

Metaphorical User Simulators for Evaluating Task-oriented Dialogue Systems

Weiwei Sun¹ Shuyu Guo¹ Shuo Zhang² Pengjie Ren¹

Zhumin Chen¹ Maarten de Rijke^{3,4} Zhaochun Ren^{1†}

¹Shandong University, Qingdao, China ²Bloomberg, London, United Kingdom

³University of Amsterdam ⁴Ahold Delhaize Research, Amsterdam, The Netherlands

sunweiwei@gmail.com, guoshuyu225@gmail.com, szhang611@bloomberg.com

{renpengjie, chenzhumin}@sdu.edu.cn, m.derijke@uva.nl, zhaochun.ren@sdu.edu.cn

ABSTRACT

Task-oriented dialogue systems (TDSs) are assessed mainly in an offline setting or through human evaluation. The evaluation is often limited to single-turn or very time-intensive. As an alternative, user simulators that mimic user behavior allow us to consider a broad set of user goals to generate human-like conversations for simulated evaluation. Employing existing user simulators to evaluate TDSs is challenging as user simulators are primarily designed to *optimize* dialogue policies for TDSs and have limited evaluation capability. Moreover, the evaluation of user simulators is an open challenge. In this work, we propose a metaphorical user simulator for end-to-end TDS evaluation. We also propose a tester-based evaluation framework to generate variants, i.e., dialogue systems with different capabilities. Our user simulator constructs a metaphorical user model that assists the simulator in reasoning by referring to prior knowledge when encountering new items. We estimate the quality of simulators by checking the simulated interactions between simulators and variants. Our experiments are conducted using three TDS datasets. The metaphorical user simulator demonstrates better consistency with manual evaluation than Agenda-based simulator and Seq2seq model on three datasets; our tester framework demonstrates efficiency, and our approach demonstrates better generalization and scalability.

CCS CONCEPTS

• **Information systems** → **Users and interactive retrieval**; • **Human-centered computing** → **Human computer interaction (HCI)**.

KEYWORDS

User simulation, Task-oriented dialogue, Conversational recommendation, Conversational information access

ACM Reference Format:

Weiwei Sun, Shuyu Guo, Shuo Zhang, Pengjie Ren, Zhumin Chen, Maarten de Rijke, and Zhaochun Ren. 2022. Metaphorical User Simulators for Evaluating Task-oriented Dialogue Systems. In *Proceedings of the 45th International*

[†]Corresponding author.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SIGIR '22, July 11–15, 2022, Madrid, Spain

© 2022 Copyright held by the owner/author(s).

ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain. ACM, New York, NY, USA, 12 pages.

1 INTRODUCTION

Task-oriented dialogue systems (TDSs) assist users in solving a specific task during a conversation [51]. They are being considered in an increasing number of information retrieval scenarios, such as conversational information seeking, conversational Q&A, and conversational recommendation [13, 41, 42]. In a TDS, dialogues follow a clearly defined structure that build on domain knowledge relevant for the task(s) at hand. The evaluation of TDSs is a crucial part of the development process. Recent studies on evaluating TDSs are either through offline evaluation or human evaluation [22, 25]. Offline evaluation evaluates a dialogue system based on test sets, whereas human evaluation reflects the overall performance of the agent through in-field experiments [3, 25] or crowd-sourcing [22]. However, offline evaluation is often limited to single turn assessments, while human evaluation is intrusive, time-intensive, and does not scale [9].

User simulations. User simulation can potentially mitigate the above issues via simulated interactions, and thus a viable choice for large-scale automatic evaluation for TDS [9]. The structured nature of task-oriented dialogues allows us to build user simulators as we can exhaustively enumerate user goals to generate human-like conversations for simulated evaluation [54]. Recent studies have employed user simulation for evaluating conversational information access tasks, including complex information seeking [24], conversational item recommendations [52], and multi-step task completion [47]. User simulators have also been successfully applied as a reinforcement learning environment for dialogue policy optimization [45]. However, user simulators as evaluation methods for TDSs are still under-explored [2].

We identify the following challenges when employing user simulators to evaluate TDSs: (1) First, employing existing simulators for evaluating TDSs suffers from limited realism and evaluation capability [2]. E.g., commonly used agenda-based simulators [44] rely on human-curated rules that are domain-specific; thus, their responses are far from human-like [52]. End-to-end simulators [35, 48] have been proposed to improve the domain generalizability of simulators. Still, their evaluation capabilities are limited at the semantic level (e.g., if the system provides a novel entity) [48]. (2) Besides, there is a lack of automatic methods to assess user simulators' realism and

evaluation capabilities. Most existing methods rely on manual evaluation, which is costly and hard to reproduce. Thus the evaluation of user simulators is still an open challenge [38].

A metaphorical user simulator. In the face of the challenges identified above, we aim to provide a solution for improving the realism and transferability of user simulators and thus serve as an evaluation method for TDSs. Our solution includes a simulator based on metaphorical user modeling for end-to-end TDS evaluation and an automated evaluation method based on testers.

Specifically, we propose a metaphorical user simulator (MetaSim) that improves the realism of the conversation strategies by simulating the user’s analogical thinking. The metaphorical user model is designed to assist the simulator in the reasoning process by referring to historical dialogue strategies when encountering new items. MetaSim comprises: (1) a preference module that initiates diverse user preferences and updates them during conversations; (2) a natural language understanding (NLU) module that tracks the dialogue state; (3) a metaphor module that retrieves similar dialogue strategies related to the current dialogue state; (4) a policy module that simulates user satisfaction and predicts the user action according to context and retrieved strategies; and (5) a natural language generation (NLG) module that generates a natural response.

To evaluate user simulators, we construct a tester-based comparative evaluation framework inspired by the paradigm of evaluating interactive IR system [24] and conversational recommenders [52]. A tester defines how to make variants for a dialogue system with distinguishable performance and how to evaluate user simulators by interacting with them. We leverage a base dialogue system based on SOLOIST [37]. Its variants, defined by testers, are constructed by configuring parameters of the context history, retrieval or recommender methods, and dialogue domains. We evaluate the simulators by checking if they can rank the system variants in the same way as humans and respond to new items in a human-like way. We rely on manual experiments to evaluate the reliability of the tester framework. A well-built tester can repeatedly evaluate the evaluation capabilities of simulators without the need for manual testing.

Experiments. We conduct experiments on three benchmark datasets, MultiWOZ [12], ReDial [29], and JDDC [7]. We implement MetaSim based on T5 [40], a pre-trained transformer model, and adopt a unified data format that allows it to be generalized to multiple tasks. We optimize each module on the dataset and connect them during testing. We evaluate MetaSim by checking the naturalness of the generated responses by user simulators, the quality of interactions, and consistency between user simulators and humans when comparing the variants of TDSs. Our experiments show that MetaSim consistently outperforms previous comparable simulators in terms of evaluation consistency and human-likeness. Our experiments verify that (1) metaphorical user simulators (MetaSim) can generate human-like dialogues and achieve evaluation results that are consistent with human expectations; (2) a tester-based framework is a valid tool for automatic assessment of simulator evaluation capabilities; and (3) our method generalizes well across domains.

Contributions. The contributions of this paper are:

- We introduce a simulation approach for evaluating task-oriented dialogue systems. In particular, we propose a metaphorical user

simulator (MetaSim) that leverages the dialogue records to improve the dialogue reasoning ability.

- We introduce a tester-based framework for evaluating user simulators. We verify its reliability by manual evaluation; and
- We validate MetaSim through automatic and human experiments on three datasets and find that MetaSim improves the evaluation accuracy and the human-likeness of the user simulation.

2 RELATED WORK

2.1 Task-oriented dialogue systems

Task-oriented dialogue systems are developed either via module-based or end-to-end approaches [19]. Research on the former boils down to advancing the components of TDS including Natural Language Understanding (NLU), Dialogue State Tracking (DST), Dialogue Policy Learning (DPL), and Response Generation (RG) [16, 32, 33, 43]. The latter generates responses relying upon end-to-end neural dialogue generation techniques [20, 27, 49]. Recently, pre-trained language models have been employed in end-to-end dialogue modeling with superior performance [17, 37]. TDSs have also been considered in an increasing number of interactive information retrieval scenarios, such as conversational recommendation [13, 29], conversational information seeking [42], and conversational question answering [41].

2.2 Evaluation of dialogue systems

The typical evaluation methods of dialogue systems include automatic evaluation and human evaluation [9]. Commonly used automatic evaluation methods include component-level evaluation and task-level evaluation. Component-level evaluation concerns the assessment of each component, e.g., joint goal accuracy for DST [12], the action prediction accuracy for DPL; Recall, NDCG metrics for item recommendation [26]; BLEU [36], F1, perplexity, and Distinct for text generation evaluation [6]. Task-level evaluation evaluates the overall performance of the dialogue system; commonly used metrics include task success rate, inform success rate, and average turn [12].

The above methods based on a static test set are limited to a single turn and do not inform us about the overall usefulness of the system in interaction [52]. Human evaluation can address this drawback [22], and popular human evaluation metrics include Satisfaction, Fluency, Coherence, Task success, Engagingness, etc. However, human evaluation is intrusive, time-intensive, and does not scale [46]. Employing simulation-based evaluation can tackle the above issues and be a viable choice for large-scale automatic evaluation [9].

2.3 User simulation

User simulators are designed to simulate the user’s behavior, which can be used either as an environment to train a reinforcement learning-based system or to evaluate a functioning system to find weaknesses (or assess) dialogue quality and potentially replace human evaluation [9]. For the sake of evaluating spoken dialogue systems, Eckert et al. [10] propose the first statistical user simulator. Using a Markov model [14], Cuayáhuitl et al. [8] propose a hidden Markov model for the same purpose. In later work, the agenda-based user simulator [44] has been widely accepted as it elegantly represents the user state as a stack of necessary user actions.

User simulation is not foreign to information retrieval evaluation; its importance has been confirmed in the Sim4IR workshop at SIGIR 2021 [2]. Plenty of methodologies have been proposed for building simulators, e.g., employing Bayesian procedure [5], cognitive state for interactive information retrieval [31], and for different tasks like search sessions [53], online news recommendation [4], conversational recommender systems via an agenda-based user simulator [52], biases present in the logged data [18]. In addition, Shi et al. [45] investigate the design of user simulators as an reinforcement learning environment. Tseng et al. [48] introduce a reinforcement learning approach based on end-to-end modeling. Labhishetty and Zhai [24] introduce a tester-based evaluation approach to evaluating the reliability of user simulators.

Existing simulators mechanically inform the system of the slot, which limits the realism and the evaluation capability [47]. Thus, the main challenge of employing simulation is to build a realistic user simulator that can mimic natural user behavior to a realistic extent, i.e., “human-likeness.” Among the efforts towards human-likeness for either simulators [47, 52] or dialogue systems [11], building a human-like dialogue system calls for detailed consideration of human behavior in dialogues as humans have complicated mental activities and specific behaviors when interacting with the machine. Studies show that users build “mental models” in interacting with a machine, and that such mental models are formed through analogical thinking [21]. We aim to prototype an initial mental model in this work, namely the metaphorical user simulator.

2.4 Evaluation of user simulators

Evaluation of user simulators is an open challenge [38]. Existing evaluation methods of user simulators include: (1) Simulators are treated as dialogue systems and then evaluated similarly through text generation metrics (e.g., BLEU, PPL), component-level metrics (e.g., action prediction accuracy), and task metrics (e.g., task success rate) [34]. (2) Alternatively, the assessments are performed based on the realism of simulated dialogues [52]. There, descriptive statistics (e.g., the number of turns, words, slots, actions) of machine-machine dialogues data are compared data from with human-human dialogues, or use humans are asked to perform a Turing test. (3) In addition, human evaluation is a critical component that assesses the consistency between humans and simulators. Typical metrics include scoring-based methods (e.g., Cohen’s Kappa, KL divergence, Spearman’s Rho) and comparison-based methods (e.g., the rank of systems) [46]. (4) Lastly, some work evaluate simulators by checking the performance of dialogue systems trained with the simulators by reinforcement learning [45, 48]. To directly assess the evaluation capabilities of the simulator, recently, a tester-based evaluation framework has been used in conversation search [24], which has shown good consistency with human expectations while costing less and being more stable than human evaluation. Focusing on assessing user simulators for TDS, we introduce a tester-based framework, where we can design reliable system variants from different perspectives.

3 PROBLEM FORMULATION

Most evaluation methods for TDSs follow the PARADISE [50] framework and estimate user satisfaction based on dialogue cost and task

success via automatic, simulated, or human evaluation. Simulation-based evaluation methods efficiently evaluates the system with interactions, but its current capacity is too limited to compensate for human evaluation [2]. Building a human-like user simulator does not mean that we are after a perfect mirror of human behavior, let alone the replacement of humans. The simulator should be good enough to be an assessor that correlates well with human assessment in certain aspects and potentially reduces the reliance on human effort in the loop of evaluation.

Our goal is to build an evaluation framework for comparing task-oriented dialogue systems via user simulation. This framework, first, should be capable of configuring *testers* for a given base dialogue system, i.e., a *base model*.

DEFINITION 1 (Base Model): *The base model is a standard dialogue system that is configurable, thus leading to potentially different capabilities.*

DEFINITION 2 (Tester): *A tester, \mathbb{T} , is an instance of a configurable base system, and it defines the objective to configure a base system, e.g., via the choices of a TDS model.*

DEFINITION 3 (Variant): *A system variant, S_i , is an instance of \mathbb{T} that follows the directed control of a specific capability of the tester.*

Given a tester system \mathbb{T} , any pair of its variants $S_1^{\mathbb{T}}$ and $S_2^{\mathbb{T}}$ should have a sensible difference in components, training data, and/or methods, and thus display distinguishable system performance.

Second, the user simulators can be configured to evaluate dialogue systems on certain aspects by interacting with the systems. Suppose S_1 has a better ability in recommending new items than S_2 , and the simulator is expected to measure this difference. The measures by the simulator should be consistent and stable with human assessments, and can be reproduced under this framework.

Lastly, this framework enables us to explore more human-like aspects to see how realistic simulators can be. In this study, we investigate the possibility of building a simulator that responds to new knowledge in a human-like fashion by exploring using a human metaphor. Here, *metaphor* is the process where the user draws on relevant conversational strategies and knowledge in the interaction [23] and is found to be the primary element of the construction of *mental model* [21]. The simulation framework will be detailed in Sect. 4.

4 FRAMEWORK

We introduce a framework aimed at improving the realism and transferability of user simulators for end-to-end TDS evaluation. This framework includes a metaphorical user simulator (MetaSim) for end-to-end TDS evaluation and an automated evaluation method based on testers. In this section, we first introduce the MetaSim (Sect. 4.1) that improves the evaluation capability of the simulator by simulating user’s analogical thinking. Then, we introduce the tester-based evaluation framework for evaluating the evaluation capability of the simulators (Sect. 4.2).

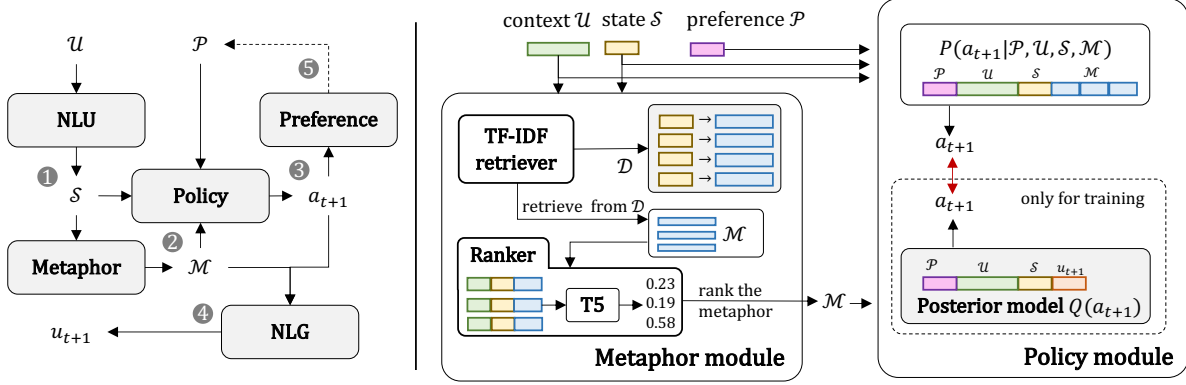


Figure 1: The architecture of metaphorical user simulator (MetaSim). The left side shows the five modules of MetaSim (i.e., Preference, NLU, Metaphor, Policy, and NLG) and how they are connected. The right side details the Metaphor module and Policy module, where the Metaphor module retrieves M from \mathcal{D} , and the Policy module decides the next user action.

4.1 Metaphorical user simulators

We propose a metaphorical user simulator, **MetaSim**, that constructs a metaphorical user model to assist dialogue simulation by referring to prior knowledge. The metaphorical user model simulates the user’s analogical thinking in interacting with machines to improve the realism of conversation strategies and improve the model’s ability to generalize to new domains and items. As shown in Fig. 1, the simulator consists of five modules: Preference module, NLU, Metaphor module, Policy module, and NLG.

We introduce the modules one-by-one using the following notation: $\{\mathcal{P}, \mathcal{U}, \mathcal{S}, \mathcal{M}, a_{t+1}, u_{t+1}\}$, where \mathcal{P} denotes user preferences, $\mathcal{U} = \{u_1, \dots, u_t\}$ is the dialogue context of t turns, and u_i denotes an utterance of the system or user at turn i , $\mathcal{S} = \{a_1, \dots, a_t\}$ is the dialogue state while each element a_i is the action of the utterance at turn i , $\mathcal{M} = \{m_1, \dots, m_k\}$ is the metaphor strategies and each m_i in \mathcal{M} is a utterance, a_{t+1} is the next user action concatenated with user satisfaction, and u_{t+1} is the next user response. Besides, the metaphorical user simulator relies on a database \mathcal{D} that contains dialogue records from which the model retrieves a metaphor M , and we detail it in Sect. 4.1.3.

4.1.1 Preference module. The user preferences \mathcal{P} define the user requirements or the goal of the conversation, and the preference module tells how \mathcal{P} is initialized and updated during the conversations. We represent the user preferences \mathcal{P} as

$$\{(\text{slot}_1, \text{value}_1), \dots, (\text{slot}_n, \text{value}_n)\},$$

where slot_i denotes an attribute of user requirements and value_i is its value. For example, the preference for hotel reservation “*want to book a cheap hotel where parking is available*” is represented as: $\{(\text{hotel_parking}, \text{“yes”}), (\text{hotel_price}, \text{“cheap”})\}$. \mathcal{P} is initialized by sampling the slots based on the distributions in a dialogue dataset, and their values are randomly selected. During the conversation, the preference module updates the preference \mathcal{P} according to the following rules: (1) If the user “informs” the system about a particular slot, delete it from \mathcal{P} ; (2) If the system “recommends” an item, add it to \mathcal{P} .

4.1.2 NLU. The NLU module keeps track of the states (i.e., the slots have been informed) of the dialogue. We use *actions* to represent the intent of an utterance in dialogue. We define the format of an

action as follows:

$$\{(\text{act}_1, \text{slot}_1, \text{value}_1), \dots, (\text{act}_n, \text{slot}_n, \text{value}_n)\},$$

where each item $(\text{act}_i, \text{slot}_i, \text{value}_i)$ includes: (1) act_i , the name of action (e.g., *inform*, *bye*); (2) slot_i , a slot, and (3) value_i , its value. Note that some actions that do not have slots or values, such as *bye*; in such cases the slots and values are set to NULL. Given user preferences \mathcal{P} and dialogue context \mathcal{U} , the NLU module predicts the action a_t of the last system utterance u_t . Then, we append a_t to the historical actions, which forms the dialogue state \mathcal{S} . Thus the dialogue state \mathcal{S} records all the slots that have been informed from the system. We optimize the NLU module using supervised learning and minimize the log-likelihood $\mathcal{L}_{NLU} = -\log(P(a_t | \mathcal{P}, \mathcal{U}))$.

4.1.3 Metaphor module. The metaphor module is responsible for retrieving similar dialogue records from a database (terms as \mathcal{D}). The retrieved dialogue records are used to support the policy module. To construct database \mathcal{D} , we index the dialogue state for each utterance in the training data and take the user utterance as the value. Formally, $\mathcal{D} = \{(d_1^k, d_1^v), \dots, (d_n^k, d_n^v)\}$, where n denotes that \mathcal{D} has n records, d_i^k is the index of the i -th record, and d_i^v is the value of the i -th record.

For any new dialogue, we retrieve k records $\mathcal{M} = \{m_1, \dots, m_k\}$ (we call \mathcal{M} a *metaphor*) from \mathcal{D} , where each m_i is a record retrieved from \mathcal{D} .

As illustrated in Fig. 1, the retrieval follows a two-stage pipeline:

- (1) **Candidate generation:** For any record (d_i^k, d_i^v) , we retrieve k candidate records from \mathcal{D} using $\text{TF-IDF}(\mathcal{S}, d_i^k)$, which is the TF-IDF score between the index of the record d_i^k and the current dialogue state \mathcal{S} . These candidates form \mathcal{M} .
- (2) **Ranking:** We rank the candidates by a learnable *ranker* for improving the retrieval accuracy. The *ranker* is defined as $P(r_i | m_i, \mathcal{U}, \mathcal{S})$. Specifically, for an utterance m_i in \mathcal{M} , we input the dialogue context \mathcal{U} and dialogue state \mathcal{S} to the *ranker*, and estimate the relevance score r_i of them. In training, we set the relevance score of a ground-truth utterance to 1.0 and the remaining utterances to 0.0, and optimize the *ranker* by maximum the likelihood of

the relevance score:

$$\mathcal{L}_{Metaphor} = - \sum_{i=1}^k \log(P(r_i | m_i, \mathcal{U}, \mathcal{S})).$$

In testing, we estimate the relevance scores for each utterance in \mathcal{M} and rank them based on the scores.

4.1.4 Policy module. The policy module is responsible for predicting user satisfaction and the next user action. We define it as $P(a_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{S}, \mathcal{M})$. The inputs of policy module are the user preferences \mathcal{P} , dialogue context \mathcal{U} , dialogue state \mathcal{S} , and metaphor \mathcal{M} ; the output is the user action a_{t+1} . To predict the user satisfaction and action jointly, we make the user satisfaction a particular slot in the original action following [47]. The annotation of user action and satisfaction is usually incomplete or noisy [15, 47], thus we introduce a *posterior network* $Q(a_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{S}, u_{t+1})$ to facilitate the training of the policy module (as shown in Fig. 1). The *posterior network* utilizes the ground-truth user utterance; thus, its predictions are more accurate than the policy module. We first optimize the *posterior network* on partially annotated data by a negative log-likelihood (NLL) objective:

$$\mathcal{L}_{Policy}^Q = -\log(Q(a_{t+1})),$$

and then optimize the policy module by learning from the *posterior network*. The objective is defined as the Kullback–Leibler divergence between the $Q(a_{t+1})$ and $P(a_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{S}, \mathcal{M})$:

$$\mathcal{L}_{Policy}^P = -Q(a_{t+1}) \log \left(\frac{Q(a_{t+1})}{P(a_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{S}, \mathcal{M})} \right).$$

4.1.5 NLG. The NLG is responsible for generating natural language responses that incorporate system requests and individual preferences. We define it as: $P(u_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{M}, a_{t+1})$, where the NLG module inputs the user preferences \mathcal{P} , dialogue context \mathcal{U} , metaphor \mathcal{M} and next action a_{t+1} , generate the next utterance u_{t+1} in an autoregressive fashion. We optimize it by supervised learning; the training objective of the NLG module is defined as:

$$\mathcal{L}_{NLG} = -\log(P(u_{t+1} | \mathcal{P}, \mathcal{U}, \mathcal{M}, a_{t+1})),$$

where u_{t+1} is the next user utterance, \mathcal{P} is user preference, a_{t+1} is the next user action, and \mathcal{U} is the dialogue context.

4.1.6 Inference. The operation of our model is shown in Fig. 1. We first draw a preference \mathcal{P} from the preference module, that represents the initial user needs that lead the dialogue. Then the simulator interacts with the system according to \mathcal{P} . For each turn, with a dialogue history \mathcal{U} , the model first predicts the system action a_t by the NLU module and composes the dialogue state \mathcal{S} . Then the metaphor module retrieves \mathcal{M} and ranks the sentences using the ranker. After that, the policy module predicts the user action and satisfaction based on the metaphor and the dialogue context. Then the NLG module converts the action to a natural language utterance u_{t+1} and responds to the system. Finally, the preference module updates \mathcal{P} by adding new items and removing informed slots. The simulators generate “[END]” as an indicator to end the dialogue.

At the end of the dialogue, the simulator rates the system based on the simulated dialogue. Previous studies [47] show that human assessment of TDS are mainly driven by the success of the task

and the quality of system response. Thus we define the calibrated satisfaction by the average of **SUCCESS** (that equals 1 if the system solves the user’s problem and equals 0 otherwise) and **SATISFACTION** (that averages the turn-level user satisfaction during the dialogue).

4.1.7 Implementation. We use T5 [40] to instantiate each module. T5 is a sequence-to-sequence model with an encoder-decoder structure. We process the inputs and outputs of each module as a sequence. See Fig. 2 for some examples of our sequence processing.

\mathcal{P}	{domain=hotel; parking=yes; price=cheap;}
\mathcal{U}	{user: I am looking for a place. . . . system: Okay, do you have. . . }
\mathcal{S}	{user: inform type=hotel, price=cheap. system: request area. }
a_{t+1}	{user: <neutral>; inform parking=yes. }
u_{t+1}	{user: No, I just need to have parking. }
a	
\mathcal{P}	{Interstellar=like; Inception=like; Kill Bill=unlike;}
\mathcal{U}	{user: I really like interstellar, do. . . . system: Of course, have you. . . }
\mathcal{S}	{user: inform Interstellar=like. system: recommend Tenet. }
a_{t+1}	{user: <positive>; inform Tenet=like. }
u_{t+1}	{user: Yes, that’ s my favorite. I love Nolan’ s movies. . . }
b	
\mathcal{P}	{品类=手机; 交付=物流; 营销服务=包邮;}
\mathcal{U}	{user: 手机什么时候到货. . . . system: 亲, 您的订单XXXX正在. . . }
\mathcal{S}	{user: 配送周期流 品类=手机. system: 订单 营销服务=定制. }
a_{t+1}	{user: <negative>; 联系配送. }
u_{t+1}	{user: 这都一周了怎么 还没到货? ? 怎么. . . }
c	

Figure 2: Examples of processed sequences. We highlight attributes, values, action, satisfaction, utterance, and speaker. (a) is in hotel domain from MultiWOZ, (b) is in movie domain from ReDial, and (c) is in delivery domain from JDDC.

For modules with multiple inputs, we concatenate them in order with a special token <SEP> and clip to the maximum input length of the model. Thus for each module, assuming the input text is \mathcal{I} and the output text is $\mathcal{O} = \{o_1, \dots, o_m\}$ with m tokens, the probability is defined as:

$$P(\mathcal{O} | \mathcal{I}) = \sum_{i=1}^m P(o_i | \mathcal{O}_{<i}, \mathcal{I}).$$

4.2 Tester-based evaluation framework

Existing simulated evaluation methods for a TDS typically rely on third-party systems [52]. This dependency suffers from the following issues: the limited choices of third-party systems and the difficulty of comparing them due to differences in training data and model architecture. Drawing inspiration from [24], we introduce a tester-based evaluation framework for evaluating both dialogue systems and user simulators, see Fig. 3. The tester-based framework

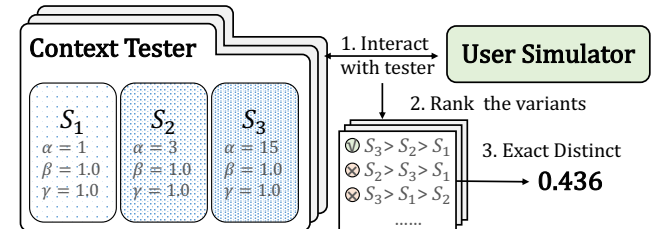


Figure 3: In tester-based evaluation, the simulator interacts with system variants in the tester, ranks them, and calculates the Exact Distinct score (i.e., to accurately distinguish between the three systems).

is constructed to evaluate the simulator automatically—the simulator talks to and evaluates the variants within a tester so that we can see whether a simulator can fairly compare the variants in a tester or not. The tester-based framework has the following benefits in evaluating simulators: (1) the constructed system variants are well-calibrated and thus comparable to each other, and there are unambiguous and stable performance differences among the variants in a trusted tester; (2) a tester can evaluate any specific capability of a system, leading to a more comprehensive evaluation; and (3) a well-constructed tester does not require repeated human evaluation and can be reused to evaluate multiple simulators.

4.2.1 Evaluating user simulators by testers. Simulators are built for evaluating systems. Simulators also need to be assessed appropriately. Here, we discuss the latter case. The main idea to evaluate a simulator is to check whether the evaluation results by simulator upon variants are consistent with human expectations. For example, given a simulator US , and three variants S_1 , S_2 and S_3 , US interacts with S_1 , S_2 and S_3 separately and scores them based on the interactions. The performance scores are on certain aspects like recommendation accuracy and domain understanding. A comprehensive assessment requires various testers and variants (see Sect. 4.2.2). We compare these scores against human expectations. A better consistency score indicates a better simulator.

We calculate the EXACTDISTINCT (ED for short) score to measure the consistency between simulators’ assessment scores and human expectations. EXACTDISTINCT is defined as the proportion of examples that ranked the same as expectations by the human evaluator (i.e., $S_1 < S_2 < S_3$). An ED score of 0.5 means a rank accuracy of 50%.

4.2.2 Tester bootstrapping. We introduce the base system and three strategies to configure the testers. We leverage a base model based on SOLOIST [37], which employs a sequence-to-sequence model for dialog state tracking and response generation. The testers enable us to configure the base model and make variants based on the needs. Specifically, we can bootstrap testers by changing the context in the dialogue history, exploring different choices for components like the recommender module, leveraging domain-specific training data, and supplying various amounts of supervision for training, to name a few. Here, we introduce three strategies that we explore in this work; other options are left for future investigation:

- **Context tester.** Context refers to the dialogue history. The amount of dialogue history that is used for training has an impact on the model performance [13]. The Context tester, T_C , makes variants by giving different context memory capabilities. We define a hyperparameter α indicating the number of sentences in the dialogue history. For example, $S_{\alpha=2}$ represents a model that keeps two sentences of dialogue history.
- **Recommender tester.** The Recommender tester, T_R , consists of systems with different retrieval capabilities. This tester demonstrates how to make variants by changing a system component. The recommender retrieves from an item database using a query that contains a set of keywords. We then control the capacity of the recommender system by removing the keywords in the query. We define a hyperparameter β in the range 0 to 1. $S_{\beta=0.2}$ represents the model that keeps 0.2 of features in query.
- **Domain tester.** Domain tester, T_D , makes variants by training modules that use different amounts of training data so that we

Table 1: Statistics of the training datasets.

Dataset	MultiWOZ	ReDial	JDDC
Language	English	English	Chinese
Num. dialogues	9,438	10,006	47,389
Num. utterance	147,176	142,502	633,826
Avg num. turns	15.59	14.24	13.37
Num. Action	13	4	19
Num. Slots	15	6,925	23

can test the system’s ability in a particular domain. We control it by removing training data from the training set. We define a hyperparameter γ to indicate the number of dialogues used for training a dialogue system; $S_{\gamma=0.8}$ represents the model that uses 80% of data.

5 EXPERIMENTAL SETUP

5.1 Datasets

We conduct experiments on three datasets:

- **MultiWOZ** [12]: MultiWOZ is a multi-domain task-oriented dialogue dataset spanning seven domains. It contains about 10K dialogues. Each dialogue can span up to three domains.
- **ReDial** [29]: ReDial is an annotated dataset consisting of over 10K conversations, where users recommend movies to each other. Following [30], we identify entities in the dialogues by the Spacy toolkit¹ and link the entities to a movie knowledge graph provided by DBpedia[1]. We then define a user’s preferences as a list of movies the user has seen and whether they like them.
- **JDDC** [7]: JDDC is a large-scale, real-world Chinese e-commerce conversation corpus with over 1 million dialogues. We identify entities in the dialogues by LTP toolkit² and link the entities to an e-commerce knowledge graph provided by Aliyun.³ We define the user’s preference as the entities mentioned in the dialogues and design an action based on the informed entity. We sample the dialogues related to the “delivery” domain from JDDC, resulting in 47K dialogues.

We additionally leverage the USS [47] dataset, which provides user satisfaction annotations (1-5 scale) both on the exchange level and for entire conversations for the three datasets listed above. Summary statistics of our training datasets are in Table 1.

5.2 Implementations of the evaluation framework

Implemented user simulators. We use T5-base [40] as the backbone of MetaSim for MultiWOZ and ReDial, and optimize the model by AdaFactor with a learning rate of 1e-3. For the Chinese dataset JDDC, we use Mengzi-T5-base [55] and use AdamW with a learning rate of 5e-4. We set the batch size to 64, use a linear lr scheduler with 500 warm-up steps, and train up to 20 epochs. We set the max length to 512 for MultiWOZ and 256 for ReDial and JDDC. We perform greedy decoding for simulators. Due to the characteristics of the data, we replaced the NLU module on the ReDial dataset with a rule-based model that identifies and links the entities in

¹<https://spacy.io/>

²<http://www.ltp-cloud.com/>

³<https://www.aliyun.com/>

Table 2: Evaluation results of user simulators. Dist denotes for DISTINCT, Coh. denotes COHERENCE, and Hum. denotes for HUMAN-LIKE. Boldface indicates the best results in the corresponding metric; significant improvements over the best baseline are marked with * (t-test, $p < 0.05$).

Datasets	MultiWOZ								ReDial								JDDC				
	Test set			Interaction			Human	Test set			Interaction			Human	Test set			Interaction	Human		
	F1	Dist	Slot	Success	Dist	Coh.	Hum.	F1	Dist	Slot	Success	Dist	Coh.	Hum.	F1	Dist	Slot	Dist	Hum.		
Agenda [44]	27.25	18.18	27.73	12.19	14.15	1.96	1.84	16.58	24.36	7.44	4.84	22.97	2.45	1.95	13.18	22.42	8.95	29.37	1.81		
AgendaGen [45]	34.51	16.91	24.68	32.17	13.58	2.20	1.98	21.06	18.81	7.15	6.78	23.04	2.42	2.10	17.32	25.36	10.53	31.49	1.97		
SL+Template [45]	35.54	20.20	42.02	59.78	12.74	2.20	1.75	27.83	23.27	15.07	7.30	19.26	2.64	2.30	12.91	25.28	40.55	28.74	1.86		
Seq2seq [40]	42.02	14.98	49.41	76.75	13.76	2.18	2.04	29.75	16.97	11.28	6.18	16.45	2.30	2.35	23.45	21.29	37.26	23.29	2.22		
MetaSim	44.13*	17.29	57.65*	80.25*	15.65	2.31	2.21	33.79*	20.43	15.72*	13.70*	20.97	2.70	2.41	23.85	26.37*	44.00*	34.81*	2.38		
MetaSim-Metaphor	43.31	16.83	55.27	78.96	15.32	2.25	2.13	32.59	18.94	15.21	12.74	19.36	2.66	2.35	23.42	25.41	43.65	32.65	2.31		
MetaSim-Policy	43.11	14.65	49.46	71.54	14.18	2.20	2.17	33.67	15.80	13.01	12.87	17.39	2.35	2.14	22.35	20.46	39.55	30.11	2.14		
MetaSim-Preference	33.84	12.71	9.76	0.93	5.20	1.98	1.72	32.18	14.25	4.35	3.43	5.09	2.05	2.05	18.13	19.76	22.32	0.09	1.77		

the system utterance to the knowledge graph [1], and serialize the linked subgraphs as system actions.

We additionally implement the following baselines on three datasets for comparing against MetaSim:

- **Agenda model** [44] constructs an agenda (that consist of a stack of actions) and pops the top action at each dialogue turn. We build an agenda model by statistically measuring the distribution of action transitions in the training dataset. We retrieve the templates that maximize the override of the slots in action and choose the one with the highest TF-IDF score between the dialogue context.
- **Agenda Gen model** [45] uses the same agenda-based strategy as the Agenda model but generates responses using a generative model instead of retrieved templates. We use a T5-base as the generative model and input the dialogue history and the action drawn from the agenda.
- **SL+Template** [45] predicts employing a retriever in the same way as the Agenda model but using the action predicted by a neural model instead. We use an NLU module and a Policy module to predict the next user action, retrieve a template from the training data, and fill the slot in the template as the response.
- **Seq2seq model** [40] generates response based on dialogue history and preference with a sequence-to-sequence model. We process the data into sequence format and train a T5-base to perform text-to-text generation. Compared to our model, the Seq2seq model doesn't model the policy and metaphor of the user explicitly.

We also compare MetaSim with three modifications:

- **MetaSim-Metaphor** removes the metaphor module and always sets \mathcal{M} to "none".
- **MetaSim-Policy** removes the policy module and always sets the user action $a_{t=1}$ to "none".
- **MetaSim-Preference** removes the \mathcal{P} and let the model response without constrain.

Implemented testers. We use SOLOIST [37] as the base dialogue system, which is a sequence-to-sequence model based on GPT2 [39]. We implement the base system with hyperparameters of $\{\alpha = 15, \beta = 1, \gamma = 1\}$. Then we set the system variants in context tester as $\mathbb{T}_C = \{S_{\alpha=3}, S_{\alpha=1}\}$; set the system variants in recommender tester as $\mathbb{T}_R = \{S_{\beta=0.4}, S_{\beta=0.1}\}$; and set the system variants in domain tester as $\mathbb{T}_D = \{S_{\gamma=0.1}, S_{\gamma=0.01}\}$. We re-implement the SOLOIST model for ReDial and JDDC. For the ReDial dataset, we use movie mentions in context as the dialog states and implement the movie

recommender as a classifier following [29] (i.e., retrieve the movie by calculating the inner-product of the hidden state produced by the GPT model the movie embeddings). For the JDDC dataset, we use the user actions list as the dialog states. Since the Database used in JDDC data is not publicly available [7], we employ end-to-end modeling to generate the response in our implementation and do not implement a recommender tester on JDDC. We optimize the systems using AdamW with a learning rate of $1.5e-4$, batch size 32. The systems are trained up to 20 epochs. We set the max length to 512 for MultiWOZ and to 256 for ReDial and JDDC. Following [37], we generate five delexicalized candidates [34] through nucleus sampling, choose the one that contains more slots in the utterance, and back-fill the slots with the retrieval results from the recommender.

6 EVALUATION AND RESULTS

6.1 Evaluating the realism of simulators

Evaluation methods. We evaluate the realism of the simulators by evaluating the response generated by the simulators and the quality of interactions following three strategies:

- **Test set evaluation** evaluates the simulators on the test set. Using the following metrics, we measure the next user utterance generated based on dialogue context by user simulators: $F1^4$ is a text generation metric that measures the word overlap between the generated responses and ground-truth responses. DISTINCT [28] estimates the proportion of distinct 3-gram in the data, which indicates the lexical diversity. SLOTACC measures the slot accuracy. SLOTACC equals 1 if all the slots (or entities) in the ground-truth response appear in the generated text, otherwise 0.
- **Interaction evaluation** measures the interaction quality of the simulators. The simulators generate 10K dialogues by interacting with the *base system*, and are evaluated by the SUCCESS rate and DIVERSITY of the simulated dialogues. SUCCESS [12] measures if the recommended item meets all user preferences (we discard SUCCESS for JDDC due to the lack of definition). DISTINCT measures the diversity of simulated dialogues.
- **Human evaluation** evaluates the simulators by the human assessment on simulated dialogues. We ask ten workers to rate each of the simulated dialogues in terms of the COHERENCE and HUMAN-LIKE [45]. Specifically, we randomly sample about 700

⁴<https://github.com/facebookresearch/ParlAI/blob/master/parlai/core/metrics.py>

Table 3: Results of tester-based evaluation. \mathbb{T}_C denotes the Context tester, \mathbb{T}_R denotes the Recommender tester, and \mathbb{T}_D denotes the Domain tester. Human evaluation results are only included for MultiWOZ as detailed in Sect. 6.2.

Datasets	MultiWOZ			ReDial			JDDC	
Tester	\mathbb{T}_C	\mathbb{T}_R	\mathbb{T}_D	\mathbb{T}_C	\mathbb{T}_R	\mathbb{T}_D	\mathbb{T}_C	\mathbb{T}_D
Agenda	16.93	17.28	19.47	16.41	17.12	17.05	13.76	14.52
Seq2seq	30.84	37.15	32.81	16.52	17.46	19.16	18.47	16.89
MetaSim	33.23	40.11	35.21	17.48	20.61	20.43	19.17	17.47
Human	43.63	40.54	42.54	-	-	-	-	-

dialogues from the simulated data and have people rate a COHERENCE in range 1 to 3, as well as a HUMAN-LIKE score in range 1 to 3 (higher scores are better.).

Results. The experimental results are provided in Table 2. MetaSim consistently outperforms the baseline methods in terms of SLOTACC, which indicates that MetaSim can inform the system about individual requirements more accurately and has a superior conversational strategy reasoning capability. MetaSim exceeds Seq2seq on SLOTACC with an absolute improvement of +8.24 in MultiWOZ, +4.47 in ReDial and +6.47 in JDDC. The rule-based methods (e.g., Agenda) perform worse than data-driven methods on the test set, especially on JDDC, which indicates that human-curated rules are difficult to cover all the conversation situations. MetaSim also outperforms baselines in terms of F1: the responses generated by MetaSim are more natural than baseline methods. In terms of DISTINCT, MetaSim surpasses all generative methods but it is inferior to the retrieval-based techniques (i.e., Agenda and SL+Template) in MultiWOZ and ReDial. The retrieved responses are more diverse since they are based on templates extracted from human-curated responses from the corpus. Note these responses may not be appropriate for the current conversation.

The experiments on Interaction evaluation indicate that MetaSim achieves a higher task success rate than the baselines, demonstrating better conversational skills (understanding system questions and informing system requirements). MetaSim also exceeds the baselines in terms of diversity except for the agenda model on ReDial data. The human evaluation shows that the simulated dialogue by MetaSim is more coherent to humans and human-like than the baselines. The ablation experiments show that removing modules brings a consistent decrease in performance. Specifically, MetaSim beats MetaSim-Metaphor, indicating a better reasoning ability and diversity via adding the module of Metaphor. Similarly, removing the Policy module leads to a significant performance drop. The MetaSim-Preference is less capable of completing a task-oriented dialogue without preference constraints and a clear task goal.

6.2 Evaluating the evaluation ability of simulators

We conduct experiments with our testers and compare several interactive evaluators (i.e., simulators and humans) in terms of EXACTDISTINCT to evaluate the evaluation ability of them.

Evaluation methods. Following the strategies described in Sect. 4.2.2, we implement three testers for each of the three datasets except for JDDC (cf. Sect. 5.2). We initialize the preferences from the preference module and let the simulators interact with system

Table 4: Evaluation results of variants on the MultiWOZ dataset. Dist denotes DISTINCT, Sat. denotes SATISFACTION, Eff. denotes EFFICIENCY, and Nat. denotes NATURALNESS.

	Test Set			Interaction			Human		
	BLEU	Success	Slot	Success	Dist	Sat.	Success	Eff.	Nat.
System	20.42	75.00	51.92	80.25	15.32	3.54	39.50	84.57	2.37
$\alpha = 3$	18.84	57.65	51.75	40.20	13.76	3.07	29.09	74.54	2.15
$\alpha = 1$	16.81	46.70	44.59	27.04	15.16	2.32	12.20	51.22	1.66
$\beta = 0.4$	20.49	38.17	51.69	28.60	15.27	3.26	24.53	77.36	2.13
$\beta = 0.1$	20.53	34.90	52.33	7.40	10.17	3.24	13.33	71.11	2.07
$\gamma = 0.1$	16.10	42.53	33.66	31.70	15.84	3.10	16.33	75.51	2.06
$\gamma = 0.01$	13.18	38.54	20.02	25.30	14.51	2.55	2.13	59.57	1.89

variants defined by testers. The simulators give a rating of the system after the interaction. We use user calibrated user satisfaction, i.e., the average of satisfaction and task success (as described in Sect. 4.1.6) as the ratings of models. If the scoring of the two system variants is the same, we further distinguish them by the number of dialogue turns (the one with fewer rounds is better). We then rank the variants based on the ratings and check if the order is the same as expected (i.e., $S_1 < S_2 < S_3$). The simulator will be scored 1 on EXACTDISTINCT if the order given by the simulator is consistent with expectations, otherwise scored as 0. We repeated the testing on 1K different preferences and calculated the averaged EXACTDISTINCT score. We also conduct a human evaluation on MultiWOZ to validate the reliability, i.e., if the human can distinguish the system variants. Specifically, we develop an annotation system and ask 40 annotators to talk with the system variants to complete a specific goal. We randomly assign systems and goals to users to avoid bias caused by order of operations. After the dialogue, the annotator is asked to rate the system in terms of SATISFACTION and SUCCESS. We use the averaged SATISFACTION and SUCCESS as the ratings of the system.

Results. Table 3 shows the results of the tester evaluation. First, MetaSim is more capable of distinguishing the system variants defined in a tester and achieving the best in terms of the EXACTDISTINCT score than Agenda model and Seq2seq model. This might be because the agenda model is limited to domain-specific human-curated rules and difficult to generalize, and end-to-end simulators can only evaluate the systems on the semantic level. Second, the difficulty of the tester varies, as does the difficulty of the tester on different data (e.g., Recommender tester is more difficult for people than others, because the responses generated by system defined by Recommender tester look reasonable and may confuse users). Third, people can better distinguish between different systems compared to simulators (e.g., Human get an EXACTDISTINCT of 43.63, 40.54, and 42.54 for the three testers on MultiWOZ, compared to 33.23, 40.11, and 35.21 for MetaSim). Note there still exists a performance gap between MetaSim and Humans, indicating that building human-like simulators is still a challenge. We leave it for future work of investigating the reasons behind it.

6.3 Evaluating the reliability of testers

Evaluation methods. To assess the variants of these testers that have sensible performance differences and validate the reliability

Table 5: Evaluation results of variants on the ReDial and JDDC datasets; same conventions as in Table 4. No results are included for the β (recommender tester) for JDDC as JDDC does not have a recommender component due to the unavailability of its database (cf. Sect. 5.2).

Datasets	ReDial					JDDC				
	Test Set		Interaction			Test Set		Interaction		
	BLEU	Success	Slot	Success	Dist	BLEU	Dist	Slot	Dist	
System	15.42	8.20	3.74	13.70	20.97	25.45	10.68	8.72	34.81	
$\alpha = 3$	14.87	8.20	3.37	11.45	23.01	21.76	9.49	6.80	31.10	
$\alpha = 1$	14.34	6.93	2.91	9.62	21.13	14.64	6.84	1.14	29.74	
$\beta = 0.4$	15.44	1.86	2.96	6.54	21.43	-	-	-	-	
$\beta = 0.1$	14.80	0.75	2.75	2.45	27.14	-	-	-	-	
$\gamma = 0.1$	14.91	4.62	3.03	5.95	18.52	22.48	10.05	6.18	33.68	
$\gamma = 0.01$	13.99	1.19	1.45	2.27	17.26	17.5	9.94	1.92	34.25	

of the tester evaluation framework, we evaluate their performance in the test set, interaction quality, and human assessment.

- **Test set evaluation** evaluates the system variants defined by testers on the test set. Specifically, similar to the way to evaluate simulators, we measure the generated utterances using the following metrics: BLEU [36] measures the word-overlap between the generated utterance the ground-truth utterance. SUCCESS measures if the recommended item meets all user preferences. SLOTACC measures the slot accuracy.
- **Interaction evaluation** evaluates the system by interacting with MetaSim. We generate 10K dialogue and calculate the SATISFACTION and DIVERSITY. SUCCESS measures if the system recommend an item that meets all the user needs. DISTINCT measures the diversity of simulated dialogue.
- **Human evaluation** evaluates the systems by human experiments. Human evaluation was operated on the annotation system we built for MultiWOZ and invited 40 users to talk to the system based on the predefined preference. We collected about 500 dialogues, where the users were instructed to label the following aspects at the end of each dialogue: (1) SUCCESS measures if the system solves the user’s problem, rating as: 1=Fail, 2=Success; (2) EFFICIENCY measures the efficiency of the system, rating as: 1=Inefficient, 2=Efficient; (3) NATURALNESS measures the human-likeness of the system, rating as: 1=Unnatural, 2=Fair, 3=Natural; (4) SATISFACTION is labeled on a 5-point scale: 1=Very bad; 2=Bad; 3=Fair; 4=Good; 5=Very good.

Results. Table 4 shows the evaluation results on MultiWOZ and Table 5 shows the evaluation results on ReDial and JDDC. First, the system variants are distinguishable to humans. The user satisfaction annotations (as an overall rating for the systems) are consistent with our expectations, while other metrics demonstrate the same results but with a focus on the specific capabilities of the system (e.g., NATURALNESS concerns the quality of the response). Second, testers can evaluate a specific capability of the system. For example, the recommender tester largely disrupts the system’s ability to make recommendations (low SUCCESS), while keeping the ability to generate natural responses (relatively good BLEU and NATURALNESS). Third, the interaction experiments also exhibit results consistent with expectations, but the difference between good and bad systems

Preference	domain = Hotel ; type = guesthouse ; stars = 4 ; area = south domain = Restaurant ; name = Riverside Brasserie ; people = 6 ; day = Sunday ; time = 14:30
Context	User: I’m also looking for a place to stay. In the south preferably. System: What price range were you thinking?
Metaphor	Any price point is fine as long as it is [stars] star and has free parking. I do not have a preference on price as long as it has a [stars] star rating and is a [type]. Not particularly considered about price range, just as long as it’s a [stars] star hotel.
MetaSim	User: Any price range is fine as long as it is 4 star and a guesthouse.
Agenda	User: I would like a 4 star guesthouse, please.
Seq2seq	User: Not really. I’m just looking for a 4 star hotel.
Human	User: No particular price range , but I would like it to be a 4 star hotel.

Figure 4: This figure compares the responses generated by MetaSim and Agenda, Seq2seq, and Human with the same dialogue history.

<p>1 MetaSim vs $S_{\gamma=1}$, rating = 0.875</p> <p>User: Hi, I need a taxi to come to curry prince and take me to sidney sussex college.</p> <p>System: I would be happy to help! Where are you coming from?</p> <p>User: I need to arrive by 12:15.</p> <p>System: Your booking is complete. A toyota will pick you up. The contact number is XXX.</p> <p>User: Thank you! That’s all I need.</p>	<p>3 MetaSim vs $S_{\gamma=0.01}$, rating = 0.125</p> <p>User: Hi, I need a taxi to come to curry prince and take me to sidney sussex college.</p> <p>System: I can help with that. What type of ride do you prefer, pool, or a different type of ride?</p> <p>User: I need to arrive at the college by 12:15.</p> <p>System: I have 1 located 23.60 pounds pounds. you can get there at 05:00 and arrive by 05:51.</p> <p>User: Can you give me the car type and contact number?</p> <p>System: I am sorry, I was unable to book a taxi for you. Would you like to try a different part of town?</p> <p>User: No, can you try to book a taxi in the same area?</p> <p>System: I can help with that. What type of ride do you prefer, pool, or a different type of ride?</p> <p>User: No I want to book a taxi.</p> <p>System: Okay, I have you booked. You will be at pool way, whitehill road, off newmarket road in the east.</p> <p>User: Can I have the contact number?</p> <p>System: Sure, the contact number is XXX. Is there anything else I can help you with?</p> <p>User: No, that’s all I need. Thank you so much.</p>
<p>2 MetaSim vs $S_{\gamma=0.1}$, rating = 0.375</p> <p>User: Hi, I need a taxi to come to curry prince and take me to sidney sussex college.</p> <p>System: Yes, what time would you like to leave and when would you want to leave by?</p> <p>User: I need to arrive at the college at 12:15.</p> <p>System: TR7075 will arrive at 05:51 on monday. Would you like me to book that for you?</p> <p>User: Can I get the car type and contact?</p> <p>System: Yes, it is a taxi and the contact number is XXX. Is there anything else I can help you?</p>	

Figure 5: Simulated dialogues of MetaSim with Domain tester, as well as ratings by the simulator.

are more significant. This is because the accumulation of errors in the interaction amplifies the difference in system performance.

6.4 Case study

Fig. 4 displays the responses predicted by MetaSim, Agenda, and Human, under the same user preference. We can see that the response generated by the MetaSim is more natural and coherent to the dialogue than the Agenda model. Fig. 5 shows the simulated dialogues of MetaSim with three variants defined by the Recommender tester, as well as the rating of the systems. These variants have sensible capabilities, witnessed by these examples, i.e., $S_{\gamma=1}$ successfully books a taxi that meets the user’s requirements, $S_{\gamma=0.1}$ completes a book, but the booking time is wrong, and $S_{\gamma=0.01}$ fails in this task. MetaSim distinguishes three systems by expected scoring.

7 CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we have studied the problem of evaluating task-oriented dialogue systems using user simulators. Based on the identified challenges when employing user simulators in evaluating TDSs, i.e., the lack of automatic methods to evaluate the user simulators, we have proposed a solution target at improving the realism of the current user simulator for better evaluating TDS. Specifically, we have proposed the metaphorical user simulator (MetaSim) for end-to-end TDS evaluation and a tester-based evaluation framework for evaluating the evaluation capability of the simulators.

We have conducted extensive experiments on three benchmark TDS datasets, i.e., MultiWOZ, ReDial, and JDDC. Our experiments have verified that the proposed MetaSim achieves better consistency with manual evaluation on three datasets and can generate more realistic dialogue than Agenda-based simulator and Seq2seq model.

We also have found that the tester-based evaluation framework is distinguishable to humans and demonstrates efficiency.

Limitations of this paper include the preliminary combination of user simulation and satisfaction assessment and the relatively heavy simulator design that limits the inference efficiency. In future work, we would like to extend the tester framework to support more prosperous simulator evaluation demands and achieve a better combination of user simulation and satisfaction assessment.

REFERENCES

- [1] Soren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC/ASWC 2007*.
- [2] Krisztian Balog, David Maxwell, Paul Thomas, and Shuo Zhang. 2021. Sim4IR: The SIGIR 2021 Workshop on Simulation for Information Retrieval Evaluation. In *SIGIR 2021*.
- [3] Alan W. Black, Susanne Burger, Alistair Conkie, H. Hastie, Simon Keizer, Oliver Lemon, Nicolas Merigaud, Gabriel Parent, Gabriel Schubiner, Blaise Thomson, J. Williams, Kai Yu, Steve J. Young, and Maxine Eskénazi. 2011. Spoken Dialog Challenge 2010: Comparison of Live and Control Test Results. In *SIGDIAL 2011*.
- [4] Dimitrios Bountouridis, Jaron Harambam, Mykola Makhortykh, Mónica Marrero, Nava Tintarev, and Claudia Hauff. 2019. SIREN: A Simulation Framework for Understanding the Effects of Recommender Systems in Online News Environments. In *FACCT 2019*.
- [5] Ben Carterette, Evangelos Kanoulas, and Emine Yilmaz. 2011. Simulating simple user behavior for system effectiveness evaluation. In *CIKM 2011*.
- [6] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of Text Generation: A Survey. *ArXiv abs/2006.14799* (2020).
- [7] Meng Chen, Ruixue Liu, Lei Shen, Shaozu Yuan, Jingyan Zhou, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. The JDDC Corpus: A Large-Scale Multi-Turn Chinese Dialogue Dataset for E-commerce Customer Service. In *LREC 2020*.
- [8] Heriberto Cuayáhuil, Steve Renals, Oliver Lemon, and Hiroshi Shimodaira. 2005. Human-computer Dialogue Simulation using Hidden Markov Models. *IEEE Workshop on ASRU 2005* (2005), 290–295.
- [9] Jan Dერი, Álvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2020. Survey on Evaluation Methods for Dialogue Systems. *Artificial Intelligence Review* 54 (2020), 755 – 810.
- [10] Wieland Eckert, Esther Levin, and Roberto Pieraccini. 1997. User Modeling for Spoken Dialogue System Evaluation. In *IEEE Workshop on ASRU 1997*. 80–87.
- [11] Jens Edlund, Joakim Gustafson, Mattias Heldner, and Anna Hjalmarsson. 2008. Towards human-like spoken dialogue systems. *Speech Commun.* 50 (2008), 630–645.
- [12] Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Anuj Kumar Goyal, Peter Ku, Sanchit Agarwal, and Shuyang Gao. 2020. MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines. In *LREC 2020*.
- [13] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and Challenges in Conversational Recommender Systems: A Survey. *AI Open* 2 (2021), 100–126.
- [14] Kallirroi Georgila, James Henderson, and Oliver Lemon. 2005. Learning user simulations for information state update dialogue systems. In *INTERSPEECH 2005*.
- [15] Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2021. MultiWOZ 2.3: A Multi-domain Task-Oriented Dialogue Dataset Enhanced with Annotation Corrections and Co-Reference Annotation. In *NLPC 2021*.
- [16] Homa Baradaran Hashemi. 2016. Query Intent Detection using Convolutional Neural Networks. In *WSDM 2016*.
- [17] Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2021. GALAXY: A Generative Pre-trained Model for Task-Oriented Dialog with Semi-Supervised Learning and Explicit Policy Injection.
- [18] Jin Huang, Harrie Oosterhuis, Maarten de Rijke, and Herke van Hoof. 2020. Keeping Dataset Biases out of the Simulation: A Debiased Simulator for Reinforcement Learning based Recommender Systems. In *RecSys 2020*.
- [19] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A Survey on Conversational Recommender Systems. *ACM Computing Surveys (CSUR)* 54 (2021), 1 – 36.
- [20] Xisen Jin, Wenqiang Lei, Zhaochun Ren, Hongshen Chen, Shangsong Liang, Yihong Eric Zhao, and Dawei Yin. 2018. Explicit State Tracking with Semi-Supervised Neural Dialogue Generation. In *CIKM 2018*.
- [21] Natalie A Jones, Helen Ross, Timothy Lynam, Pascal Perez, and Anne Leitch. 2011. Mental Models: an Interdisciplinary Synthesis of Theory and Methods. *Ecology and Society* 16 (2011), 46.
- [22] Filip Jurcicek, Simon Keizer, Milica Gasic, François Mairesse, Blaise Thomson, Kai Yu, and Steve J. Young. 2011. Real User Evaluation of Spoken Dialogue Systems Using Amazon Mechanical Turk. In *INTERSPEECH 2011*.
- [23] Anna Kaal. 2012. *Metaphor in Conversation*. Ph.D. Dissertation. Vrije Universiteit Amsterdam.
- [24] Sahiti Labhishetty and ChengXiang Zhai. 2021. An Exploration of Tester-based Evaluation of User Simulators for Comparing Interactive Retrieval Systems.. In *SIGIR 2021*.
- [25] Lori Lamel, Sophie Rosset, Jean-Luc Gauvain, Samir Bannacef, Martine Garnier-Rizet, and Bernard Prouts. 2000. The LIMSI ARISE system. *Speech Commun.* 31 (2000), 339–353.
- [26] Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2020. Conversational Recommendation: Formulation, Methods, and Evaluation. In *SIGIR 2020*.
- [27] Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying Task-oriented Dialogue Systems with Single Sequence-to-Sequence Architectures. In *ACL 2018*.
- [28] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *NAACL 2016*.
- [29] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Christopher Joseph Pal. 2018. Towards Deep Conversational Recommendations. *ArXiv abs/1812.07617* (2018).
- [30] Wenchang Ma, Ryuichi Takanobu, and Minlie Huang. 2021. CR-Walker: Tree-Structured Graph Reasoning and Dialog Acts for Conversational Recommendation. In *EMNLP 2021*.
- [31] David Maxwell and Leif Azzopardi. 2016. Agents, Simulated Users and Humans: An Analysis of Performance and Behaviour. In *CIKM 2016*.
- [32] Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Pei hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2015. Multi-domain Dialog State Tracking using Recurrent Neural Networks. In *ACL 2015*.
- [33] Nikola Mrksic, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve J. Young. 2017. Neural Belief Tracker: Data-Driven Dialogue State Tracking. In *ACL 2017*.
- [34] Tomáš Nekvinda and Ondrej Dusek. 2021. Shades of BLEU, Flavours of Success: The Case of MultiWOZ. *ArXiv abs/2106.05555* (2021).
- [35] Alexandros Papangelis, Yi-Chia Wang, Piero Molino, and Gökhan Tür. 2019. Collaborative Multi-Agent Dialogue Model Training Via Reinforcement Learning. In *SIGDial 2019*.
- [36] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *ACL 2002*.
- [37] 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 (2021), 807–824.
- [38] Olivier Pietquin and Helen Hastie. 2012. A Survey on Metrics for the Evaluation of User Simulations. *The Knowledge Engineering Review* 28 (2012), 59 – 73.
- [39] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.
- [40] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *ArXiv abs/1910.10683* (2020).
- [41] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics* 7 (2019), 249–266.
- [42] Pengjie Ren, Zhongkun Liu, Xiaomeng Song, Hongtao Tian, Zhumin Chen, Zhaochun Ren, and Maarten de Rijke. 2021. Wizard of Search Engine: Access to Information Through Conversations with Search Engines. In *SIGIR 2021*.
- [43] Lina Maria Rojas-Barahona, Milica Gasic, Nikola Mrksic, Pei hao Su, Stefan Ultes, Tsung-Hsien Wen, Steve J. Young, and David Vandyke. 2017. A Network-based End-to-End Trainable Task-oriented Dialogue System. In *EACL 2017*.
- [44] Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve J. Young. 2007. Agenda-Based User Simulation for Bootstrapping a POMDP Dialogue System. In *NAACL 2007*.
- [45] Weiyan Shi, Kun Qian, Xuwei Wang, and Zhou Yu. 2019. How to Build User Simulators to Train RL-based Dialog Systems. In *EMNLP 2019*.
- [46] Eric Michael Smith, Orion Hsu, Rebecca Qian, Stephen Roller, Y-Lan Boureau, and Jason Weston. 2022. Human Evaluation of Conversations is an Open Problem: comparing the sensitivity of various methods for evaluating dialogue agents. *ArXiv abs/2201.04723* (2022).
- [47] Weiwei Sun, Shuo Zhang, Krisztian Balog, Zhaochun Ren, Pengjie Ren, Zhumin Chen, and Maarten de Rijke. 2021. Simulating User Satisfaction for the Evaluation of Task-oriented Dialogue Systems. In *SIGIR 2021*.
- [48] Bo-Hsiang Tseng, Yinpei Dai, Florian Kreyszig, and Bill Byrne. 2021. Transferable Dialogue Systems and User Simulators. *ArXiv abs/2107.11904* (2021).
- [49] Oriol Vinyals and Quoc Le. 2015. A Neural Conversational Model. *ArXiv abs/1506.05869* (2015).
- [50] Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A Framework for Evaluating Spoken Dialogue Agents. In *ACL 1997*. 271–280.
- [51] Steve Young, Milica Gasic, Blaise Thomson, and Jason D Williams. 2013. POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proc. IEEE* 101, 1160–1179.
- [52] Shuo Zhang and Krisztian Balog. 2020. Evaluating Conversational Recommender Systems via User Simulation. In *ACM SIGKDD 2020*.
- [53] Yinan Zhang, Xueqing Liu, and ChengXiang Zhai. 2017. Information Retrieval Evaluation as Search Simulation: A General Formal Framework for IR Evaluation. In *SIGIR 2017*.

- [54] Zheng Zhang, Ryuichi Takanobu, Minlie Huang, and Xiaoyan Zhu. 2020. Recent Advances and Challenges in Task-oriented Dialog System. *ArXiv abs/2003.07490* (2020).
- [55] Zhuosheng Zhang, Hanqing Zhang, Keming Chen, Yuhang Guo, Jingyun Hua, Yulong Wang, and Ming Zhou. 2021. Mengzi: Towards Lightweight yet Ingenious Pre-trained Models for Chinese. *ArXiv abs/2110.06696* (2021).