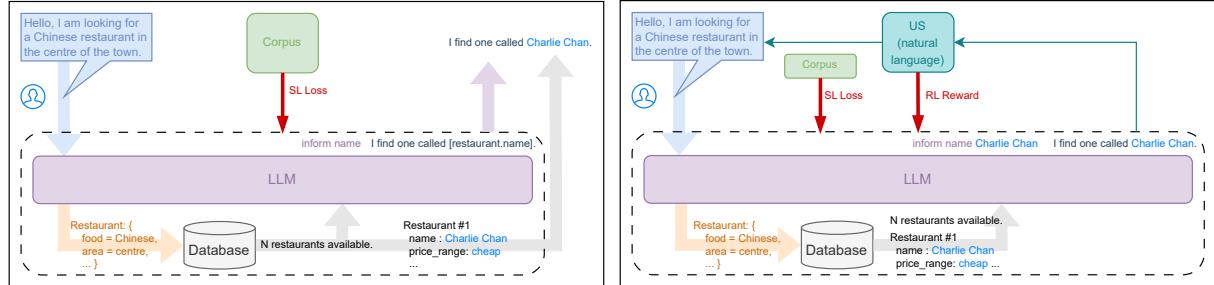
## 2 Related Work

### 2.1 Modular Systems

Traditional ToD systems typically follow a modular pipeline, with separate components: NLU for user intent understanding, DST for user goal estimation, policy for system action selection, and



(a) Modular system with individually optimised components.  
(b) Modular system with policy optimised in the natural-language interactive loop.



(c) (L)LM-based end-to-end system trained to imitate corpus.  
(d) (L)LM-based end-to-end system trained with both corpus imitation and RL.

Figure 2: ToD system training paradigms. Red arrows indicate learning signals for components in dotted boxes.

NLG for converting actions into natural utterances. Systems also interface with a database (DB) and rely on an ontology to structure symbolic queries and information abstraction.

This divide-and-conquer architecture (Figure 2a) ensures clear separation of objectives and allows component-level supervision. Modules are typically trained independently on annotated datasets such as MultiWOZ (Budzianowski et al., 2018), with SL applied to NLU, DST, and NLG, and RL applied to the policy. The interaction is framed as a partially observable Markov decision process (POMDP) (Young et al., 2013), and policy learning is carried out with simulated (Schatzmann et al., 2007; Kreyssig et al., 2018; Lin et al., 2023) or real (Gašić et al., 2013) users.

However, policies trained in isolation often misalign with other pipeline components during full-system interaction. To mitigate this issue, Feng et al. (2024a) optimise the policy within the natural-language interactive loop (Figure 2b).

To manage the complexity of the policy output space, modular dialogue policies typically produce delexicalised dialogue actions, omitting actual slot values. This abstraction simplifies policy learning by reducing action variability. The NLG module then realises these actions, together with slot values from database query results, into natu-

ral language utterances (Figures 2a and 2b).

## 2.2 End-to-end Systems

To overcome challenges in information passing and module coordination of modular pipelines, researchers have developed end-to-end systems that unify learning and eliminate the need for information transmission between modules. Early approaches demonstrate the feasibility of training dialogue systems as a single monolithic model (Wen et al., 2017). With advances in language models (LMs), researchers (Hosseini-Asl et al., 2020; Peng et al., 2021; Stricker and Paroubek, 2024) use decoder-only LMs to map dialogue history to belief states, database queries, and final responses in an auto-regressive manner (Figure 2c).

End-to-end systems typically retain the practice of using delexicalised dialogue actions from modular pipelines. They also generate responses containing placeholders (e.g., <v.food>) rather than actual entity values. This allows weight sharing across values of the same slot type and facilitates corpus-based supervised learning and evaluation. The substitution of placeholders with retrieved slot values is handled as a separate, post-processing operation (Figure 2c).

While these models perform competitively on datasets, the lexicalisation of these responses is

not robust and often leads to failures in interactive settings (Feng et al., 2024a).

Several efforts have incorporated RL into the training of end-to-end ToD systems. Jang et al. (2022) combine offline behavior cloning with a critic model that evaluates and filters self-generated delexicalised responses from an LM-based end-to-end ToD system. Lubis et al. (2020) tackle the challenge of the large action space inherent in word-level sequential decision-making by introducing a variational model that defines a latent action space. This abstraction enables more tractable RL training via simulated yet still delexicalised interactions for an end-to-end ToD system.

### 2.3 Agentic Systems

The advent of general-purpose LLMs has enabled agentic ToD systems that perform complex dialogue reasoning via prompting rather than task-specific training. These models adapt to diverse dialogue tasks via prompt engineering using task descriptions, demonstrations, or dialogue history under a zero- or few-shot setting. Techniques such as prompt chaining (Wei et al., 2022) and self-refinement loops (Yao et al., 2023) enhance reasoning and consistency, though at the cost of higher computational complexity.

ToD system developers have leveraged this ability to construct both individual components, such as slot filler (Sun et al., 2024), dialogue state tracker (Heck et al., 2023), and emotion recognition (Feng et al., 2024b), as well as agentic systems using carefully designed prompts (Li et al., 2024). Vukovic et al. (2024) demonstrates that chain-of-thought prompting can also be used to automate ontology construction, significantly reducing human effort in domain specification.

Despite their flexibility, agentic systems come with trade-offs. LLMs operating in zero- or few-shot settings remain sensitive to prompt phrasing, often exhibit unstable behaviour across sessions, and lack consistent control over generated outputs (Errica et al., 2025). They also underperform compared to models with task-specific optimisation, particularly in high-stakes or domain-intensive scenarios where robustness and consistency are critical (Hudeček and Dusek, 2023).

**In summary**, modular systems offer interpretability but suffer from information loss across modules; end-to-end systems provide unified learning but face challenges in out-of-corpus gen-

eralisation; agentic approaches offer flexibility and low training overhead but still lag in reliability and controllability. Their complementary strengths motivate a unified approach explored in this work to combine all potential benefits.

## 3 LUSTER: LLM-based Unified System for Task-oriented dialogue with End-to-end Reinforcement Learning

### 3.1 Inference with LUSTER

As illustrated in Figure 3, LUSTER performs sequential inference over core ToD subtasks, appending each output to the prompt to guide the next generation step. This auto-regressive process enables dynamic integration of contextual and affective cues. Motivated by prior work in affective modelling in ToDs (Feng et al., 2024a), LUSTER augments traditional ToD subtasks (DST, policy optimisation, and NLG) with two additional tasks: user emotion recognition and system conduct (affective behaviour) selection. This supports emotionally intelligent and goal-aligned responses.

Unlike prior (L)LM-based end-to-end systems that separate generation steps with special tokens (e.g., <|state|>), LUSTER adopts a lightweight prefix-based format inspired by Li and Liang (2021). Each task starts with a natural-language cue (e.g., “state :”) and ends with a standard end-of-sequence token. This simplifies implementation and improves compatibility with LLM tokenisers. We also explored rare or reserved tokens as markers, but they are often skipped or misgenerated due to LoRA-induced numerical imprecision, destabilising downstream behaviour.

**Step 0: Dialogue History Encoding** At the  $t$ -th turn, LUSTER receives a dialogue history  $\mathcal{H}_t$ , constructed as an alternating sequence of user and system utterances, concluding with the most recent user input:

$$\mathcal{H}_t := \{\text{user} : u_{\text{usr},1}, \text{system} : u_{\text{sys},1}, \dots, \text{user} : u_{\text{usr},t}\} \quad (1)$$

**Step 1: User Emotion Recognition** The model first predicts the user’s emotional state  $e_t$  by selecting the token from a discrete set of emotions  $E$  corresponding to the highest logit score:

$$e_t = \arg \max_{e \in E} \text{LLM}(\mathcal{H}_t, \text{“emotion :”}).\text{logit}(e) \quad (2)$$

**Step 2: Active Domain Recognition** Next, the model identifies the active domain  $d_t$  from the pre-defined set of domains  $D$ :

$$d_t = \arg \max_{d \in D} \text{LLM}(\mathcal{H}_t, e_t, \text{“domain :”}).\text{logit}(d) \quad (3)$$

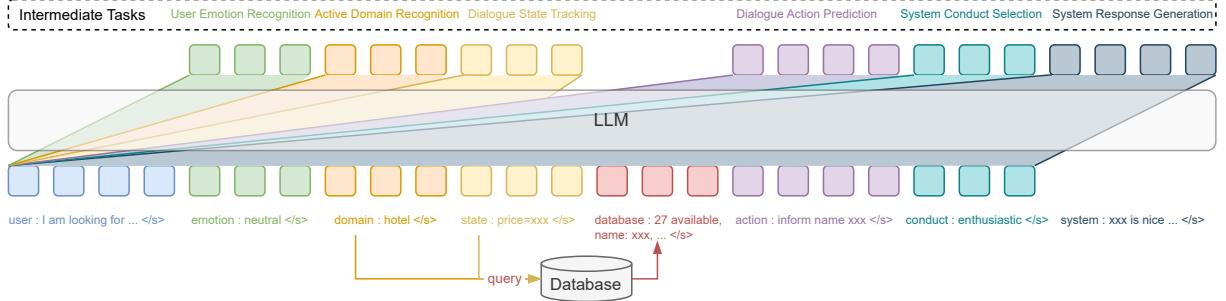


Figure 3: LUSTER model architecture and supervised training objectives.

**Step 3: Dialogue State Tracking** The dialogue state  $ds_t$  is generated for the selected domain only, in order to limit memory usage and maintain relevance. All  $m$  slots of the active domain are filled according to the ontology with either: 1) a freely generated value for non-categorical slots, 2) a selected value from a candidate list for categorical slots, or 3) “unknown” if the slot is not mentioned by the user.

$$ds_t := \{slot_1 : value_1; \dots; slot_m : value_m\} \quad (4)$$

$$ds_t = \text{LLM}(\mathcal{H}_t, e_t, d_t, "state :") \quad (5)$$

**Step 4: Database Query** The generated state is then used to retrieve the list of matching entries  $ent$  from the database. The database description  $db_t$  includes the number of matching entities and all attribute-value pairs of the first matched entity:

$$ent = \text{database.query}(d_t, ds_t) \quad (6)$$

$$db_t := \{\text{len}(ent) \text{ found} : \text{attr val } \forall (\text{attr}, \text{val}) \in ent[0]\} \quad (7)$$

**Step 5: Dialogue Action Prediction** Next, the model generates dialogue actions  $da_t$  as a set of intent-slot-value tuples:

$$da_t := \{\text{intent}_1 \text{ slot}_1 \text{ value}_1; \dots\} \quad (8)$$

$$da_t = \text{LLM}(\mathcal{H}_t, e_t, d_t, ds_t, db_t, "action :") \quad (9)$$

**Step 6: System Conduct Selection** The system’s emotional stance  $c_t$  is then selected from a set of conducts  $C$ :

$$c_t = \arg \max_{c \in C} \text{LLM}(\mathcal{H}_t, e_t, d_t, ds_t, db_t, da_t, "conduct :").\logit(c) \quad (10)$$

**Step 7: System Response Generation** Finally, LUSTER generates the system’s natural language response  $u_{sys,t}$  conditioned on the full context:

$$u_{sys,t} = \text{LLM}(\mathcal{H}_t, e_t, d_t, ds_t, db_t, da_t, c_t, "system :") \quad (11)$$

### 3.2 Fully Lexicalised Representation

Different from how lexicalisation is involved in the generation process of modular and end-to-end

systems as discussed in Section 2.1 and 2.2, LUSTER uses fully lexicalised representation throughout the generation, leveraging the world knowledge and generative strength of large pretrained LLMs (Figure 2d and 3): (1) **Dialogue actions** include slot values along with intents and slots. (2) **Responses** are generated directly in lexicalised form, eliminating placeholders and subsequent substitution. (3) The **database results** are represented as structured, value-rich entity descriptions, not just query summaries (e.g., “3 options found”).

LUSTER’s lexicalised approach simplifies inference by eliminating the need for post-hoc lexicalisation. By using fully lexicalised representations and natural-language prefixes (e.g., state:”), we avoid modifying the LLM’s vocabulary or resizing its embedding and output layers. This means these layers do not need to be fully re-trained, even when using low-rank adaptation (LoRA Hu et al. 2022). As a result, the number of trainable parameters is significantly reduced, making RL via online interaction feasible. Moreover, the lexicalised setup enhances modularity and future compatibility, enabling easy substitution of the LLM backbone as newer models emerge.

## 3.3 Training Objectives

### 3.3.1 Supervised Learning

An overview of LUSTER’s training procedure is shown in Figure 2d. We begin by training the model to imitate successful demonstrations from human operators recorded in the EmoWOZ corpus (Feng et al., 2022). Each training example is formatted from a dialogue turn as a fully lexicalised sequence  $x = \{\mathcal{H}_t, e_t, d_t, ds_t, db_t, da_t, c_t, u_{sys,t}\}$ , following the inference structure described in Section 3.1. Here,  $x$  is a token sequence of length  $n = |x|$ , denoted by  $(x_1, x_2, \dots, x_n)$ .

We train LUSTER using the standard causal language modelling objective (Radford et al.,

2019). Given a dataset  $\mathcal{X} = \{x^1, x^2, \dots, x^{|\mathcal{X}|}\}$ , we optimise the model parameters  $\theta$  to minimise the negative log-likelihood of the tokens in each sequence:

$$p(x) = \prod_{i=1}^n p(x_i|x_{<i}), \quad \mathcal{L}_{SL} = -\sum_{j=1}^{|\mathcal{X}|} \sum_{i=1}^{n_j} \log p_\theta(x_i^j|x_{<i}^j) \quad (12)$$

This objective provides a reliable initialisation by training the model to generate coherent completions based on dialogue context and intermediate representations from dataset demonstrations.

### 3.3.2 Reinforcement Learning with Emotion Reward Signal

Beyond large-scale pretraining and model size, LLMs' effectiveness also stems from RL-based alignment with human preferences (Ouyang et al., 2022b). Language generation can be framed as a sequential decision-making process, where the model (policy  $\pi$ ) generates tokens (actions  $a$ ) based on context (states  $s$ ) following  $a \sim \pi(\cdot|s)$ , and a reward model, often trained on human preference, guides optimisation. We adopt a similar RL set-up, using user sentiment as a reward to encourage system responses that are both task-effective and emotionally appropriate.

**Episode Formulation** Here, an episode corresponds to the generation of the system response at the  $t$ -th user-system interaction in a dialogue, resulting in the following full sequence:

$$\{\mathcal{H}_t, e_t, d_t, dst_t, db_t, da_t, c_t, u_{sys,t}\} \quad (13)$$

To keep episodes manageable and avoid excessive context length, we define the initial state as:

$$s_t^0 = \{\mathcal{H}_t, e_t, d_t, dst_t, db_t\} \quad (14)$$

Note that superscripts denote the time step in the generation of a single turn, while the subscript  $t$  refers to the dialogue-level time step, which remains fixed during that turn.

Starting from the initial state, each action token  $a_t^l$  is generated from the policy  $\pi_\theta(a_t^l|s_t^l)$ . We obtain the reward  $r_t^{l+1}$ , and the state is incrementally updated:

$$s_t^l = \{s_t^l, a_t^{<l}\} \quad (15)$$

At the end of the turn-level generation, we have a list of  $L$  action tokens:

$$a_t^{0:L-1} = \{da_t, c_t, u_{sys,t}\} \quad (16)$$

The LLM-based policy  $\pi_\theta$  then generates an action (i.e., text token) according to  $\pi_\theta(a_t^l | s_t^l)$ .

### Reward Modelling via Emotion Advantage

We expect the availability of user utterance and emotion prediction from a user simulator (US). Given the system response  $u_{sys,t}$ , the US produces the next user turn and emotion prediction:

$$u_{usr,t+1}, e_{t+1} = \text{US}(u_{sys,t}) \quad (17)$$

To explore diverse behaviours, we sample multiple turn-level trajectories from the same initial state:

$$\tilde{a}_t^{0:L-1} = \{\tilde{d}_{at}, \tilde{c}_t, \tilde{u}_{sys,t}\} \sim \prod_{l=1}^L \pi(a_t^l | s_t^l) \quad (18)$$

One of the sampled sequences is then randomly selected to continue the conversation with the US. Each sampled response  $\tilde{u}_{sys,t}$  is evaluated by the US to produce an emotion label  $\tilde{e}_{t+1}$ , which is mapped to a valence-based numeric reward with the mapping function  $\mathcal{M}(\cdot)$ : +1 for positive, 0 for neutral, and -1 for negative. Intermediate rewards  $r_t^{1:L-1} = 0$ , and the final reward is normalised across samples:

$$\tilde{r}_t^L = \frac{\mathcal{M}(\tilde{e}_t) - \mathbb{E}[\mathcal{M}(\tilde{e}_t)]}{\sigma(\mathcal{M}(\tilde{e}_t))} \quad (19)$$

We refer to this signal as the **emotion advantage**, as it captures how much emotionally better (or worse) a generated response is compared to alternatives under the same context. In practice, we drop the set of sampled responses from the same context that lead to the same user emotion (zero variance in  $\mathcal{M}(\tilde{e}_t)$ ).

**User Emotion Optimisation** We aim to maximise the expected return of the policy  $\pi_\theta$  across the sampled trajectories with turn-level discount factor  $\gamma_{\text{turn}}$ :

$$J(\theta) = \mathbb{E}_{\pi_\theta} \sum_{l=0}^{L-1} \gamma_{\text{turn}}^l r_t^{l+1} \quad (20)$$

Gradients are estimated using the policy gradient theorem and the advantage function  $A_{\pi_\theta}(s_t^l, a_t^l)$  (Sutton et al., 1999):

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \sum_{l=0}^{L-1} \nabla_\theta \log \pi_\theta(a_t^l | s_t^l) A_{\pi_\theta}(s_t^l, a_t^l) \quad (21)$$

In practice, we apply the Proximal Policy Optimization (PPO) algorithm (Schulman et al., 2017) to approximate the direction of the gradient above with a gradient of the surrogate loss. PPO stabilises training by clipping large policy updates and penalising Kullback–Leibler (KL) divergence between successive policies. For full implementation details, please refer to Appendix A.2.

### 3.3.3 Reinforcement Learning with Multi-turn Reward Signal

While single-turn RL can encourage the system to elicit positive user emotion, effective goal fulfillment requires planning over longer horizons. We therefore extend our training set-up to incorporate multi-turn RL with dialogue-level reward.

**Episode Formulation** An episode corresponds to a full dialogue, consisting of  $T$  user-system turns. At the  $t$ -th step in the dialogue, we derive the state and action with a formatter function  $\mathcal{F}(\cdot)$  from Equations 14 and 16: the dialogue-level state  $\mathcal{S}_t = \mathcal{F}(s_t^0)$  and dialogue-level action  $\mathcal{A}_t = \mathcal{F}(a_t^{0:L-1})$  (see Appendix A.3 for formatter details). This forms a sequence of state-action pairs  $\{(\mathcal{S}_0, \mathcal{A}_0), \dots, (\mathcal{S}_{T-1}, \mathcal{A}_{T-1})\}$  over the course of the interaction.

**Reward Modelling** To guide long-term behaviour, we introduce dialogue-level reward signals based on task success and interaction efficiency. We train two dialogue-level critic networks for our policy  $\pi$  parameterised by  $\phi$  and  $\psi$ : state-action value function  $Q_\phi^\pi(\mathcal{S}_t, \mathcal{A}_t)$  and state value function  $V_\psi^\pi(\mathcal{S}_t)$ . Both critics are based on ALBERT (Lan et al., 2020), with a linear projection on the [CLS] token embedding to produce scalar value estimates. Following the Archer framework (Zhou et al., 2024), they are trained to minimise temporal difference (TD) error and ensure Bellman consistency using online experience  $\mathcal{E} = \{\mathcal{S}_t, \mathcal{A}_t, r_{t+1}, \mathcal{S}_{t+1}\}_{t=0}^{T-1}$  collected during interactions with a user simulator:

$$\mathcal{L}_Q = \mathbb{E}_{s,a,r,s' \sim \mathcal{E}} [(Q_\phi(s, a) - r - \gamma_{\text{dial}} V_\psi(s'))^2] \quad (22)$$

$$\mathcal{L}_V = \mathbb{E}_{s \sim \mathcal{E}} [\mathbb{E}_{a \sim \pi_\theta(\cdot|s)} [(V_\psi(s) - Q_{\bar{\phi}}(s, a))^2]] \quad (23)$$

The delayed target models  $Q_{\bar{\phi}}$  and  $V_{\bar{\psi}}$  are updated towards their current counterparts with Polyak averaging (Haarnoja et al., 2018). For full critic training details, refer to the work of Zhou et al. (2024) and Appendix A.3.

**Task Completion Optimisation** Given the trained critics, we compute the advantage estimate at each  $t$ -th turn using:

$$A_t = Q_\phi^\pi(\mathcal{S}_t, \mathcal{A}_t) - V_\psi^\pi(\mathcal{S}_t) \quad (24)$$

This value measures how much better a particular action is compared to the expected return of the current state. We then use this advantage signal in the same PPO optimisation framework as described in the set-up for emotion reward (Section 3.3.2), replacing the emotion-based advantage with this task-oriented multi-turn advantage.

**Combined Reward** The single-turn (Equation 19) and multi-turn reward (Equation 24) signals are combined for selected sequences from Equation 18 using a weighted sum with mixing parameter  $\rho$  (set to 0.1 in practice):

$$\mathcal{R}_{\text{combine},t} = \rho A_t + r_t^L \quad (25)$$

**Off-policy Experience Replay** Due to the high computational cost of simulated interactions, we maintain an experience buffer and apply importance sampling (IS) (Precup et al., 2000) to reuse off-policy data. A constant buffer size is enforced using reservoir sampling (Isele and Cosgun, 2018). The IS weight is defined as:

$$\alpha_t = \frac{\pi_{\theta_{\text{new}}}(\mathcal{A}_t | \mathcal{S}_t)}{\pi_{\theta_{\text{old}}}(\mathcal{A}_t | \mathcal{S}_t)} \quad (26)$$

In practice, computing IS weights over long sequences can be numerically unstable due to the sharp trajectory distribution of LLMs. This often leads to high variance in IS weight estimates, even when the behavior and target policies only diverge slightly. To mitigate this, we compute IS weights only over intents and slots of dialogue actions  $da_{t,\text{intent+slot}}$ , and clip them to  $[1 - \eta, 1 + \eta]$  with  $\eta = 0.9$ :

$$\hat{\alpha}_t = \text{clip}\left(\frac{\pi_{\theta_{\text{new}}}(da_{t,\text{intent+slot}} | \mathcal{S}_t)}{\pi_{\theta_{\text{old}}}(da_{t,\text{intent+slot}} | \mathcal{S}_t)}, 1 - \eta, 1 + \eta\right) \quad (27)$$

When deploying experience replay, we multiply the corresponding reward for past experiences (Equation 24 or 25) with IS weight  $\hat{\alpha}_t$ .

## 4 Experiments Set-up

### 4.1 ToD Systems

Based on design considerations discussed in Section 1, we consider the following ToD systems. System configurations are summarised in Table 1.

#### Naive Baseline

**Chitchat-ToD** has a 3B Phi-3.5-mini backbone and is trained to generate response directly, omitting all sub-tasks and the interaction with the database. We aim to investigate if an LLM can memorise the database solely from the corpus.

#### Agentic System

**FnCTOD** (Li et al., 2024) is an agentic system with a prompt-based zero-shot generation process: DST, database query, and response generation. We follow the prompt specified in the original paper and use GPT-4o-mini (2025-04-16).

	Architecture	Optimisation	Dialogue State	Dialogue Action	Emotion	RL Objectives Sent	Training Env	#Param
						Succ		
Chitchat-ToD	E2E	E2E	Implicit	Implicit	Implicit	-	NA	3.82B
FnCTOD (2024)	Agentic	No	Explicit	Implicit	Implicit	-	NA	8B(est.)
Modular-base (2022)	Modular	Modular	Explicit	Delexicalised	No	+	EmoUS	413M
Modular-emotion (2024a)	Modular	Modular	Explicit	Delexicalised	Explicit	+	EmoUS	547M
SimpleLoop (2022)	Modular	Loop	Explicit	Delexicalised	No	+	langEmoUS	413M
EmoLoop (2024a)	Modular	Loop	Explicit	Delexicalised	Explicit	+	langEmoUS	547M
SimpleLLAMA (2024)	E2E	E2E	Explicit	Delexicalised	Implicit	-	NA	7.02B
EmoLLAMA (2024)	E2E	E2E	Explicit	Delexicalised	Explicit	-	NA	7.02B
LUSTER-base	E2E	E2E	Explicit	Lexicalised	Implicit	-	NA	3.82B
LUSTER-base-emotion	E2E	E2E	Explicit	Lexicalised	Explicit	-	NA	3.82B
LUSTER-RL(sent)	E2E	E2E	Explicit	Lexicalised	Explicit	-	langEmoUS	3.82B
LUSTER-RL(succ)	E2E	E2E	Explicit	Lexicalised	Explicit	+	langEmoUS	3.82B
LUSTER	E2E	E2E	Explicit	Lexicalised	Explicit	+	langEmoUS	3.82B

Table 1: System configurations of ToD systems as discussed in Section 1. **Architecture:** end-to-end (*E2E*), *Modular*, or *Agentic*. **Optimisation:** *Modular* optimisation following Figure 2a, *Loop* optimisation following Figure 2b, end-to-end (*E2E*) following Figure 2c,2d or *No* task-specific optimisation. **Dialogue State:** *Explicit* representation or *Implicit* model hidden representations. **Dialogue Action:** *Delexicalised* where no slot values in policy output, *Lexicalised* where slot values are included, or *Implicit* model hidden representations. **Emotion:** *Explicit* policy input/output, *Implicit* consideration from natural language, or *Not* perceived nor expressed by the policy. **RL Objectives:** RL using *Sentiment* or task *Success* reward signal. **Training Environment** for RL training: *NA* when no RL involved, *EmoUS* for semantic interaction, and *langEmoUS* for natural-language interaction. For number of parameters **#Param**, the size of GPT-4o-mini used as FnCTOD backbone is *estimated*.

### Modular Systems with Modular Optimisation

**Modular-base** has SetSUMBT DST (van Niekerk et al., 2021), DDPT policy (Geishauser et al., 2022), and SEC-BART NLG (Feng et al., 2024a). The policy is trained via RL using *semantic* interaction with an emotional user simulator EmoUS (Lin et al., 2023), receiving a success signal (Figure 2a).

**Modular-emotion** has all components of Modular-base with additionally user emotion (Feng et al., 2023) as policy input and system conduct as policy output. The policy also receives turn-level user sentiment as reward.

### Modular Systems with Loop Optimisation

**SimpleLoop** has the same architecture as Modular-base but its RL training involves success reward from natural-language interaction (Figure 2a) with an emotional user simulator langEmoUS (Feng et al., 2024a).

**EmoLoop** has the same pipeline set-up as Modular-emotion and the same training set-up as SimpleLoop with an additional turn-level sentiment reward signal.

### Delexicalised End-to-end Systems

**SimpleLLAMA** (Stricker and Paroubek, 2024) follows the best set-up of SimpleTOD (Hosseini-Asl et al., 2020) to

auto-regressively generate the state, dialogue action, and response. It uses the 7B LLAMA-2-chat (Touvron et al., 2023) as the backbone. It generates delexicalised dialogue actions and responses, and it is trained via SL on the corpus.

**EmoLLAMA** has two more emotion-related sub-tasks than SimpleLLAMA: user emotion recognition and system conduct selection.

### LUSTER and its Variants

**LUSTER-base** completes non-affective ToD modelling sub-tasks as listed in Figure 3. It uses a 3B Phi-3.5-mini and generates lexicalised actions and responses.

**LUSTER-base-emotion** has two more affective sub-tasks than LUSTER-base: user emotion recognition and system conduct selections.

**LUSTER-RL(sent)** is based on LUSTER-base-emotion and incorporates RL with turn-level *sentiment* reward as discussed in Section 3.3.2.

**LUSTER-RL(succ)** is based on LUSTER-base-emotion and incorporates RL with dialogue-level *success* reward as discussed in Section 3.3.3.

**LUSTER** is based on LUSTER-base-emotion and it incorporates RL using both turn-level sentiment reward and dialogue-level success reward.

## 4.2 Training and Evaluation Environment

All systems or sub-modules are trained on EmoWOZ (an extended version of MultiWOZ). Interactive evaluation is carried out using ConvLab-3 toolkit (Zhu et al., 2023). The natural-language user model consists of T5NLU (Zhu et al., 2023) and langEmoUS (Feng et al., 2024a). T5NLU reports a slot F1 of 86.5%, putting extra emphasis on the clarity of the system’s natural-language responses. LangEmoUS is a state-of-the-art user simulator and reports a macro F1 of 52.1% on emotion prediction. Details about hyperparameters and computing resource can be found in Appendices A.1 and A.5.

We simulate 3000 dialogues per system during evaluation. We report the success rate (judged by the ConvLab-3 evaluator) and the average sentiment (predicted by the user simulator). For success evaluation, the evaluator verifies whether the user’s goal has been met. This includes checking database constraints and ensuring that a booking was made as specified. The sentiment score is assigned for each turn as follows: +1 if the user is classified as “satisfied”, -1 if “dissatisfied” or “abusive”, and 0 otherwise.

## 5 Results and Discussion

### 5.1 Model Performance

	Success ( $\uparrow$ )	Sentiment ( $\uparrow$ )	Error ( $\downarrow$ )
Chitchat-ToD	20.5%	0.368	NA
FnCTOD	32.3%	0.229	NA
Modular-base	31.8%	0.335	5.35%
Modular-emotion	33.0%	$\dagger$ 0.345	$\dagger$ 5.14%
SimpleLoop	35.2%	0.346	4.74%
EmoLoop	37.2%	$\dagger$ 0.380	$\dagger$ 3.36%
SimpleLLAMA	33.0%	0.214	5.11%
EmoLLAMA	34.2%	$\dagger$ 0.250	$\dagger$ 3.68%
LUSTER-base	47.2%	0.298	1.73%
LUSTER-base-emotion	47.5%	$\dagger$ 0.310	$\dagger$ 1.38%
LUSTER-RL(sent)	$\dagger$ 49.8%	$\dagger$ 0.345	1.79%
LUSTER-RL(succ)	$\dagger$ 50.8%	$\dagger$ 0.324	$\dagger$ 1.51%
LUSTER	$\dagger$ 51.2%	$\dagger$ 0.316	1.77%

Table 2: Task **success**, user **sentiment**, and concept **error** rate of all systems. Best scores across all systems are **bolded**.  $\dagger$ denotes statistically significant improvement ( $p < 0.05$ ) from the first system within each group.

**Chitchat-style or agentic model lacks the optimisation necessary for goal-oriented task completion.** Chitchat-ToD performs poorly on success rate (20.5%) despite generating positive

sentiment, as it often generates ungrounded content that appears helpful but fails to meet the user’s search constraints. This underscores that smooth, engaging interactions alone are insufficient. FnCTOD, despite comparable success rate to some systems (32.3%), falls short on user sentiment, likely due to its overly informative and less engaging language (see Section 5.2). These findings suggest effective task fulfillment requires both fact-driven database access and emotionally attuned interaction strategies.

**Loop-based optimisation enables more coherent behaviour.** SimpleLoop and EmoLoop, which train policies within the full natural-language interaction loop, outperform their modularly trained counterparts despite having the same architecture. This shows that loop optimisation provides holistic feedback, leading to more coherently optimised behaviours.

**Emotion modelling improves user sentiment across architectures.** Across both modular (e.g., SimpleLoop vs. EmoLoop) and end-to-end (e.g., SimpleLLAMA vs. EmoLLAMA) systems, incorporating user emotion recognition and system conduct prediction consistently improves user sentiment. EmoLoop achieves the highest sentiment score, highlighting the effectiveness of affect-aware modelling for generating emotionally intelligent responses.

**Lexicalised systems enable better grounding and task success.** LUSTER variants significantly outperform delexicalised ones in both success and error rates. Providing full lexical access to slot values improves grounding and overall task effectiveness. This suggests that while symbolic abstractions such as dialogue states and actions remain useful, the enhanced reasoning and representation capacity of LLMs benefits from direct access to lexical content, enabling more accurate and contextually consistent responses.

**RL enhances task success and user sentiment of LUSTER variants’ performance.** Incorporating turn-level sentiment rewards boosts user sentiment, while dialogue-level success signals increase task success to 50.8%. The full model, LUSTER, optimises both objectives, achieving the highest task success (51.2%) while maintaining stable sentiment and concept error. This demonstrates the effectiveness of hierarchical RL in balancing short-term emotional response with long-term goal achievement.

## 5.2 Linguistic Style

	#Turns	#Tokens	Vocab	LogTTR
Chitchat-ToD	17.4	12.9	4,114	0.853
FnCTOD	6.3	29.9	1,301	0.831
Modular-base	16.3	14.0	2,333	0.823
Modular-emotion	16.2	14.2	2,486	0.824
SimpleLoop	15.3	12.8	3,628	0.833
EmoLoop	14.5	11.6	2,256	0.845
SimpleLLAMA	14.6	13.7	2,109	0.834
EmoLLAMA	14.0	12.8	2,035	0.838
LUSTER-base	11.1	11.2	1,841	0.881
LUSTER-base-emotion	11.4	11.3	1,928	0.873
LUSTER-RL(sent)	11.3	11.0	2,043	0.877
LUSTER-RL(succ)	11.6	12.4	2,048	0.870
LUSTER	11.5	12.5	2,026	0.872

Table 3: Average number of user-system exchanges per dialogue (#Turns), average number of tokens per system turn (#Tokens), average number of unique tokens (Vocab), and average log type-token ratio (LogTTR) per dialogue of all systems.

Chitchat-ToD takes the most turns per dialogue (17.4 on average) due to the lack of database grounding for effective user goal fulfillment. In contrast, FnCTOD, completes dialogue on average in 6.3 turns using overly informative responses (nearly 30 tokens per turn). Loop-optimised systems ({Simple,Emo}Loop), compared with their modularly trained counterparts (Modular-{base,emotion}), achieve better performance with fewer turns, highlighting the better efficiency of policies as a result of natural-language optimisation. LUSTER variants, benefiting from lexicalised end-to-end modeling and RL, are the most efficient. They achieve overall best success with an average around 11 turns.

Most systems use around 2,000 unique tokens, reflecting lexical compactness for efficient and goal-directed interactions. SimpleLoop uses over 3,600 tokens due to frequent but unnecessary mentions of phone numbers. Chitchat-ToD, being open-ended and unconstrained, uses the largest vocabulary (4,114). FnCTOD has the smallest (1,301) because of long templated responses.

We also measure the within-dialogue log ratio between the number of unique and total tokens (LogTTR) for a fairer comparison of lexical diversity. LUSTER variants generally achieve high LogTTR ( $> 0.87$ ), indicating greater within-dialogue lexical variation. In contrast, systems like Modular-base exhibit lower diversity despite larger vocabularies than LUSTER.

## 5.3 System Comparison Using LLM Raters

LUSTER variants outperform other models in task success and concept error (Table 2). However, differences among LUSTER variants are small, likely due to limitations of the rule-based evaluator and pretrained user simulator in capturing fine behavioural nuances. To obtain more sensitive and human-aligned assessment, we employ proprietary LLMs as judges. Each system interacts with the user simulator under the same user goal. Their dialogues are presented in random order to GPT-4o-mini (2025-04-16) and Deepseek-R1 (2025-05-28), tasked with identifying the better system (see Appendix A.6.1).

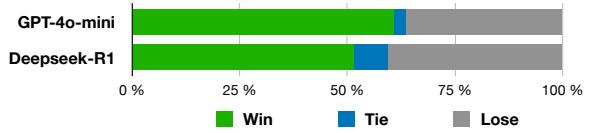


Figure 4: LUSTER vs. LUSTER-base-emotion.

Figure 4 shows the win-rate comparison between LUSTER and its non-RL counterpart, LUSTER-base-emotion for 500 dialogues. Both LLM raters exhibit a clear preference for LUSTER: GPT-4o-mini attributes a strong majority of wins to LUSTER, while Deepseek-R1 also favors LUSTER, albeit with a more moderate margin. These results confirm that RL contributes to nuanced yet perceptible improvements in conversational quality. Appendix A.6.2 exemplifies how LLMs identify behavioural nuances.

## 6 Conclusion

In this work, we systematically investigate the design dimensions of ToD systems, spanning architectural choices, optimisation strategies, representation, and emotion modeling. Through experimentation across modular, end-to-end, and agentic systems, we find that no single design axis alone guarantees optimal performance. Instead, efficient task success, user satisfaction, and semantic accuracy emerge from their thoughtful combination. Our proposed LUSTER system demonstrates this by integrating lexicalised generation, emotion-aware modeling, and hierarchical reinforcement learning, achieving a significantly higher task success rate and overall balanced performance. These findings underscore the importance of holistic, end-to-end, yet structured design in building conversational agents capable of effective and emotionally intelligent task completion.

## References

- Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, et al. 2024. *Phi-3 technical report: A highly capable language model locally on your phone*. Technical Report MSR-TR-2024-12, Microsoft.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.
- Federico Errica, Davide Sanvito, Giuseppe Siracusano, and Roberto Bifulco. 2025. What did I do wrong? quantifying LLMs' sensitivity and consistency to prompt engineering. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1543–1558, Albuquerque, New Mexico. Association for Computational Linguistics.
- Shutong Feng, Hsien-chin Lin, Christian Geishauser, Nurul Lubis, Carel van Niekerk, Michael Heck, Benjamin Matthias Ruppik, Renato Vukovic, and Milica Gasic. 2024a. Infusing emotions into task-oriented dialogue systems: Understanding, management, and generation. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 699–717, Kyoto, Japan. Association for Computational Linguistics.
- Shutong Feng, Nurul Lubis, Christian Geishauser, Hsien-chin Lin, Michael Heck, Carel van Niekerk, and Milica Gasic. 2022. EmoWOZ: A large-scale corpus and labelling scheme for emotion recognition in task-oriented dialogue systems. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4096–4113, Marseille, France. European Language Resources Association.
- Shutong Feng, Nurul Lubis, Benjamin Ruppik, Christian Geishauser, Michael Heck, Hsien-chin Lin, Carel van Niekerk, Renato Vukovic, and Milica Gasic. 2023. From chatter to matter: Addressing critical steps of emotion recognition learning in task-oriented dialogue. In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 85–103, Prague, Czechia. Association for Computational Linguistics.
- Shutong Feng, Guangzhi Sun, Nurul Lubis, Wen Wu, Chao Zhang, and Milica Gasic. 2024b. Affect recognition in conversations using large language models. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 259–273, Kyoto, Japan. Association for Computational Linguistics.
- Artur S. d’Avila Garcez, Dov M. Gabbay, and Krysia B. Broda. 2002. *Neural-Symbolic Learning System: Foundations and Applications*. Springer-Verlag, Berlin, Heidelberg.
- M. Gašić, C. Breslin, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis, and S. Young. 2013. On-line policy optimisation of bayesian spoken dialogue systems via human interaction. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8367–8371.
- Christian Geishauser, Carel van Niekerk, Hsien-chin Lin, Nurul Lubis, Michael Heck, Shutong Feng, and Milica Gašić. 2022. Dynamic dialogue policy for continual reinforcement learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 266–284, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.
- Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR.
- Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-chin Lin, Carel van Niekerk,

- and Milica Gasic. 2023. ChatGPT for zero-shot dialogue state tracking: A solution or an opportunity? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 936–950, Toronto, Canada. Association for Computational Linguistics.
- Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS ’20, Red Hook, NY, USA. Curran Associates Inc.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.
- Vojtěch Hudeček and Ondrej Dusek. 2023. Are large language models all you need for task-oriented dialogue? In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 216–228, Prague, Czechia. Association for Computational Linguistics.
- David Isele and Akansel Cosgun. 2018. Selective experience replay for lifelong learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI’18/IAAI’18/EAAI’18. AAAI Press.
- Youngsoo Jang, Jongmin Lee, and Kee-Eung Kim. 2022. GPT-critic: Offline reinforcement learning for end-to-end task-oriented dialogue systems. In *International Conference on Learning Representations*.
- Daniel Jurafsky and James H. Martin. 2025. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*, 3rd edition. Online manuscript released January 12, 2025.
- Florian Kreyssig, Iñigo Casanueva, Paweł Budzianowski, and Milica Gašić. 2018. Neural user simulation for corpus-based policy optimisation of spoken dialogue systems. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 60–69, Melbourne, Australia. Association for Computational Linguistics.
- Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.
- Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.
- Zekun Li, Zhiyu Chen, Mike Ross, Patrick Huber, Seungwhan Moon, Zhaojiang Lin, Xin Dong, Adithya Sagar, Xifeng Yan, and Paul Crook. 2024. Large language models as zero-shot dialogue state tracker through function calling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8688–8704, Bangkok, Thailand. Association for Computational Linguistics.
- Hsien-Chin Lin, Shutong Feng, Christian Geishauser, Nurul Lubis, Carel van Niekerk, Michael Heck, Benjamin Ruppik, Renato Vukovic, and Milica Gasić. 2023. Emous: Simulating user emotions in task-oriented dialogues. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’23, page 2526–2531, New York, NY, USA. Association for Computing Machinery.
- Hsien-Chin Lin, Chia-Hao Shen, Benjamin Matthias Ruppik, Carel van Niekerk, Michael Heck, Nurul Lubis, Renato Vukovic, Shutong Feng, and Milica Gasic. 2024. Text as parameter: interactive prompt optimisation for large language models.

- Nurul Lubis, Christian Geishaeuser, Michael Heck, Hsien-chin Lin, Marco Moresi, Carel van Niekerk, and Milica Gasic. 2020. **LAVA: Latent action spaces via variational auto-encoding for dialogue policy optimization**. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 465–479, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Carel van Niekerk, Andrey Malinin, Christian Geishaeuser, Michael Heck, Hsien-chin Lin, Nurul Lubis, Shutong Feng, and Milica Gasic. 2021. **Uncertainty measures in neural belief tracking and the effects on dialogue policy performance**. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7901–7914, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- OpenAI. 2023. **Gpt-4 technical report**.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, et al. 2022a. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA. Curran Associates Inc.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, et al. 2022b. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA. Curran Associates Inc.
- Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. **Soloist: Building task bots at scale with transfer learning and machine teaching**. *Transactions of the Association for Computational Linguistics*, 9:807–824.
- Rosalind W. Picard. 1997. *Affective Computing*. MIT Press, Cambridge, MA.
- Doina Precup, Richard S. Sutton, and Satinder P. Singh. 2000. Eligibility traces for off-policy policy evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML ’00, page 759–766, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- David Premack and Guy Woodruff. 1978. **Does the chimpanzee have a theory of mind?** *Behavioral and Brain Sciences*, 1(4):515–526.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.
- Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. **Agenda-based user simulation for bootstrapping a POMDP dialogue system**. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152, Rochester, New York. Association for Computational Linguistics.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. **Proximal policy optimization algorithms**.
- Armand Stricker and Patrick Paroubek. 2024. **A Unified Approach to Emotion Detection and Task-Oriented Dialogue Modeling**. In *IWSDS*, Sapporo (Japon), Japan.
- Guangzhi Sun, Shutong Feng, Dongcheng Jiang, Chao Zhang, Milica Gasic, and Phil Woodland. 2024. **Speech-based slot filling using large language models**. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6351–6362, Bangkok, Thailand. Association for Computational Linguistics.
- Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS’99, page 1057–1063, Cambridge, MA, USA. MIT Press.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, et al. 2023. **Llama 2: Open foundation and fine-tuned chat models**.
- Renato Vukovic, David Arps, Carel van Niekerk, Benjamin Matthias Ruppik, Hsien-chin

- Lin, Michael Heck, and Milica Gasic. 2024. **Dialogue ontology relation extraction via constrained chain-of-thought decoding**. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 370–384, Kyoto, Japan. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA. Curran Associates Inc.
- Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. **A network-based end-to-end trainable task-oriented dialogue system**. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.
- Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. **Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning**. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 665–677, Vancouver, Canada. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. 2013. **Pomdp-based statistical spoken dialog systems: A review**. *Proceedings of the IEEE*, 101(5):1160–1179.
- Steve J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. 2006. *The HTK Book Version 3.4*. Cambridge University Press.
- Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. 2024. **ArCHer: Training language model agents via hierarchical multi-turn RL**. In *Forty-first International Conference on Machine Learning*.
- Qi Zhu, Christian Geishäuser, Hsien-chin Lin, Carel van Niekerk, Baolin Peng, Zheng Zhang, Shutong Feng, Michael Heck, Nurul Lubis, Dazhen Wan, Xiaochen Zhu, Jianfeng Gao, Milica Gasic, and Minlie Huang. 2023. **ConvLab-3: A flexible dialogue system toolkit based on a unified data format**. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 106–123, Singapore. Association for Computational Linguistics.

## A Supplementary Information

### A.1 Computing Resources

All modular systems were trained and deployed using a single NVIDIA RTX 2080 (8GB) graphics processing unit (GPU). The end-to-end systems SimpleLLAMA and EmoLLAMA were trained and evaluated on an NVIDIA A100 (80GB) GPU. LUSTER variants were trained on a single NVIDIA A100 (40GB) GPU, with inference requiring at least an NVIDIA RTX 6000 (24GB) GPU. While training LUSTER demands more computational resources than modular systems, it remains more resource-efficient than LLAMA-based end-to-end models.

### A.2 Proximal Policy Optimisation for Single-turn RL

The advantage function  $A_\pi(s, a)$  denotes the difference between the action-value function  $Q_\pi(s, a)$ , which is the expected return when taking action  $a$  in state  $s$  and from then on following the policy  $\pi$  and the state value function  $V_\pi(s)$ , which is the expected return of state  $s$  for policy  $\pi$ .

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s) \quad (\text{A.1})$$

In the context of single-turn response generation with an episode length of  $L$  (Equation 14) for the  $t$ -th interaction of a dialogue with a policy parameterised with  $\theta$ ,

$$Q_{\pi_\theta}(s, a) = \mathbb{E}_{\pi_\theta} \sum_{k=0}^{L-l-1} [\gamma^k r_t^{k+1} | s = s_t^l, a = a_t^l] \quad (\text{A.2})$$

$$V_{\pi_\theta}(s) = \mathbb{E}_{\pi_\theta} \sum_{k=0}^{L-l-1} [\gamma^k r_t^{k+1} | s = s_t^l] \quad (\text{A.3})$$

$$\hat{A}_t^l = Q_{\pi_\theta}(s, a) - V_{\pi_\theta}(s) \quad (\text{A.4})$$

The surrogate loss consisting of the clipping term and the KL-penalty term is given by:

$$\begin{aligned} \mathcal{L}_{\text{PPO}} = & \mathbb{E}_t \left[ \min \left( \frac{\pi_\theta(a_t^l | s_t^l)}{\pi_{\theta_{\text{old}}}(a_t^l | s_t^l)} \hat{A}_t^l, \text{clip} \left( \frac{\pi_\theta(a_t^l | s_t^l)}{\pi_{\theta_{\text{old}}}(a_t^l | s_t^l)}, 1 - \epsilon, 1 + \epsilon \right) \hat{A}_t^l \right) \right. \\ & \left. - \beta \cdot \sum_a \pi_{\theta_{\text{old}}}(a | s_t^l) \log \left( \frac{\pi_{\theta_{\text{old}}}(a | s_t^l)}{\pi_\theta(a | s_t^l)} \right) \right], \end{aligned} \quad (\text{A.5})$$

where  $\text{clip}(x, a, b)$  ensures that  $x$  is within the interval  $[a, b]$ . The gradient of the surrogate loss (Eq A.5) approximates the direction of the policy gradient (Eqn 21).

To estimate advantage function  $\hat{A}_t$  needed for the surrogate loss, generalised advantage estimation is used:

$$\hat{A}_t^l = \sum_{j=0}^{L-j-1} (\gamma_{\text{turn}} \lambda)^j \delta_t^{l+j} \quad \text{where} \quad \delta_t^l = r_t^l + \gamma_{\text{turn}} V_\pi(s_{t+1}^l) - V_{\pi_\theta}(s_t^l) \quad (\text{A.6})$$

Parameter  $\lambda \in [0, 1]$  trades off bias (lower  $\lambda$ ) and variance (higher  $\lambda$ ).  $V_{\pi_\theta}$  is initialised with the reward model  $R_\phi$  for complete sequences, and parameters  $\phi$  are updated in such a way so that  $V_{\pi_\theta}$  estimates observed returns for partial sequences. In this way, the critic parameterised with  $\phi$  influences the actor parameterised with  $\theta$ . We follow default hyperparameters in the PPO trainer of TRL library, which follows Ouyang et al. (2022a).

PPO is an online learning algorithm, which means it needs to adopt exploration during the process of optimisation. That is achieved by applying a temperature parameter to the softmax layer of policy  $\pi_\theta$  to induce more varied responses.

In practice, the critic is modelled by a single-layer value head attached to the output of the language model. It projects the logits of the token into a scalar. The input for calculating  $V_{\pi_\theta}(s)$  is the logit of the last token in  $s$ , and that for calculating  $Q_{\pi_\theta}(s, a)$  is the logit of  $a$ , since the action  $a$  is the next token generated.

### A.3 Dialogue-level Critics

#### A.3.1 Architecture

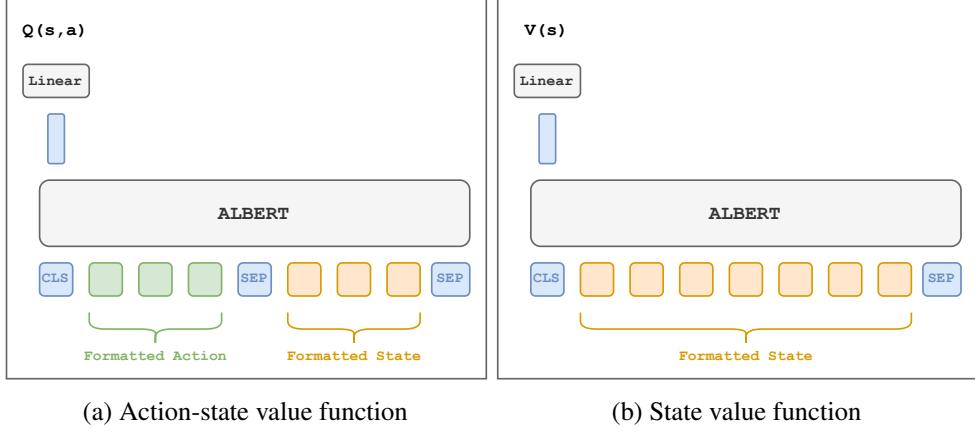


Figure A.1: Dialogue-level Critics

Figure A.1 illustrates the modelling of dialogue critics. We format the dialogue-level state  $S_t$  and action  $A_t$  from the generated sequence  $s_t^0$  and  $a_t^{1:L-1}$  to obtain a compact input to the critic.

**Textual State Representation** The input to the state value function is in the form of the following:

$<\text{domain}> <\text{slot}> <\text{"inactive"}|<\text{"informed"}|<\text{"unknown"}> ; \dots \text{[SEP]} <\text{N } <\text{domain}> \text{found"}|<\text{"no entity found"}|<\text{"database not available"} \text{[SEP]} \text{"system :"} u_{\text{sys},t-1} \text{[SEP]} \text{"user :"} u_{\text{usr},t}$

The first segment contains the system’s belief about the user’s goal. Instead of actual informed value, we limit the status of each slot to be one of the three: *informed*, *unknown* if not informed, or *inactive* if the  $<\text{domain}>$  is not the active domain determined by the system  $<\text{domain}> \neq d_t$ .

The second segment contains a summary of the database query status. The status informs the number of entities returned  $N$  for  $N > 0$  or informs “no entity found” if  $N = 0$ . There is a special case when the active domain is “general” for small-talk and greetings. Under such circumstances, the status informs “database not available”.

To account briefly for the dialogue history, we also include the previous system and user turns  $u_{\text{sys},t-1}$  and  $u_{\text{usr},t}$ .

**Textual Action Representation** The formatted action for the state-action value function is in the form of the following. Values are excluded.

$<\text{domain}> <\text{intent}> <\text{slot}> ; \dots$

#### A.3.2 Training

Following the objective functions defined in Equation 22 and 23, for each critic update step:

$$\phi \leftarrow \phi - \nabla J_\phi(Q) \quad (\text{A.7})$$

$$\psi \leftarrow \psi - \nabla J_\psi(V) \quad (\text{A.8})$$

Parameters of target critics are then updated via Polyak averaging (Haarnoja et al., 2018):

$$\bar{\phi} \leftarrow (1 - \tau)\bar{\phi} + \tau\phi \quad (\text{A.9})$$

$$\bar{\psi} \leftarrow (1 - \tau)\bar{\psi} + \tau\psi \quad (\text{A.10})$$

## A.4 Pseudo Code

---

**Algorithm 1** LUSTER Training

---

```

1: Initialise parameters  $\theta, \phi, \psi, \bar{\phi}, \bar{\psi}$ 
2: ## Actor Pretraining using EmoWOZ
3: for each actor step do
4:    $\theta \leftarrow \theta - \nabla \mathcal{L}_{\text{SL}}$                                      ▷ Equation 12
5: end for
6: ## Critic Pretraining using EmoWOZ, not applicable when using emotion advantage only.
7: for each critic step do
8:   ## Update utterance-level Q and V functions by target function bootstrapping.
9:    $\phi \leftarrow \phi - \nabla \mathcal{L}_Q$                                          ▷ Equation 22
10:   $\psi \leftarrow \psi - \nabla \mathcal{L}_V$                                          ▷ Equation 23
11:  ## Update target Q and V functions via Polyak averaging.
12:   $\bar{\phi} \leftarrow (1 - \tau)\bar{\phi} + \tau\phi$                                 ▷ Equation A.9
13:   $\bar{\psi} \leftarrow (1 - \tau)\bar{\psi} + \tau\psi$                                 ▷ Equation A.10
14: end for
15: ## Initialise and fill experience replay buffer  $\mathcal{D}$ 
16: for each dialogue while buffer_size not reached do
17:   for each turn do
18:     Execute  $a_t^{0:L-1} \sim \pi_\theta(\cdot | s_t^0)$ , obtain the next state  $s_{t+1}^0$  and reward  $\{r_t^L, A_t, \text{or } \mathcal{R}_{\text{combine}, t}\}$ , add to buffer  $\mathcal{D}$ .
19:   end for
20: end for
21: for each iteration using buffer  $\mathcal{D}$  do
22:   Repeat line 17-21 until  $\mathcal{D}$  is full.
23:   Repeat line 7-14 for critic update.
24:   for each off-policy experience do
25:     Replay experience and scale reward with importance sampling weight.          ▷ Equation 27
26:   end for
27:   ## Update token-level actor with utterance-level critic.
28:   for each actor step do
29:      $\theta \leftarrow \theta - \nabla \mathcal{L}_{\text{PPO}}$                                          ▷ Equation 21, approximated with A.5
30:   end for
31:   Discard oldest experiences based on specified Training Interval to make room for new experiences
32: end for

```

---

## A.5 Training Configurations

The predefined set for token selection in the following equations are:

In Equation 2,  $E = \{\text{neutral, fearful, dissatisfied, apologetic, abusive, excited, satisfied}\}$ , defined in the EmoWOZ (Feng et al., 2022).

In Equation 3,  $D = \{\text{general, restaurant, hotel, train, attraction, taxi, police, hospital}\}$ , which includes the seven EmoWOZ domains plus an additional “general” domain to handle small-talk and greetings.

In Equation 10,  $C = \{\text{neutral, compassionate, apologetic, enthusiastic, appreciative}\}$  as defined in EmoWOZ 2.0 (Feng et al., 2024a).

Parameter	Value
Learning Rate	3e-5
Training Epochs	10
Batch Size	32
LoRA Rank	32
LoRA Alpha	32
LoRA Dropout	0.1
LoRA Bias	None
LoRA Target Modules	o_proj, qkv_proj

Table A.1: Supervised Training Hyperparameters

Parameter	Value
Critic Learning Rate	5e-5
Critic Batch Size	32
Critic Epochs	5
Critic Gradient Norm Clipping	40
Actor Learning Rate	1e-7
Actor Batch Size	16
Actor Sample Size (Equation 18)	6
Actor Sample Temperature (dialogue action $da$ )	0.5
Actor Sample Temperature (response $u_{sys}$ )	0.9
Actor History Length	5
Training Interval	10 Dialogues
Experience Buffer Size	50 Dialogues
Reservoir Sampling	True
User Utterance and Emotion Sample Size (Equation 18)	6
Dialogue-level Discount Factor $\gamma_{dial}$ (Equation 22)	0.99
Reward Combination Weight $\rho$ (Equation 25)	0.1
Importance Sampling Weight Clipping Range $\eta$ (Equation 27)	0.9
Turn-level Discount Factor $\gamma_{turn}$ (Equation A.6)	1.0
Decay Rate $\lambda$	0.99
Dialogue-level Reward (Equation (22))	$r_t = -1$ for $0 < t < T$ , $r_T = +80$ if success else -40
Polyak Averaging Weight $\tau$ (Equation A.9, A.10)	0.01

Table A.2: Reinforcement Learning Configurations

## A.6 System Comparison

### A.6.1 Prompt Template

*There are two systems system\_A and system\_B. The user talks to each of them to complete the same information retrieval goal. You are now act as a judge that determines which system has better overall performance.*

*The dialogue between the user and system\_A in json list form is: {system\_a\_dialogue}*

*The dialogue between the user and system\_B in json list form is: {system\_b\_dialogue}*

*Now, output your judgement together with explanation in the following format:*

```
{
  "judgement": "which system is better, system_A or system_B? If you really cannot decide, you can output 'tie'.",
  "explanation": "Why one system is better. Your analysis should be based on the success of the dialogue, the informativeness of the system responses, and the system emotional performance to enhance user experience. You can also consider the length of the dialogue, the diversity of language, and whether the system provided relevant information to the user."
}
```

### A.6.2 Case Study

The simulated user goal is described in the JSON format below where the user's goal is to retrieve **requestable** information based on **informable** constraints:

```
{
  "restaurant": {
    "informable": {
      "area": "centre",
      "pricerange": "moderate"
    },
    "requestable": {"postcode": "?"}
```

```

        },
        "train": {
            "infomable": {
                "arriveBy": "20:45",
                "day": "thursday",
                "departure": "broxbourne",
                "destination": "cambridge"
            },
            "requestable": {"trainID": "?"}
        }
    }
}

```

The conversation between the simulated user and LUSTER-base-emotion system is:

**User** : I'm looking for a restaurant in the centre of town.  
**LUSTER-base-emotion** : there are 69 restaurants in that area. what type of food would you like?  
**User** : I would like a moderately priced one please.  
**LUSTER-base-emotion** : there is 21 of those. what type of cuisine would you like?  
**User** : Could I have the postcode of one of them?  
**LUSTER-base-emotion** : **sure, the postcode is cb21db.**  
**User** : Thank you. I also need a train. I need to leave for cambridge on Thursday and arrive by 20:45.  
**LUSTER-base-emotion** : where will you be departing from?  
**User** : I will be departing from broxbourne going to cambridge.  
**LUSTER-base-emotion** : tr0393 leaves at 05:32 and arrives at 06:32. **would you like to book that for 1 person?**  
**User** : thank you. bye  
**LUSTER-base-emotion** : **thank you for using our service.**

Under the same goal, the conversation between the simulated user and LUSTER system is:

**User** : I am looking for information on a place to eat that is in the centre.  
**LUSTER** : there are 69 restaurants in that area. what type of food would you like?  
**User** : Any kind of food in the moderate price range would be fine.  
**LUSTER** : there is 21 of those. what type of cuisine would you like?  
**User** : I don't have a preference on cuisine, I just need the postcode.  
**LUSTER** : **pizza express has the postcode cb21db.**  
**User** : Great! Thank you. I also need a train on Thursday that is arriving by 20:45.  
**LUSTER** : where will you be departing from?  
**User** : I will be departing from broxbourne going to cambridge.  
**LUSTER** : tr0393 leaves at 05:32 and arrives at 06:32. **would you like me to book that for you? how many tickets do you need?**  
**User** : thank you. bye  
**LUSTER** : **thank you for using our service. goodbye.**

Both systems fulfill the user goal by correctly providing requestable information: postcode=cb21db and trainID=tr0393. Convlab-3 evaluator marks both conversation as successful, and the user simulator, langEmoUS, gives the same sentiment rating in each turn. Despite this, GPT-4o-mini prefers LUSTER over LUSTER-base based on more nuanced system behaviours as highlighted in blue colour above with the following explanation:

LUSTER outperforms LUSTER-base in several key areas.

Firstly, **LUSTER provides a specific restaurant name** (Pizza Express) along with the postcode, which adds a layer of useful information for the user that LUSTER-base lacks, as it only provides the postcode without a specific restaurant name associated with it.

Secondly, LUSTER uses slightly more user-friendly language, such as '**how many tickets do you need?**' which invites the user into the conversation and indicates a more engaging approach.

In terms of emotional performance, LUSTER ends the conversation with a more polite '**goodbye**' which may enhance user experience by making it feel more personal. Overall, LUSTER demonstrates better informativeness, engagement, and emotional connection with the user, leading to a more successful interaction.