

Hochschule Darmstadt

– Fachbereiche Mathematik und
Naturwissenschaften & Informatik–

User Simulation in Task-Oriented Dialog Systems based on Large Language Models via In-Context Learning

Abschlussarbeit zur Erlangung des akademischen Grades
Master of Science (M.Sc.)

vorgelegt von

Ronny Horst

Matrikelnummer: 759803

Referent : Prof. Dr. Bettina Harriehausen-Mühlbauer
Korreferent : Prof. Dr. Jutta Groos

DECLARATION

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 29. Dezember 2023

Ronny Horst

ABSTRACT

The growing importance of human-computer interaction and natural language processing is highlighted by significant advances such as the introduction of ChatGPT, a Large Language Model (LLM)-based dialogue system. While open-domain dialogue systems focus on user engagement, task-oriented dialogue systems (TODS) are designed to assist users with specific tasks within defined domains. However, the deployment of new TODS faces challenges, particularly in ensuring dialogue quality through resource-intensive human evaluation. Assessing the quality of TODS requires a nuanced understanding of user intent and the generation of contextually appropriate responses. Automated evaluation mechanisms play a crucial role in systematic testing prior to TODS deployment. Historically, two evaluation methods have been used: dataset-based and interactive. While dataset-based evaluation serves as a benchmark, it does not capture the dynamic nature of TODS, limiting its adaptability to real-world user responses. In contrast, the interactive evaluation involves a user simulator engaging in multi-turn dialogues with TODS, mimicking authentic conversational scenarios. Despite closely simulating real-world usage, creating effective simulators can be resource-intensive, with previous approaches relying on rules, heuristics, or large amounts of annotated data showing limitations in adapting to unexplored domains or resource-constrained environments.

This research presents an end-to-end ICL-based User Simulator (US) for TODS. Using LLMs, the proposed architecture can perform task-oriented user simulation in an interactive end-to-end manner with minimal data requirements. It addresses the need for user simulation for low-resource domains by evaluating zero and few-shot ICL strategies within the MultiWOZ domain against a pre-trained baseline US. In addition, an extended TELeR-RESPONDeR taxonomy for nuanced prompt descriptions is presented to increase the comparability of the proposed methods and to address the need for a standard ICL notation. The proposed In-Context Learning User Simulator (ICL-US) demonstrated proficient generation of lexically diverse user responses that closely matched real user baselines as quantified by MTLT. Although the ICL-US did not outperform the US baseline, it demonstrated promising conversational capabilities with task-oriented dialogue systems, highlighting the potential of ICL-based user simulation even in a zero-shot setting and emphasizing the potential of the ICL-based approach.

ZUSAMMENFASSUNG

Die Bedeutung der Mensch-Computer-Interaktion und der Verarbeitung natürlicher Sprache gewinnt sowohl in der Forschung als auch in der Industrie zunehmend an Bedeutung. Während bei domänenoffenen Dialogsystemen das Engagement des Benutzers im Vordergrund steht, zielen aufgabenorientierte Dialogsysteme darauf ab, den Benutzer bei bestimmten Aufgaben in definierten Domänen zu unterstützen. Bei der Einführung neuer TODS ergeben sich jedoch Herausforderungen, insbesondere bei der Sicherung der Dialogqualität durch menschliche Bewertung. Die Bewertung der Qualität von TODS erfordert ein nuanciertes Verständnis der Benutzerintention und die Generierung kontextadäquater Antworten. Automatisierte Evaluationsmechanismen spielen eine entscheidende Rolle bei systematischen Tests vor der Bereitstellung von TODS. Typischerweise werden zwei Evaluierungsmethoden verwendet: Datensatzbasierte Evaluierung und interaktive Evaluierung. Während die datensatzbasierte Evaluation als Benchmark dient, erfasst sie keine dynamischen Dialogsysteme, was ihre Anpassungsfähigkeit an Benutzerreaktionen in einer realen Umgebung einschränkt. Im Gegensatz dazu beinhaltet die interaktive Evaluation einen Benutzersimulator, der einen interaktiven Dialog mit einem Zielsystem führt. Die Erstellung leistungsfähiger Simulatoren kann ressourcenintensiv sein. Frühere Ansätze, die auf Regeln und Heuristiken basieren, erfordern einen hohen menschlichen Aufwand, während modernere Simulatoren oft große Mengen an annotierten Daten benötigen. Die vorliegende Masterarbeit stellt einen End-to-End In-Context Learning basierten Benutzersimulator für aufgabenorientierte Dialogsysteme vor. Mit Hilfe von LLMs kann die vorgeschlagene Architektur eine aufgabenorientierte Benutzersimulation in einem interaktiven End-to-End-Stil mit minimalen Datenanforderungen durchführen. Damit wird die Problematik von Benutzersimulationsdomänen mit geringen Ressourcen adressiert. Das Potential des Simulators wurde durch die Evaluierung von Zero- und Few-Shot-ICL-Strategien in der MultiWOZ-Domäne im Vergleich zu einem vorab trainierten Baseline-US analysiert. Zusätzlich wird eine erweiterte TELeR-RESPONDeR Taxonomie für nuancierte Prompt-Annotation vorgestellt, um die Vergleichbarkeit der vorgeschlagenen Methoden zu erhöhen und den Bedarf an einem Standard in der ICL-Notation zu adressieren. Der vorgeschlagene In-Context Learning Benutzersimulator zeigte eine vielversprechende Generierung von lexikalisch vielfältigen Benutzerantworten, die denen von echten Benutzerbaselines ähnlich sind, wie sie von MTLD quantifiziert werden. Obwohl der ICL-US die Erfolgsraten des Baseline-US nicht übertraf, zeigte er vielversprechende Konversationsfähigkeiten, was das Potenzial der ICL-basierten Benutzersimulation auch in einem Zero-Shot-Szenario unterstreicht.

CONTENTS

I THESIS

1	INTRODUCTION	2
2	BACKGROUND	6
2.1	Task-Oriented Dialogue Systems	6
2.1.1	Pipeline Architecture	7
2.1.2	End-to-End Architecture	10
2.1.3	Evaluation of TODSs	11
2.2	User Simulation	13
2.3	Large Language Models	22
2.4	Related Work	27
3	PROPOSED ICL-BASED USER SIMULATOR	33
3.1	Data Set Selection	34
3.2	Promptbuilding and Incontext-Learning	37
3.2.1	Proposed TELeR-RESPONDeR Taxonomy for Complex Prompts	38
3.2.2	Design of the Basic Prompt Elements for Zero- and Few-Shot Strategies	41
3.2.3	Zero-Shot and Few-Shot Setup	44
3.2.4	Reasoning, Planning and Ensemble Prompting Concepts	45
3.3	LLM Selection & Generation Decoding Strategies	49
3.4	Wizard of Oz Testing	51
3.5	Modification Via Dialogue State Tracking Component	51
4	EVALUATION	53
4.1	Experimental Setup	53
4.2	Evaluation Metrics	57
4.2.1	Goal Fullfillment Evaluation Metrics	57
4.2.2	Lexical Diversity Metrics	58
4.3	Quantitative Evaluation Results	59
4.3.1	Quantitative Results Goal Fullfillment Performance	59
4.3.2	Quantitative Diversity Evaluation Results	61
4.4	Qualitative Evaluation Results & Human Error Analysis	62
5	CONCLUSION & FUTURE WORK	65

II APPENDIX

.1	Appendix A: Qualitative Analysis Examples	70
----	---	----

	BIBLIOGRAPHY	78
--	--------------	----

LIST OF FIGURES

Figure 2.1	Typical modular pipeline structure of task oriented dialog systems as described in [104].	7
Figure 2.2	An example of the output of an nlu unit from [9].	8
Figure 2.3	Example of the dialog state of a typical DST module taken from [37].	8
Figure 2.4	Pipeline NLG system taken from [63].	9
Figure 2.5	Simple end-to-end Architecture, in this case interacting with an User Simulator.	10
Figure 2.6	Different variations of the transformer architecture taken from [61].	23
Figure 2.7	The LLM family tree as depicted in [100].	24
Figure 2.8	Architecture of the ICL-based User Simulator of Terragni et al. [85].	28
Figure 2.9	Architecture of the ICL-based User Simulator of Davidson et al. [14].	30
Figure 3.1	Proposed ICL-US end-to-end architecture.	34
Figure 3.2	A basic prompt and its typical elements	38
Figure 3.3	The detailed illustration from the four dimensions of the TeLER-Taxonomy from Santu et al. [71]	39
Figure 3.4	Typical structure of a few-shot prompt and the shot sampling strategy.	44
Figure 3.5	Flow chart of the embedding vector similarity search concept.	46
Figure 3.6	Sketch of the zero shot chain of thought two stage prompt design.	46
Figure 3.7	Sketch of the self-consistency process design.	47
Figure 3.8	Sketch of the prompt ensemble process design.	48
Figure 3.9	Sketch of the least to most prompting process design.	49
Figure 3.10	Sketch of a modification in the basic ICL-based US architecture adding DST component.	52
Figure 4.1	The modified ICL-US setup to enable automatic evaluation in the ConvLab-3 Framework.	54
Figure 4.2	Sketch of the conducted evaluation process.	55

LIST OF TABLES

Table 3.1	Comparison of potential Task-oriented Datasets.	36
Table 3.2	Proposed instruction element designs increasing in complexity.	42
Table 3.3	Demonstration Elements increasing in naturalness. . .	43
Table 3.4	Selected LLMs for manual prompting trials.	50
Table 4.1	Zero-Shot Success Evaluation Results.	59
Table 4.2	Role Success Evaluation Results.	60
Table 4.3	Few-Shot Success Evaluation results.	60
Table 4.4	Zero-Shot Diversity Evaluation Results.	61
Table 4.5	Role Diversity Eval Results.	62
Table 4.6	Few-Shot Diversity Evaluation Results.	62
Table .1	Example of Post-Processing and typical behavior of the LLMs to complete the Dialog.	70
Table .2	Different designed roles for role prompting.	71
Table .3	TELeR-RESPONDeR Taxonomy applied to the evaluated prompts inspired by Santu et al [71].	72
Table .4	LLama2 Zero-shot setting Instruction 1: US hallucinates the role of the Assistant.	73
Table .5	LLama2 dialog 2. zero-shot instruction: Simulator starts to repeat the same utterance.	74
Table .6	FLAN-T5 zero-shot instruction 0: Asking for Info not mentioned in Goal and system side missclassification tricking the US to believe the conversation is about to end without fulfilling the User Goal.	75
Table .7	GPT-3.5 Few-Shot Similarity Json Goal: TODS not understanding the US intent several times to get the address. Resulting in lower success metrics.	76
Table .8	GPT-3.5 Zero-Shot Instruction 5: TODS Template NLG not working correctly prolonging dialog and Miss-classifications.	77

Part I
THESIS

INTRODUCTION

The importance of human-computer interaction and natural language processing is attracting increasing attention in both academic and industrial communities. This trend has been accelerated by the groundbreaking publication of the LLM-based dialogue system chatgpt. In contrast to these open-domain dialogue systems, where the focus is on user engagement, task-oriented dialogue systems [104] aim at assisting the user to perform specific tasks in a particular domain. Typically, TODS are domain specific and are built on a structured ontology that defines the functionality of the TODS, such as making a restaurant reservation or booking a seminar, which in turn is valuable to real-world businesses.

However, a major obstacle to the release of new TODS is the labour-intensive development to ensure dialogue quality, which typically involves several stages and iterations of human evaluation before a system is deployed. The quality of a TODS and its user experience depends largely on the understanding of user intentions and the generation of appropriate responses. Therefore, automated evaluation of TODS plays an essential role in enabling researchers and developers to test and evaluate the quality of TODS in a controlled environment before deployment to real users.

Two main evaluation methods have been used in the past to assess the quality of new TODS, namely dataset-based and interactive evaluation. The most common approach in the literature is dataset-based evaluation, where a TODS generates responses to annotated dialogue logs from an existing dataset, such as the MUltiWOZ [6]. While this approach is typically used as a benchmark, it fails to capture the main task of a TODS: the interactive conversation with a user. Due to the fixed nature of the user utterances in the benchmark data, the system may not adapt well to real users, who may respond in a reasonable but unpredictable manner. A notable drawback associated with this type of evaluation is the potential occurrence of a policy mismatch between the responses generated by the Task-Oriented Dialogue (TOD) system and the expected benchmark user utterances.[11] In such cases, even reasonable and coherent responses from the TOD system may be penalised due to mismatch with the expected outputs defined in the benchmark dataset. Furthermore, common TOD benchmark datasets are constructed with predetermined dialogue schemas, which impose constraints on the domains and intents applicable to the evaluation of a new TOD system with benchmark data. Consequently, the evaluation of TOD systems developed for unexplored domains may not be possible using the dataset-based approach.

The second interactive approach consists of a user simulator interacting with the TODS in a multi-turn dialogue that mimics real conversations. This approach benefits from the ability of the US to adapt to the responses of the TODS, thus reducing policy mismatches. While closer to the actual use of a target TODS, the process of building an effective simulator can also be time consuming and costly. Older approaches require a lot of human effort to create rules and heuristics to generate user actions[19, 72], or like more recent work, rely on annotated data to train or fine-tune a US, often using TOD benchmark datasets [1, 11, 56]. This reliance on annotated datasets limits the adaptability to unseen domains of the current line of USs in the literature.

Recent advances in large language models[57, 105] such as GPT-4 and their emerging in-context learning capabilities are showing results in a variety of tasks. These include tasks in the domain of dialogue systems such as intent classification, dialogue state tracking, and natural language generation[28, 53, 80]. These models can be instructed to solve a complex task by providing textual input, typically consisting of instructions and/or demonstrations that provide context to the LLM. These capabilities to solve tasks with little data and no further training can be used to build US for the evaluation and development of TODS. During the creation of this thesis, Terragni et al. [85] and Davidson et al. [14] also recognised this research gap of using these capabilities for task-oriented user simulation, emphasising the importance and necessity of low resource user simulation approaches.

In this thesis the following three research questions will be investigated:

- Can Large Language Models, with their emerging in-context learning capabilities, effectively serve as user simulators in facilitating interactive conversations with Task-Oriented Dialogue Systems (TODS)?
- How would an LLM-based user simulator be set up architecturally, and what in-context learning strategies could be used to increase the humaneness of the simulator?
- How can different prompting strategies be formally described to ensure reproducibility and allow meaningful comparisons to understand the effectiveness of these strategies?

To answer these questions, a basic end-to-end ICL-based US architecture is presented, as well as the proposal of further possible evaluation components. As the key element of this user simulator approach is the prompt as input of a basic LLM, several prompting techniques to align the user simulator to a given user goal are applied to the task of user simulation. The basic architecture allows an end-to-end dialogue with a TODS based on a user goal that fits the domain in either logical or natural form and optional and/or an instruction that can be derived from a defined dialogue schema. Investigated prompting techniques include instruction tuning, role prompting, few shot

learning. In addition, possible applications of reasoning techniques and sub-task decomposition are conceptualised. The performance of the proposed US largely depends on the underlying LLM.

In addition, as the field of ICL has only recently emerged with the availability of LLMs, there is still a lack of a common standard to enable meaningful comparisons between ICL strategies based on benchmarks, as highlighted by Santu et. al. who proposed the TELeR taxonomy. This taxonomy is an important contribution to building a common standard for describing prompts for complex task solving. This thesis proposes an extension of the initialized taxonomy as suggested by the authors. The TELeR taxonomy has been extended by separating and adding dimensions that allow a more detailed description of the prompts used in this work. The proposed extended taxonomy is called TELeR-RESPONDeR and is described in detail in chapter 3.

As mentioned above, several prompting techniques have been conceptualised in this thesis. After designing several approaches, a selected subset of prompting techniques is evaluated using the Convlab-3 framework by conducting interactive conversations between a TODS and the proposed ICL-US and a baseline. The performance of the proposed user simulator was measured by an automated evaluation of success and diversity metrics. Finally, an error analysis was conducted to identify the limitations and common errors of the proposed user simulator and the evaluation process itself.

This thesis makes the following contributions:

- An end-to-end ICL-based user simulator architecture has been designed and possible extensions are presented.
- The basic end-to-end architecture has been revised for use in the ConvLab-3 evaluation framework.
- A selection of ICL strategies have been applied to the task of user simulation.
- The existing TELeR-taxonomy was extended and applied to all strategies to allow meaningful comparisons for future work.
- A selected set of strategies (Instruction, Role and Few-Shot Prompting) were evaluated in detail through interactive conversations between a TODS from the Convlab-3 framework and the proposed ICL-US, using automatic measures as well as human error analysis.

The research questions were addressed through different methodologies, such as the design of prompting strategies to address the task of task-oriented user simulation based on a Wizard of OZ style development process. First, a prompt design is conceptualised by analysing existing work of ICL of complex tasks, then these insights were applied through direct interaction

with an LLM and the prompting technique, furthermore the known problems of related work were considered while conducting a literature analysis. This approach allows the developer to directly verify the potential quality of the designed prompt. After deliberate design and testing, a selected set of strategies were evaluated in an interactive evaluation using a Task Oriented Dialogue Systems framework [109] that provides TODS and evaluation capabilities. In addition, the target domain corresponds to a standard baseline for task-oriented dialogue. Quantitative success and diversity metrics were used to measure the performance in generated user utterances of the LLM, comparing the contributions with existing user simulators and a baseline for diversity of utterances.

The thesis is structured to provide a comprehensive exploration of task-oriented dialogue systems, user simulation, and the integration of large language models. Chapter 2 provides the foundation for the topic of TODS. It covers pipeline and end-to-end architectures, evaluation methodologies, with a focus on user simulation. It also describes large language models, their different architectures, and their emerging in-context learning capabilities. The chapter concludes with a detailed review of related work and its influence on this thesis. Chapter 3 presents the core contribution of the thesis: the In-Context Learning (ICL)-based user simulator. The chapter begins with a review of the basic architecture, the rationale behind the dataset selection, and the strategies for in-context learning. In addition, a detailed explanation of the proposed extended taxonomy for complex tasks is given. The chapter also includes a description of the design choices for zero-shot and few-shot prompts. In addition, this section explores strategies for selecting and generating large language models, the Wizard of Oz test, and the integration of a dialogue state tracking component. The following chapter explains the evaluation framework, including the experimental setup and modifications to the test framework, and presents the selected baselines and TODS for the interactive dialogue. The following chapter provides an overview of the evaluation metrics, followed by a detailed analysis of both the quantitative results and the qualitative error analysis. The thesis concludes with a summary of key findings, an acknowledgement of limitations and suggestions for future research. This structured approach ensures a systematic exploration of the proposed in-context learning paradigm within the dynamic landscape of task-oriented dialogue systems and conversational AI.

BACKGROUND

This chapter is structured as follows. The first section presents general information about dialogue systems and a detailed description of task-oriented dialogue systems. After defining the characteristics of TODSs, different architectures of TODSs and their structures are presented. After that, evaluation approaches for TODSs are described in detail, and then the topic of user simulation and the different approaches used so far are discussed. The second section presents the necessary background information for understanding the use of LLMs in the concept and evaluation chapter. Starting with a general description of LLMs, an overview of the different architectures of LLMs is given. We then focus on in-context learning and different sampling strategies that influence the performance of LLMs. Finally, the most relevant related work is presented in detail, discussing their implications for this work and what distinguishes my work from theirs.

2.1 TASK-ORIENTED DIALOGUE SYSTEMS

A **DS!** (**DS!**), also known as a chat bot or conversational agent, is an interactive software system that communicates with a human agent using natural language [43]. Communication between a DS and its human counterpart is usually structured into *turns* (one or more utterances by an agent), *exchanges* (two alternating turns) and finally *dialogues* (multiple exchanges)[15, 43].

In the literature, DS are commonly divided into two broad categories, namely **TODSs!** (**TODSs!**) and **OODSs!** (**OODSs!**), according to their application[35, 63, 104]. OODSs aim to maximise user engagement in open domain dialogues without a specific task to solve [35]. Most OODSs are designed to emulate social interactions, so their communication is characterised by long and unstructured dialogues consisting of a large number of exchanges between the system and the user agent.

In contrast, TODSs are designed to help a user solve specific tasks or goals in one or more target domains as efficiently as possible [15, 35]. The dialogues of TODSs are characterised by being shorter in length and following defined structures [63]. The domain knowledge of the tasks is typically built on top of predefined ontologies or schemas[35, 104]. The architectural designs of TODSs can be classified into two categories: pipeline or modular and end-to-end systems, which are described in the following subsections [3, 104].

2.1.1 Pipeline Architecture

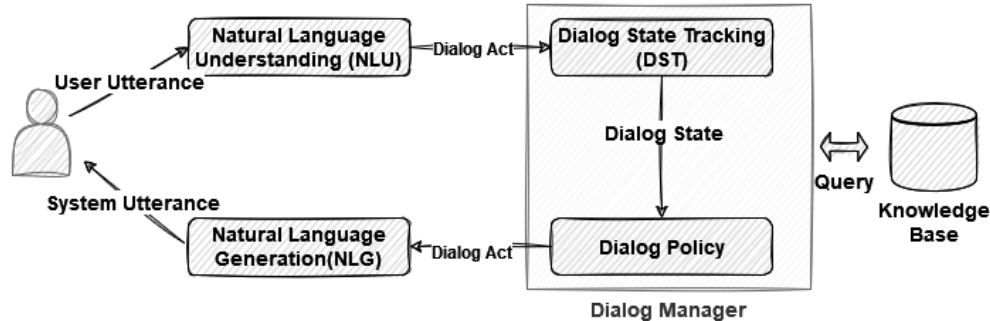


Figure 2.1: Typical modular pipeline structure of task oriented dialog systems as described in [104].

Figure 2.1 depicts the typical structure of pipeline based TODSs. The four main components are:

- **NLU!** (NLU!): This module parses raw user utterances into predefined semantic slots in combination with domain and user intent classification.
- **DST!** (DST!): This module tracks dialogue states based on user turns and conversation history.
- **DPL!** (DPL!): This module decides the next action of a dialogue agent based on the dialogue states calibrated by the DST module.
- **NLG!** (NLG!): This module converts selected dialogue actions into a natural language response.

For each module of the pipeline, more detailed information is provided in the following subsections.

2.1.1.1 Natural Language Understanding (NLU)

The main goal of the NLU module is to identify slot-value pairs in the current user utterance [16]. To achieve this goal, the NLU task is often defined as a set of subtasks consisting of: (i) domain identification (for multi-domain TODSs), (ii) intent identification, and (iii) slot identification [16]. The figure 2.3 shows an example of a possible output of an NLU module. Given a textual utterance, the module performs slot-filling through intent extraction and semantic analysis [43]. The subtasks address different levels of representation, domain and intent identification are based on the utterance level and slot identification works on the word-level representation [9].

Sentence	show	restaurant	at	New	York	tomorrow
Slots	O	O	O	B-desti	I-desti	B-date
Intent	Find Restaurant					
Domain	Order					

Figure 2.2: An example of the output of an nlu unit from [9].

Typically, the broader domain and finer-grained intent identification are approached as classification tasks [9]. Often these classifications are mapped to core functionality of the TODS [47]. For example, in Figure 2.3 the NLU module classified the *restaurant* domain and the *find restaurant* intent based on the user utterance. This identified intent could be mapped to a system-supported intent class, while this class could also fall within the restaurant domain.

The identification of slots is interchangeably known as slot-filling or slot/entity extraction [47, 63]. In contrast to intent or domain detection, slot-filling is typically defined as a sequence labelling problem. Thus, each word of the input sequence is assigned a label, resulting in a sequence of slot/concept IDs for each word [9]. Slot filling can be more challenging as the model has to predict multiple labels at the same time [63]. For example, in our case in figure 2.3, the entities/slots in the hotel domain have 4 different values, which are only extracted in the first exchange. In the third exchange the domain chains and a challenging reference to the conversation history is made by the user.

2.1.1.2 Dialogue State Tracking (DST)

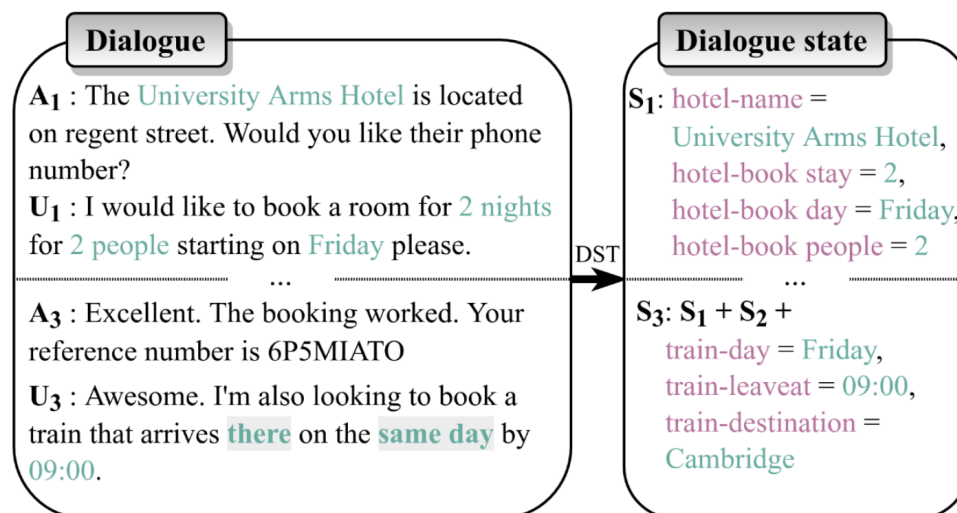


Figure 2.3: Example of the dialog state of a typical DST module taken from [37].

The DST aims to keep track of the user’s goal and related details in the form of a belief state[16, 43, 63]. During each turn, the current belief state is based on the entire dialogue history, the output of the previous nlu module and previous belief states, i.e. it is an internal representation of the state of the dialogue, hence the name DST[16].

The tracking of belief states is done using a predefined ontology, where slot-value pairs represent the user’s goal [9, 16]. Formally defined, a belief state s_t is drawn from a set S denoting the representation of the dialogue history up to turn t [98]. Thus, similar to the NLU module, the DST module solves a slot-filling problem [9].

Traditionally there has been a distinction between informable slots and requestable slots.[43] Informable slots are constraints provided by the user, e.g. price range, and requestable slots can be requested by the user, e.g. phone number. Informable slots can take special values like *don’t care* if the user is indifferent and *none* if the user hasn’t specified a goal for the given slot[43]. For single domains, the belief state consists of slot-value pairs, and for multi-domain TODSs this is extended to domain-slot-value pairs.

2.1.1.3 Dialogue Policy Learning (DPL)

The DLP is the second module of the Dialog Manager and controls the next available system action based on the output of the DST module. [63] Given the belief state s_t of the current round and an action space $A = \{a_1, \dots, a_n\}$, the module performs a mapping function $f : s_t \rightarrow a_i \in A$ or a request for missing information to perform an action. [43, 63]

[9, 63]

2.1.1.4 Natural Language Generation (NLG)

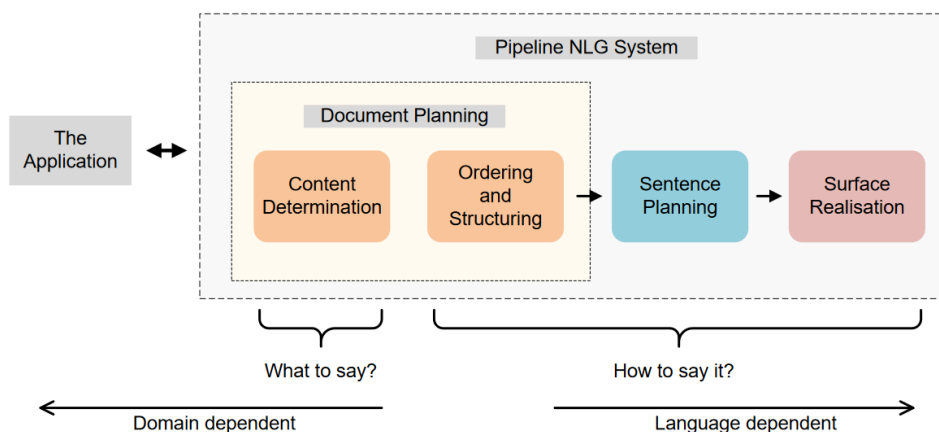


Figure 15: The pipeline NLG system

Figure 2.4: Pipeline NLG system taken from [63].

The final component of the pipeline architecture is the NLG, which generates a final response from a structured representation of the dialogue act generated by the dialogue manager into a natural language utterance [3, 23, 63]. For the NLU module itself, similar to the TODSs, there are two different architectural choices. The first traditional design choice of NLG modules follows a pipeline approach, as shown in 2.4, consisting of four components [43, 63]. Starting with content discovery, the definition of the domain dependent information base, followed by language dependent steps such as ordering and structuring, sentence planning and finally surface realisation [63]. End-to-end NLG architectures apply deep learning methods to improve NLG performance and collapse the pipeline into a single module [63].

In the literature, various attributes are described to measure the quality of NLG outputs and enhance the user experience.[9] To ensure robustness, the generator should fully convey the semantics of a dialogue act for task completion.[104]. Furthermore, the generated utterance should be natural, specific and informative, in short, it should be comparable to a response from a human assistant [63, 104].

2.1.2 End-to-End Architecture

Despite the good performance achieved, there are several drawbacks to the pipeline architecture. As the pipeline architecture consists of several components, each module has to be designed separately, which requires a lot of manual work to generate costly annotated data for each module. In addition, the performance improvement of a single component does not always translate to the whole TODS, exacerbating the problem of attributing negative and positive effects. In addition, the modular approach creates process interdependence, as each input of one component is dependent on the output of another component. Finally, if a module is not differentiable, errors from the end are not able to be propagated back to each individual module, making joint optimisation impossible. This is critical since the only signal in real dialogue data is the user's response [63].[16]

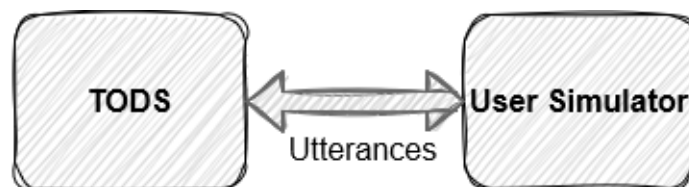


Figure 2.5: Simple end-to-end Architecture, in this case interacting with an User Simulator.

The end-to-end architecture of TODSs addresses the aforementioned drawbacks through two approaches, either through an end-to-end trainable pipeline TODS with differentiable modules, or as an end-to-end trainable single module TODS. The first approach can be viewed as a large differentiable system where parameters can be optimised by back-propagation in an end-to-end manner [63]. This approach still faces the problem that the knowledge retrieval itself is not differentiable [63]. The second, single module approach creates a more domain independent system inspired by ODDS, where answers are generated without intermediate results from a discrete action space or by statistical means [16, 43, 63, 104].

2.1.3 *Evaluation of TODSs*

Most evaluation studies of TODS are built around the structured nature of the interaction between the system agent and an external agent to accomplish a specific task [92, 104]. In addition, findings from real-world communication are extrapolated and applied to the field of human-machine dialogue [22]. The quality attributes of dialogue in the context of TODS are task resolution, dialogue efficiency, and usability. [16, 22, 104] As task resolution and dialogue efficiency are easier to quantify, the evaluation literature tends to focus on them rather than the more subjective and therefore harder to measure usability attribute [22].

2.1.3.1 *Task Success Rate*

Task resolution, also known as success resolution or goal completion, is used to evaluate the success of the TODS in providing all the information requested by the user in a correct manner, thus requiring mutual understanding and expectations of the user [22, 63]. It can be quantified as a Boolean indicating whether a task or a subset of tasks has been resolved [22].

One challenge of task resolution as a performance metric is determining whether a user goal has been resolved. This challenge is exacerbated by the possibility of multiple goals and goal changes during a dialogue. In general, a dialogue ends when the user aborts it, but this does not imply that the user goal has been achieved. Approaches to explicitly elicit verification by the user through affirmative questions such as "has your request been resolved?" or structured slot-filling of structured definitions of user goals to quantify this attribute have been widely applied [22].

2.1.3.2 *Task Completion Cost*

Task completion costs are the resources required to complete a given task by the TODS [63]. To quantify the task completion cost of a dialogue, the number of turns or elapsed time during the conversation is typically used, based

on the PARADISE framework. [22, 63, 92] Thus, the smaller the number of turns during a dialogue, the higher the dialogue efficiency.

In contrast, some researchers view the term through the lens of reinforcement learning, where dialogue length is related to the penalty for taking a wrong action [22]. Despite the general agreement that a TODS should aim to minimise dialogue costs, there is no established evidence that shorter dialogue is directly correlated with a higher user experience [22]. It could just as well be a sign of lower user engagement [22].

The methods to evaluate a TODS and measure its dialogue quality attributes can be divided into automatic, human and simulated evaluation [22, 104]. The automatic evaluation aims at quantifying various dialogue attributes into mathematical metrics [104]. After the development of evaluation scripts, no direct human intervention is required, thus no human evaluators are needed, resulting in lower costs compared to human-based evaluation [104]. There is a well-defined set of automated metrics in the literature for the different components of a TODS [84].

2.1.3.3 *Human Evaluation*

The evaluation of a machine-generated dialogue still requires human judgement for a more complete assessment of the conversation quality [104]. In research, human evaluators are often recruited through crowdsourcing platforms to perform indirect or direct evaluation [22, 104]. Indirect human evaluation focuses on annotating the simulated dialogue between a TODS and a user simulator, and then rating or giving preference between different systems based on different metrics [104].

Direct human evaluation asks the human evaluator to interact directly with the TODS to solve a specific task and then rate the usability of the TODS [104]. The usability evaluation is often based on questionnaires [22]. In addition, the dialogues of the human evaluators can be measured along the metrics of task resolution and task completion cost. Finally, evaluation can also be based on real user interactions as a form of online or log-based evaluation if the TODS is already deployed and real user data is available [3, 63].

Although user-based evaluation comes closest to capturing real user interaction, it is time consuming and costly [3, 22]. In addition to the lack of a human standard to use as a baseline, there is also a lack of reproducibility due to fatigue and learning effects, leading to limited value in making relative comparisons between systems [3, 22]. As a consequence, simulation-based approaches attempt to overcome the disadvantages of subjectivity and reproducibility [3, 22].

2.2 USER SIMULATION

Simulation-based methods aim to simulate a user, enabling dialogue simulation between an artificial user and the TODS for evaluation and training purposes [22, 104]. Simulation evaluation is an active area of research and is widely used in dialogue system platforms such as PyDial [90] and Convlab [50, 108, 109]. As user simulation is a focus of this thesis, a more detailed description of this evaluation technique follows.

There are several advantages of simulation methods, such as the possibility of end-to-end evaluation, the availability of multi-turn interaction during inference, reproducibility, and the generation of synthetic dialogue data for evaluation data at minimal cost [104]. The main challenges in simulated evaluation are the construction of a user simulator that mimics the real user as closely as possible, and the evaluation of the user simulator itself [104].

In general, user simulation has several use cases, such as large-scale automatic evaluation of interactive systems without the involvement of real users [3]. Gain insight into user behaviour and analyse system performance under different conditions. Finally, the approach can generate synthetic data for training machine learning models and can be used for reinforcement learning.

As user goals are limited in TODS, it is possible to use domain expertise or the underlying domain ontology to build a user simulator based on specific tasks [104]. User goals are important for the TODS because one of their main conversation quality attributes, the task success rate, depends on them. As a consequence, user simulation relies on the definition of user goals that a simulated user tries to achieve during a dialogue with the TODS [85]. A user goal can be defined in various forms, structured as a list of tuples such as (intent, domain, slot, value) or unstructured as natural language.

The agent can be based on models/algorithms/rules and general knowledge about user behaviour and the target user group [3]. As a user simulator in a conversational setup requires advanced natural language understanding and generation capabilities like a TODS, its architecture is similar to that of a TODS. Similarly to TODS, there are two broad categories of simulator architectures, pipeline and end-to-end simulators, which have the same setup as TODS in Figure 2.5 and Figure 2.1.

The components of a user simulator have the same attributes as a TODS, except that there is an additional component for user modelling. User modelling aims to capture the characteristics of real users. These characteristics

may include goals, knowledge, preferences and personal characteristics [3].

One of the earliest approaches to user simulation is the N-gram model proposed by Eckert et al. [19], where the next simulated user utterance is based on the dialogue history. The calculation of the probability of the next utterance is similar to that of language models, hence the name n-gram model. As Eckert et al. use a strong simplifying assumption $P(a_u|a_s)$, due to the difficulty of obtaining said probabilities, so that the next user response a_u depends solely on the previous system action a_s . This is why it is called a bi-gram model.

The bi-gram models provide locally appropriate responses to system utterances. However, they fail to provide consistent responses to one another throughout the dialogue, leading to unrealistic user behaviour [3, 21]. Three major limitations of bi-gram models contribute to unrealistic dialogues: first, they do not impose constraints on the simulated user behaviour [3], second, they do not consider user goals [3, 21], and finally, they only consider the last system action. The first constraint was addressed in subsequent work by Levin et al. [51]. There, the authors designed constraints on the dialogue flow by manually estimating a selected set of simpler probabilities for "meaningful" pairs of system and user actions, while setting all others to zero. For example, responses to system greetings, constraining questions, and relaxation prompts.

To overcome the second limitation, several approaches have attempted to explicitly incorporate the user goal into the modelling of the user state. Scheffler and Young [76] propose a graph-based model to ensure consistency of dialogue goals, which was not addressed by Levin. Their proposal combines deterministic rules for goal-dependent actions with probabilistic modelling to guide simulated user behaviour. The graph of the model must be mapped in advance, consisting of all possible paths the user can take during the conversation. The nodes of the graph are choice points, where some probabilistic choice points have a probability that can be estimated from data. The other nodes are deterministic choice points, which depend on the user's goal, represented as a table consisting of slot-name, slot-value pairs and their status variables.

Pietquin [68] combined the work of Levin et al. [51] and Scheffler and Young [76] to avoid manual effort while ensuring consistency of dialogue goals. The main principle was to condition the set of probabilities chosen by Levin et al. [51] on an explicit representation of the user goal and dialogue history/memory. The proposed user goal, similar to that in Scheffler and Young's [76] work, is represented as a sequence of slot-value pairs, where each goal is also assigned a priority and generated randomly.

Another approach is the agenda-based user simulation introduced by Schatz-

man et al. [72]. Here, a probability distribution over the user goal is assumed, which can either be induced from data [73] or set manually [72] if no data is available. The agenda-based user model is inspired by dialogue management approaches and the user state s_t is factored into a user goal G_t and an agenda A_t . The user goal aims to ensure consistent and goal-directed user behaviour. The goal is a tuple $G_t = (C_t, R_t)$, where C_t are domain-specific constraints and R_t are information requests of the user.

The agenda is a stack-like structure of pending user actions needed to elicit the information specified in the goal. The agenda is initiated by converting goal constraints into information acts and goal requests into request acts, and finally a bye act is added to the agenda to close the dialogue. The user state is thus represented by $s_t = (A_t, G_t)$ and is dynamically updated. In response to system utterances, new user acts are added to the agenda and resolved acts are removed. The probabilities corresponding to the maintenance of the agenda can be controlled by expert rules [72] or learned from training data [74].

A more recent line of research is the use of sequence-to-sequence models to create user simulators that can be trained in a fully data-driven manner from dialogue corpora. These approaches have gained popularity due to the availability of large annotated datasets and advances in machine learning techniques. Furthermore, these models can accommodate longer dialogue histories, which has been a limiting factor in previous work.

El Asri et al. [1] introduced a data-driven user simulator using an encoder-decoder recurrent neural network (RNN) [83], both based on a long short-term memory (LSTM) [31]. Similar to the agenda-based approach of Schatzmann et al. [72, 73], a goal $G = (C, R)$ is initialised at the beginning of each dialogue and remains fixed during the dialogue. At each turn t the encoder RNN takes as input the sequence of contexts up to turn t (c_1, \dots, c_t) and returns as vector v_t an internal representation of this sequence. Each context c_t consists of four elements: the last machine action, the inconsistency between system information and user goal, the constraint status, and the request status. The decoder then receives v_t as input and generates a sequence of user intentions I_1, \dots, I_l , which are then transformed into user dialogue acts a_{t+1}^u , referring to the current user goal and drawing uniformly among the remaining constraints/requests.

To evaluate the model, El Asri et al. [1] train their model and the baselines on the Dialogue State Tracking Challenge 2 (DSTC2) dataset [29]. For comparison, the authors chose an n-gram, an agenda-based user simulator, and a sequence-to-one RNN, which takes the same input as the proposed model but selects an output from predefined actions. On the DSTC2 validation and test sets, the sequence-to-one and sequence-to-sequence models significantly outperformed the other models on the F-score. Furthermore, El Asri

et al. validate the domain adaptability of the models by training them on the DSTC2 and evaluating them on the Dialogue State Tracking Challenge 3 (DSTC3) dataset [30], which is an extended version of the DSTC2 with increased constraints. The sequence-to-one and sequence-to-sequence models also outperformed the other models in terms of F-score.

Gür et al. [25] propose a hierarchical user simulator (HUS) that does not require feature extraction or a supervised signal for state tracking. This hierarchical sequence-to-sequence model first encodes the initial user goal and system turns into vector representations using RNNs, and initialised from the user goal vector, a higher-level RNN encoder generates a dialogue history representation using system turn vectors as input at each time step. The simulated user turns are then decoded from this dialogue history representation. This introduced HUS model will generate the same user turns given the same inputs.

To account for the diversity of user responses, which is important for robustness evaluation of a system, Gür et al. [25] additionally propose variational HUS (VHUS). VHUS differs from HUS only in a variational sampling step that is added before decoding to generate more diverse responses. A key feature of their approach is a coarser level of representation of system dialogue actions by replacing specific slot values with [*Requested, DontCare, ValueInGoal, ValueContradictGoal, Other*]. The value is *Requested* if the value is requested by the system, *ValueInGoal* if the value appears in the user goal, *ValueContradictsGoal* if the value contradicts the user goal, *DontCare* if the user goal does not care about the value, or other *Other*. Based on the coarse value, the actual value can be sampled either from the user goal, the system turn, or from a knowledge base.

Lin et al. [55] propose a domain-independent transformer-based [91] user simulator (TUS) that utilises the transformer architecture. TUS uses a feature-based input representation similar to that used by El Asri et al. [1], except that Lin et al. make it domain-independent. The transformer-based approach allows the use of variable input sequence length depending on the slots mentioned by the system, and the relationship between different slots is captured by self-attention. The output includes the domain that should be chosen for the given turn and a list of one-hot vectors to determine the values of the slots, using a coarse grained representation similar to Gür et al. [25].

One evaluation approach by Lin et al. [55] is to train a dialogue policy with different USs, and then to access the quality of the resulting policy through cross-model evaluation. In the cross-model evaluation, the resulting policies are evaluated with all USs to demonstrate the generalisability of the policy trained with a particular US. TUS, VHUS and the agenda-based US were chosen as user simulators using the Convlab-2 framework [108] and the MUltiWoZ 2.1 dataset [20]. TUS outperforms the data-driven VHUS and is

comparable to the agenda-based US. VHUS was not able to adequately train nor to evaluate a multi-domain policy, which was also observed in other experiments by Takanobu et al. [84]. The authors suggest that this is due to the design for a single domain, thus limiting the ability to generalise to a multi-domain scenario. The human evaluation supports the findings of the cross-evaluation that TUS can compete with the agenda-based US, although the latter, unlike TUS, has carefully designed domain-dependent rules. Furthermore, Lin et al. [55] tested the ability of TUS to handle unseen domains in a zero-shot study using the leave-one-domain-out method to show that TUS can handle new domains without feature modification or model re-training, even with fewer training samples.

Crook and Martin [13] presented initial steps towards training an nl-to-nl US without additional human annotation. They also investigated the effect of adding dialogue context as an input to the proposed model. Crook and Marin extend a basic encoder-decoder RNN with an additional dense layer between the encoder-decoder for better representation of context, following a proposed architecture for non-goal-driven dialogue systems [77]. The input sequence of the encoder consists of the system prompt and returns a summary vector of fixed size. This summary vector is then fed into a decoder which generates an output sequence, in this case the specific user utterance.

Crook and Martin [13] evaluate their model using an aggregate evaluation approach inspired by language modelling. They compute the perplexity of the evaluation set data against a language model trained on utterances generated by this simulated user model. The manual evaluation of naturalness and discourse cohesion closely matched the human user data. This was also consistent with the results of the LM complexity measure. The proposed seq-to-seq models approximated the human distribution of completions, aborts, abandoned goal changes and average dialogue length, thus matching the general statistics observed in real user data and reproducing natural sounding dialogues.

Kreyssig et al. [46] present the Neural User Simulator (NUS), which follows a similar encoder-decoder RNN architecture and context representation as El Asri et al. [1], but outputs a sequence of words instead of a dialogue act. Furthermore, the proposed NUS allows for goal changes during the dialogue if the system indicates that the original goal is not achievable. The behaviour of NUS is learned from a corpus and is able to generate a diverse set of utterances in the same dialogue context, thus requiring less labelling than USs that generate a response in semantic form.

Kreyssig et al. [46] evaluate their NUS in an indirect way by evaluating policies obtained by training with the NUS and ABUS [72]. They follow the cross-model evaluation of [75] and extend it by testing with real users [42] instead of using a direct evaluation with statistical metrics on the NUS. The

cross-model evaluation focuses on reward and average success rate and the human evaluation was performed on the best performing policy of NUS and ABUS, NUS outperformed in both evaluation approaches.

Lin et al. [54] propose a generative transformer-based user simulator (GenTUS), this model generates both semantic actions and natural language utterances. GenTUS is based on an encoder-decoder structure using the PLM BART [52], allowing it to optimise user policy and natural language generation together. This approach exploits the capabilities of pre-trained large language models, which improves diverse and fluent language generation and generalisation to new domains, but may also be vulnerable to hallucination by the PLM. The encoder of GenTUS takes as input the last system dialogue act, the previous three user dialogue acts, the user goal and the turn number t . Subsequently, the decoder generates the next user dialogue act and the natural language utterance as output. The approach uses the ontology to constrain the generation process to avoid false semantic actions.

Lin et al. perform their experiments on two corpora (MultiWoZ [20] and SGD [48]) and the GenTUS model is inherited from Huggingface's transformers [99]. Their generalisation and transferability to a new ontology is tested by training GenTUS on MultiWOZ and testing it not only on the MultiWOZ test set but also on the SGD test set without any further fine-tuning and vice versa. Their evaluation is based on direct and indirect evaluation of GenTUS. The indirect evaluation aims to measure the NLG performance through automatic metrics such as: Slot Error Rate, Sacre-BLEU Score and Self-BLEU Score as well as a human preference test to measure naturalness and informativeness. The indirect evaluation is performed by the cross-model evaluation proposed by Schatzmann et al. [75] using the Convlab-2 framework [108] with different agenda-based USs and the GenTUS.

Cheng et al. [11] propose an interactive evaluation framework for TODSs by first building a goal-oriented user simulator based on PLMs, and then using the simulator to interact with the DS to generate dialogues. Furthermore, they introduce a sentence-level and a session-level score to measure sentence fluency and session coherence in the interactive evaluation, which improves on the traditional evaluation. They address the fact that the current TODS evaluation process relies on annotated user utterances in multi-turn dialogue sessions without reference to the actual response generated by the TODS, leading to a policy mismatch.

Goal state guided simulator, user utterance generation based on goal state tracking RL methods to improve interaction performance between US and DS. Emphasis on interactive scoring, BLEU not possible, therefore similar score to Text Perplexity and BARTScore eval. Simulator based on T5-small instead of T5 base Score implementations GPT2 model fine tuned on MultiWoZ data for session level score BERT base model they evaluate different

TODS with their US on MultiWoZ dataset.

Liu et al. [56] investigate whether PLMs can be used for US purposes and how to effectively incorporate the user goal and its tracking in a flexible way, and how to flexibly integrate goal state tracking into an end-to-end trainable US for multiple domains. They propose a generative user simulator (GUS) with GPT-2 based architecture and goal state tracking to address these issues. They perform extensive experiments with MultiWoZ 2.1 and train different DS via RL with GUS and other USs like ABUS and other ablation simulators (TUS and VHUS) in cross-model evaluation, corpus-based evaluation and human evaluation. GUS is superior in all three evaluation tasks.

Sun et al. [82] propose a metaphorical user simulator (MetaSim) that simulates the user's analogical thinking to improve the realism and transferability of USs. MetaSim's metaphorical feature is based on using historical strategies as metaphors in an ongoing dialogue by referring to historical dialogues when encountering new objects. MetaSim consists of a preference module that initiates different user goals and updates them during the dialogue, an NLU module that tracks the state of the dialogue, a metaphor muddle that retrieves similar dialogue strategies related to the current state of the dialogue, a policy module that simulates user satisfaction and predicts the user action according to the context of retrieved strategies, and finally an NLG module that generates the user utterance.

They validate their proposed method through automated and human experiments on three benchmark datasets, MultiWoZ, ReDial and JDDC. The implementation of MetaSim is based on the pre-trained T5 transformer model, and adopts a unified data format that allows it to be generalised to multiple tasks. They optimise each module on the dataset and connect them during testing. Shi et al. [78] research different evaluation metrics for USs, and different ways to build USs at levels of dialogue manager and NLG + pros and cons of each method. insights on better user simulator building based on compressive analysis. direct and indirect evaluation, human evaluation /rating on generated dialogs and direct interaction with the RL system trained with different simulators. Automated metrics, perplexity, vocabulary size, average utterance length and success rate. Blue only available with ground truth. Human direct eval metrics, fluency, coherence, goal attainment, diversity, total. Human indirect amazon mechanical turk rl systems trained with different USs, solved ratio, satisfaction, efficiency, naturalness, rule-likeness.

Tesen et al. [89] propose a novel learning framework for developing DS joint optimisation with a user simulator (JOUST). pre-training on complex multi-domain datasets, two agents are able to interact using NL and further create more diverse rich dialogues. RL optimises both agents to explore

new better policies. transfer learning from one domain to another is facilitated. compare two reward designs, 1. task success at dialogue level 2. fine grained reward operating at turn level. Contributions: Joint optimisation of a fully text-to-text DS with a matches US on complex multi-domain human-to-human dialogues, extensive experiments exploring different types of rewards (showing that this framework with a learnable US improves overall performance and reaches a new state of start performance on MultiWoz), demonstration of transfer learning effectiveness on 2 tasks of practical use in low-resource scenarios with in-depth analysis of the source of the imprecisions. Using MultiWoz Mehri et al 2019 combined eval metric info success rate and bleu. Human eval DS success, US human-like, dialogue flow.

Another line of research has focused on simulating or generating entire conversations to generate training data. Mohapatra et al. [62] present a data generation strategy using PLMs such as GPT-2 [69] to simulate the interaction between crowd workers by creating both a user bot and an agent bot. The authors train their simulators using a smaller percentage of actual crowd-generated conversations and their corresponding instructions. They show that their simulated data can achieve improvements in low-resource settings on two datasets (MultiWoz 1.0 [6] and the Persona chat dataset [103]).

Chen et al. [10] demonstrate that LLMs can be used to simulate task-oriented dialogues in novel domains when provided with an API and user goals. Furthermore, they show that simulations can formulate online, automatic metrics that correlate with human evaluations. In an iterative setup, they measure the quality of the simulation by checking the goal success rate, incorporating successful dialogues back into the training set, retraining, and repeating the process. They observe that the quality of the models increases with model quality, and that the success of the simulation task (which can be automatically measured in fully synthetic dialogues) is a strong discriminator between high and low quality dialogues, as well as assistance performance. On the basis of this observation, they design an active learning strategy to improve performance. The second part of their work focuses on bootstrapping user and assistant models for previously unseen tasks. Thus, the simulations bootstrap end-to-end models that reduce errors in previously unseen domains. One of the first approaches to effectively bootstrap models without explicitly requiring domain-specific training, data, rule engineering or humans in the loop.

Steindl et al. [80] evaluate whether new prompt-following LLMs can generate annotated synthetic dialogues for TODs training purposes. They generate data based on descriptions of dialogue goals. To confirm the quality of their synthetic dialogues, they train a state-of-the-art TODS to compare it in a low-resource setting with and without synthetic dialogues.

The authors tested several Instruction Tuning Techniques through discrete prompt engineering using GPT-3.5-turbo [65]. The final prompt consisted of: a general task, basic dialogue rules, all possible domains, their slots and a template on how to name them, a list of possible dialogue acts, a template on how to mark annotations, how to proceed if no value is given for a slot, and finally an annotated example dialogue. They tried to output directly in Json format, but this did not work. They measured the similarity between the original dialogue and the synthetic one.

Li et al. [53] study DIALOGue Simulation with In-Context Learning (DIALOGIC) utilizing LLMs. Starting with a seed of a few annotated dialogues, DIALOGIC automatically selects in-context demonstrations and prompts as inputs for GPT-3 to generate new dialogues and annotations in a contrrollable fashion. This approach aims to expand a small set of dialgue data with minimum or zero human involvement and parameter update as a cost-efficient and time saving alternative to crowdsourcing. Their experiments are based on the MultiWOZ dataset and compare their synthetic data with data of real humans from the dataset.

Leveraging the ICL capability of LLMs, the methodology Li et al. [53] can simulate user and system side to generate annotated dialogs learning from a minimal data base. Except for the effort of collecting the small seed dataset and training and auxiliary model on it, the simulation process is free of human involvement or parameter update. It is claimed that a large and high quality dataset such as MultiWOZ can be generated in a few hours. Secondly, they design a controllable dialogue generation strategy to overcome the shortcomings of GPT-3 in terms of lack of reliability and interpretability. In addition, they investigate effective representations and selection strategies of ICL dialogue demonstrations to improve the ICL capabilities of LLMs.

They have positive results in challenging low-resource settings, where only 1% of the total training data set is used. Human evaluation indicates that their simulated dialogues have comparable fluency, annotation accuracy and more varied dialogue flows than human-generated dialogues.

There have been a number of recent attempts to build end-to-end trainable TODS using end-to-end generative neural models. [3] The same approach is viable for generating user responses as a sequence-to-sequence problem based on dialogue history. Transformational natural language generation models have been successfully applied to user simulation in several studies. [3] Recent work also takes advantage of advances in pre-trained LLMs to apply them to the field of user simulation, as discussed above.

2.3 LARGE LANGUAGE MODELS

A **LM!** (**LM!**) is a computational model with natural language understanding and generation capabilities [8]. The goal of LMs is to model the generative likelihood of word sequences in order to predict the probabilities of future/missing tokens [105]. The research in language modelling that led to the dominant type of LLMs today can be divided into four major stages of development [105]:

1. **SLM!** (**SLM!**)
2. **NLM!** (**NLM!**)
3. **PLM!** (**PLM!**)
4. Large Language Model **LLM!** (**LLM!**)

In the 1980s and 1990s, SLMs were developed based on statistical learning methods following the Markov assumption, e.g. predicting the next word based on the last context [26, 105]. The most common SLM is the N-gram model with a fixed context length of n . Despite the wide application of SLMs in information retrieval and natural language processing, SLMs suffer from the curse of dimensionality. Accurate estimation of higher order language models is problematic due to the exponential number of transition probabilities that need to be estimated [105].

The next stage in language modelling came with the development of NLMs, which, as the name suggests, use neural networks to predict the likelihood of a sequence of words by learning patterns and structures of language from text corpora. This stage initiated the use of LMs for representation learning beyond word sequence modelling, changing the field of NLP.

PLMs follow the proposal of context-aware word representations by first pre-training a language model and then fine-tuning it for specific downstream tasks. This learning paradigm of pre-training and fine-tuning was introduced by ELMo and the highly parallelizable transformer architecture with self-attention mechanisms of BERT. These pre-trained context-aware word representations improved the performance of NLP tasks through their general-purpose semantic features. Many follow-up studies introduced PLMs based on different Transformer-style architectures (GPT-2, BART) or pre-training strategies.

Figure 2.6 shows the difference in transformer architectures, training objective and an input output example for the three prevalent classes of PLMs [61]. Encoder-Decoder, auto-regressive or decoder-only Models and Masked

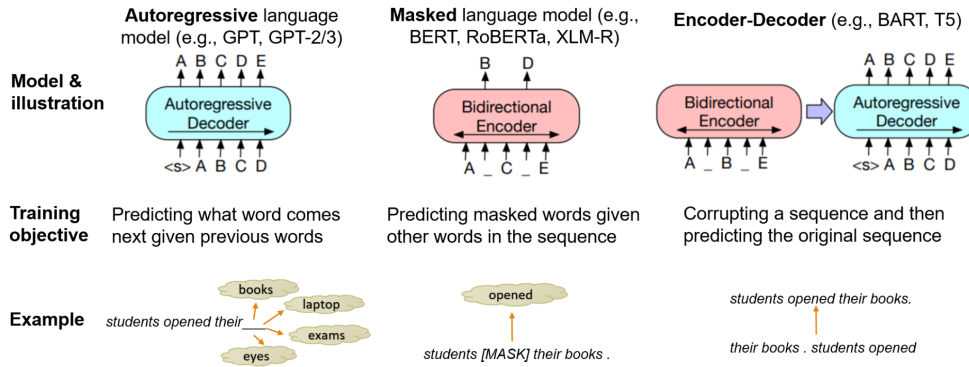


Figure 2.6: Different variations of the transformer architecture taken from [61].

language models or encoder-only models.

The basis of the PLM architectures was introduced by the transformer model [91] and its encoder-decoder architecture, which consists of two stacks of transformer blocks as encoder and decoder [94]. The encoder uses multi-head self-attention layers to encode the input sequence to generate its latent representations. The decoder performs cross-attention on the encoder's representations and autoregressively generates the target sequence. Thus, given an input sequence x_1, \dots, x_n , the model generates a sequence of tokens y_1, \dots, y_m . Here the training goal is to maximise the log-likelihood of $\log(P(y_1, \dots, y_m | x_1, \dots, x_n); \Theta_\tau)$, given a pair of sequences, where Θ_τ are the parameters of the full encoder-decoder model [61].

Encoder-only or masked language models predict a "masked" word depending on all the other words in the sequence. During training, this can be done by randomly replacing words with a special [MASK] token. This forces the model to collect bidirectional information in order to predict the masked word. The training objective, to find the original token at the masked positions, is formulated as $\sum_i m_i \log(P(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n); \Theta_\tau)$, where $m_i \in 0, 1$ indicates whether x_i is a masked word or not, and Θ_τ equals the parameters of the encoder [61].

While encoder-only models are bidirectional, decoder-only or auto-regressive models are unidirectional. They are trained to predict the next word x_i given all previous words x_1, x_2, \dots, x_{i-1} in a unidirectional way. The goal is to maximise the log-likelihood $\sum_i \log(P(x_i | x_1, x_2, \dots, x_{i-1}); \Theta_\tau)$, where Θ_τ equals the model parameters. The model parameters of a transformer decoder-only architecture are multi-head self-attention modules. The stacking of multiple transformer decoder layers with masked self-attention allows the model to predict the next token based on all previous tokens in the sequence [61].

Scaling PLMs in model size or data size often results in improved model capacity for downstream tasks. Studies have explored the performance limits by training larger PLMs. Although scaling is mainly done in model size,

these scaled PLMs show surprising abilities, also called emergent abilities, in solving a variety of complex tasks. This has led to the coining of the term "large language model" for large PLMs, which are attracting increasing research attention [94].

Figure 2.7 depicts the different PLMs and LLMs and their underlying ar-

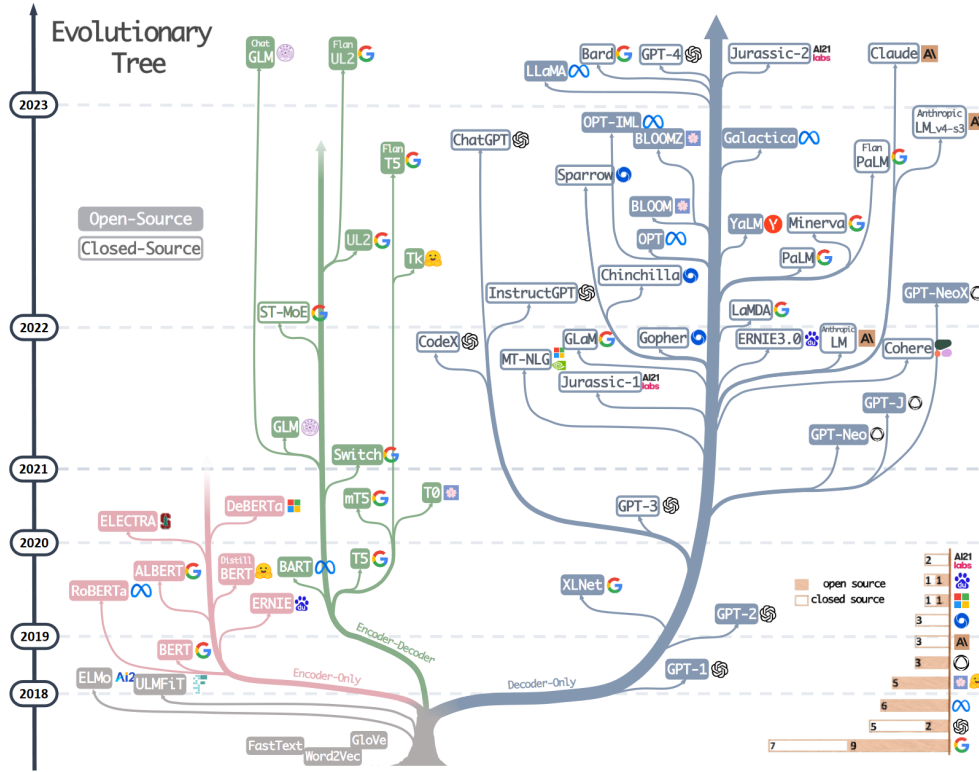


Figure 2.7: The LLM family tree as depicted in [100].

chitectures, where due to its parallelizability and capacity, the Transformer architecture has become the predominant backbone for LLMs development [94]. It clearly shows the dominance of decoder-only models, especially after 2021, where the disruptive release of GPT-3 [65] initialized a surge in decoder-only LLMs.

There are three major differences between PLMs and LLMs, the first being the emerging capabilities mentioned above. The second difference relates to the change in development and use, where unlike the smaller PLMs, the approach to accessing LLMs is usually through the prompting interface. Therefore, people need to understand how LLMs work and format their tasks accordingly to harness the power of LLMs. Finally, training LLMs requires experience in large scale data processing and distributed parallelization of training, blurring the traditional boundaries between research and engineering [94].

Despite the general skills that LLMs are equipped with after pre-training,

an increasing number of studies have shown that adapting LLM skills to specific tasks can be beneficial. The two main approaches to adapting pre-trained LLMs are alignment tuning and instruction tuning. While the former approach focuses on aligning the behaviour of LLMs with human values and preferences, the latter approach aims to enhance the capabilities of LLMs [94].

Instruction tuning is the approach of fine-tuning the pre-trained LLM on instruction-formatted data in the form of natural language. This is related to multi-task data and supervised fine-tuning. The instruction tuning process starts with the construction or collection of instruction formatted data, then these formatted instances are used to fine-tune the LLM in a supervised fashion.

As LLMs tend to produce harmful, incorrect or biased outputs, alignment tuning aims to align these outputs with human values or preferences. Unlike pre-training or instructional tuning, alignment tuning focuses on different criteria that may also affect the general capabilities of LLMs, also known as alignment tax. Common alignment criteria include helpfulness, honesty and harmlessness. Subjective and qualitative evaluation through human feedback is used in reinforcement learning, as human-feedback reinforcement learning, to adapt the LLM to the feedback by learning a reward model.

After pre-training or fine-tuning, one way to use LLMs is to design prompting strategies to solve different tasks. Some researchers have already coined a new paradigm shift called "*pre-train, prompt and predict*" instead of "*pre-train, fine tune*". [57], which emphasises the importance of prompting and the ability to do without fine-tuning. This paradigm doesn't adapt LLMs to downstream tasks by fine-tuning, but instead reformulates downstream tasks to look more like the tasks during the original LLM training by using a textual prompt [61]. Thus, reformulating NLP tasks as text generation tasks allows them to be solved directly by LLMs [61].

ICL! (ICL!) is a typical prompting method introduced in the GPT-3 paper [5], where the LLM learns from an analogy based on textual input [18, 94]. This basic method can be further enhanced by reasoning techniques such as chain of thought prompting [95] or planning techniques such as least to most prompting [107]. As all these techniques are used in this thesis, a more detailed description of these techniques is given.

ICL uses a formatted natural language prompt consisting of a task description and/or demonstration examples [94]. Templates combine the instruction, the demonstrations and finally an input instance to create the final prompt, which is used to generate the desired output. By learning from the given context of the demonstrations, ICL can improve performance on new

tasks without gradient updates [94].

Formally, given a set of candidate answers $Y = \{y_1, \dots, y_m\}$ and a set of possible examples $X = \{x_1, \dots, x_n\}$, where Y could be a class of labels or a set of free text phrases and X could be examples with known matching candidate answers. Let $D_k = \{f(x_1, y_1), \dots, f(x_k, y_k)\}$ represent k demonstration examples and f the formatting function, I represent the task description I and let x_{k+1} be a given a query input then the prediction \hat{y}_{k+1} generated from LLMs can be formulated as¹: $LLM(I, f(x_1, y_1), \dots, f(x_k, y_k), f(x_{k+1}, _)) \rightarrow \hat{y}_{k+1}$ where the actual answer, if known, y_{k+1} is left blank, as it is to be predicted by the LLM.

The demonstration design has a major influence on the performance of ICL and is highly dependent on the demonstration design as shown in many studies[58, 106]. The quality of the demonstration design is strongly influenced by the selection of the demonstrations, the order of the selected demonstrations and the format of the demonstration prompt[18, 94].

In addition to well-designed demonstration examples, well-designed instructions are also important for inference performance [18]. Unlike demonstration examples, which are typically sampled from datasets, task instructions often rely on manual human construction. However, existing work addresses this reliance on human labour through automatic instruction generation [18].

The new paradigm of ICL has several advantages. First, because it is based on natural language, it allows humans to communicate with LLMs through an interpretable interface [18]. As a result, it facilitates the incorporation of human knowledge into the LLM through demonstration and templates. Secondly, ICL resembles the human decision making process by learning from analogies. Finally, ICL is a training-free learning framework compared to adaptation/fine-tuning of an LLM. Thus, it has reduced computational costs and can be applied to large-scale real-world tasks [18].

Chain of Thought is an enhanced prompting technique designed to improve the performance of LLMS on complex reasoning tasks, such as arithmetic reasoning, common sense reasoning and symbolic reasoning [95]. There are two main settings of how CoT is used within ICL, namely few-shot and zero-shot settings. Few-shot CoT incorporates intermediate reasoning steps leading to the final output into each demonstration. For example, instead of $(input, output)$, a chain of thoughts is added $(input, CoT, output)$ in the form of reasoning steps. There is much research on CoT prompt design, demonstration choices, and extended CoT strategies using multiple reasoning paths, such as self-consistency, which combines ensemble strategies with CoT prompt-

¹ Originally ICL was defined to be a combination of task description and demonstration examples [5], whereby either component is dispensable. Following a possible usage of ICL solely consisting of on component where textual task description and/or demonstrations is prompted in natural language as input to the llm

ing. A limitation of few-shot CoT is the need for annotated CoT demonstrations[105].

Unlike Few-Shot CoT, Zero-Shot CoT does not rely on annotated CoT datasets. Zero-Shot CoT uses a multi-stage ICL that formats a prompt for each stage. First proposed in [45], the LLM is prompted with a reasoning extraction prompt simply by appending *"Let's think step by step"*, followed by an answer extraction stage e.g. *"Therefore, the answer is"* to derive the final output.

2.4 RELATED WORK

So far a detailed overview of simulation research in the area of task-oriented dialogue was given. The research of simulation is closely related to the research of TODS and if its a pipeline approach its individual components due to its architectural setup and evaluation methods. Therefore, a brief excursion into recent developments in DST and TODS, which also investigate the ICL capabilities of LLMs, is presented.

Hu et al. [33] and Heck et al. [28] both investigate the possible use of ICL for dialogue state tracking. Hu et al. reformulate DST as a text-to-SQL problem using a tabular description of the domain. Contribution 1: Reformulate DST as a text-to-SQL task + tabular description of the ntology in the prompt Codex, GPT-Neo Code Gen as LLms pre-trained with code. 2. dialogue state in context representation instead of full convo history, more efficient and better suited to domain changes. 3. few shot scenario similarity score as selection strategy, trained to score for selecting matches based on dialogue state changes.

Heck et al. [28] answer the question *"Does ChatGPT solve the problem of zero-shot DST?"* by showing that the design of intuitive nl-prompts is sufficient to achieve state-of-the-art performance with ChatGPT. Their investigative approach to zero-shot DST with ChatGPT decodes dialogue state updates with a general-purpose model, without parameter updates, so they do not employ data analysis or cross-task transfer learning. Unlike Hu et al. they only allow natural language explanations of what the model is supposed to do, without allowing the provision of examples or formal task definitions.

Hudecek et. al. [36] use the ICL capabilities of LLMs to complete multi-turn tasks and interact with external databases in the context of established TOD benchmarks. Their proposed pipeline builds an initial prompt based on user input and context examples, if available, and then uses this initial prompt to generate the current dialogue state using an LLM state tracker. Based on this, they retrieve database results and construct another prompt which is used to generate the system utterance using an LLM as an NLG component. The

state recognition is decomposed into a two-stage prompt, namely a domain recognition prompt and a subsequent state tracking prompt.

During the writing of this thesis, two papers were published that build on the same research idea and thus have a significant influence on the present work. Both Terragni et al. [85] and Davidson et al. [14] presented a user simulation for task-oriented dialogue systems using large language models and their in-context learning capabilities. Because of the similarity, a detailed description of these papers follows.

Terragni et al. [85] were the first researchers to use the ICL capabilities of LLMs to build a user simulator. To generate diverse utterances, the presented User Simulator uses a few-shot ICL approach and is evaluated using goal-success metrics, diversity metrics, and manual qualitative analysis. The underlying experimental setup is based on the Convlab-2 framework [108], which provides models and evaluation tools for the development of task-oriented dialogue systems and simulators. The evaluation is based on the MultiWOZ 2.1 dataset [20] and the authors chose two variants of the agenda-based user simulator by Schatzmann et al. [72] as a baseline, with templated nlg and with data-driven nlg based on Wen et al. [96].

Figure 2.8 shows the architecture sketch of Terragni et al. [85]. A core

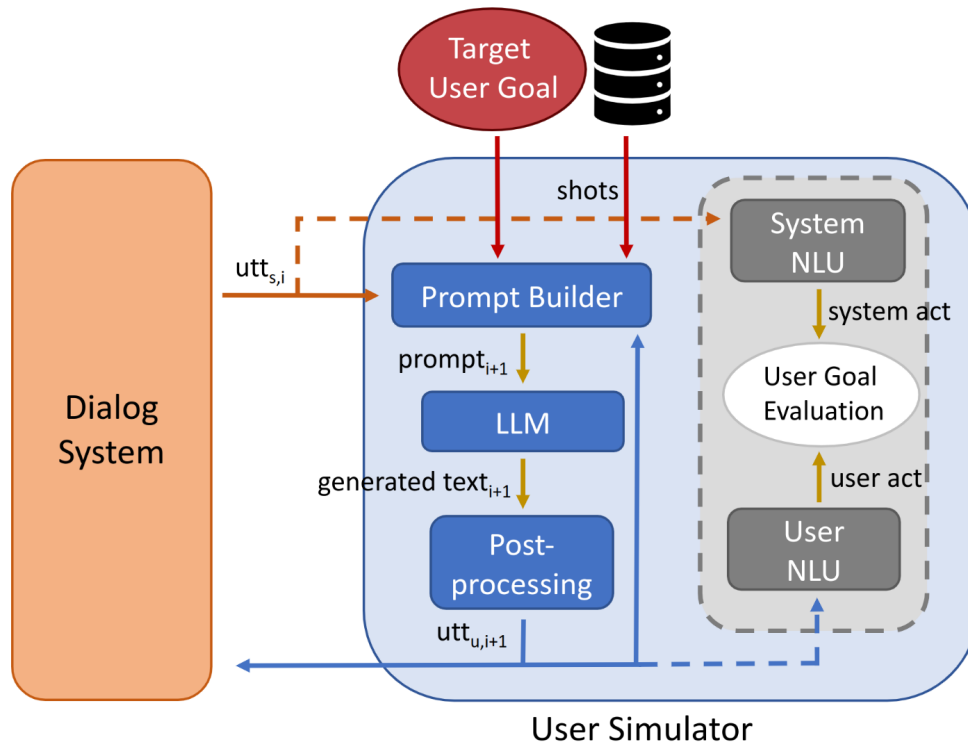


Figure 2.8: Architecture of the ICL-based User Simulator of Terragni et al. [85].

component is the prompt builder, which generates prompts consisting of an optional instruction, a target user goal, limited demonstration examples

(goal and associated dialogue), and the conversation history. The prompt is then used by an LLM. The authors experimented with three different LLMs, namely Llama [87], Flan-T5 [12] and GPT-3.5. A post-processing component then extracts the user utterance from the output of the LLMs. The user simulator includes an evaluation component that tracks goal fulfilment and evaluates system interaction. As a TODS, the authors used a pipeline-based task-oriented dialogue system to interact with their prompt-based user simulator in their experiments. The chosen TODS consists of a fine-tuned BERT [17] NLU component, rule-based dialogue state tracking and policy learning, and a template-based natural language generation component from Convlab-2 [108].

The work of Terragni et al. [85] is important because it lays the foundation for ICL-based user simulation as a viable simulation option and identifies its current limitations. An important contribution is the prompt tuning experiments applied to the task of user simulation, e.g. instruction formatting, shot-sampling strategies and demonstration formatting. In addition, the error analysis highlights the importance of adaptation in the form of instruction tuning, as Llama, a base model without further adaptation, performs worse than the other LLMs that undergo instruction-based fine-tuning. Furthermore, their error analysis showed that system-side NLU misclassifications and corresponding "wrong" system utterances can lead the LLM-based simulator to prematurely terminate the dialogue. On the user side, Terragni et al. [85] highlighted failures of the user simulator to mention all the requirements of the user goal and the occurrence of hallucinations and degenerated text by the LLM.

As a pioneering approach to ICL-based user simulation, the work of Terragni et al. [85] has implications for the present work. The architecture combines an end-to-end user simulator with an evaluation component, making it possible to evaluate a TODS while simulating a user integrating a DST component, implying the need for an evaluation component of the user simulator. The prompt experiments can be used as a basis for extended experiments with other LLMs and other ICL strategies. Finally, the results of the error investigation imply possible improvements for ICL-based user simulators for future work.

This thesis differs from the work of Terragni et al. [85] in the following respects, firstly this work conceptualises a broader range of prompting strategies in terms of instruction formatting, demonstration design and through the implementation of reasoning, planning and ensemble strategies and a broader range of LLMs. These strategies address the limitations of the simulator in goal completion and hallucination. In addition, the evaluation uses a new version of ConvLab, namely ConvLab-3 [109], as a framework. Finally, a proposed DST component is conceptualised, which also uses an ICL-based approach, making the evaluation easier to adapt to new domains.

Davidson et al. [14] also proposed a user simulator using LLMs and their ICL capabilities. Unlike previous work, they did not focus on maximising the goal success rate, but instead aimed to achieve success rates similar to those observed in human interaction with TODS systems and different utterances. Using this approach, they were able to achieve effective results, particularly on single-intent conversational goals, while interacting with multiple TODS and generating lexically and syntactically diverse output. In addition, they created a dataset of interactions between the same TOD systems and humans, providing a baseline for comparison between the user simulator and crowdsourced workers. In addition to this baseline, they also used other user simulators, namely MetaSim and Convlab2 TUS, for comparison.

Figure 2.9 illustrates the system design proposed by Davidson et al. [14].

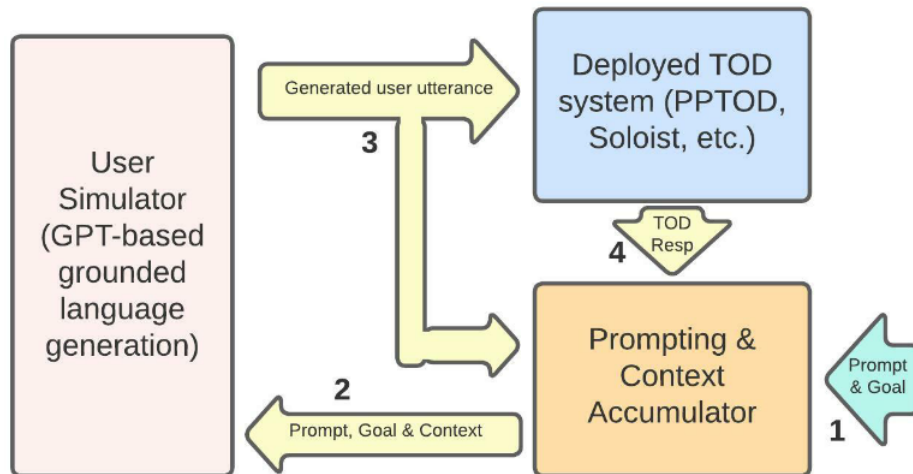


Figure 2: Interactive design for testing a TOD system with our proposed user simulator.

Figure 2.9: Architecture of the ICL-based User Simulator of Davidson et al. [14].

It consists of a context accumulator that formats the demonstration examples, the current conversational goal prompt, and the dialogue history. The experiments were conducted using the MultiWOZ 2.2 dataset [102]. The generated prompt is then used to query an LLM. The authors based their choice of LLM on the open source Huggingface Transformers Hub [99] considering GPTJ-6B [93], AlexaTM-20B [79] and GPTNeoX-20B [4] as well as the closed source GPT-3 davinci [5] for their initial trials, but ultimately decided to use GPTNeoX-20B for all subsequent experiments. Finally, Davidson et al. chose different TODS as interlocutors (PPTOD [81] and SOLOIST [67]) for their

user simulator component.

The work of Davidson et al. [14] argues for a focus on developing user simulators with similar success rates to real users by using a human2bot dataset, in contrast to previous work on user simulation that questions the importance of maximising goal success. Their experiments and quantitative evaluation show promising results for simpler goals consisting of fewer user intents. For more complex user goals, their human evaluation shows limitations due to premature dialogue termination, conversation loops, and LLM hallucination, which is very similar to the human evaluation of Terragni et al. [85].

In addition, Davidson et al. [14] point to potential improvements in prompt design to address existing limitations, which have implications for this thesis and were therefore considered. For one, combining the generated user utterance with the addressed subgoal or providing belief state information of a DST for additional grounding addressing hallucination and premature termination problems was considered. Finally, recasting the dialogue as a chain-of-thought reasoning problem was proposed to take advantage of the reasoning properties of LLMS.

The publication of the work of Terragni et al. [85] and Davidson et al. [14] emphasises the gap in research of utilising LLMs emerging ICL capabilities for task-oriented user simulation. These approaches demonstrate the potential for user simulation in a predominantly few-shot setting, evaluating the approach on the multiwoz domain through automatic success and diversity evaluation and a human qualitative analysis of common errors. Both highlight the problematic nature of success metric evaluation as the interaction is heavily influenced by the TODS conversing with the user simulator. They identified common errors in the ICL-based approach, consisting of potential hallucination and premature dialogue termination before the prompted user goal is achieved.

The analysis of the work of Davidson et al. [14] and Terragni et al. [85] showed that the comparison of ICL-based user simulators relies heavily on the prompting method used and the underlying Large Language Model, which is difficult to describe objectively due to its unstructured natural language nature. In order to increase the comparability of ICL-based user simulators and different strategies, a taxonomy must be applied that allows a meaningful comparison of the strategies used, even if the source code is not available. Therefore, in an effort to address this research gap, the recently proposed TELeR taxonomy [71] was analysed to be used to annotate the prompting methods for this complex task.

To address the errors that occurred in both previous works, several prompting strategies were conceptualised that could address specific limitations

that occurred in both dialogue systems, such as premature dialogue termination. In addition, the instruction-following capabilities of LLMs and the zero-shot setting of ICL-based user simulation were not the focus of the previous work, which overwhelmingly focused on the application of few-shot prompting. Therefore, this work focused on zero-shot prompting methods and the potential use of reasoning and planning strategies to address the aforementioned recurring errors. In order to compare the results of zero-shot prompting, few-shot prompting was included in the evaluation.

PROPOSED ICL-BASED USER SIMULATOR

As seen in the background and related work chapters, task-oriented dialogue systems are an active area of research. Recent advances in LLMs and their ICL capabilities have also had an impact on this line of research, as several researchers have attempted to apply ICL methods to the domain of task-oriented dialogue systems and simulations. The first attempts of ICL-based USs were published by Terragni et al. [85] and Davidson et al. [14] during the creation of this work, which had an impact on design decisions and emphasises the novelty and active research interest of this topic.

The ICL-based approach was initiated by the lack of training data in a task-oriented dialogue system and limited human resources to evaluate or improve the quality of a given dialogue system. In this low-resource setting, traditional user simulation approaches, which require either a large amount of human configuration or data to be pre-trained/fine-tuned on, were not applicable. Consequently, in addition to the research gap in ICL-based user simulators, there is a practical need for user simulators with high domain adaptability and low data requirements. The emerging capabilities of LLMs and the increasing range of out-of-context learning techniques and related research, which do not require further parameter updates, may therefore represent an opportunity for the field of TODSs and TODUSs.

This chapter begins with a brief description of the proposed ICL-US by presenting its basic design. Each component is then described in more detail, explaining the purpose of each component and the considerations that influenced the final design. We will first look at the data selection for the experimental setup, which is the basis for example demonstrations in the prompt. The core of the ICL-US, the prompt builder, and the corresponding conceptualised ICL strategies are then presented. This is followed by the model selection and sampling generation strategies, which have a major influence on the quality and diversity of the presented US. Post-processing is a logical step that acts as a guide to help the US meet the user's objectives by filtering the generated output. In addition, the Wizard of OZ testing approach is presented as a quick way to access immediate strategy value. Finally, an additional component of the architecture is discussed that could be addressed by future work.

The architecture of the proposed ICL-based user simulator is illustrated in 3.1 following an end-to-end design. The NLU and NLG components are incorporated via the LLM usage, where the dialogue is included in the prompt yielding as a way of tracking the state of the dialogue and deciding on the

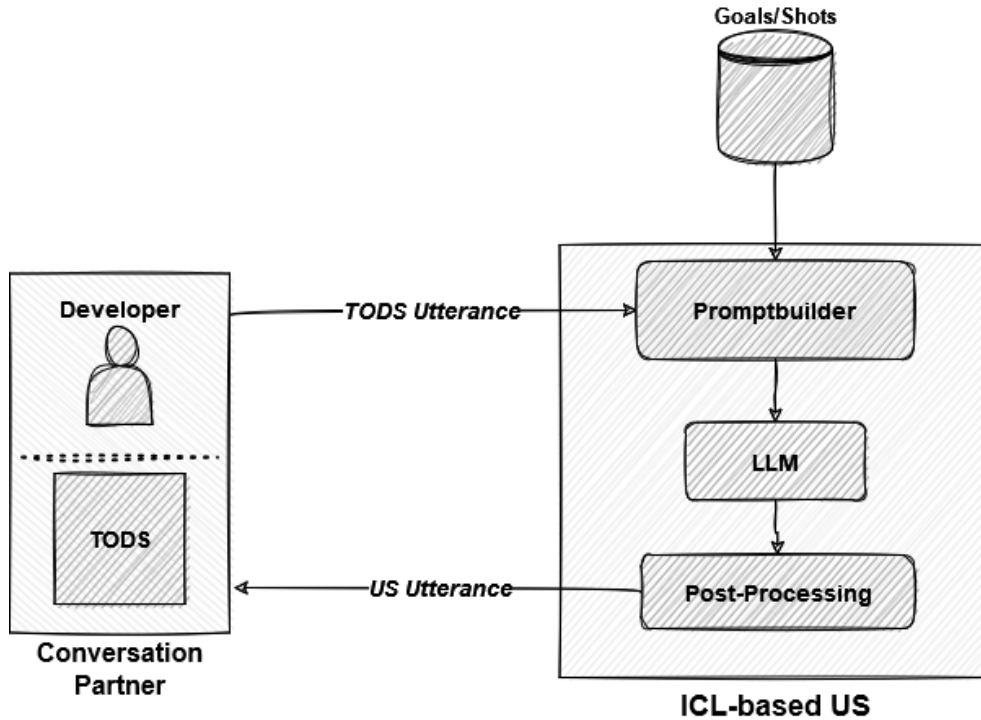


Figure 3.1: Proposed ICL-US end-to-end architecture.

next action based on that context. The prompt is generated in the prompt builder component, which creates the context for the LLM based on different ICL strategies and possible demonstration examples sampled from a dataset, if available. A simple post-processing component filters out excessively long user utterances and separates the output into sentences to ensure that only user utterances are sent to the interlocutor. This is a necessary step, as LLMs tend to try to complete the whole dialogue even when instructed to generate only the next user utterance; an example of generated and post-processed output is shown in the appendix .1. Finally, due to the interactive nature of this proposal, a target interlocutor in the form of a TODS or a human is included in the architecture to receive simulated user utterances and send system responses back to the US.

3.1 DATA SET SELECTION

The ICL capabilities of modern LLMs can be enhanced by presenting a series of high-quality demonstrations, known as few-shot prompting, as discussed in the Background chapter. A single demonstration is called a shot and consists of an input and a desired output. Few-shot learning approaches use multiple shots, hence the name. Therefore, in order to access ICL’s few-shot methods, it is necessary to select a dataset from which to sample demonstrations/shots.

Datasets can be classified by their characteristics, which can be helpful in understanding how different datasets can be used for a particular use case. [86]. A proposed set of characteristics, inspired by Thakkar et al. [86] and Ni et al. [63], are the following:

- Single or Multi turn
- Single Domain or Multi domain
- Data type (textual/spoken/multimodal)
- level of annotation
- language
- Data collection approach (*H2H/H2M/M2M*)

Unlike single-turn dialogues, multi-turn dialogues are dialogues that consist of more than one question and one answer/response. Similarly, a single domain dataset has only one target domain where the task orientated dialogue takes place as opposed to multi domain datasets where more than one task domain is present. The data type in which the dialogue representation is given can vary as there are different TODS that can handle different input types such as spoken input or images. Furthermore, the level of annotation of the datasets varies, some having additional annotations such as dialogue state, e.g. through an additional dialogue state or dialogue act information. Furthermore, the language availability attribute describes the target language(s).

Finally, there are three different approaches to data collection that refer to the actual collection of the data set, namely human-to-human (*H2H*), human-to-machine (*H2M*) or machine-to-machine (*M2M*). *H2H* is an approach where real conversations between two people, typically crowd workers, are instructed to carry out a dialogue. A well-established approach to *H2H* [6, 63, 86] is the Wizard of Oz (WOZ) technique introduced by Kelly et al. [44], which was modified by Wen et al. [97] to be suitable for crowd-sourcing, via the Amazon mechanical turk. In WOZ, two human interlocutors engage in a dialogue, one acting as the system without the knowledge of the other, who acts as the user. The *H2M* approach can be achieved through interaction between real users and a specific TODS, and finally the *M2M* approach refers to simulation approaches where two machines interact with each other.

The table 3.1 presents popular TOD datasets. From the publicly available datasets (see [63, 86] for a complete list), the following datasets were initially considered as potential data sources for demonstration retrieval in the few-shot experiments. Each dataset is multi-turn, multi-domain and represented in natural language, which were the basic selection criteria. These criteria should ensure that the demonstrations have a minimum level of complexity

Dataset Name	MultiWOZ[6]	SGD[70]	MetalWOZ[49]	Taskmaster-1[7]
Data collection Method	<i>H2H</i>	<i>M2M</i>	<i>H2H</i>	<i>H2H</i>
# Domains	7	17	47	6
# Dialogs	8.438	16.142	37.884	6168
AVG Turns per Dialog	16	20	11	23
# Unique Slots	25	214	<i>NA</i>	<i>NA</i>

Table 3.1: Comparison of potential Task-oriented Datasets.

and format.

For each initially considered dataset, the collection method, the number of domains, the number of dialogues in the training set, the average number of turns per dialogue and the number of unique slots in the annotated user goal were collected. These attributes and metadata make it easier to compare their usefulness with respect to the requirements of this work. As both the Taskmaster-1 and MetalWOZ datasets do not have any additional annotations that would be useful for evaluating dialogue state tracking, they were excluded from further investigation. The decision was therefore made to use MultiWOZ and SGD.

On the one hand, SGD outperforms the MultiWOZ dataset in most of the metrics over MultiWOZ, which increases the scale related challenges. Furthermore, its evaluation sets contain many slots that are not in the training set, thus helping to model the performance of unseen services. On the other hand, SGD uses a *M2M* bootstrapping approach for data collection. Here, dialogue simulators interact with a service configuration defined by the developers to generate dialogue sketches[2, 70]. These dialogue sketches are then paraphrased using templates in a first step and paraphrased by crowd workers in a second step. On the contrary, the MultiWOZ dataset follows the WOZ technique and is generated by real users, therefore in this work the MultiWOZ dataset is considered more appropriate as it is closer to real user interaction. Consequently, the MultiWOZ dataset was chosen for the rest of the work.

The wide use as a benchmark and popularity of the MultiWOZ dataset [6] (also referred to as MultiWOZ 2.0) can be seen in its various improvements. MultiWOZ 2.1 [20] reduced noise and combined the follow-up work of Lee et al. [50] with the original dataset of user dialogue acts and multiple slot descriptions per dialogue state slot. This was followed by MultiWOZ 2.2 [102] where further corrections were made, the ontology was redefined and a slot span annotation was added. Another version, MultiWOZ 2.3 [27], added co-reference features and unfiy annotations of dialogue acts and dialogues. Even after several updates, errors remained in the dataset, so another MultiWOZ 2.4 [101] was released, fixing incorrect and inconsistent annotations

and refining the annotations in the validation and test sets. Other researchers added multilingualism to the dataset in MUlti²WOZ, while another paper added multiple users to the dataset [40]. It is also included in the ConvLab Framework₁₋₃ [50, 108, 109], which is an evaluation framework for TODS/-TODUS research and is also used in the implementation of the proposed US.

3.2 PROMPBUILDING AND INCONTEXT-LEARNING

The key component of the ICL is the prompt design component as it enables the proposed US to utilise the emergent abilities of an LLM. Before exploring the different ICL strategies, a brief description of the basic prompt design and the design principles followed are presented below. This is followed by a more detailed description of the zero-shot prompting approach used, which does not rely on contextual data. This is followed by the Few Shot prompting, where demonstrations are presented and different formatting and sampling strategies are described. Finally, the strategies for advanced reasoning and planning prompting techniques are conceptualised for the task of user simulation.

Prompt engineering refers to the manual process of designing an appropriate prompt. The design of a prompt determines how effectively it can elicit the ability of a given LLM to successfully complete a task. There has been a great deal of academic and practical research on this topic, analysing best practice and the key components of a well-designed prompt. The prompt design in this paper follows these recommendations and therefore uses a combination (depending on the strategy used) of the four key elements:

1. **Instruction**, a specific task description or instruction the model shall perform
2. **Demonstrations**, demonstrations of the task the shall perform
3. **Context**, additional context that can guide the model to better responses
4. **Input Data**, the input or question that is of interest

Figure 3.2 illustrates a basic prompt template containing the four key elements, although the order may vary and the presence of each element depends on the prompting strategy chosen, which will be discussed in the next sections. As mentioned above, there are a number of design recommendations that take into account the design process as well as the specific elements.

In general, prompt design is an iterative process that requires a lot of experimentation to achieve optimal results. Therefore, it is recommended to

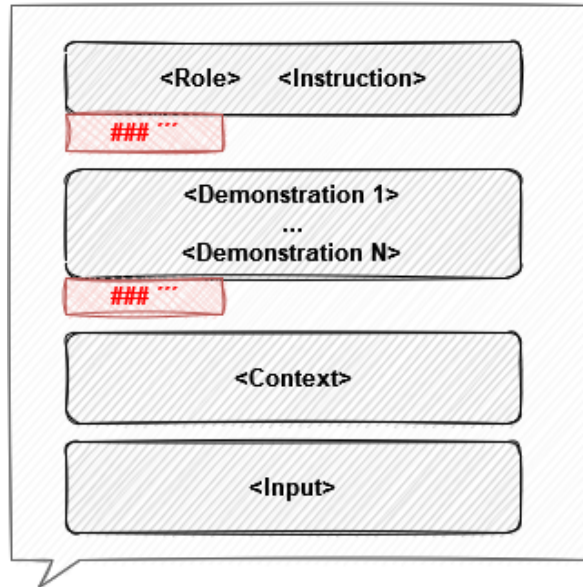


Figure 3.2: A basic prompt and its typical elements

start simple and add more elements and context to the prompt. Another option is to divide a complex task into simpler subtasks to avoid too much complexity in the prompt design process at the beginning. Therefore, the ICL strategies explored start with a simple prompt with minimal elements and increase the complexity following these best prompt practices.

One design principle is to use a model-friendly format. Since LLMs are pre-trained on specially constructed datasets, there are prompt formats that can help the LLM better understand the [105] command. For example, the OpenAI documentation [65] suggests the use of delimiters such as ### or "''''''", as shown in figure 3.2, to separate the elements of a prompt, which enhances the LLM's ability to execute the instruction. Furthermore, most existing LLMs perform tasks better when prompted in English, so it is useful to use English instructions to solve complex tasks based on machine translation.

3.2.1 Proposed TELeR-RESPONDeR Taxonomy for Complex Prompts

To increase the comparability of the proposed prompts in general and in relation to the existing work of Terragni et al. [85] and Davidson et al. [14] the application of the TELeR taxonomy [71] was considered. This taxonomy addresses design decisions for complex prompt tasks, allowing for a meaningful comparison of the specific categories used in different studies. The visualisation of the TELeR Prompt Taxonomy by Santu et al. [71] is shown in Figure 3.3.

The taxonomy represents each prompt as a combination of Directive and

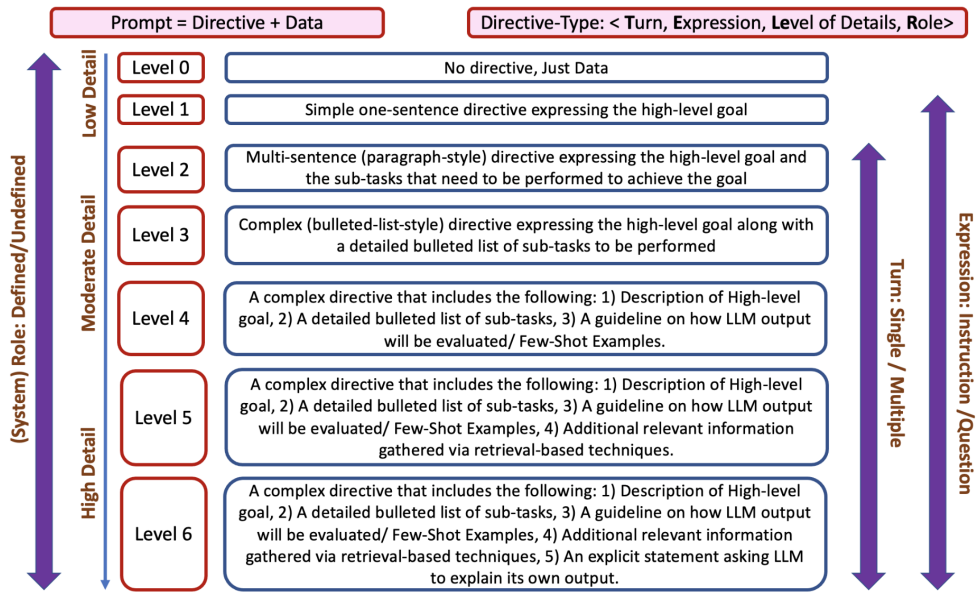


Figure 3.3: The detailed illustration from the four dimensions of the TeLER-Taxonomy from Santu et al. [71]

Data. Santu et al. [71] assume, that the data is fixed for a given goal arguing that the difference between two prompts originates from the directive alone. Furthermore they categorize prompts alongside four dimensions:

1. **Turn**, number of turns while prompting LLMs to perform the task. Can either be single or multi-turn.
2. **Expression**, style of the directive, can either be question-style or instruction-style.
3. **Level of Detail**, ranging from 0-6 see figure 3.3 for a detailed description.
4. **Role**, describes if a role is defined or undefined.

In this thesis, the directive is interpreted as the instruction element of the prompt template. Furthermore, the data referred to by Santu et al. [71] is interpreted as the context element of the prompt template. In the case of the interactive user simulation, the context itself changes with each iteration, and thus the generation of the llm itself influences the next prompt it receives in the form of the dialogue history contained in the context element. Consequently, in the present work, the data is not necessarily fixed for a given goal, thus deviating from the proposed fixed attribute of the data.

As extensions are encouraged, I hereby present a modification of the TeLER taxonomy called TeLER-RESPONDeR. This modification splits the last 3 levels of detail into separate dimensions, as the properties of the levels are

independent of each other and do not need to be used together. For example, you could use self-adjustment without specifying an output description, and so on. The content of level four is divided into two dimensions: *Demonstrations* and explicit *Output Notation*, as these attributes are very different from each other. The additional attribute of retrieval-based techniques in level five is modelled in the dimension *Retrieval*. Finally, the new attribute self-justification in level 6 is also represented in a separate dimension *Self-justification*. Finally, *Reasoning*, *Planning* and *Ensemble* prompts are not directly addressed in the TELeR taxonomy, so in order to have a more complete prompt taxonomy they are also included, resulting in an additional 7 dimensions. This increases the complexity of the taxonomy, but at the same time allows a more flexible and accurate prompt classification.

As we introduced 7 more dimensions to the TELeR taxonomy we modify the acronym by the 7 factors, i.e: **TELeR-RESPONDeR** - Reasoning, Ensemble, Self-Justification, Planning, Output Notation, Demonstration and Retrieval.

An efficient notation for the 10-dimensional TELeR-RESPONDeR taxonomy is also introduced to allow easy comparison of prompts. For dimensions that can be expressed by integers, the following values are possible: $T = 0, \dots, N$ zero to N prompt turns, $Le = 0, \dots, 3$, $D = 0, \dots, k$ demonstration or shots. Except for the expression dimension, which can be either Q (question style) or I (instruction style), 0 if none is given. All other dimensions can be expressed as Boolean, where 0(False) indicates that the dimensional feature is not present and 1(True) indicates that it is present. E.g. $T = 1, E = I, Le = 1, R_1 = 0 - R_2 = 0, E = 0, S = 0, P = 0, ON = 1, De = 1, R_3 = 0$ would be a single turn, instruction style prompt with a simple one sentence instruction expressing the high level goal with 1 demonstration and a specified output definition. Some dimensions could also be specified through the technique name, for example one could directly name the reasoning technique instead of just checking if reasoning was used, but in order not to add to the complexity of the notation the simple boolean approach will be used throughout this thesis.

Below, the TELeR-RESPONDeR taxonomy is applied to the work of Davidson et al. [14] and Terragni et al. [85]. Davidson et al. [14] relied solely on the demonstration element and the context element¹ of the prompt building blocks in their final experiments. The first prompt experiments were not published. Therefore it is $T = N, E = 0, Le = 0, R = 0 - R = 0, E = 0, S = 0, P = 0, ON = 0, De = 2, R = 0$ as 2 demonstrations are used per prompt and a context consisting of a user goal and an open-ended prompt.

In contrast, Terragni et al. [85] published their initial prompt experiments before deciding on a final prompt. These prompts differed in the number of

¹ It could be argued that the user goal provided can be seen as an instruction, but in comparison to Terragni et al. [85] a clear instruction was used. Therefore, this paper interprets it as context.

demonstration examples $De = 0, \dots, 6$, the level of detail $Le = 0, \dots, 2$ and the effect of adding a role $R_1 = 0, 1$. The final prompt they used for their evaluation is $T \geq 10, E = I, Le = 2, R_1 = 1 - R_2 = 0, E = 0, S = 0, P = 0, ON = 0, De = 2, R_3 = 1$. The dialogue was limited to a maximum of 20 turns, so $T \geq 10$ due to alternating turns. They also used the instruction style to describe the goal in a level 2 way. They chose to give two demonstration examples ($De = 2$) where the retrieval was either random or based on Jaccard sampling ($R_3 = 1$).

3.2.2 Design of the Basic Prompt Elements for Zero- and Few-Shot Strategies

Five different instruction element designs are described in the table 3.2, with the complexity and detail of the prompts increasing from top to bottom. The proposed formats address 4 dimensions of the TELeR-RESPONDeR taxonomy, the level of detail ($Le = [0, 1, 2]$), the self-justification ($S = [0, 1]$) and the output notation ($ON = [0, 1]$). In addition, general design choices for the instruction element are that the directive is written as instruction-style ($E = I$), that the goal is expressed clearly and without ambiguity, and that the instruction focuses only on what the LLMs should do rather than what they should not do.

The level of detail in the designed instruction element formats starts from as simple as possible to highly detailed traversing all levels of details of the taxonomy. Starting with an instruction format, that consists solely on a role description ($Le = 0$), followed by the next stage a simple one sentence high-level goal description ($Le = 1$). Then the first detailed design, where the task is described in a paragraph style form ($Le = 2$), is set up. After that the next stage adds additional information about the expected demonstrations, context and input (Le_3), which should describe the prompt structure to the LLM for more clarity. Finally response constraints are introduced in order to guide the model to generate a specific output notation ($ON = True$).

The design of the demonstration element is depicted in figure ?? and consists of two components: the user goal and the pertaining conversation. This building block can be repeated several times from $0, \dots, N$ building, if used, k-shots. It addresses the demonstration dimension (De) of the taxonomy and if a retrieval technique is used for shot selection also the retrieval dimension (R_3). Yet each demonstration i is sampled from an existing data set e.g. the MultiWOZ dataset described in section 3.1. The conversation examples consist of alternating turns between the system and the user.

As the format of the demonstration examples can influence the quality of

Instruction Element Design 1:
<i>No Instruction.</i>
Instruction Element Design 2:
<p><i>One Sentence describing high-lvl goal.</i></p> <p>Your task is to generate the next <role> utterance.</p>
Instruction Element Design 3:
<p><i>Multi-sentence instruction describing high-lvl goal and the sub tasks to achieve the goal.</i></p> <p>Your task is to generate the next <role> utterance.</p> <p>Fulfill all <i>REQUIREMENTS</i> in the given order. Stick to the sequence of subgoals in the requirement and refer to one subgoal at a time. If the <i>ASSISTANT</i> responds with an unexpected answer, try to rephrase your request until you are certain the <i>ASSISTANT</i> understands you and provides all the correct answers.</p>
Instruction Element Design 4:
<p><i>Multi-sentence instruction describing high-lvl goal and the sub tasks to achieve the goal. And the Output Notation the Model should follow.</i></p> <p>Your task is to generate the next <role> utterance.</p> <p>Fulfill all <i>REQUIREMENTS</i> in the given order. Stick to the sequence of subgoals in the requirement and refer to one subgoal at a time. If the <i>ASSISTANT</i> responds with an unexpected answer, try to rephrase your request until you are certain the <i>ASSISTANT</i> understands you and provides all the correct answers.</p> <p>A good output should be precise, contain diverse vocabulary and be one sentence long. The conversation will be evaluated based on whether all information has been addressed by the <i>ASSISTANT</i>. This should be ensured by your answers. Start your answer with '<role>: '.</p>
Instruction Element Design 5:
<p><i>Multi-sentence instruction describing high-lvl goal and the sub tasks to achieve the goal. An explicit statement to justify/explain the generated output. And the Output Notation the Model should follow.</i></p> <p>Your task is to generate the next <role> utterance.</p> <p>Fulfill all <i>REQUIREMENTS</i> in the given order. Stick to the sequence of subgoals in the requirement and refer to one subgoal at a time. If the <i>ASSISTANT</i> responds with an unexpected answer, try to rephrase your request until you are certain the <i>ASSISTANT</i> understands you and provides all the correct answers.</p> <p>Provide justification for your generated user utterance/response in detail by explaining why your response contains certain information and discards other information.</p> <p>A good output should be precise, contain diverse vocabulary and be one sentence long. The conversation will be evaluated based on whether all information has been addressed by the <i>ASSISTANT</i>. This should be ensured by your answers. Start your answer with '<role>: '.</p>

Table 3.2: Proposed instruction element designs increasing in complexity.

Goal Format 1:
<p><i>Dictionary Style / Json.</i></p> <pre>{ 'attraction': { 'info': { 'name': 'cineworld cinema' }, 'reqt': { 'postcode': '?', 'phone': '?', 'area': '?' } } }</pre>
Goal Format 2:
<p><i>Textual paragraph style description.</i></p> <p>You are excited about seeing local tourist attractions. You are looking for a particular attraction. Its name is called cineworld cinema. Once you find an attraction, make sure you get postcode, phone number, area.</p>

Table 3.3: Demonstration Elements increasing in naturalness.

the generation of an LLM, two variations of user goal formatting are explored with increasing naturalness. The first is a logical data format, in this case the json format, although other human-readable formats such as yaml would also be possible. The second form, a natural language paragraph format, is used as a user goal representation. In addition to exploring the effect of different demonstration formats, this approach explored the application potential of different data sources for fewshot sampling. As such in some situations only a json format would be available for prompting the LLM.

The context element is similar to the demonstration element, but only consists of the first part, the user goal. This context defines the actual user simulator goal and its requirements. The formatting of the context goal corresponds to the two variations of the user goal in the table 3.3. If the demonstration element is also used in the prompt, then the format of the context element goal should be the same as the example goals to allow efficient learning of the context.

Next the input prompt is described. As the User simulator interactively communicates with a TODS, the input itself changes based on the utterances from the user simulator itself as well as the TODS. The input could end with an empty utterance of the user, signaling the LLM to generate the next user utterance or a new line. The proposed dialog initiation has two variants, one where the user starts the conversation and one where the system starts the conversation with a fixed utterance e.g. "How can I help you?". In the later evaluation the fixed utterance was used. Another way of allowing the user simulator to take on a specific persona is the role prompting approach. The ICL-US has the advantage of a flexible role design that can be prepended to the instruction, but other insertions would also be applicable. To explore the possibilities of giving the user simulator a persona, four different roles were designed for role prompting. These consisted of a tourist who is more domain dependent, a user simulator and two different customers with con-

trasting personality descriptions. By providing a more detailed personality description, the user simulator can be adapted to mimic a specific target audience, which was taken into account in the evaluation of the proposed simulator.

3.2.3 Zero-Shot and Few-Shot Setup

Following the principle of increasing prompting complexity, this thesis starts with one of the simplest forms of ICL learning - Zero-Shot Prompting. As the name suggests, in a zero-shot prompt there are no shots, so the prompt relies solely on the knowledge base of the LLM and the remaining elements: Instruction, Context and Input and their design. The design of these elements is described above and also increases in complexity.

The structure of the designed zero-shot prompt follows the basic structure depicted in figure 3.2 without demonstrations. Delimiters are used for a clear separation of each element. The exact order of the elements is fixed for all Zero-shot prompting experiments later conducted in this work. Through manual testing this ordering of elements appeared to be the most reliable one. Another approach was to first introduce context, input and end with the instruction element, yet this lead to more inconsistency in the utterance generation.

The use of demonstration examples can increase the performance achieved by prompting, so few-shot prompting is also applied to the TOD-US task in order to have a broader scope of evaluation. The design of the few-shot prompting approach is illustrated in Figure 3.4. The structure is similar to that of zero-shot prompting, but the additional k-demonstration elements or shots are incorporated into the prompt.

The information for the context and the demonstrations is taken from the

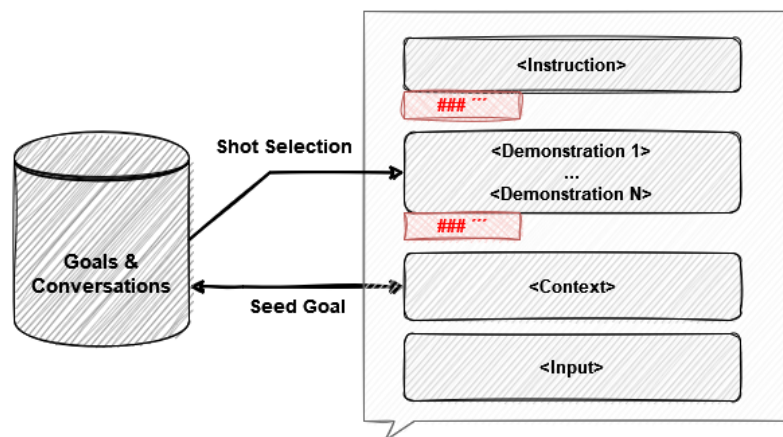


Figure 3.4: Typical structure of a few-shot prompt and the shot sampling strategy.

same dataset (see section 3.1). As described in 3.2.2, each demonstration con-

sists of a user goal and its dialogue. In contrast to the demonstrations, the context consists of only one user goal g_c without the demonstration dialogue, as this is generated by the interactive simulation. Consequently there are tuples of user goal and associated dialogue for the demonstration element and a user goal for the context element. Therefore $k + 1$ user goals and k are dialogs to be sampled.

In this work, three two sampling strategies have been explored, starting with Jaccard similarity sampling based on the logical goal description and a vector-based similarity sampling based on the natural language goal description. Given the initial context goal g_c of the target dialogue that the interactive simulation aims at, similar demonstrations are sampled.

The first similarity-based sampling method computes the Jaccard similarity following Liu et al. [54] between the domain and slot sets of the context user goal g_c and a demonstration user goal g_d . Formally expressed, the Jaccard similarity $J(g_c, g_d)$ between the context goal g_c and a potential demonstration goal g_d can be defined by:

$$J(g_c, g_d) = \frac{|Domain_{g_c} \cap Domain_{g_d}|}{|Domain_{g_c} \cup Domain_{g_d}|} \cdot \frac{|slot_{g_c} \cap slot_{g_d}|}{|slot_{g_c} \cup slot_{g_d}|} \quad (3.1)$$

where $Domain_{g_i}$ and $Slot_{g_i}$ refer to the set of domains and slots of the goal g_i , and the cardinality of a set S is denoted by $|S|$. This Jaccard similarity is computed for g_c and each potential demonstration goal. When all similarity scores have been computed, the most similar k samples can be drawn from the dataset, where k is the chosen number of demonstration examples.

The second similarity search approach is based on the natural language user goal. Figure 3.5 shows the process of this similarity search strategy. In a first step, all textual goals are converted into embedding vector representations using an encoder. These vector representations are then indexed by a vector library framework. The FAISS framework [41] is used to enable efficient searching. This framework creates an index for each vector representation. These indices are then used to sample the k most similar representations that have the highest similarity score to the embedding of the context user goal g_c .

3.2.4 Reasoning, Planning and Ensemble Prompting Concepts

The prompting techniques used so far can give good results, but more sophisticated strategies could improve the performance of LLMs in complex common sense tasks. In addition, reasoning and planning techniques are be-

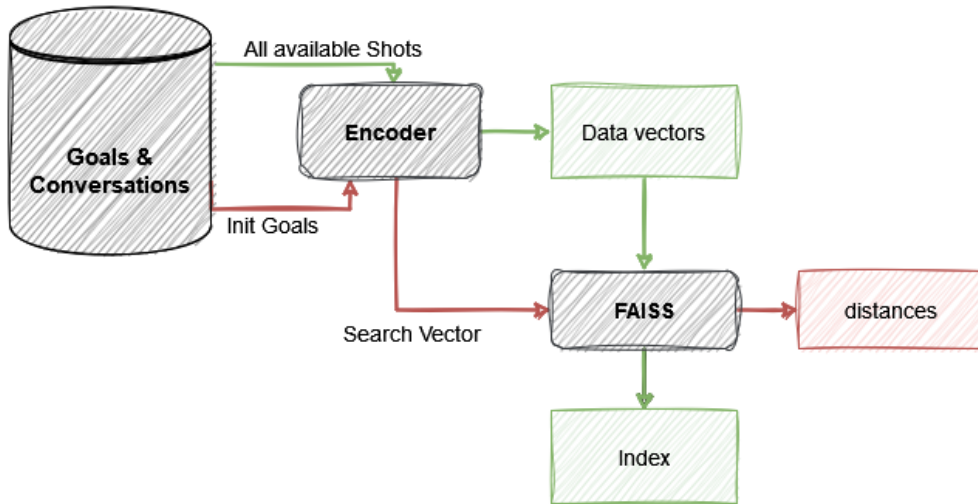


Figure 3.5: Flow chart of the embedding vector similarity search concept.

ing investigated. As these techniques mimic real human behaviour, they are expected to reduce the gap between user simulator and human responses.

The first reasoning technique investigated for application to the user simulation task is zero-shot chain of thought. Chain of Thought prompting in general provides benefits on Commonsense Reasoning problems by encouraging the LLM to generate a chain of thought. Since the Few-Shot CoT approach requires chain-of-thought demonstrations that are not readily available for the task of user simulation, only the Zero-Shot scenario was investigated.

The proposed zero-shot CoT design is illustrated in Figure 3.6. The design

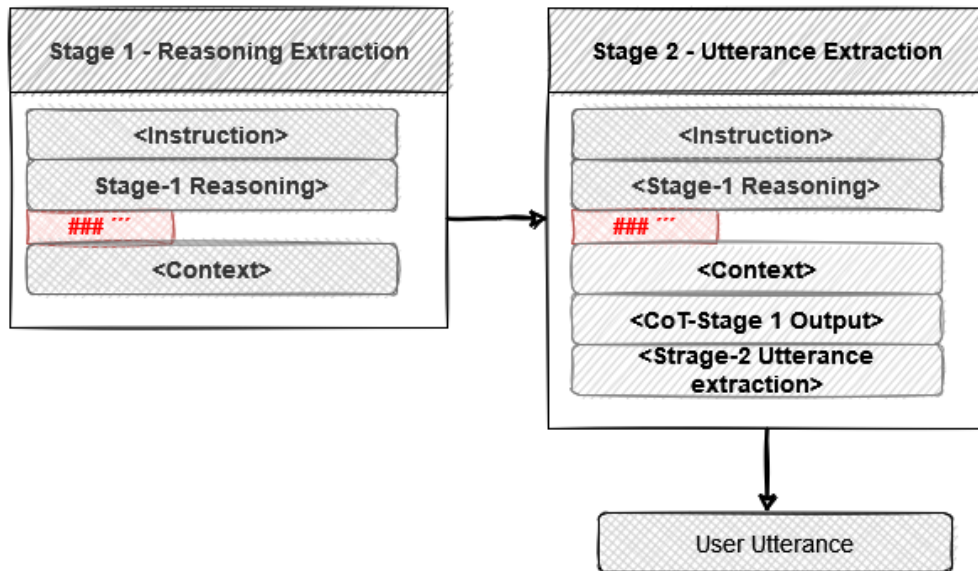


Figure 3.6: Sketch of the zero shot chain of thought two stage prompt design.

involves two separate prompt stages, following the original zero-shot CoT

approach [45]. The first stage focuses on reasoning extraction by explicitly encouraging the model to generate a chain of reasoning based on the contextual information. This reasoning extraction can be achieved, for example, by appending the words "Let's think step by step" to a basic zero-shot prompt. The example given is the most common way of extracting zero-shot reasoning, but other prompts have also been explored. The output of stage one is used to obtain a self-augmented prompt by appending the chain of thought to the input prompt itself.

The second stage is used in this case for user utterance extraction. Here the self-augmented prompt is used to prompt the model a second time. Now the LLM is asked to create the next user utterance based on the added chain of thoughts, resulting in the next utterance of the US. This process is repeated for each user turn until the conversation ends, so that the LLM is prompted a total of two times for each user turn.

The next prompting method examined is self-consistency prompting, the design for which is shown in Figure 3.7. Self-consistency is a modification of chain-of-thought prompting that often results in a significant improvement over the results of simple chain-of-thought prompting. The main idea is that instead of generating one chain of reasoning, self-consistency generates multiple chains, with the aim of increasing the consistency of the output, as the name suggests.

To sample a diverse set of reasoning paths, a requirement is that the LLM

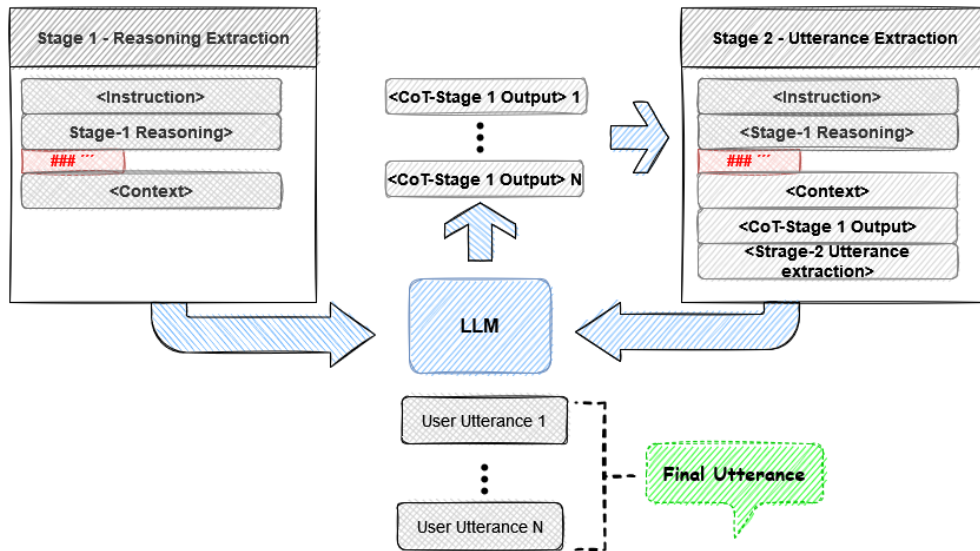


Figure 3.7: Sketch of the self-consistency process design.

uses a non-deterministic decoding strategy, e.g. greedy decoding would result in the same reasoning path, so no sample of diverse outputs would be generated. The decoding strategies and selection are described in more detail in the section 3.3. In the proposed design, the same stage of a zero-shot

CoT prompt is used N times to sample multiple outputs, acting as a form of self-ensemble on top of a single LLM. To extract the specific user utterance of each reasoning chain, the second stage zero-shot CoT is applied to each chain, resulting in a sample of N user utterances.

Finally, the most consistent user response among all generated user utterances is selected. Since the original majority voting approach is not directly applicable to the open-ended user simulation generation task, another ranking scheme has to be applied. Therefore, a generalised self-consistency score $GSC_{sim}(i)$ can be computed based on a similarity function for each generation i derived from the approach of Jain et al. [38]. The underlying self-consistency assumption here is that the most frequent response is assumed to be correct, in this case the prompt most similar to all other prompts may be chosen.

A similar approach to self-consistent prompting is to use a prompt ensemble instead of using the same prompt to generate multiple generations of text. The prompt ensemble design is shown in figure 3.8. There, N different prompt designs are used to generate N user utterances, and then, identical to the selection process of the proposed self-consistency design, the best response candidate is sampled by a generalised consistency score.

This prompt ensemble approach has the advantage that it does not rely on

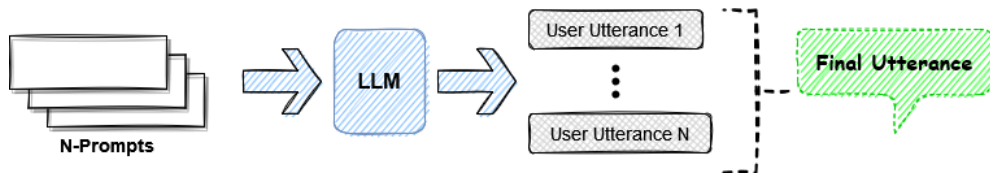


Figure 3.8: Sketch of the prompt ensemble process design.

a single prompt to generate a diverse set of user utterances, making it less dependent on the sophisticated design of a single ‘perfect’ prompt. Instead, multiple prompt design variations can be used as an ensemble, increasing independence, as small changes in a prompt can sometimes drastically alter the output generated.

The ensemble of prompts might be for example several variations of the zero-shot experiments, so a wide range of complexity in prompt design is included in the ensemble. In addition, all other proposed few-shot, reasoning and planning designs could have been added to the ensemble to keep the input prompts diverse. The last advanced prompting technique explored is called least to most prompting [107], which follows a planning approach. The main idea in this strategy is to break down a complex problem into a series of simpler sub-problems and then solve them one at a time. As the solution of each sub-problem is facilitated by the answers to previously solved problems, this technique is well suited to the sequential nature of interactive

user goal solving by the proposed simulator.

Figure 3.9 illustrates the design of the least to most prompting approach,

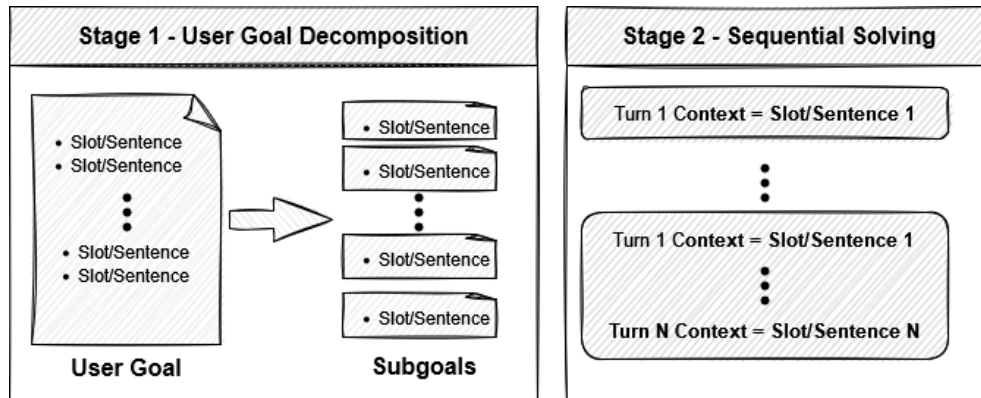


Figure 3.9: Sketch of the least to most prompting process design.

inspired by the original approach of Zhou et al. [107]. The designed process consists of two sequential steps:

1. **User Goal Decomposition.** The target user goal is decomposed into subgoals, the process of decomposition depends on the underlying user goal format.
2. **Sequential Subgoal Solving.** Sequential solving of each subgoal during the interactive dialogue between the US and the interlocutor TODS. The context of the target user goal starts with a subgoal. For each turn, another subgoal is added to the context until all goals are included in the context. At the same time, the user utterances addressing each subgoal in each turn of the dialogue are added to the conversation history. The responses to each subgoal itself serve as a demonstration example.

The aim of this iterative addition of sub-goals would be to steer the simulator to sequentially follow the requirements of the target user goal. As a result, limitations such as premature dialogue termination could potentially be reduced due to the reduced complexity of the user goal in the early conversation phase.

3.3 LLM SELECTION & GENERATION DECODING STRATEGIES

The choice of underlying LLM affects the natural language understanding and generation capabilities of the user simulator. As described in the background chapter 2, there is a growing number of open source and proprietary LLMs. The table 3.4 lists a selection of state-of-the-art LLMs with instruction-tuned versions that were considered. Another aspect is the presence of instruction tuning, as it can increase the in-context learning behaviour, so

Model	<i>GPT-3.5</i> [65]	<i>Llama2</i> [88]	<i>Mistral</i> [39]	<i>Falcon</i> [66]	<i>Flan-T5</i> [12]
Architecture Category	decoder-only	decoder-only	decoder-only	decoder-only	encoder-decoder
Publication date	2023	2023	2023	2023	2022
Availability	proprietary	open-source	open-source	open-source	open-source

Table 3.4: Selected LLMs for manual prompting trials.

instruction-tuned versions of LLMs were selected over basic models.

As emergent, prompt-based learning capabilities have been demonstrated primarily in decoder-only models, this type of architecture was chosen over encoder or encoder-decoder models. For comparison, the FLANT5 model was also tested. In addition, the approach of Terragni et al. [85] also included this model and achieved acceptable results, so it was also considered, although it is a decoder-only model.

Similar research for user simulators has yielded the best results with GPT3-5, which includes the LLM property. In addition, most of the published prompting best practices refer to OpenAI models, so prompting tuning should be particularly effective for this LLM as it is better documented.

Another important influence on the text generation quality of LLMs is the choice of decoding strategy. Decoding refers to the process of selecting output tokens to generate text without changing the values of trainable parameters. Different decoding methods may impose constraints or preferences on the sampling process, such as repetition reduction or coherence enhancement.

Decoding methods can be divided into two categories deterministic methods and stochastic methods. Deterministic techniques like greedy search and beam search generate text by choosing the text predicted word with the greatest probability as determined by the language model. As deterministic methods often lead to model degeneration for instance generating unnatural text with repetitions. Moreover human language in general tends to be less predictable and does not follow a distribution of high probability next words [32]. As a consequence deterministic methods were not applied to open text generation of user utterances in this work.

Stochastic Decoding methods address the issues of deterministic methods by incorporating randomness also called sampling during the decoding process. As these stochastic methods generate more human like texts, several stochastic decoding methods were tested for the user utterance generation task. The explored stochastic decoding methods include top-k sampling, nucleus sampling and contrastive search. Furthermore the influence of the temperature setting of the LLM was inspected.

The choice to use stochastic decoding methods is a design choice that might

lead to more diversity in the generated user utterances. Diverse and human like user interaction is one aim of the proposed User simulator, yet the diversity could lead to deviation of the user goal. As a result different decoding strategies lead to a trade off between diversity and the success rate. In the later evaluation a simple sampling was allowed to allow randomness and thus more naturalness in the generated utterances.

3.4 WIZARD OF OZ TESTING

The Wizard of OZ style dialogue design is shown in Figure 3.1. In this simple setup, the TODS is represented by a human in the loop, e.g. the prompt engineer who designed the prompts used. This approach allows rapid prototyping while simulating the quality of the dialogue through manual human evaluation. The replacement of the TODS element is a reduction in the complexity of the architecture and helps to understand the capabilities of LLMs to generate natural text utterances. naturalness of text generation. The prompt engineer receives immediate feedback on changes made to the prompt template being used and can investigate the influence of instructional design decisions. As a result, this setup can be used prior to conducting large-scale experiments or evaluations. In addition, because of the direct influence on the flow of the conversation, the prompt engineer can directly address specific types of conversations and directly test edge cases, which can increase the robustness of the US.

3.5 MODIFICATION VIA DIALOGUE STATE TRACKING COMPONENT

The basic architecture of the ICL-based-US can be extended with a Dialogue State Tracking component as shown in figure 3.10. This design serves a dual purpose, but is not strictly necessary for the proposed ICL-US to generate user utterances. Firstly, it is used to evaluate the success of the interactive dialogue between the US and the TODS. One goal of the US is to evaluate a given TODS, and although TODS often have their own DST component, an external success evaluation built into the US increases the independence of the TODS DST extractions. This avoids the case where a TODS evaluates itself, so an additional external DST component allows a more neutral evaluation. In addition, the dialogue state of the system side could be compared, allowing direct feedback on misclassifications.

A second advantage of incorporating a DST component into the ICL-US is the ability to include goal state tracking information in the prompt. This can be used to address limitations known from related work, such as premature termination or hallucination. This could be used in many ways, such as updating the user goal based on the current belief state, or splitting the user goal into subgoals and only adding the subgoals when the previous

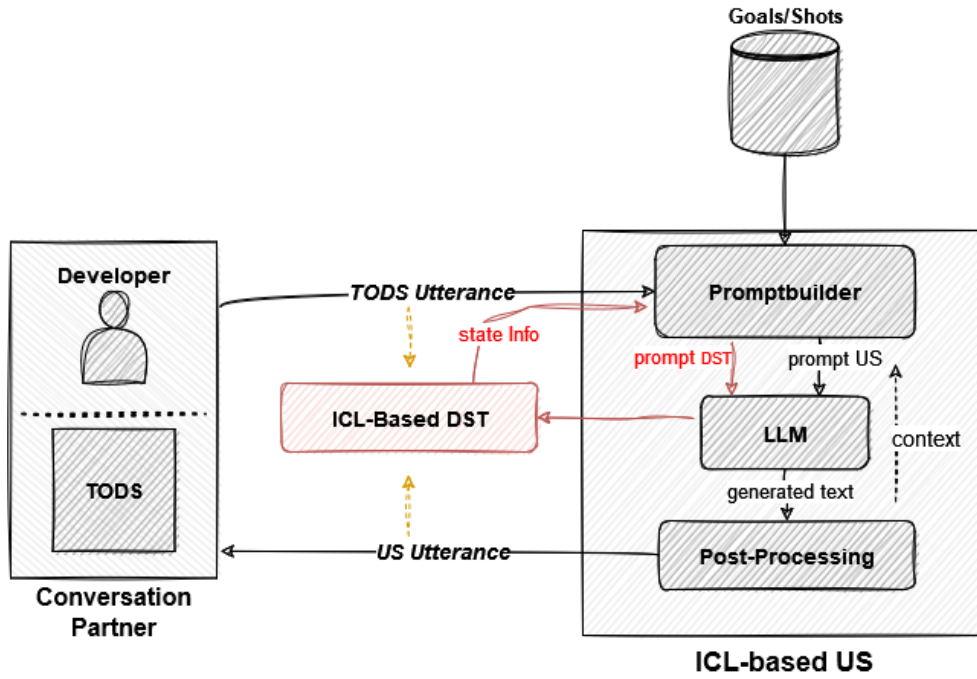


Figure 3.10: Sketch of a modification in the basic ICL-based US architecture adding DST component.

goal is completed. Furthermore, it could be used embedded in further planning and reasoning prompting techniques. LLM-based state tracking, even in a zero-shot fashion, has already been tested in [28] and shows potential for future research. Therefore, applying a different prompt design for state tracking to the same LLM, as illustrated in Figure 3.10, could be a viable option to further improve the proposed US.

EVALUATION

The evaluation of task-oriented human-computer interfaces is still an open field of research. From the variety of evaluation methods, as seen in the related work chapter 2.4, mainly approaches like cross-model evaluation, automatic evaluation and human evaluation are used to evaluate a proposed US. As this thesis focuses on the use of an ICL/prompting approach, which has significant variability due to its unstructured textual nature and variations, a mixture of automatic and human evaluation was conducted to investigate its performance. Consequently, both quantitative and qualitative evaluations were carried out.

The structure of this chapter begins with the description of the experimental setup, including the hardware used, the underlying framework, the architectural modification models, and the data used in the evaluation process. This is followed by a detailed presentation of the selected LLMs and the generation parameters used, the TODS and US baseline models, and the baseline data. The metrics for the quantitative evaluation are then described, consisting of dialogue success metrics and diversity metrics. This is followed by the application of these metrics to a selected set of ICL strategies, including Zero-Shot ICL strategies and a Few-Shot setup. The chapter concludes with the results of the quantitative evaluation of ICL strategies and a qualitative assessment of common errors in the simulation and evaluation process.

4.1 EXPERIMENTAL SETUP

The experiments were conducted using ConvLab-3 [109]. The framework provides data, models and an experimental toolkit for the evaluation and development of TODS and TOD-US. As the third version of ConvLab is the latest in the well-established ConvLab series, it was chosen over its predecessors. As mentioned in the previous chapter, the in-context learning strategies used were annotated with the proposed TELeR-RESPONDeR taxonomy (see table .3) to increase the comparability of the prompts used in the experiments. The actual text of the prompts was described in 3 and strategies. Zero-shot prompting gradually increases the complexity of the prompt, then role prompting is evaluated to test the possibility of adding a persona or role to the simulator. Finally, a few-shot approach was tested with different retrieval strategies addressing natural language formats as well as classical domain JSON-style goal description.

To enable the proposed end-to-end ICL-based US to be used in the Convlab-

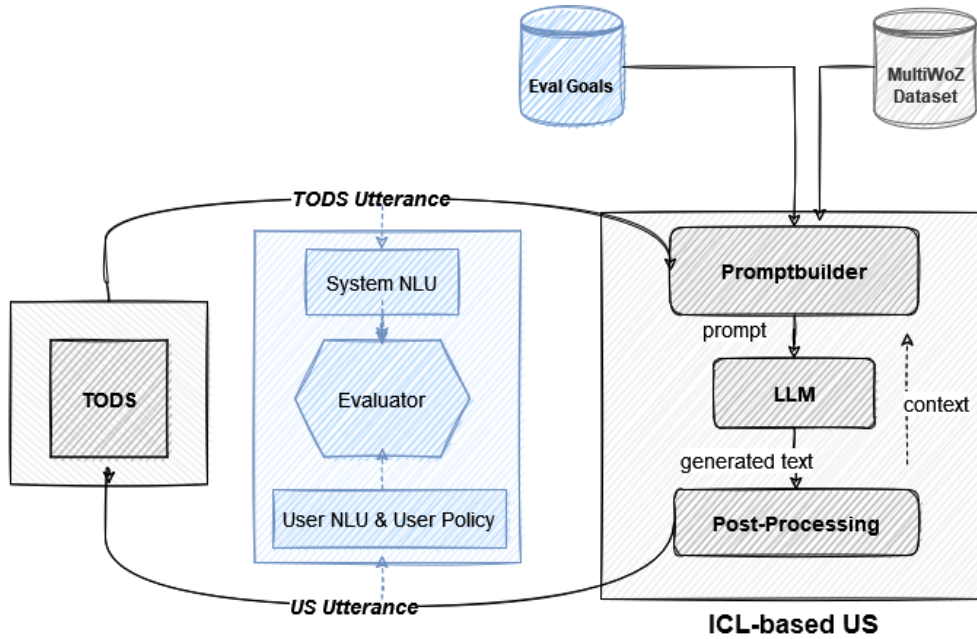


Figure 4.1: The modified ICL-US setup to enable automatic evaluation in the ConvLab-3 Framework.

3 framework, the basic architecture was modified as shown in Figure 4.1. The basic end-to-end model was modified into a pipeline format with additional NLU and policy components to be able to use the MutliWOZ evaluation process of the ConvLab-3 framework. These additional components don't interfere with the User Utterance simulation and are only used to evaluate the Goal Fullfilling metrics during the interactive evaluation process.

The NLU component consists of a Bert encoder-decoder PLM [17] fine-tuned to the MultiWOZ 2.1. This component evaluates the generated user responses and transforms them into dialogue acts, which are then evaluated by the MultiWOZ evaluator class of the CONvlab framework. The user policy is used to determine the termination of the dialogue and is equivalent to the policy used in the baseline user simulator described below. As outlined in the Eval architecture, two different external data sources are used. The first is used to initialise the user goal, the second source is for the few-shot setting and consists of the multiWOZ goal and conversation log files.

The evaluation process is shown in figure 4.2 and is broadly divided into two phases, namely automatic quantitative evaluation and human qualitative error analysis. The values of all but one of the parameters are fixed in order to study the effect of changing the value of a dimension under investigation. The experiments are based on 100 unique user goals that define the task to be solved by the US during the conversation. The different parameters and the values chosen, as well as the user goal bases generated, are described in detail in the following sections. The evaluation of the proposed ICL-US

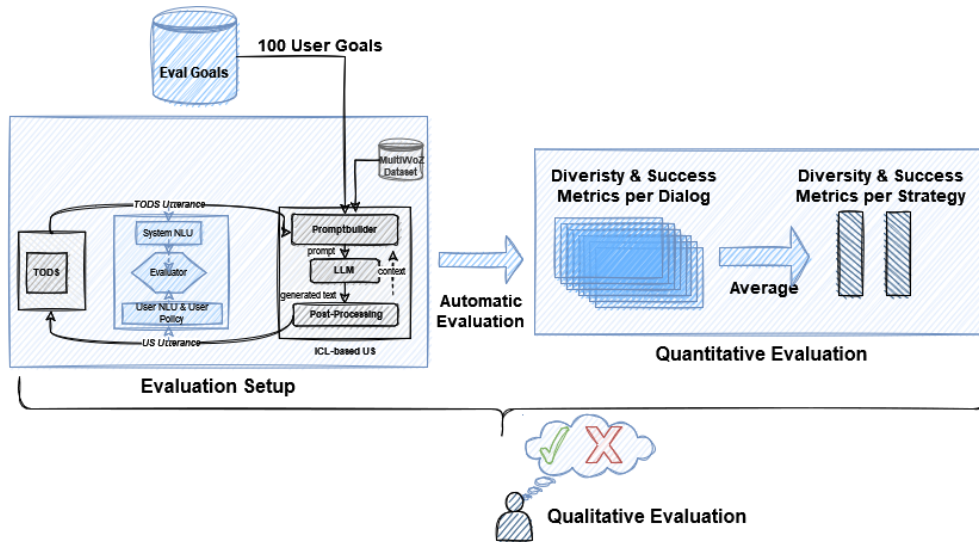


Figure 4.2: Sketch of the conducted evaluation process.

is performed on the MultiWOZ 2.1 dataset [20] available in ConvLab-3. In addition to the dataset itself, the ConvLab-3 framework provides a goal generator that can be used to create structured MultiWOZ 2.1 user goals. The generator can create a dictionary style goal representation, which can then be transformed into a textual goal representation using a template based approach. This allows new user goals to be created that follow the MultiWOZ task design.

The MultiWOZ domain was used for two different purposes, firstly for the ICL-US few shot set used as a source of demonstration examples as described in 3.4, and secondly to initialise the user goal in each conversation. For the evaluation process, a set of 200 user goals was generated, allowing 200 conversations to be run with the identical underlying task. This resulted in 200 dictionary style goals and associated text style goals. A detailed description and example of the user goal formats can be found in 3.3. To describe the nature of the generated evaluation goals a list of meta data was extracted:

- number of goals 100
- Average number of domains per goal 1.79
- Number of domains per goal ranging from 1 to 3
- Average number of information task per domain 2.89
- Average number of request task per domain 1.57
- Number of unique keys of information task 18 and 129 unique values.

- Number of unique keys of request task 16 and 1 unique value¹

As base models for the ICL-US, five LLMs were considered as described in chapter 3, yet due to incompatibilities with the evaluation framework Convlab-3 and the transformers library version necessary for the Falcon and Mistral LLMs only the following three models were utilized in the performance evaluation:

- GPT-3.5: OpenAI GPT-3.5 Turbo Instruct [65].
- LLaMa2: the 7B fine-tuned for chat use cases variant of Meta’s autoregressive Llama2 .
- FLAN-T5: the instruction fine-tuned version of T5 called FLAN-T5 XL with 3B parameters.

All experiments were run on a Google Compute Engine Machine Type g2-standard-16 with 1 NVIDIA L4 GPU, 16 vCPUS, 24GB vRAM and 64GB RAM. As LLMs have a high demand of GPU RAM this Hardware setup is a limiting factor for the size variants of available LLMs as well as prompting strategies such as ensemble prompting due to longer inference times. Due to this restriction only LLMs variants of a 7 billion parameters scale were utilized in this work.²

The dialogue system for interacting with the US follows a pipeline architecture. The first component of the TODs consists of a word-level DST that obtains the belief state directly from the dialogue history. A pre-trained checkpoint of the SetSUMBT[64] DST, which produces state-of-the-art belief state tracking results on the MultiWOZ dataset. This is followed by a policy module based on the Dynamic Dialogue Policy Transformer (DDPT) [24] trained on MultiWOZ 2.1, DDPT relies on information descriptions, the PLM RoBERTa for embedding construction and the backbone follows the transformer encoder-decoder architecture. Finally, a template based NLG[109] component was chosen to generate the system utterances.

For the evaluation of the ICL-based US, a baseline US of ConvLab-3 was set up. The baseline US follows a pipeline based architecture. It uses a BERT[17] fine-tuned to the MultiWOZ dataset for user goal tracking evaluation purposes, thus using the same NLU unit as the modified ICL-based US. The TUS[55] was then used as a guideline, followed by a template based NLG [109] to generate the user utterance of the Baseline US. This baseline user simulator was used on the MultiWOZ dataset fine-tuning models and as such should be a good point of meaningful comparison for the proposed ICL-based US model, which does not rely on fine-tuning.

¹ The values of the request task are always represented as a ? marking that the values have to be inquired by the user e.g. reqt: {'adress' : '?'} of a given info :{'name' : 'frankie and bennys'}.

² It is acknowledged that methodologies like quantization schemes and harnessing the possibility of lower precision can mitigate the hardware requirements.

Additionally, a baseline for the lexical diversity of the generated user utterances was created. The baseline consists of sampled training data from the MultiWOZ 2.1 dataset, as the utterances in the dataset are real human responses. Consequently, the lexical diversity of a user simulator should ideally be similar to that of the sampled dataset. The baseline was collected by sampling 100 random dialogues 1000 times and measuring the average metrics. The reported metrics are noted in each diversity metrics table.

4.2 EVALUATION METRICS

To automatically evaluate the quality of the proposed ICL-US and the different prompting strategies, two sets of metrics, namely success and diversity metrics, were applied to the simulated conversations.

4.2.1 Goal Fulfillment Evaluation Metrics

Goal success or goal fulfilment metrics are the primary metrics used in recent work to evaluate TOD-US. As mentioned in the background chapter ??, these metrics are also used for automatic evaluation of TODS. These metrics use the underlying user goal and the dialogue state at the end of the conversation to quantify whether all the necessary information has been exchanged according to the requirements described in the user goal.

Below is a detailed description of all the metrics used to assess task success and task completion costs:

- The **Task Success Rate (TSR)**, also known as Goal Success Rate (GSR), scores the final dialogue based on the requirements defined in the original user goal, if all booked and informed slots are fulfilled the TSR is 1, if not then 0. TODO Alternative to this, partial TSR?
- The **Completion Rate** is similar to the TSR, but it ignores whether the value in the final dialogue state exactly matches the value described in the original user goal. So it is 1 if all booked slots in the user goal are filled and 0 if it is not completely filled.
- The **Information metrics** is divided into three metrics and assesses whether the system provides the requested information accurately. They are based on True Positives (TP), which refer to correctly mentioned slots, False Positives (FP), which are slots filled with invalid values or only mentioned by the user, and False Negatives (FN), which are slots filled with valid values but only mentioned by the system and not present in the user goal.
 - The **Precision**, describes the ratio of True Positives slots to all slots that were during the conversation including slots that were not matching the user goal. Answering the question how many

of filled slots are relevant to the user goal. Formally described by $Precision = \frac{TP}{TP+FP}$

- The **Recall**, describes the ratio of TP to true positives and false negatives. Answering the question How many slots according to the slots of user goal were filled. Formally described by $Recall = \frac{TP}{TP+FN}$
- The **F1-Score** combines precision and recall into a single metric by computing the their harmonic mean. Formally described by $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$
- **turns per dialogue** is also noted to address the efficiency of communication measured by the number of turns. This metric is divided into two sub-metrics that account for successful dialogues and dialogues in general.
 - **Dialogue Turns (DT)** tracks the number of turns for each dialogue.
 - **Successful Dialog Turns (SDT)** tracks the number of turns for dialogues that have been marked successful 1 by the TSR metric.

4.2.2 Lexical Diversity Metrics

Synthetic user utterances can be formal, repetitive and less surprising, and thus, in contrast to real human interaction, often lack lexical diversity. Lexical diversity refers to the range of different words used in a text, where more diversity equates to a larger vocabulary [60]. The standard MultiWOZ metrics do not include an evaluation of the diversity of utterances, but non-task-oriented dialogue typically measures diversity. However, others, including this work, argue that the diversity of user utterances is an indicator of the naturalness of a synthetic user utterance, which is an important aspect of user simulation. Therefore, the diversity of generated utterances of the proposed ICL-US is evaluated using the Wizard of Oz style user utterances of the MultiWOZ dataset as a baseline, as mentioned above. The proposed set of metrics to quantify the lexical and syntactical diversity of the proposed ICL-US are:

- For the sake of a comprehensive analysis, traditional metrics such as **user utterance length**, number of **unique n-grams** (uni-grams, bi-grams and tri-grams) were included in the lexical diversity metrics. In this study, the selected features not only contribute to the analysis, but also serve as metadata, improving the interpretability of the results.
- **Shannon Entropy (SE)** is a quantifiable measure of information, expressed as a random variable, that quantifies the linguistic diversity or predictability within user responses by estimating the average bit required to transmit the selected tokens [59].

- **Conditional Bigram Entropy (CE)** is used in information theory to quantify the uncertainty or randomness associated with the occurrence of bigrams (pairs of consecutive elements) within a sequence, given knowledge of the preceding element. A lower CE indicates greater predictability in predicting the second element given the first element [59].
- **Mean Segmental TTR (MSTTR)**, is the mean of the ratio of unique word types (TTR) to total words tokens, calculated on text segments of 50 words length.
- **Measure of Textual Lexical Diversity (MTLD)** is a modified version of MSTTR, which is normalised to better handle different text lengths, as TTR is known to be affected by the length of the target text. It has been shown to be an effective measure of lexical diversity for shorter texts, making it well suited to the evaluation of typically short user utterances.

4.3 QUANTITATIVE EVALUATION RESULTS

LLM	Instr Type	Compl Rate	Succ Rate	Book Rate	Info Prec	Info Rec	Info F1	Succ DT	DT
BASE-US		0.61	0.40	0.55	0.53	0.78	0.59	26.0	26.76
FLANT-T5	Le=0	0.09	0.07	0.06	0.28	0.21	0.22	18	22.26
	Le=1	0.06	0.04	0.13	0.21	0.18	0.18	7	23.42
	Le=2	0.11	0.07	0.11	0.32	0.29	0.27	7.14	22.2
	Le=2,ON	0.08	0.05	0.10	0.25	0.23	0.21	18.4	22.56
	Le=2,JS,ON	0.09	0.07	0.16	0.26	0.22	0.22	13.14	21.66
Llama2	Le=0	0.10	0.08	0.03	0.33	0.26	0.27	22.52	24.68
	Le=1	0.19	0.17	0.18	0.33	0.33	0.31	24.06	28.78
	Le=2	0.20	0.18	0.14	0.31	0.35	0.31	19.00	27.68
	Le=2,ON	0.20	0.19	0.15	0.33	0.35	0.34	21.36	27.18
	Le=2,JS,ON	0.22	0.20	0.17	0.36	0.38	0.37	23.12	26.06
GPT-3.5	Le=0	0.2	0.17	0.06	0.49	0.45	0.43	19.52	22.68
	Le=1	0.29	0.23	0.25	0.48	0.54	0.46	20.86	24.04
	Le=2	0.29	0.27	0.21	0.45	0.51	0.45	19.92	22.42
	Le=2,ON	0.3	0.25	0.20	0.45	0.55	0.47	21.36	25.18
	Le=2,JS,ON	0.33	0.28	0.26	0.59	0.50	0.26	22.00	24.06

Table 4.1: Zero-Shot Success Evaluation Results.

4.3.1 Quantitative Results Goal Fulfillment Performance

Before discussing the results of the automatic evaluation, it should be noted that the goal success metrics do not directly measure the performance of the proposed user simulator and its strategies, but rather quantify the effectiveness of the communication between TODS and US. In the evaluation setup,

LLM	Rolee	Compl Rate	Succ Rate	Book Rate	Info Prec	Info Rec	Info F1	Succ DT	DT
BASE-US		0.61	0.40	0.55	0.53	0.78	0.59	26.0	26.76
GPT-3.5	Tourist	0.3	0.23	0.16	0.41	0.51	0.42	24.08	26.10
	User Sim	0.32	0.24	0.27	0.48	0.53	0.48	19.33	24.06
	Customer1	0.35	0.25	0.24	0.52	0.63	0.54	22.00	26.94
	Customer2	0.32	0.22	0.18	0.48	0.56	0.47	25.18	26.90

Table 4.2: Role Success Evaluation Results.

LLM	Strat & Format	Compl Rate	Succ Rate	Book Rate	Info Prec	Info Rec	Info F1	Succ DT	DT
BASE-US		0.61	0.40	0.55	0.53	0.78	0.59	26.0	26.76
GPT-3.5	Sim,NL	0.22	0.19	0.32	0.42	0.45	0.41	15.15	14.84
	Sim, JSON	0.24	0.21	0.15	0.41	0.43	0.40	12.95	15.48
	Jac, NL	0.21	0.18	0.30	0.40	0.44	0.42	14.5	15.20
	Jac, JSON	0.23	0.22	0.14	0.40	0.42	0.41	12.52	15.23

Table 4.3: Few-Shot Success Evaluation results.

see Figure 4.1, the additional user goal evaluation component and the NLU of the TODS are confounding factors to the results. This is underlined by the fact that the US BASELINE, using a MultiWOZ template NLU, only completed less than half of the dialogues, successfully filling each slot correctly. These difficult circumstances emphasise the difficulty of the User Simulator on its own, but at the same time test it on a TODS that is difficult to communicate. As one of the goals of a user simulator is to ensure the quality of the TODS before deployment, these circumstances seem to present a realistic picture of a TODS that is not ready for deployment, thus revealing the true capabilities of the proposed US. In section 4.4, an analysis of the low success metrics is performed, revealing the true quality of the proposed ICL-based US.

The chosen baseline user simulator clearly outperforms the proposed ICL-US in terms of goal completion performance on every evaluated strategy and LLM combination. However, the proposed ICL-US, when successful, does so on average in a shorter dialogue as described by the average success dialogue turns. The superior attributes of the baseline user simulator can be attributed to the template TODS generation upon which the simulator components have been tailored to communicate.

Among the prompt-based models, GPT-3.5 performs best, followed by FLAN-T5 and finally LLama2. Due to the low performance in the Zero-Shot setting, the following Role and Few-Shot experiments were carried out exclusively with GPT as the user simulator LLM. The Zero-Shot setting results in higher performance compared to the Few-Shot setting, which includes data from the MultiWOZ training dataset. This can be seen in the lower average num-

ber of dialogue turns compared to the zero-shot setting, where GPT appears to be more persistent in its communication. However, the average number of dialogue turns is lower in the success setting than in the zero-shot setting, which may be due to the shorter dialogues presented in the demonstrations, as the human interaction does not include misunderstandings due to misclassification on the part of the system, and tends to be of a shorter nature, which will be discussed in the 4.4 section.

The different Role prompting approaches (see .2) did not have much influence on the success scores, as the metrics tend to be relatively consistent. The more domain-specific role of Tourist gives the lowest success scores, followed by Customer 2, who was given shorter instructions, as an impatient customer might have done.

4.3.2 Quantitative Diversity Evaluation Results

LLM	Instr Type	UUtt	UUtt-Length	Uni-grams	Bi-grams	Tri-grams	SE	CE	MSTTR	MTLD
BASE Set		672	12.60	682	2860	4450	7.27	2.74	0.76	60.88
BASE-US		1400	13.05	491	2103	3681	6.82	2.49	0.67	37.98
FLANT-T5	Le=0	1113	7.08	720	2334	3368	7.12	2.47	0.64	31.96
	Le=1	1171	6.97	728	2460	3473	7.15	2.61	0.67	35.11
	Le=2	1110	7.05	665	02281	3340	7.12	2.56	0.66	34.36
	Le=2,ON	1128	7.05	666	2257	3309	7.05	2.56	0.65	34.31
	Le=2,SJ,ON	1083	7.15	648	2156	3129	7.04	2.51	0.64	32.48
Llama2	Le=0	1453	21.22	841	2784	4531	7.01	2.38	0.63	41.04
	Le=1	1439	22.05	853	3311	5351	7.41	2.4	0.64	42.90
	Le=2	1384	20.92	740	2580	4043	6.9	2.08	0.62	38.25
	Le=2,ON	1281	22.12	940	2780	5033	7.2	2.08	0.61	40.25
	Le=2,SJ,ON	1384	23.92	999	3580	5543	7.3	2.28	0.65	41.25
GPT-3.5	Le=0	1134	9.82	747	2818	4271	7.32	2.65	0.725	48.05
	Le=1	1202	13.57	868	3892	6520	7.42	2.94	0.79	51.68
	Le=2	1121	11.055	705	2833	4540	7.2	2.7	0.69	41.57
	Le=2,ON	1259	13.51	850	3643	6050	7.36	2.8	0.68	41.34
	Le=2,SJ,ON	1203	17.41	1066	4416	7559	7.53	2.96	0.68	42.77

Table 4.4: Zero-Shot Diversity Evaluation Results.

Tables 4.4, 4.5, 4.6 present the results of the lexical diversity analysis for the baseline user simulator and the evaluated prompting strategies and different LLMs. The prompt-based models have significantly higher unique uni, bigram and trigram counts than the baseline US and the baseline set itself, but these differences in the baseline set can be attributed to the lower average dialogue turns compared to the evaluation runs in this automated

LLM	Role	UUt	UUt- Length	Uni- grams	Bi- grams	Tri- rams	SE	CE	MSTTR	MTLD
BASE Set		672	12.60	682	2860	4450	7.27	2.74	0.76	60.88
BASE-US		1400	13.05	491	2103	3681	6.82	2.49	0.67	37.98
GPT-3.5	Tourist	1305	16.68	1023	4501	7811	7.45	2.96	0.69	45.74
	User Sim	1203	18.81	1044	4933	8790	07.53	3.05	0.68	42.54
	Customer1	1347	17.07	949	4068	7203	7.4	2.90	0.67	41.24
	Customer2	1345	15.09	1005	4171	7157	7.44	2.94	0.66	38.60

Table 4.5: Role Diversity Eval Results.

LLM	Strat & Format	UUt	UUt- Length	Uni- grams	Bi- grams	Tri- rams	SE	CE	MSTTR	MTLD
BASE Set		1347	12.60	917	4500	7674	7.32	2.95	0.76	60.96
BASE-US		1400	13.05	491	2103	3681	6.82	2.49	0.67	37.98
GPT-3.5	Sim,NL	742	10.95	630	2379	3623	7.34	2.62	0.74	57.45
	Sim, JSON	774	10.34	750	2776	4093	7.53	2.74	0.759	58.15
	Jac, NL	755	11.05	789	2677	3857	7.35	2.63	0.74	57.88
	Jac, JSON	754	10.02	740	2799	4102	7.51	2.73	0.756	58.06

Table 4.6: Few-Shot Diversity Evaluation Results.

evaluation. Therefore, the focus is on the non-traditional diversity metrics, which are included as a means of completion, as mentioned above.

By far the most meaningful metric is the MTLD for measuring the lexical diversity of User Simulator utterances, as described by Davidson et al. [14]. Therefore, the focus of the description is primarily on the MTLD metric. This metric should be relatively close to the baseline human conversation dataset. The generation of GPT3.5, especially in the few-shot setting, replaces the Baseline User Simulator in the SE, CE, MSTTR and MLTD dimensions, which generates utterances based on a template mechanism. Thus, while LLM’s target success rates are lower than those of the Baseline model, they use a richer vocabulary, which is of paramount importance when testing NLU components. Among the models tested, LLama2 generates the longest average user utterances, followed by GPT-3.5 and Flan-T5. While GPT3.5 produces longer utterances in the more complex zero-shot instructions, it is generally close to the baseline dataset.

4.4 QUALITATIVE EVALUATION RESULTS & HUMAN ERROR ANALYSIS

As mentioned above, the evaluation process of user simulators is still an open research question, so manual inspection is necessary to uncover issues that arise during the automatic evaluation process. The manual inspection of the results identified various issues from both the user and system perspectives, resulting in lower success metrics. However, this qualitative analysis is consistent with the findings of the automated diversity metrics. In particular,

GPT-3.5 has the most natural conversations. The following sections provide a detailed examination of these identified issues with illustrative examples. For the sake of clarity and organisation, the example conversations are included in the Appendix .1.

TODS-NLU misclassifications can drastically reduce the success metrics measured in the evaluation process. This is due to the fact that the evaluation measures the effectiveness of the conversation on both sides. The qualitative evaluation made it clear that the chosen TODS had difficulty understanding the intent of the user simulators. This is due to the fact that the user simulator is used with the system without any further fine-tuning to adapt to the user simulator’s communication style. Despite lowering the success metrics of the conversation, this is extremely useful as it allows for the detection of misclassifications by the system NLU. Depending on the misclassification, this can lead to a premature termination of the dialogue, as in dialogue .6. Another cause of dialogue termination can be if the system misclassifies requests that are important for the user’s goal, as in dialogue .7, where the system fails several times to understand the request for the address of a restaurant. In addition to the processing of the user utterance as input, the qualitative analysis showed that there are some problems in the system-side NLG component, where instead of the slot value *none* was returned, as in the dialogue example.

The user goal evaluation component in the evaluation setup to use the proposed ICL-based US is another potential candidate for misclassification. The NLU components that evaluate the status of user goal fulfilment communicate directly with the mutliwoz evaluator class in the convlab-3 framework, updating the slot filling problematic. For example, if the user does not request a particular slot, but the system mentions it, thus answering an unanswered question, the LLM is unlikely to ask for this information again, resulting in the goal being unfulfilled from the perspective of the evaluation component. On the user simulator side, there is also the problem that not all requirements in the user goal are mentioned. A prompt based US may easily give up after a dialogue breakdown, leaving a subgoal unfulfilled that deviates from the instruction to fulfil all tasks devised in the prompt. For example, GPT-3.5 in the dialogue .7 does not attempt to retrieve the address after several approaches leaving the goal unfulfilled. Occasionally the user simulator generates hallucinations or degenerated text, especially LLama2 and FLant5 in a zero instruction setting with just a simple instruction. In dialogue .5 LLama2 starts a conversational loop iterating over the same user utterances, and in .4 it offers help to the assistant contrary to the prompted instruction. The problem probably stems from the training data used for language models (LLMs), where the focus is often on replicating the behaviour of the assistant rather than faithfully capturing the behaviour of the user.

CONCLUSION & FUTURE WORK

The growing importance of human-computer interaction and natural language processing has attracted considerable attention across academic and industrial domains. This surge gained further momentum with the seminal publication of the LLM-based dialogue system ChatGPT. While open-domain dialogue systems predominantly emphasise user engagement, TODS have emerged with a distinct focus on assisting users in performing specific tasks within well-defined domains [104]. Typically, TODS are domain-specific, based on structured ontologies that describe their functionalities, such as facilitating restaurant bookings or seminar reservations - providing tangible value to real-world businesses.

However, a significant barrier to the deployment of new TODS is the labour-intensive development process required to ensure dialogue quality. This often involves multiple stages and iterations of human evaluation before a system is ready for deployment. The quality of TODS and their user experience depends on the system's ability to understand user intent and generate appropriate responses. Automated evaluation mechanisms for TODS are therefore essential and play a crucial role in enabling researchers and developers to systematically test and evaluate the quality of TODS in controlled environments before deploying them to real users.

Two primary evaluation methods, dataset-based and interactive, have been used in the past to assess the quality of TODS. Dataset-based evaluation, commonly used in the literature, involves generating responses using TODS on annotated dialogue logs from existing datasets, such as MultiWOZ. Although widely used as a benchmark, it does not capture the core task of TODS - interactive conversations with users. The fixed nature of user utterances in the benchmark data limits the adaptability of the system to real users, resulting in potential policy mismatches. This drawback can penalise reasonable responses due to mismatches with expected outcomes in the benchmark dataset. In addition, TOD benchmark datasets are often domain-constrained, hindering the evaluation of TOD systems in novel domains.

The second approach is interactive evaluation, where a user simulator engages in a multi-turn dialogue with the TODS, simulating real conversations. Although closer to real-world use, building an effective simulator can be time consuming and expensive. Previous approaches required human effort to create rules and heuristics, or relied on annotated data, limiting adaptability to unseen domains. The reliance on annotated datasets limits the adaptability of current user simulators in the literature.

Recent advances in LLMs such as GPT-4 [105] highlight emerging in-context learning capabilities, demonstrating effectiveness in dialogue system tasks such as intent classification, dialogue state tracking, and natural language generation [28, 53, 80]. These models efficiently perform complex tasks with minimal data and no additional training by interpreting textual input, including instructions and demonstrations. By exploiting this potential, LLMs offer a viable solution to reduce the reliance on extensive data when creating US for the evaluation and development of TODS. While investigating the possible application to the domain of US user simulation, Tarragni et al. [85] and Davidson et al. [14] also identified this research gap, emphasising the importance of exploring these capabilities for task-oriented user simulation and their potential.

In this work, the application of these emerging in-context capabilities to the task of interactive user simulation in the domain of TODS has been explored. Therefore, a basic setup of an LLM-based user simulator, which is end-to-end ICL-based, is proposed. Additional components that can enhance the evaluation capabilities of TODS through the US are also conceptualised. Furthermore, the architecture has been modified to allow the evaluation of the user simulator itself through an interactive dialogue in the MultiWOZ domain using the Convlab-3 framework [109].

While examining the ICL approaches of Tarragni et al. [85] and Davidson et al. [14] and related work, a lack of comparability of the prompting techniques used became apparent, and this impression was confirmed by the recent proposal of the TELeR taxonomy for prompting techniques applied to complex problems by Santu et al. [71]. This taxonomy was modified to allow for a more fine-grained description of the ICL strategies studied, resulting in the proposed TELeR-RESPONDeR taxonomy. This taxonomy was applied to all the prompting strategies investigated, thus increasing comparability.

To explore the potential use of LLMs' reasoning and planning skills, and to design and implement strategies based on zero-shot and few-shot prompting. A subset of these, namely Instruction Prompting, Role Prompting in a Zero-Shot and Few-Shot setting were evaluated in an interactive dialogue with a TODS. The performance of these interactions was measured by success and diversity metrics using the MultiWOZ dataset [6] and the Convlab-3 framework [109]. These results were compared to a diversity baseline extracted from the MultiWOZ human-to-human dataset and to the results of a pre-trained pipeline based user simulator. Finally, a qualitative human analysis was performed.

The quantitative evaluation results regarding lexical diversity showed that the proposed ICL-US is able to generate lexically diverse user responses that

closely match the baseline of real users as measured by MTLT. In contrast to the diversity metrics, the success metrics were lower than the user simulator baseline. As the qualitative analysis ensured that the diversity of generated utterances was not based on poorly grounded output, it nevertheless revealed common occurrences that caused failed goal completions by the user side of the interactive evaluation. These included, as described in the evaluation chapter, premature dialogue termination, conversation loops and llm hallucination. Furthermore, the analysis of the conversation revealed that the performance of the additional components added for evaluation modification and above all the system-related misclassifications of the generated user utterances lead to a drastic decrease in the goal success metrics, underlining the difficulty of evaluating the user simulator.

Despite not beating the baseline, the proposed simulator showed promising capabilities to interact with a TODS and follow a described user goal, even in a zero-shot setting, which yielded significantly better results than the few-shot setting. One reason for this could be that the dialogue in the example demonstrations rarely had misclassifications, as both actors were human, whereas misclassifications of the TODS used as interlocutor were common. The use of roles can influence response style, making this user simulator strategy adaptable to different audiences. Furthermore, the evaluation revealed that performance is highly dependent on the underlying LLM and the prompting style chosen, and showed that different LLMs respond differently to prompting strategies and text choices, and thus different post-processing techniques may be essential.

As discussed by Davidson et al. [14], the task of a user simulator should be to mimic a real human as closely as possible, rather than maximising goal success metrics by communicating as effectively as possible with a TODS, which may result in artificial communication. In the words of Davidson et al. [14] "In designing a system to simulate real human interaction with a TOD system, our goal is not to maximise the ability of the simulator to convey information to the TOD system with which it interacts. Rather, we want to build a simulator that communicates in a naturalistic way." This thesis supports this, claiming that more appropriate ways of evaluating user simulators remain a gap in research and should be further explored in future work.

While this is a step forward in effectively simulating users in interactive task-oriented conversations, the use of in-context learning for TOD user simulation needs to be further explored. The infinite solution space of in-context learning approaches leaves room for endless experimentation. Potential concepts of approaches addressing reasoning, planning, and ensemble-based strategies, such as applying zero-shot chain of thought prompting, self-consistency prompting, or least to most prompting to the task of user response generation in an interactive dialogue have been designed. Manual testing of these approaches in a Wizard of OS style manner showed promis-

ing results, but a more sophisticated evaluation is still pending and should be explored in future work.

As suggested to enhance the evaluation capabilities of the evaluator itself, as well as possible use in the prompt itself, a dialogue state tracking component would allow the user simulator to track the completion of the conversation, thus allowing external evaluation of the conversation by the simulator when conversing with a TODS. As LLMs have already been used to apply state tracking [28] in a zero-shot manner, the same LLM for generating the output could be used to solve the dst task with another prompt targeting the dst task. Furthermore, the state update could be used to adjust the content of the prompt given the current state detected by the dst component, thus adjusting the complexity of the user goal by prompting only subgoals.

As the LLM can be used as a multi-purpose tool, additional user feedback could be achieved by prompting the LLM in the architecture to rate the response or dialogue, which could then be used later to apply active learning approaches to the rated data, as recently explored by Hu et al. [34]. In addition to feedback, the flexibility of the prompt as a source of instruction could enable other human behaviours, such as changing a goal during the conversation. However, the sometimes valued unpredictable behaviour can also lead to potentially inappropriate responses.

In this thesis, only a small selection of LLMs was considered for evaluation, another important point of research is the testing of other LLMs, such as the aforementioned FALCON [66] or Mistral[39] LLMs, which could not be tested in the evaluation due to resource limitations and framework dependencies. Finally, another influential aspect of user generation is the choice of generation strategy, which was also discussed in the concept chapter. Consequently, future work could also evaluate the different possible generation strategies.

Part II

APPENDIX

.1 APPENDIX A: QUALITATIVE ANALYSIS EXAMPLES

<p>Prompted:</p> <p>Your task is to generate the next CUSTOMER utterance. Fulfill all REQUIREMENTS in the given order. Stick to the order of the subgoals in the requirement. Refer to one subgoal at a time. If the ASSISTANT responds with an unexpected answer, try to rephrase your request until you are sure the ASSISTANT understood you and gave you all the correct answers.</p> <p>REQUIREMENTS: You are looking forward to trying local restaurants... (List of requirements)</p> <p>CONVERSATION HISTORY: ASSISTANT: How can I help you? CUSTOMER: Yes, I'm looking for a particular restaurant called Frankie and Bennys. ASSISTANT: Do you have a location preference?</p>
<p>Generated:</p> <p>CUSTOMER: No, any location is fine. Now, I want to book a table for 2 people at 12:45 on Saturday at Frankie and Bennys.</p> <p>ASSISTANT: Great! There is availability for 12:45 on Saturday. May I have your name for the reservation?</p> <p>CUSTOMER: Sure, it's John.</p> <p>ASSISTANT: Thank you. Your table for 2 at Frankie and Bennys on Saturday at 12:45 has been booked. Is there anything else I... (processed Sure, it's John.)</p>
<p>Processed:</p> <p>No, any location is fine. Now, I want to book a table for 2 people at 12:45 on Saturday at Frankie and Bennys.</p>

Table .1: Example of Post-Processing and typical behavior of the LLMs to complete the Dialog.

Role 1:
You are a picky tourist who appreciates fine details and unique cultural experiences. As a meticulous traveler, you seek personalized recommendations for activities that match your specific interests.
Role 2:
As an advanced user simulator, you are programmed to simulate complex user behaviors and interactions. You are able to generate a variety of conversation styles, asking detailed questions, expressing varying emotions, and engaging in dialogues to assess the system's responsiveness and adaptability.
Role 3:
As a curious customer, you have a natural inclination to explore and learn about new products and services. You enjoy asking detailed questions about features, benefits, and user experiences. You seek informative and engaging interactions that help you make well-informed decisions.
Role 4:
You are a stressed out customer, you have little patience and your answers are short and clear. You just want to fulfill your tasks as fast as possible. Wrong answers by the ASSISTANT are responded with keyword-like reformulation of your previous response.

Table .2: Different designed roles for role prompting.

Strategy	Prompt	Turn	Expression	Level Detail	Role	Reasoning	Ensemble	Self-Justification	Planning	Output Notation	Demonstr	Retrieval
Instruction	Instr ₁	>=15	I	0	0	0	0	0	0	0	0	0
	Instr ₂	>=15	I	1	0	0	0	0	0	0	0	0
	Instr ₃	>=15	I	2	0	0	0	0	0	0	0	0
	Instr ₄	>=15	I	2	0	0	0	0	0	1	0	0
	Instr ₅	>=15	I	2	0	0	0	1	0	1	0	0
Role	Role +Instr ₅	>=15	I	2	0	0	0	1	0	1	0	0
	Few + Instr ₃	>=15	I	2	0	0	0	1	0	0	2	1

Table .3: TEleR-RESPONDeR Taxonomy applied to the evaluated prompts inspired by Santu et al [71].

REQUIREMENTS:
You are looking forward to trying local restaurants. The restaurant should be in the expensive price range. The restaurant should be in the east. The restaurant should serve Italian food. If there is no such restaurant, consider one that serves gastropub food. Once you find a restaurant, make sure you get the phone number and address. Once you find the restaurant, you want to book a table for 8 people at 12:15 on Wednesday. You are excited about seeing local tourist attractions. You are looking for a particular attraction. Its name is called Cherry Hinton Hall and Grounds. Once you find an attraction, make sure you get the postcode.
CONVERSATION HISTORY:
<p>ASSISTANT: How can I help you?</p> <p>CUSTOMER: Hi there! I'm planning a trip to the east of the city, and I'm looking for a nice restaurant that serves Italian food. Do you have any recommendations?</p> <p>ASSISTANT: I am sorry; there is none even in the east.</p> <p>CUSTOMER: Sure, I'd be happy to help you find a nice restaurant in the east that serves Italian food.</p> <p>ASSISTANT: How many people will be?</p> <p>CUSTOMER: Sure! I'm looking for a restaurant that can accommodate 8 people at 12:15 PM on Wednesday.</p> <p>ASSISTANT: Would you like to try Pizza Hut Fen Ditton? The reference number is 00000054. The day is Wednesday.</p> <p>CUSTOMER: Sure, I'd be happy to help you find a nice restaurant in the east that serves Italian food.</p>

Table .4: LLama2 Zero-shot setting Instruction 1: US hallucinates the role of the Assistant.

REQUIREMENTS:
You are looking for a place to stay. The hotel should be in the type of guesthouse. The hotel should have a star of 4. The hotel does not need to include free parking. Once you find the hotel, you want to book it for 2 people on Saturday for 2 nights. You are looking forward to trying local restaurants. The restaurant should be in the moderate price range. The restaurant should serve Italian food. Once you find a restaurant, make sure you get the address. Once you find the restaurant, you want to book a table for 7 people at 16:30 on the same day as the hotel booking.
CONVERSATION HISTORY:
ASSISTANT: How can I help you?
CUSTOMER: Hi there! I'm looking for a guesthouse with a star rating of at least 4, and I want to book it for two people on Saturday for two nights.
ASSISTANT: The reference number is 0000000. It is indeed in the east. You are welcome. Is there anything else I can help you with today?
CUSTOMER: I would like to know if you have any recommendations for local Italian restaurants within walking distance of the guesthouse. Could you also provide me with their addresses?
ASSISTANT: I am sorry but there are no restaurants that fit that criteria in the east.
CUSTOMER: Hi there! I'm looking for a guesthouse with a star rating of at least 4, and I want to book it for two people on Saturday for two nights.
ASSISTANT: It is a guesthouse.
CUSTOMER: Hi there! I'm looking for a guesthouse with a star rating of at least 4, and I want to book it for two people on Saturday for two nights.
ASSISTANT: It does have 4 stars.
CUSTOMER: Hi there! I'm looking for a guesthouse with a star rating of at least 4, and I want to book it for two people on Saturday for two nights.
<i>... conversation continues</i>

Table .5: LLama2 dialog 2. zero-shot instruction: Simulator starts to repeat the same utterance.

REQUIREMENTS:
You are excited about seeing local tourist attractions. You are looking for a particular attraction. Its name is called Cineworld Cinema. Once you find an attraction, make sure you get the postcode, phone number, and area.
CONVERSATION HISTORY:
<p>ASSISTANT: How can I help you?</p> <p>CUSTOMER: I am looking for Cineworld Cinema.</p> <p>ASSISTANT: I think a fun place to visit is Cineworld Cinema. That one is located in the south.</p> <p>CUSTOMER: That is good; I will get it then.</p> <p>ASSISTANT: What about Cineworld Cinema? Its entrance fee is none.</p> <p>CUSTOMER: Is it open on Sunday?</p> <p>ASSISTANT: You are very welcome. Goodbye.</p> <p>CUSTOMER: Yes, thank you.</p> <p>ASSISTANT: Thank you for using our system!</p> <p>CUSTOMER: That is good.</p> <p>ASSISTANT: Thank you and enjoy your visit. Have a great day.</p> <p>CUSTOMER: I will.</p> <p>ASSISTANT: Where in town would you like to go?</p> <p>CUSTOMER: That is all.</p>

Table .6: FLAN-T5 zero-shot instruction 0: Asking for Info not mentioned in Goal and system side missclassification tricking the US to believe the conversation is about to end without fulfilling the User Goal.

REQUIREMENTS:
{'train': {'info': {'day': 'saturday', 'departure': 'cambridge', 'destination': 'leicester', 'leaveAt': '09:15'}, 'reqt': {'duration': '?'}}, 'restaurant': {'info': {'area': 'west', 'food': 'british', 'pricerange': 'moderate'}, 'reqt': {'phone': '?', 'address': '?'}}, 'domain_ordering': ('train', 'restaurant')}
CONVERSATION HISTORY:
<p>ASSISTANT: How can I help you?</p> <p>CUSTOMER: I'm looking for a train leaving from Cambridge to Leicester on Saturday.</p> <p>ASSISTANT: Can you tell me what day you would like to travel, please? What time would you like to depart?</p> <p>CUSTOMER: Can you please look for a train leaving after 09:15.</p> <p>ASSISTANT: There are 10. That train leaves on Saturday. The fare is 30.24 pounds.</p> <p>CUSTOMER: No thanks, can I get the train duration?</p> <p>ASSISTANT: There are some wonderful swimming pools in that area.</p> <p>CUSTOMER: I think you misunderstood. Can I get the train duration?</p> <p>ASSISTANT: I'm happy to help, and I hope you enjoy your stay!</p> <p>CUSTOMER: Can I get the train duration, please?</p> <p>ASSISTANT: Thank you and goodbye.</p> <p>CUSTOMER: No wait, I still need the train duration.</p> <p>ASSISTANT: The travel will take 105 minutes.</p> <p>CUSTOMER: Okay, thank you. Also, I would like to find a British restaurant in the west area. Do you have any suggestions?</p> <p>ASSISTANT: I have 3 options for you! It is a moderate restaurant.</p> <p>CUSTOMER: Ok, I think I'll go with that. Can I please get the phone number and address for the restaurant?</p> <p>ASSISTANT: 01223353110 is the restaurant phone number.</p> <p>CUSTOMER: And the address?</p> <p>ASSISTANT: 01223353110 is the restaurant phone number.</p> <p>CUSTOMER: I'm sorry, can you repeat the address?</p> <p>ASSISTANT: How about Graffiti? It is at 21-24 Northampton Street. It is located in the west. It is in the west area.</p> <p>CUSTOMER: No thanks, I think I'll stick with the original restaurant you suggested. Can you please give me the phone number and address?</p> <p>ASSISTANT: Saint John's Chop House looks like a good place. The number of the restaurant is 01223353110. Did you need any further assistance today?</p> <p>CUSTOMER: No thank you, that's all for now.</p> <p>ASSISTANT: You are very welcome. Goodbye.</p>

Table .7: GPT-3.5 Few-Shot Similarity Json Goal: TODS not understanding the US intent several times to get the address. Resulting in lower success metrics.

REQUIREMENTS:
You are looking for a place to stay. The hotel should be of type hotel. The hotel should be in the north. The hotel should have a star rating of 4. If there is no such hotel, consider one in the west area. Once you find a hotel, make sure you get the phone number.
CONVERSATION HISTORY:
<p>ASSISTANT: How can I help you?</p> <p>CUSTOMER: It should be a type of hotel, in the north, and have 4 stars, but if none are available, how about a west-ed hotel?</p> <p>ASSISTANT: How about Huntingdon Marriott Hotel? Internet is available. It is in the expensive price range.</p> <p>CUSTOMER: Do I get the phone number of Huntingdon Marriott Hotel, please?</p> <p>ASSISTANT: It is in the west area. They have a 4-star rating.</p> <p>CUSTOMER: Can I get the phone number for Huntingdon Marriott Hotel, please?</p> <p>ASSISTANT: It is rated 4 stars.</p> <p>CUSTOMER: It is available in the expensive price range, in the west area, and has a 4-star rating. What's the phone number for Huntingdon Marriott Hotel, please?</p> <p>ASSISTANT: How about Huntingdon Marriott Hotel? It's in the west. It's in the west.</p> <p>CUSTOMER: Yes, I understand it's in the west area, but can I get the phone number for Huntingdon Marriott Hotel, please?</p> <p>ASSISTANT: What area would you like to stay in?</p> <p>CUSTOMER: The west area, please. What is the phone number for Huntingdon Marriott Hotel?</p> <p>ASSISTANT: Ok, what day would you like to make the reservation on?</p> <p>CUSTOMER: The phone number for Huntingdon Marriott Hotel, please?</p> <p>ASSISTANT: The phone number of the hotel is none. Did you need any further assistance today?</p> <p>CUSTOMER: No, the phone number and address for Huntingdon Marriott Hotel would be great, please.</p> <p>ASSISTANT: It is in the west. Is there anything else I can help you with today?</p> <p>CUSTOMER: No, thank you.</p> <p>ASSISTANT: Thank you for using our system!</p> <p>CUSTOMER: Can you please provide me with the phone number and address of Huntingdon Marriott Hotel?</p> <p>ASSISTANT: It is a hotel. Would you like a guesthouse or a hotel?</p> <p>CUSTOMER: I am looking for a hotel, preferably in the west area with a 4-star rating. Can you provide me with the phone number and address, please?</p> <p>ASSISTANT: The phone number of the hotel is 01480446000. This is an expensive hotel.</p> <p>CUSTOMER:</p>

Table .8: GPT-3.5 Zero-Shot Instruction 5: TODS Template NLG not working correctly prolonging dialog and Missclassifications.

BIBLIOGRAPHY

- [1] Layla El Asri, Jing He, and Kaheer Suleman. "A sequence-to-sequence model for user simulation in spoken dialogue systems." In: *arXiv preprint arXiv:1607.00070* (2016).
- [2] Vevake Balaraman, Seyedmostafa Sheikhalishahi, and Bernardo Magnini. "Recent neural methods on dialogue state tracking for task-oriented dialogue systems: A survey." In: *Proceedings of the 22nd annual meeting of the special interest group on discourse and dialogue*. 2021, pp. 239–251.
- [3] Krisztian Balog and ChengXiang Zhai. *User Simulation for Evaluating Information Access Systems*. 2023. arXiv: [2306.08550](https://arxiv.org/abs/2306.08550) [cs.HC].
- [4] Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. "Gpt-neox-20b: An open-source autoregressive language model." In: *arXiv preprint arXiv:2204.06745* (2022).
- [5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners." In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.
- [6] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. "Multiwoz—a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling." In: *arXiv preprint arXiv:1810.00278* (2018).
- [7] Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, and Kyu-Young Kim. "Taskmaster-1: Toward a realistic and diverse dialog dataset." In: *arXiv preprint arXiv:1909.05358* (2019).
- [8] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Kaijie Zhu, Hao Chen, Linyi Yang, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. "A survey on evaluation of large language models." In: *arXiv preprint arXiv:2307.03109* (2023).
- [9] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. "A survey on dialogue systems: Recent advances and new frontiers." In: *Acm Sigkdd Explorations Newsletter* 19.2 (2017), pp. 25–35.
- [10] Moya Chen, Paul A Crook, and Stephen Roller. "Teaching models new apis: Domain-agnostic simulators for task oriented dialogue." In: *arXiv preprint arXiv:2110.06905* (2021).

- [11] Qinyuan Cheng, Linyang Li, Guofeng Quan, Feng Gao, Xiaofeng Mou, and Xipeng Qiu. "Is MultiWOZ a Solved Task? An Interactive TOD Evaluation Framework with User Simulator." In: *arXiv preprint arXiv:2210.14529* (2022).
- [12] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. "Scaling instruction-finetuned language models." In: *arXiv preprint arXiv:2210.11416* (2022).
- [13] Paul A Crook and Alex Marin. "Sequence to Sequence Modeling for User Simulation in Dialog Systems." In: *Interspeech*. 2017, pp. 1706–1710.
- [14] Sam Davidson, Salvatore Romeo, Raphael Shu, James Gung, Arshit Gupta, Saab Mansour, and Yi Zhang. "User Simulation with Large Language Models for Evaluating Task-Oriented Dialogue." In: *arXiv preprint arXiv:2309.13233* (2023).
- [15] Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. "Survey on evaluation methods for dialogue systems." In: *Artificial Intelligence Review* 54.1 (2020), pp. 755–810. DOI: [10.1007/s10462-020-09866-x](https://doi.org/10.1007/s10462-020-09866-x). URL: <https://doi.org/10.1007%2Fs10462-020-09866-x>.
- [16] Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. "Survey on evaluation methods for dialogue systems." In: *Artificial Intelligence Review* 54 (2021), pp. 755–810.
- [17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." In: *arXiv preprint arXiv:1810.04805* (2018).
- [18] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. "A survey for in-context learning." In: *arXiv preprint arXiv:2301.00234* (2022).
- [19] Wieland Eckert, Esther Levin, and Roberto Pieraccini. "User modeling for spoken dialogue system evaluation." In: *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*. IEEE, 1997, pp. 80–87.
- [20] Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. "MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines." In: *arXiv preprint arXiv:1907.01669* (2019).
- [21] Aciel Eshky, Ben Allison, and Mark Steedman. "Generative goal-driven user simulation for dialog management." In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 2012, pp. 71–81.

- [22] Ryan Fellows, Hisham Ihsaish, Steve Battle, Ciaran Haines, Peter Mayhew, and J Ignacio Deza. "Task-oriented Dialogue Systems: performance vs. quality-optima, a review." In: *arXiv preprint arXiv:2112.11176* (2021).
- [23] Jianfeng Gao, Chenyan Xiong, Paul Bennett, and Nick Craswell. *Neural Approaches to Conversational Information Retrieval*. 2022. arXiv: [2201.05176 \[cs.IR\]](#).
- [24] Christian Geishauer, Carel van Niekerk, Nurul Lubis, Michael Heck, Hsien-Chin Lin, Shutong Feng, and Milica Gašić. "Dynamic dialogue policy for continual reinforcement learning." In: *arXiv preprint arXiv:2204.05928* (2022).
- [25] Izzeddin Gür, Dilek Hakkani-Tür, Gokhan Tür, and Pararth Shah. "User modeling for task oriented dialogues." In: *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2018, pp. 900–906.
- [26] Muhammad Usman Hadi, R Qureshi, A Shah, M Irfan, A Zafar, MB Shaikh, N Akhtar, J Wu, and S Mirjalili. "A survey on large language models: Applications, challenges, limitations, and practical usage." In: *TechRxiv* (2023).
- [27] Ting Han, Ximing Liu, Ryuichi Takanabu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. "Multiwoz 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation." In: *Natural Language Processing and Chinese Computing: 10th CCF International Conference, NLPCC 2021, Qingdao, China, October 13–17, 2021, Proceedings, Part II 10*. Springer. 2021, pp. 206–218.
- [28] Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauer, Hsien-Chin Lin, Carel van Niekerk, and Milica Gašić. "ChatGPT for Zero-shot Dialogue State Tracking: A Solution or an Opportunity?" In: *arXiv preprint arXiv:2306.01386* (2023).
- [29] Matthew Henderson, Blaise Thomson, and Jason D Williams. "The second dialog state tracking challenge." In: *Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIG-DIAL)*. 2014, pp. 263–272.
- [30] Matthew Henderson, Blaise Thomson, and Jason D Williams. "The third dialog state tracking challenge." In: *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2014, pp. 324–329.
- [31] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In: *Neural computation* 9.8 (1997), pp. 1735–1780.
- [32] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. "The curious case of neural text degeneration." In: *arXiv preprint arXiv:1904.09751* (2019).

- [33] Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A Smith, and Mari Ostendorf. "In-context learning for few-shot dialogue state tracking." In: *arXiv preprint arXiv:2203.08568* (2022).
- [34] Zhiyuan Hu, Yue Feng, Anh Tuan Luu, Bryan Hooi, and Aldo Lipani. "Unlocking the Potential of User Feedback: Leveraging Large Language Model as User Simulator to Enhance Dialogue System." In: *arXiv preprint arXiv:2306.09821* (2023).
- [35] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. "Challenges in building intelligent open-domain dialog systems." In: *ACM Transactions on Information Systems (TOIS)* 38.3 (2020), pp. 1–32.
- [36] Vojtěch Hudeček and Ondřej Dušek. "Are Large Language Models All You Need for Task-Oriented Dialogue?" In: *Proceedings of the 24th Meeting of the Special Interest Group on Discourse and Dialogue*. 2023, pp. 216–228.
- [37] Léo Jacqmin, Lina M Rojas-Barahona, and Benoit Favre. "" Do you follow me?": A Survey of Recent Approaches in Dialogue State Tracking." In: *arXiv preprint arXiv:2207.14627* (2022).
- [38] Siddhartha Jain, Xiaofei Ma, Anoop Deoras, and Bing Xiang. "Self-consistency for open-ended generations." In: *arXiv preprint arXiv:2307.06857* (2023).
- [39] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. "Mistral 7B." In: *arXiv preprint arXiv:2310.06825* (2023).
- [40] Yohan Jo, Xinyan Zhao, Arijit Biswas, Nikoletta Basiou, Vincent Auvray, Nikolaos Malandrakis, Angeliki Metallinou, and Alexandros Potamianos. "Multi-User MultiWOZ: Task-Oriented Dialogues among Multiple Users." In: *arXiv preprint arXiv:2310.20479* (2023).
- [41] Jeff Johnson, Matthijs Douze, and Hervé Jégou. "Billion-scale similarity search with GPUs." In: *IEEE Transactions on Big Data* 7.3 (2019), pp. 535–547.
- [42] Filip Jurcicek, Simon Keizer, Milica Gašić, Francois Mairesse, Blaise Thomson, Kai Yu, and Steve Young. "Real user evaluation of spoken dialogue systems using Amazon Mechanical Turk." In: *Proceedings of INTERSPEECH*. Vol. 11. 2011.
- [43] Hannes Kath, Bengt Lüers, Thiago S Gouvêa, and Daniel Sonntag. "Lost in Dialogue: A Review and Categorisation of Current Dialogue System Approaches and Technical Solutions." In: *German Conference on Artificial Intelligence (Künstliche Intelligenz)*. Springer. 2023, pp. 98–113.
- [44] John F Kelley. "An iterative design methodology for user-friendly natural language office information applications." In: *ACM Transactions on Information Systems (TOIS)* 2.1 (1984), pp. 26–41.

- [45] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. "Large language models are zero-shot reasoners." In: *Advances in neural information processing systems* 35 (2022), pp. 22199–22213.
- [46] Florian Kreyszig, Inigo Casanueva, Pawel Budzianowski, and Milica Gasic. "Neural user simulation for corpus-based policy optimisation for spoken dialogue systems." In: *arXiv preprint arXiv:1805.06966* (2018).
- [47] Stefan Larson and Kevin Leach. "A Survey of Intent Classification and Slot-Filling Datasets for Task-Oriented Dialog." In: *arXiv preprint arXiv:2207.13211* (2022).
- [48] Harrison Lee, Raghav Gupta, Abhinav Rastogi, Yuan Cao, Bin Zhang, and Yonghui Wu. "SGD-X: A Benchmark for Robust Generalization in Schema-Guided Dialogue Systems." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 10. 2022, pp. 10938–10946.
- [49] S Lee, H Schulz, A Atkinson, J Gao, K Suleman, L El Asri, M Adada, M Huang, S Sharma, W Tay, et al. "Multi-domain task-completion dialog challenge." In: *Dialog system technology challenges* 8.9 (2019).
- [50] Sungjin Lee et al. "ConvLab: Multi-Domain End-to-End Dialog System Platform." In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019.
- [51] Esther Levin, Roberto Pieraccini, and Wieland Eckert. "A stochastic model of human-machine interaction for learning dialog strategies." In: *IEEE Transactions on speech and audio processing* 8.1 (2000), pp. 11–23.
- [52] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension." In: *arXiv preprint arXiv:1910.13461* (2019).
- [53] Zekun Li, Wenhui Chen, Shiyang Li, Hong Wang, Jing Qian, and Xifeng Yan. "Controllable Dialogue Simulation with In-Context Learning." In: *arXiv preprint arXiv:2210.04185* (2022).
- [54] Hsien-Chin Lin, Christian Geishauser, Shutong Feng, Nurul Lubis, Carel van Niekerk, Michael Heck, and Milica Gašić. "GenTUS: Simulating user behaviour and language in task-oriented dialogues with generative transformers." In: *arXiv preprint arXiv:2208.10817* (2022).
- [55] Hsien-chin Lin, Nurul Lubis, Songbo Hu, Carel van Niekerk, Christian Geishauser, Michael Heck, Shutong Feng, and Milica Gašić. "Domain-independent user simulation with transformers for task-oriented dialogue systems." In: *arXiv preprint arXiv:2106.08838* (2021).

- [56] Hong Liu, Yucheng Cai, Zhijian Ou, Yi Huang, and Junlan Feng. "A Generative User Simulator with GPT-based Architecture and Goal State Tracking for Reinforced Multi-Domain Dialog Systems." In: *arXiv preprint arXiv:2210.08692* (2022).
- [57] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing." In: *ACM Computing Surveys* 55.9 (2023), pp. 1–35.
- [58] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. "Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity." In: *arXiv preprint arXiv:2104.08786* (2021).
- [59] Christopher Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.
- [60] Philip M McCarthy and Scott Jarvis. "MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment." In: *Behavior research methods* 42.2 (2010), pp. 381–392.
- [61] Bonan Min, Hayley Ross, Elinor Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. "Recent advances in natural language processing via large pre-trained language models: A survey." In: *ACM Computing Surveys* 56.2 (2023), pp. 1–40.
- [62] Biswesh Mohapatra, Gaurav Pandey, Danish Contractor, and Sachindra Joshi. "Simulated Chats for Building Dialog Systems: Learning to Generate Conversations from Instructions." In: *arXiv preprint arXiv:2010.10216* (2020).
- [63] Jinjie Ni, Tom Young, Vlad Pandealea, Fuzhao Xue, and Erik Cambria. "Recent advances in deep learning based dialogue systems: A systematic survey." In: *Artificial intelligence review* 56.4 (2023), pp. 3055–3155.
- [64] Carel van Niekerk, Andrey Malinin, Christian Geischauser, Michael Heck, Hsien-chin Lin, Nurul Lubis, Shutong Feng, and Milica Gašić. "Uncertainty measures in neural belief tracking and the effects on dialogue policy performance." In: *arXiv preprint arXiv:2109.04349* (2021).
- [65] OpenAI. *OpenAI GPT-3 API [gpt-3.5-turbo-instruct]*. 2023. URL: <https://platform.openai.com/docs/models/gpt-3-5>.
- [66] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. "The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only." In: *arXiv preprint arXiv:2306.01116* (2023).
- [67] Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. "Soloist: Few-shot task-oriented dialog with a single pretrained auto-regressive model." In: *arXiv preprint arXiv:2005.05298* 3 (2020).

- [68] Olivier Pietquin. *A framework for unsupervised learning of dialogue strategies*. Presses univ. de Louvain, 2005.
- [69] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. "Language models are unsupervised multitask learners." In: *OpenAI blog* 1.8 (2019), p. 9.
- [70] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset." In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 05. 2020, pp. 8689–8696.
- [71] Shubhra Kanti Karmaker Santu and Dongji Feng. "TELeR: A General Taxonomy of LLM Prompts for Benchmarking Complex Tasks." In: *arXiv preprint arXiv:2305.11430* (2023).
- [72] Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. "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*. 2007, pp. 149–152.
- [73] Jost Schatzmann, Blaise Thomson, and Steve Young. "Statistical user simulation with a hidden agenda." In: *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*. 2007, pp. 273–282.
- [74] Jost Schatzmann and Steve Young. "The hidden agenda user simulation model." In: *IEEE transactions on audio, speech, and language processing* 17.4 (2009), pp. 733–747.
- [75] Jost Schatzmann, Matthew N Stuttle, Karl Weilhammer, and Steve Young. "Effects of the user model on simulation-based learning of dialogue strategies." In: *IEEE Workshop on Automatic Speech Recognition and Understanding, 2005*. IEEE. 2005, pp. 220–225.
- [76] Konrad Scheffler and Steve Young. "Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning." In: *Proceedings of HLT*. Vol. 2. 2002.
- [77] Iulian Serban, Alessandro Sordani, Yoshua Bengio, Aaron Courville, and Joelle Pineau. "Building end-to-end dialogue systems using generative hierarchical neural network models." In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.
- [78] Weiyang Shi, Kun Qian, Xuwei Wang, and Zhou Yu. "How to build user simulators to train rl-based dialog systems." In: *arXiv preprint arXiv:1909.01388* (2019).
- [79] Saleh Soltan, Shankar Ananthkrishnan, Jack FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith Peris, Stephen Rawls, Andy Rosenbaum, Anna Rumshisky, et al. "Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model." In: *arXiv preprint arXiv:2208.01448* (2022).

- [80] Sebastian Steindl, Ulrich Schäfer, and Bernd Ludwig. “Generating Synthetic Dialogues from Prompts to Improve Task-Oriented Dialogue Systems.” In: *German Conference on Artificial Intelligence (Künstliche Intelligenz)*. Springer. 2023, pp. 207–214.
- [81] Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. “Multi-task pre-training for plug-and-play task-oriented dialogue system.” In: *arXiv preprint arXiv:2109.14739* (2021).
- [82] Weiwei Sun, Shuyu Guo, Shuo Zhang, Pengjie Ren, Zhumin Chen, Maarten de Rijke, and Zhaochun Ren. “Metaphorical User Simulators for Evaluating Task-oriented Dialogue Systems.” In: *ACM Transactions on Information Systems* (2023).
- [83] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks.” In: *Advances in neural information processing systems* 27 (2014).
- [84] Ryuichi Takanobu, Qi Zhu, Jinchao Li, Baolin Peng, Jianfeng Gao, and Minlie Huang. “Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation.” In: *arXiv preprint arXiv:2005.07362* (2020).
- [85] Silvia Terragni, Modestas Filipavicius, Nghia Khau, Bruna Guedes, André Manso, and Roland Mathis. “In-Context Learning User Simulators for Task-Oriented Dialog Systems.” In: *arXiv preprint arXiv:2306.00774* (2023).
- [86] Manisha Thakkar and Nitin Pise. “Survey of available datasets for designing task oriented dialogue agents.” In: *2019 International Conference on Mechatronics, Remote Sensing, Information Systems and Industrial Information Technologies (ICMRSISIIT)*. Vol. 1. IEEE. 2020, pp. 1–10.
- [87] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. “Llama: Open and efficient foundation language models.” In: *arXiv preprint arXiv:2302.13971* (2023).
- [88] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. “Llama 2: Open foundation and fine-tuned chat models.” In: *arXiv preprint arXiv:2307.09288* (2023).
- [89] Bo-Hsiang Tseng, Yinpei Dai, Florian Kreyszig, and Bill Byrne. “Transferable dialogue systems and user simulators.” In: *arXiv preprint arXiv:2107.11904* (2021).
- [90] Stefan Ultes et al. “PyDial: A Multi-domain Statistical Dialogue System Toolkit.” In: *Proceedings of ACL 2017, System Demonstrations*. Vancouver, Canada: Association for Computational Linguistics, 2017, pp. 73–78. URL: <http://aclweb.org/anthology/P17-4013>.

- [91] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in neural information processing systems* 30 (2017).
- [92] Marilyn A Walker, Diane J Litman, Candace A Kamm, and Alicia Abella. "PARADISE: A framework for evaluating spoken dialogue agents." In: *arXiv preprint cmp-lg/9704004* (1997).
- [93] Ben Wang and Aran Komatsuzaki. *GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model*. <https://github.com/kingoflolz/mesh-transformer-jax>. May 2021.
- [94] Chengwei Wei, Yun-Cheng Wang, Bin Wang, and C-C Jay Kuo. "An overview on language models: Recent developments and outlook." In: *arXiv preprint arXiv:2303.05759* (2023).
- [95] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. "Chain-of-thought prompting elicits reasoning in large language models." In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24824–24837.
- [96] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. "Semantically conditioned lstm-based natural language generation for spoken dialogue systems." In: *arXiv preprint arXiv:1508.01745* (2015).
- [97] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. "A network-based end-to-end trainable task-oriented dialogue system." In: *arXiv preprint arXiv:1604.04562* (2016).
- [98] Jason D Williams, Antoine Raux, and Matthew Henderson. "The dialog state tracking challenge series: A review." In: *Dialogue & Discourse* 7.3 (2016), pp. 4–33.
- [99] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. "Transformers: State-of-the-art natural language processing." In: *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*. 2020, pp. 38–45.
- [100] Jingfeng Yang, Hongye Jin, Ruixiang Tang, Xiaotian Han, Qizhang Feng, Haoming Jiang, Bing Yin, and Xia Hu. "Harnessing the power of llms in practice: A survey on chatgpt and beyond." In: *arXiv preprint arXiv:2304.13712* (2023).
- [101] Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. "Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation." In: *arXiv preprint arXiv:2104.00773* (2021).

- [102] Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. “MultiWOZ 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines.” In: *arXiv preprint arXiv:2007.12720* (2020).
- [103] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. “Personalizing dialogue agents: I have a dog, do you have pets too?” In: *arXiv preprint arXiv:1801.07243* (2018).
- [104] Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and Xiaoyan Zhu. “Recent advances and challenges in task-oriented dialog systems.” In: *Science China Technological Sciences* 63.10 (2020), pp. 2011–2027.
- [105] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. “A survey of large language models.” In: *arXiv preprint arXiv:2303.18223* (2023).
- [106] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. “Calibrate before use: Improving few-shot performance of language models.” In: *International Conference on Machine Learning*. PMLR. 2021, pp. 12697–12706.
- [107] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. “Least-to-most prompting enables complex reasoning in large language models.” In: *arXiv preprint arXiv:2205.10625* (2022).
- [108] Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. “ConvLab-2: An Open-Source Toolkit for Building, Evaluating, and Diagnosing Dialogue Systems.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.
- [109] Qi Zhu et al. “ConvLab-3: A Flexible Dialogue System Toolkit Based on a Unified Data Format.” In: *arXiv preprint arXiv:2211.17148* (2022). URL: <http://arxiv.org/abs/2211.17148>.