

---

# Reinforcement Learning from Implicit Feedback for Conversational Question Answering

---

A dissertation submitted towards the degree  
Doctor of Engineering (Dr.-Ing.)  
of the Faculty of Mathematics and Computer Science  
of Saarland University

by

**Magdalena Theresa Kaiser**



Saarbrücken, 2025

---

---

<b>Day of Colloquium:</b>	04.02.2026
<b>Dean of the Faculty:</b>	Prof. Dr. Roland Speicher
<b>Chair of the Committee:</b>	Prof. Dr. Anna Maria Feit
<b>Reviewers:</b>	Prof. Dr. Gerhard Weikum Dr. Rishiraj Saha Roy Prof. Dr. Arjen P. de Vries
<b>Academic Assistant:</b>	Dr. Ji-Ung Lee

---



### **Declaration of original authorship**

I hereby declare that this dissertation is my own original work except where otherwise indicated. All data or concepts drawn directly or indirectly from other sources have been correctly acknowledged. This dissertation has not been submitted in its present or similar form to any other academic institution either in Germany or abroad for the award of any other degree.

### **Eidesstattliche Versicherung**

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Saarbrücken, 27. August 2025

gez. Magdalena Theresa Kaiser



# Abstract

Conversational systems that enable interactions with users in natural language to satisfy their information needs and assist them in completing their tasks have been a long-standing goal. Recent advancements in Machine Learning and Natural Language Processing have enabled the development of such systems. *Feedback* is essential to continuously improve and adapt these systems to users' needs. This thesis focuses on *Conversational Question Answering (ConvQA)*, where the task is to provide crisp answers to fact-centric questions, formulated in natural language. ConvQA models are usually trained and evaluated on benchmarks of gold-standard question-answer pairs. Manually judging answer correctness is costly and therefore often not available in real-world scenarios. If available, these judgments are often limited in scope and quality. This thesis studies forms of *implicit feedback* to effectively train and improve conversational systems from limited amounts of data, making the following contributions:

- Learning from *implicit user feedback* via *reformulations*: We present CONQUER, a ConvQA model that learns from conversational data without gold-labeled annotations but with access to user feedback using Reinforcement Learning (RL). More precisely, user reformulations are leveraged as an implicit signal that the previous answer was wrong. We train a classifier to detect if two questions are reformulations or express different intents and define rewards based on the reformulation likelihood. CONQUER comes with a conversational dataset augmented with user reformulations, named CONVREF.
- Learning from *data augmentation* guided by *performance-based feedback*: We present REIGN, an RL framework that generates reformulations in a systematic way, to increase the robustness of models to different surface forms. It judiciously selects the most helpful variants (that achieve high answering performance) as additional training data. REIGN comes with a collection of diverse reformulations for robust evaluation.
- Learning from *preferences* based on *delayed system feedback*: First, we present PRAISE, a pipeline for ConvQA based on Large Language Models (LLMs) that uses the final answering performance as feedback signal and treats intermediate information, like relevant evidence, as weakly-labeled data. Second, we present SUIT, an iterative training approach for LLM-based task-oriented dialog systems that determines relevant subgoals from final dialog-level signals by contrasting turns in successful and unsuccessful dialogs.

The proposed methods learn from implicit and delayed feedback and select data carefully to increase performance and robustness of the underlying conversational models using Reinforcement Learning.



# Kurzfassung

Systeme die mit Nutzern in natürlicher Sprache kommunizieren und sie bei ihren Aufgaben unterstützen waren ein lang angestrebtes Ziel. Die Fortschritte im Maschinellen Lernen und in der natürlichen Sprachverarbeitung haben solche Systeme ermöglicht. *Feedback* ist eine wesentliche Voraussetzung dafür, diese Systeme kontinuierlich zu verbessern und an Nutzerbedürfnisse anzupassen. Diese Dissertation adressiert dialog-basierte Frage-Antwort-Systeme (*Conversational Question Answering (ConvQA)*), deren Aufgabe es ist, knappe Antworten auf faktenbasierte, natürlich-sprachliche Fragen zu finden. ConvQA-Modelle werden in der Regel auf Datensätzen mit Frage-Antwort-Paaren in Goldstandard-Qualität trainiert und evaluiert. Eine manuelle Beurteilung der Korrektheit von Antworten ist kostenintensiv, und daher in der Realität selten verfügbar. Falls doch, sind Beurteilungen oftmals eingeschränkt hinsichtlich Menge und Qualität.

Diese Dissertation beschäftigt sich mit *implizitem Feedback* als Quelle für das Trainieren und Evaluieren von Dialogsystemen mit geringen Datenmengen, und präsentiert dazu die folgenden Forschungsbeiträge:

- Lernen aus *impliziten Nutzerbewertungen* durch *Frageumformulierungen*: Wir haben ein Modell für ConvQA namens CONQUER entwickelt, welches in der Lage ist mittels Reinforcement Learning (RL), ohne Goldstandard-Annotationen, aus Nutzerfeedback zu lernen. Umformulierungen von Nutzerfragen dienen als implizites Signal dafür, dass die vorherige Antwort des Systems falsch war. Wir trainieren einen Klassifikator, der entscheidet ob eine Frage eine Umformulierung darstellt oder eine neue Frageabsicht, und definieren die entsprechende Belohnung für das mit RL trainierte System. CONQUER beinhaltet einen Dialog-Datensatz namens CONVREF, der mit Nutzerumformulierungen angereichert wurde.
- Lernen mithilfe von *Datenaugmentation* aus *leistungsbasiertem Feedback*: Wir haben ein RL Framework namens REIGN entwickelt, welches auf systematische Weise Frageumformulierungen generiert, um die Robustheit von Modellen gegenüber unterschiedlichen Ausdrucksformen zu verbessern. Es wählt die hilfreichsten Varianten (die, die beste Antwortqualität erzielen) als zusätzliche Trainingsdaten aus. REIGN beinhaltet eine Sammlung mit diversen Frageumformulierungen für robuste Systemevaluierung.
- Lernen aus *Präferenzen* basierend auf *verzögertem System-Feedback*: Zuerst stellen wir PRAISE vor, eine Pipeline für ConvQA, welche auf großen Sprachmodellen (Large Language Models (LLMs)) basiert. PRAISE verwendet die Antwortleistung des Systems als Feedbacksignal und behandelt Zwischenergebnisse, wie relevante Textpassagen, als schwach annotierte Daten. Danach stellen wir SUIT vor, einen iterativen Trainingsansatz für LLM-basierte, aufgaben-

orientierte Dialogsysteme. SUIF ist in der Lage relevante Teilziele, die während des Dialogs erfüllt wurden, aus dem Feedback am Ende des Dialogs abzuleiten.

Die vorgestellten Methoden lernen aus impliziten und verzögertem Feedback und wählen Trainingsdaten sorgfältig aus, um die Leistung und Robustheit der Frage-Antwort-Systeme durch Reinforcement Learning zu verbessern.

# Acknowledgements

I would like to sincerely thank everyone who accompanied and supported me throughout my PhD journey. First and foremost, I am deeply grateful to my supervisors, Dr. Rishiraj Saha Roy and Prof. Gerhard Weikum for their support, guidance, and trust in my work. Thank you Rishi and Gerhard for your positive attitude, curiosity and the many insightful discussions, they greatly shaped my understanding of research. I am also grateful for the inspiring research environment and the financial support provided by the Max Planck Institute for Informatics. I am grateful to Patrick Ernst for the internship opportunity and the good collaboration. A warm thank you to Prof. Arjen P. de Vries for reviewing my thesis, and to Prof. Anna Maria Feit and Dr. Ji-Ung Lee for serving on my thesis committee. I would like to thank Petra Schaaf and my colleagues from D5, especially the “last of us” - Dang, Ghazaleh, Hiba, Philipp, Phong, Shrestha, Sneha, for creating such a wonderful working atmosphere. I greatly enjoyed our discussions, the many shared lunches, and, of course, our fun kicker sessions. My heartfelt thanks go to my friends Kalle, Jessi, Marina, Nurten, Simina, Nam, Michael, Guillermo, for their support and encouragement. I am happy to have you in my life. Finally, I am deeply grateful to my family for their unconditional love and for always being there for me: to my sister Hanni, the kindest person I know; to my father for inspiring my curiosity and love of learning; and to my mother, whose love and strength remain with me always.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Challenges . . . . .	2
1.3	Prior Work and its Limitations . . . . .	3
1.4	Contributions . . . . .	5
1.5	Ethical Considerations . . . . .	6
1.6	Publications . . . . .	6
1.7	Organization . . . . .	7
<b>2</b>	<b>Background</b>	<b>9</b>
2.1	Conversational Question Answering . . . . .	9
2.1.1	Relevant Concepts . . . . .	10
2.1.2	Conversational Context Understanding . . . . .	11
2.1.3	ConvQA over Knowledge Graphs . . . . .	12
2.1.4	ConvQA over Text . . . . .	14
2.1.5	Task-oriented Dialogs . . . . .	15
2.2	Feedback in Question Answering . . . . .	18
2.2.1	Feedback Categorization . . . . .	18
2.2.2	QA Methods with Feedback . . . . .	20
2.3	Reformulations in Question Answering . . . . .	20
2.4	Reinforcement Learning . . . . .	21
2.4.1	Relevant Concepts . . . . .	21
2.4.2	Challenges . . . . .	24
2.4.3	RL Methods . . . . .	25
2.4.4	RL in the Context of Question Answering and Dialogs . . . . .	27
<b>3</b>	<b>Learning from Implicit User Feedback via Reformulations</b>	<b>31</b>
3.1	Introduction . . . . .	32
3.2	Model and Architecture . . . . .	34
3.2.1	CONQUER KG Representation . . . . .	34
3.2.2	Concepts and Notation . . . . .	35
3.2.3	System Overview . . . . .	35
3.3	Context Entity Detection . . . . .	35
3.4	CONQUER Learning Framework . . . . .	38

3.4.1	RL Model . . . . .	38
3.4.2	RL Training . . . . .	38
3.4.3	Reformulation Prediction . . . . .	41
3.5	Answer Generation . . . . .	42
3.6	Benchmark with Reformulations . . . . .	42
3.7	Experimental Framework . . . . .	44
3.7.1	Setup . . . . .	44
3.7.2	CONQUER Configurations . . . . .	45
3.7.3	Evaluation . . . . .	45
3.8	Results and Insights . . . . .	46
3.8.1	Key Findings . . . . .	46
3.8.2	In-depth Analysis . . . . .	48
3.9	Related Work . . . . .	50
3.10	Conclusion . . . . .	51
<b>4</b>	<b>Learning from Data Augmentation Guided by Performance-based Feedback</b>	<b>53</b>
4.1	Introduction . . . . .	54
4.2	The REIGN Framework . . . . .	56
4.3	Reformulation Category Selector . . . . .	58
4.3.1	Reformulation Taxonomy . . . . .	58
4.3.2	RCS Model Training . . . . .	59
4.3.3	RCS Model Usage . . . . .	61
4.4	Reformulation Generator . . . . .	61
4.4.1	RG Model Training . . . . .	61
4.4.2	RG Model Usage . . . . .	63
4.5	Conversational Question Answering . . . . .	63
4.5.1	ConvQA Model Training . . . . .	63
4.5.2	ConvQA Model Usage . . . . .	65
4.6	Experimental Setup . . . . .	65
4.7	Results and Insights . . . . .	66
4.7.1	Key Findings . . . . .	66
4.7.2	In-depth Analysis . . . . .	68
4.8	Related Work . . . . .	70
4.9	Conclusion . . . . .	71
<b>5</b>	<b>Learning From Preferences based on Delayed System Feedback</b>	<b>75</b>
5.1	Learning from Preferences for Conversational QA . . . . .	76
5.1.1	Introduction . . . . .	76
5.1.2	The PRAISE Method . . . . .	77
5.1.3	Experimental Setup . . . . .	82
5.1.4	Results and Insights . . . . .	83
5.1.5	Related Work . . . . .	85
5.1.6	Conclusion . . . . .	86
5.2	Learning from Preferences for Task-oriented Dialogs . . . . .	90

---

5.2.1	Introduction . . . . .	90
5.2.2	The SUIT Method . . . . .	91
5.2.3	Experimental Setup . . . . .	94
5.2.4	Results and Insights . . . . .	95
5.2.5	Related Work . . . . .	97
5.2.6	Conclusion . . . . .	98
<b>6</b>	<b>Conclusion and Outlook</b>	<b>101</b>
6.1	Summary . . . . .	101
6.2	Discussion . . . . .	103
6.2.1	Lessons Learned . . . . .	103
6.2.2	Limitations . . . . .	107
6.3	Outlook . . . . .	109
	<b>List of Figures</b>	<b>111</b>
	<b>List of Tables</b>	<b>113</b>
	<b>List of Algorithms</b>	<b>115</b>
	<b>Bibliography</b>	<b>117</b>



# Chapter 1

## Introduction

### Contents

---

<b>1.1</b>	<b>Motivation</b>	<b>1</b>
<b>1.2</b>	<b>Challenges</b>	<b>2</b>
<b>1.3</b>	<b>Prior Work and its Limitations</b>	<b>3</b>
<b>1.4</b>	<b>Contributions</b>	<b>5</b>
<b>1.5</b>	<b>Ethical Considerations</b>	<b>6</b>
<b>1.6</b>	<b>Publications</b>	<b>6</b>
<b>1.7</b>	<b>Organization</b>	<b>7</b>

---

### 1.1 Motivation

Creating systems that are able to converse with humans in natural language in order to satisfy a user’s information need and assist in completing a specific task has been a long-standing goal. Learning from feedback is an essential requirement to improve such systems continuously. Recent advancements in Machine Learning and Natural Language Processing have enabled the development of such conversational systems, while the rise of personal assistants like the Google Assistant, Siri, Cortana, Alexa, and recently also ChatGPT, made this mode of interaction popular. Outputs are typically very short, compared to traditional Web search, where the user has to browse through a long list of documents. Therefore, this kind of interaction is particularly suitable when screen space is limited, like on mobile devices and for speech-based interaction.

This thesis focuses on *Conversational Question Answering (ConvQA)*, where the task is to provide *crisp* answers to *information-seeking, factoid* questions, formulated in natural language. Web sources, consisting of text documents, Web tables and structured *Knowledge Graphs (KGs)* (e.g., Wikidata [Vrandečić and Krötzsch(2014)]), are typically used to answer these questions.

Figure 1.1 shows an illustrative conversation with a sequence of turns consisting of user questions and system answers. While the first question is typically complete, the follow-up questions are often very short and certain information is left implicit. Therefore, these questions can only be understood in their conversational context. Resolving them requires more than simple co-reference resolution. In the conversation in Figure 1.1, the second question is asking about the actor who starred as *James*

<i>user question 1:</i>	Who played Claire in the TV show Outlander?
<i>system answer 1:</i>	Caitriona Balfe
<i>user question 2:</i>	What about Jamie?
<i>system answer 2:</i>	Sam Heughan
<i>user question 3:</i>	Theme song?
<i>system answer 3:</i>	The Skye Boat Song
<i>user question 4:</i>	Who wrote the books?
<i>system answer 4:</i>	Diana Gabaldon
<i>user question 5:</i>	How many are there?
<i>system answer 5:</i>	9

Figure 1.1: Conversational QA example

<i>user 1:</i>	Is <i>The Count of Monte Cristo</i> running in any cinema nearby?
<i>system 1:</i>	Yes, CineStar shows it at 6pm and 8:30pm. Which time do you prefer?
<i>user 2:</i>	8:30 works better for us.
<i>system 2:</i>	Sure, do you want me to make a reservation? If so, for how many people?
<i>user 3:</i>	Yes, for three people please.
<i>system 3:</i>	All done, your reservation number is <i>ABC</i> . Anything else I can help you with?

Figure 1.2: Task-oriented dialog example

*Fraser* in the TV series *Outlander*. Not only the name of the show needs to be extracted from the first question, but also the relation *played*. Apart from that, the character’s name is not fully spelled out, and it might be challenging to disambiguate *Jamie* to the correct entity *James Fraser*. Like the second question, there is no verb in the third one. The fourth question implicitly refers to the books the TV series is based on. Thus, the focus shifts to the book series, so that the last question is referring to the number of books (and not the number of seasons in the TV series).

This thesis also addresses *Task-oriented Dialog (ToD)* systems. These enable *goal-oriented conversations* around executing specific tasks, like making reservations or resolving customer service issues. An example dialog can be seen in Figure 1.2, where the system assists the user in making a reservation for a film screening. In these types of dialogs, the system also proactively asks information from the user (e.g., the preferred time or number of people as in Figure 1.2). These are typically predefined concepts relevant in fulfilling specific tasks (making a reservation in this case).

The overall goal of this thesis is to propose methods for effectively training conversational models by leveraging implicit feedback signals.

## 1.2 Challenges

We encounter the following challenges for training robust ConvQA models:

- **Handling the incomplete nature of conversational questions.** As pointed out, conversational questions are short, often incomplete and can be phrased in a sloppy way. Information from the conversation history is needed to correctly interpret the question intent. It is unclear how to best represent the question, considering the following factors:

- the conversation history: Relevant information can be spread over multiple turns;
  - the respective question intent: The way a question is phrased can impact the overall performance in QA. For example, for some intents, verbs are easier to handle than noun phrases, and vice versa [Buck et al.(2018)]. This affects ConvQA even more since there is a greater variability on how to formulate the conversational questions. Ideally, a model should be robust to different surface forms;
  - the underlying QA system: For example, more traditional IR-based QA methods could benefit from adding important terms from previous turns to the current query, while language models prefer fluent input. However, adding irrelevant information from the history may cause performance to drop. Also for some intents, fully-fledged question rewriting will make the question unnecessary complex.
- **Learning from limited amount of supervision data.** There is a large set of real-world conversational data without gold-labeled annotations. Nevertheless, users may give feedback: either explicitly by rating whether they were satisfied with the interaction or by implicit behavioral cues (e.g. reformulations, clicks, or task abandoning).

Supervision data or user signals are often not available for intermediate steps (e.g. no annotations indicating whether retrieved passages contain the correct answer or no user feedback for individual dialog turns but only at the end of the entire dialog). Therefore, methods are needed that are able to leverage implicit and final supervision signals.

Apart from this, augmenting existing data and obtaining supervision data for intermediate steps can be beneficial. However, this can be a costly procedure and may lead to distribution shifts when new data differs too much from the previous style.

- **Selecting high-quality training data.** Not only the quantity but also the quality of the training data can have a large impact on final model performance. Depending on how the training data was created, it is not only limited in scope, but may also contain wrong or incomplete annotations or lack naturalness. Using data augmentation or relying on implicit, potentially noisy signals, can further decrease quality. Also, a method with good performance on the benchmark it was trained on, may perform worse when deployed in the wild. Therefore, methods are needed that are able to carefully select the available data and increase the generalizability and robustness of the QA models.

### 1.3 Prior Work and its Limitations

There has been a lot of research around conversational search and conversational QA over the past years [Zamani et al.(2023)]. As pointed out, a key challenge in ConvQA is to be able to correctly interpret the questions in their conversational context. Earlier work aimed to:

- rewrite the question into completed, self-sufficient forms so that standard QA methods can be applied (e.g., [Vakulenko et al.(2021), Yu et al.(2020), Ke et al.(2022), Chen et al.(2022b)]),
- model the history as additional context to answer the current question (e.g., [Qu et al.(2019a), Qu et al.(2019b), Gupta and Sharma(2021), Qiu et al.(2021)]).

However, for reformulating the question, most works do not take the underlying QA system (and its potential style preferences) and/or the individual question intents into account.

Methods for ConvQA are typically trained and evaluated on benchmarks of gold-standard question-answer pairs. Typical ways for constructing such conversational datasets are:

- via crowdsourcing (e.g., [Choi et al.(2018), Christmann et al.(2019), Adlakha et al.(2022)]),
- applying automatic or semi-automatic methods, like rules or templates (e.g., [Talmor and Berant(2018), Saha et al.(2018)]),
- reusing existing datasets and transform them into open-domain conversational ones (e.g., [Anantha et al.(2021), Li et al.(2022a)]) and more recently,
- prompting large language models (LLMs) for data generation (e.g., [Abbasiantaeb et al.(2024)]).

All of these methods have potential drawbacks: crowdsourcing is costly and therefore the resulting data is often limited in scope. Automatic methods, or using a non-conversational dataset as basis, increase the risk of creating rather artificial datasets that largely differ from real scenarios. Using proprietary LLMs can also become very costly and using LLM-generated data for model training results in propagating LLM biases further. Additionally, this may give LLM-based ConvQA methods an unfair advantage when being trained on this data. One approach to show a QA model’s generalizability and robustness is by training and/or evaluating it on multiple benchmarks. However, the training and evaluation remain limited to the question’s surface forms appearing in these benchmarks.

Incorporating user feedback from conversations in the wild, is a more realistic setup. Existing methods leverage positive feedback in the form of correct answers to augment the training data (e.g., [Abujabal et al.(2018), Kratzwald and Feuerriegel(2019), Zhang et al.(2019), Campos et al.(2020)]). However, such explicit feedback is hard to obtain at scale and hard to ask from average users who - in a realistic setting - would not know the correct answer upfront. Implicit cues, like click logs that are well studied in Web search (e.g., [Joachims et al.(2007), Hua et al.(2013)]), are preferable since they are less intrusive. Suitable implicit feedback for the conversational setting needs to be determined and harvested.

Another aspect is the lack of training data for intermediate steps, like for evidence retrieval or for verifying correctness of different dialog flows for task-oriented systems. Supervision signals are often only available as final answers or with respect to whether a user’s goal is fulfilled at the end of a dialog. Earlier works on Task-oriented Dialog (ToD) have used Reinforcement Learning (RL) approaches to optimize for such sparse signals [Zhao et al.(2019), Lubis et al.(2020), Vlastelica et al.(2023)], while most LLM-based ToD systems neglect these signals and only optimize next turn predictions. In the context of search and QA, several works applied RL to automatically generate or retrieve reformulations that would proactively result in the best system response [Buck et al.(2018), Nogueira and Cho(2017), Das et al.(2019), Ponnusamy et al.(2020)]. In the context of LLMs, Reinforcement Learning from Human Feedback (RLHF) [Ziegler et al.(2019), Ouyang et al.(2022)] has been proposed to align LLMs to human preferences. Since this is again a costly procedure that require lots of human feedback, alternatives that use automatic feedback have been studied recently (e.g., [Yuan et al.(2024), Peng et al.(2023)]).

Reinforcement learning is particularly suitable because not only positive but also negative feedback can be leveraged [Sutton and Barto(2018)]. Additionally, non-differentiable metrics can be

used as reward signals for training RL models, while this is not possible when training end-to-end supervised models. However, applying RL also comes with many challenges: RL methods require lots of suitable interaction data for good exploration. Stable training, in which the model learns steadily without divergence in a predictable and reliable way, is often challenging. Also, how to translate the ConvQA problem into a suitable RL problem is not clear upfront.

## 1.4 Contributions

This thesis makes the following contributions towards effectively training ConvQA models, with the goal of tackling the challenges and overcoming prior work’s limitations outlined above:

- **Learning from implicit user feedback via reformulations.** In this work, we aim to learn from conversational data without gold-labeled annotations but with access to user feedback. Instead of asking the user to explicitly mark answers as correct or wrong, the approach in this dissertation opts for a more natural interaction paradigm that does not pose any overhead to the user: we leverage reformulations by the user as an implicit signal that the previous answer was wrong. More precisely, we develop a reinforcement learning model, termed CONQUER, that leverages the reformulation likelihood as reward signal. We also create a predictor that classifies whether a follow-up question is a reformulation or expresses a new intent. CONQUER models the answering process as multiple agents walking in parallel on the knowledge graph. Our knowledge graph representation is able to effectively handle large/real-world knowledge graphs. CONQUER also comes with a conversational dataset augmented with user reformulations, named CONVREF. The results of this work have been published as a full paper at SIGIR 2021 [Kaiser et al.(2021)].
- **Learning from data augmentation guided by performance-based feedback.** In our work on CONQUER, we had access to user reformulations. In this work, we generate reformulations in a systematic way to increase the robustness of models to different surface form variations. We propose REIGN, a framework that guides ConvQA models towards higher performance by feeding it only those reformulations that help improve the answering quality, using deep reinforcement learning and QA performance metrics as reward. We show that ConvQA models robustly trained with reformulations significantly outperform those trained with benchmark QA pairs only. We also investigate the model’s robustness by training on one benchmark and applying it zero-shot to another. Additionally, we collect and release large numbers of diverse reformulations generated by ChatGPT for robust evaluation. The results of this work have been published as a full paper at WSDM 2024 [Kaiser et al.(2024b)].
- **Learning from preferences based on delayed system feedback.** In the following works, we aim to create training data for individual subtasks to improve LLM-based models using the models’ own generations and task-level supervision signals only.

First, we propose a pipeline-based approach for ConvQA, named PRAISE, that trains LLMs for each ConvQA subtask. PRAISE uses the final answering performance as feedback signal without human intervention and treats intermediate information, like relevant evidence, as weakly labeled data. We apply Direct Preference Optimization (DPO) by contrasting successful and

unsuccessful samples for each subtask. The results of this work have been published as a short paper at WWW 2025 [Kaiser and Weikum(2025)].

Second, we propose an iterative training approach, termed **SUIT**, for improving task-oriented dialog systems. We sample dialogs from the model we aim to improve and evaluate success at the end of the dialogs, where successful dialogs are preferred over unsuccessful ones. Instead of using all dialog turns of successful dialogs as training data, we use distant supervision and counterfactual learning to determine subgoals that contribute to dialog success to obtain high quality training data. We show how this data improves supervised fine-tuning or, alternatively, preference learning. The results of this work have been published as a short paper at EMNLP 2024 [Kaiser et al.(2024a)].

## 1.5 Ethical Considerations

There are no negative ethical and societal concerns arising from this thesis. The data used in this thesis comes from public benchmarks and from a user study which did not include the collection of any sensitive or personally identifiable information (PII). Our prompts to large language models did not evoke adversarial, hateful, or malicious responses.

## 1.6 Publications

The results of this thesis have been published in the following conference articles, whose lead author is the author of this thesis:

- **Reinforcement Learning from Reformulations in Conversational Question Answering over Knowledge Graphs**  
Magdalena Kaiser, Rishiraj Saha Roy and Gerhard Weikum  
*The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021.*
- **Robust Training for Conversational Question Answering Models with Reinforced Reformulation Generation**  
Magdalena Kaiser, Rishiraj Saha Roy and Gerhard Weikum  
*The 17th ACM International Conference on Web Search and Data Mining, 2024.*
- **Preference-based Learning with Retrieval Augmented Generation for Conversational Question Answering**  
Magdalena Kaiser and Gerhard Weikum  
*Companion Proceedings of the ACM Web Conference, 2025*
- **Learning from Relevant Subgoals in Successful Dialogs using Iterative Training for Task-oriented Dialog Systems**  
Magdalena Kaiser, Patrick Ernst and Gyuri Szarvas  
*Findings of the Conference on Empirical Methods in Natural Language Processing, 2024.*

There are related works by the author that are not part of this thesis:

- **CROWN: Conversational Passage Ranking by Reasoning over Word Networks**

Magdalena Kaiser, Rishiraj Saha Roy and Gerhard Weikum

*Conversational Assistance track (CAST) at the Text REtrieval Conference (TREC), 2019.*

- **Conversational Question Answering over Passages by Leveraging Word Proximity Networks**

Magdalena Kaiser, Rishiraj Saha Roy and Gerhard Weikum

*System demonstration at the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020.*

- **Incorporating User Feedback in Conversational Question Answering over Heterogeneous Web Sources**

Magdalena Kaiser

*Doctoral Consortium at the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020.*

## 1.7 Organization

This dissertation is structured as follows. Chapter 2 provides the necessary background information on conversational question answering, reinforcement learning and reformulations. Chapter 3 presents CONQUER, our RL model that can learn from implicit feedback in the form of question reformulations. Chapter 4 describes the REIGN framework that employs RL for generating reformulations for model-specific data augmentation. Chapter 5 presents two methods for improving LLM-based models by learning from preference data. Finally, Chapter 6 concludes this thesis and gives an outlook.



# Chapter 2

## Background

### Contents

---

<b>2.1</b>	<b>Conversational Question Answering</b>	<b>9</b>
2.1.1	Relevant Concepts	10
2.1.2	Conversational Context Understanding	11
2.1.3	ConvQA over Knowledge Graphs	12
2.1.4	ConvQA over Text	14
2.1.5	Task-oriented Dialogs	15
<b>2.2</b>	<b>Feedback in Question Answering</b>	<b>18</b>
2.2.1	Feedback Categorization	18
2.2.2	QA Methods with Feedback	20
<b>2.3</b>	<b>Reformulations in Question Answering</b>	<b>20</b>
<b>2.4</b>	<b>Reinforcement Learning</b>	<b>21</b>
2.4.1	Relevant Concepts	21
2.4.2	Challenges	24
2.4.3	RL Methods	25
2.4.4	RL in the Context of Question Answering and Dialogs	27

---

This chapter introduces the relevant background information for this thesis. Section 2.1 presents related work and salient concepts for conversational question answering. Section 2.2 provides context for how feedback applied in question answering and how different types of feedback can be categorized. Section 2.3 touches upon the usage of reformulations in question answering. The chapter concludes with Section 2.4 which provides a short introduction to reinforcement learning, including relevant concepts and methods and its usage in question answering and related fields.

### 2.1 Conversational Question Answering

Conversational QA (ConvQA) falls under the umbrella of Conversational Information Seeking (CIS) [Zamani et al.(2023)]. In contrast to Conversational Search [Dalton et al.(2020)], where answers are ranked lists of documents or passages, answers in ConvQA are more fine-grained, short and direct. Methods can be divided based on the used knowledge source: i) structured knowledge in the

form of knowledge graphs [Saha et al.(2018), Christmann et al.(2019), Guo et al.(2018), Lan and Jiang(2021)] and tables [Iyyer et al.(2017), Müller et al.(2019)], ii) unstructured textual sources [Choi et al.(2018), Qu et al.(2020)] and iii) heterogeneous sources, combining text, tables [Deng et al.(2022)] and KGs [Christmann et al.(2022b), Min et al.(2025)] or even multimodal, with text, tables and images [Li et al.(2022a)].

In the following, we introduce relevant ConvQA concepts and discuss the important subtask of *conversational context understanding*. Then, we discuss ConvQA over knowledge graphs and ConvQA over textual sources in more detail. Finally, since we also touch upon task-oriented dialogs in this thesis, we introduce this related area, including its relevant concepts.

### 2.1.1 Relevant Concepts

In the following, we define important ConvQA concepts, using the example conversation from chapter 1.1 for illustration:

$Q_1$ : *Who played Claire in the TV show Outlander?*  
 $A_1$ : Caitriona Balfe  
 $Q_2$ : *What about Jamie?*  
 $A_2$ : Sam Heughan  
 $Q_3$ : *Theme Song?*  
 $A_3$ : The Skye Boat Song  
 $Q_4$ : *Who wrote the books?*  
 $A_4$ : Diana Gabaldon  
 $Q_5$ : *How many are there?*  
 $A_5$ : 9

**Conversation.** A conversation  $C$  is a sequence of questions and corresponding answers around a topic of interest,  $\langle Q_1, A_1, Q_2, A_2, \dots, Q_n, A_n \rangle$ . Each question-answer pair  $\langle Q_i, A_i \rangle$  constitutes a *turn*  $t_i$ . The example conversation consists of 5 turns.

**Intent.** A distinct information need in  $C$  is referred to as an intent  $I$ . The example conversation expresses five intents  $\langle I_1, \dots, I_5 \rangle$ , one per turn.

**Question.** A question  $Q$  consists of a sequence of tokens  $\langle q_1 \dots q_m \rangle$ . It is issued by the user to instantiate a specific intent in the conversation.  $Q$  can either be complete (= explicit expression of intent), like *Who played Claire in the TV show Outlander?* ( $Q_1$ ), or incomplete (= implicit expression of intent), like *What about Jamie?* ( $Q_2$ ). It can be expressed fluently in natural language, like *Who wrote the books?* ( $Q_4$ ) or a more telegraphic style, like *Theme Song?* ( $Q_3$ ), can be used.

**Question Reformulation.** In this thesis, question reformulations are defined as transformations from a question into different surface forms that express the same intent. The question *What about Jamie?* ( $Q_2$ ) can be reformulated in various different ways, for example:

$Q_{22}$ : *Who played the role of Jamie?*,  
 $Q_{23}$ : *Which actor is behind Jamie in the Outlander series?*,  
 $Q_{24}$ : *Who starred as James Fraser?*, or  
 $Q_{25}$ : *Cast member Jamie?*.

Reformulations can express the intent in an explicit or implicit way, just like the original questions.

**Answer.** An answer  $A$  is a response to the information need in question  $Q$ . Answers may have different granularity, varying from entire passages, sentences or short phrases to single named entities. In the example conversation, answers are entities ( $A_1 - A_4$ , such as *Diana Gabaldon* for  $Q_4$ ) or literals ( $A_5, 9$ ).

**Context questions.** Context questions  $Q_t^{cxt}$  are the questions in previous turns that are most relevant to the intent expressed at turn  $t$ . Typically, this set comprises a few of the immediately preceding questions  $Q_{t-1}, Q_{t-2}, \dots$ , or include the first question ( $Q_1$ ) in  $C$  as well. For example, for the question *Theme song?* ( $Q_3$ ), only the first question mentioning the name of the show is relevant, while for question *How many are there?* ( $Q_5$ ), the preceding question that sets the focus on the book series, as well as the first question, are relevant.

**Context entities.** At any given turn  $t$ , context entities  $E_t^{cxt}$  are the ones most relevant to the conversation so far. For example, the entity *Outlander*, mentioned in  $Q_1$ , is relevant for the entire conversation.

### 2.1.2 Conversational Context Understanding

There are two lines of work: i) approaches that model the conversation history and apply methods that can incorporate the respective context information, and ii) approaches that complete the conversational question with information from the conversation history and then apply standard QA methods.

**History modeling approaches.** The most straightforward way to deal with the conversational context is to append all previous QA turns to the current question [Dalton et al.(2020), Guo et al.(2018), Shen et al.(2019)]. Alternatively, for selecting the most relevant turns, methods either use heuristics (e.g. first turn or previous  $k$  turns are considered to be the most relevant) [Choi et al.(2018), Dalton et al.(2020), Kaiser et al.(2020), Qu et al.(2020)], or aim to learn the relevance of specific conversational turns [Qu et al.(2019a), Qiu et al.(2021)]. Instead of selecting a fixed set, some methods perform a soft history selection, by learning a weighting for different parts of the context [Qu et al.(2019b), Tran et al.(2024)], or by explicitly modeling the conversational flow and constructing a question-aware encoding [Chen et al.(2020), Huang et al.(2019a)].

Typically, the selected history is encoded (along with the current question) using transformer-based models and attention mechanisms [Qu et al.(2020), Choi et al.(2018), Guo et al.(2018)]. In the case of ConvQA over KGs (see section 2.1.3), it is also modeled as a contextual graph [Christmann et al.(2019), Lan and Jiang(2021)]. For open-domain ConvQA, methods sometimes also adjust the learning of the retriever [Yu et al.(2021), Mao et al.(2022), Hai et al.(2023), Mao et al.(2024)].

**Question completion approaches.** Inspired by more traditional IR approaches, question expansion strategies can be used to learn to add relevant terms from the conversation history to the current question [Voskarides et al.(2020), Kim et al.(2021)]. A semantic question represen-

tation in form of a logical structure with slots for contextual information is proposed in [Christmann et al.(2022b), Christmann et al.(2023)]. Several works perform rewriting of the conversational questions into self-sufficient forms [Vakulenko et al.(2021), Yu et al.(2020), Ke et al.(2022), Chen et al.(2022b), Ishii et al.(2022a), Raposo et al.(2022)]. Conversational question rewriting became a separate subtask for ConvQA, which was spurred by the development of question completion datasets. These datasets provide questions in its dialog context along with a context-independent rewrite of that question [Elgohary et al.(2019), Anantha et al.(2021)]. More recently, large language models have shown strong capabilities in generating fluent self-sufficient questions, given the conversation history [Mao et al.(2023), Ye et al.(2023)]. Once the question is completed, traditional non-conversational approaches, like the retriever-reader architecture [Chen et al.(2017), Karpukhin et al.(2020), Izacard and Grave(2021)] in the case of ConvQA over text [Qu et al.(2020)], can be applied to answer the conversational question.

### 2.1.3 ConvQA over Knowledge Graphs

Knowledge graphs enable answering fact-centric questions over a structured source of knowledge. In the following, we first introduce some basic KG concepts and then present the corresponding ConvQA over KG methods and benchmarks.

#### 2.1.3.1 Knowledge Graph Concepts

A Knowledge Graph (KG) consists of a set of objective facts about real-world entities. Examples of large curated KGs (equivalently, knowledge bases or KBs) include Wikidata [Vrandečić and Krötzsch(2014)], YAGO [Suchanek et al.(2007)], DBpedia [Auer et al.(2007)], or industrial ones (e.g., Google KG). A KG  $K$  is typically stored as an RDF database organized into *subject-predicate-object triples*. As of early 2025, Wikidata had 1.65 billion facts. Figure 2.1 displays an example KG snippet ((a): KG statements in triple format, (b): graph-based KG representation).

**Fact.** A KG fact is an SPO (subject, predicate, object) triple, where a subject is an *entity* (like `Sam Heughan`); an object is another entity (like `actor`), a *type* (like `human`), or a *literal* (`30 April 1980`); and a predicate is a relationship (`cast member`) between the subject and the object. Entities are real-world objects like people, locations, organizations, products, etc. In the graph representation in Figure 2.1, entities and literals are in red in squared boxes, predicates in blue in oval boxes. One example triple is `(Sam Heughan, date of birth, 30 April 1980)`. The graph is directed with edges from subject towards the object. For bidirectional facts, like parent-child relations, triples exist for both directions.

**N-ary facts.** Large KGs also contain *n-ary* facts (referred to as *qualifiers* in Wikidata) that involve more than two entities or literals. *n-ary* facts are typically represented as a main fact enhanced with qualifiers, that are (possibly multiple) auxiliary `(predicate, object)` pairs adding contextual information to the main fact. In the KG snippet in Figure 2.1, examples for such n-ary facts are highlighted in green and blue: the fact `(Outlander, cast member, Caitriona Balfe)` is the main fact that is contextualized with the additional information `(character role, Claire Fraser)` to express the relationship between series, actor and her role in the movie; the fact `(Sam Heughan, award received, Saturn Award for Best Actor)` is concretized with the qualifier facts `(for role in,`

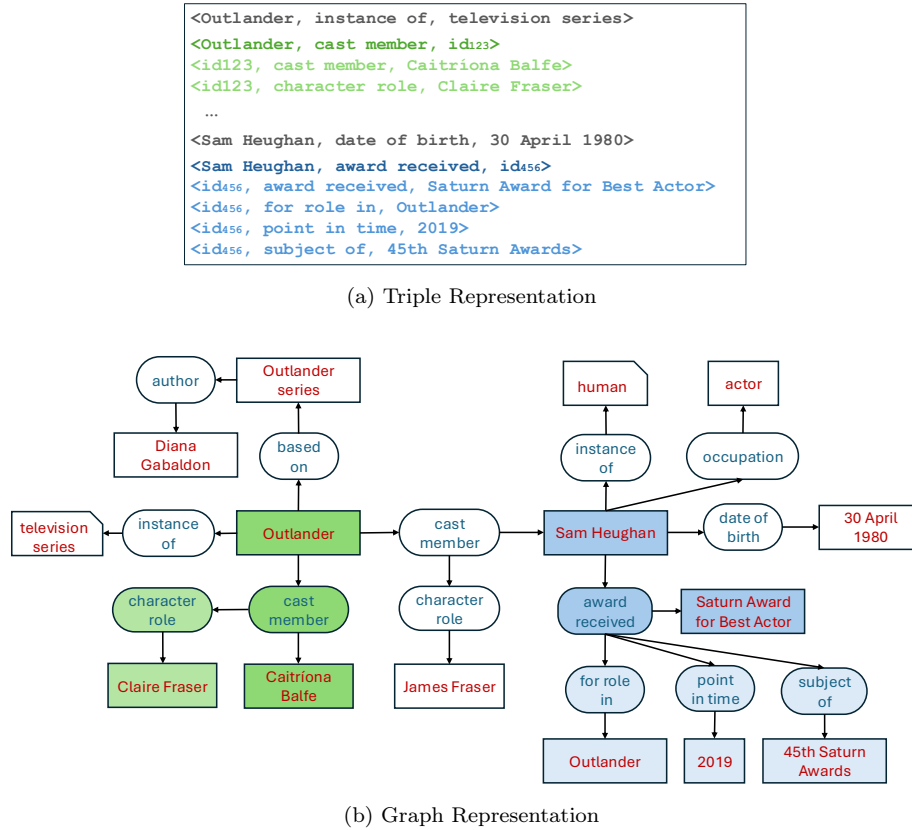


Figure 2.1: Example of a knowledge graph snippet.

Outlander),  $\langle \text{point in time, 2019} \rangle$  and  $\langle \text{subject of, 45th Saturn Awards} \rangle$  to express when the actor received which award for which role. N-ary facts are stored as triples via *reification*: the different parts of a fact are connected with each other using unique fact-specific identifiers. In the triple representation in Figure 2.1,  $id_{123}$  connects the entities Outlander, Caitriona Balfe and Claire Fraser with each other, and  $id_{456}$  connects Sam Heughan with Saturn Award for best Actor, Outlander, 2019, 45th Saturn Awards. A large part of QA research disregards qualifiers, but they contain valuable information [Nguyen et al.(2014), Oguz et al.(2020), Leeka et al.(2016), Hernández et al.(2015), Galkin et al.(2020)] and constitute a substantial fraction of Wikidata and other KGs. Qualifiers are required to be able to answer questions like *Who played Claire in Outlander?*, *Who was US president in 2018?* or *What is the second book in the series?*

**Mention.** A mention refers to a sequence of tokens in question  $Q$  that is the surface form of a KG item (entity, predicate, or type). A mention of a predicate is referred to as a *relation*. For example, in *Who played Claire in the TV show Outlander?*, the phrases *Claire*, *TV show*, *Outlander* and *played* are mentions of the KG entity *Claire Fraser*, the KG type *television series*, the entity *Outlander*, and the KG predicate *cast member*, respectively. Named Entity Disambiguation (NED) methods [Ferragina and Scaiella(2010), Christmann et al.(2022a)] are used to detect the mentions and map them to the KG.

### 2.1.3.2 ConvQA over KG Methods

There are two lines of work handling conversational questions over KGs. The first line deals with semantic-parsing based approaches [Guo et al.(2018), Shen et al.(2019), Marion et al.(2021), Kacupaj et al.(2021)]. The question is mapped to logical forms following a structured template. Logical forms are then executed over the KG to obtain answers. This way, complex questions that require reasoning and aggregations can also be handled. However, the expressiveness of the respective semantic-parsing template determines the scope of question complexity that can be answered. The second line consists of graph-based approaches [Christmann et al.(2019), Lan and Jiang(2021), Kacupaj et al.(2022), Jain and Lapata(2024)]. These operate directly on a subset of KG nodes and edges by mapping relevant entities from the conversational context to a graph representation. More precisely, [Christmann et al.(2019)] uses an unsupervised graph exploration algorithm that judiciously expand the graph to find candidate answers, while [Kacupaj et al.(2022)] proposes a contrastive learning approach to rank KG paths. Graph neural networks are employed in [Lan and Jiang(2021)], along with LLMs in [Jain and Lapata(2024)].

**Benchmarks.** There are two prominent benchmarks for ConvQA over KGs, namely ConvQuestions [Christmann et al.(2019)] and CSQA [Saha et al.(2018)]. Both of them operate over Wikidata. ConvQuestions contains 11200 conversations from five different domains, which were created by crowdworkers. The resulting questions contain complex intents requiring comparisons, aggregations and complex reasoning. CSQA is a large-scale dataset (containing 200k dialogs with 1.6M turns) that was collected through a semi-automatic process. The dataset contains simple questions and complex ones that require quantitative and logical reasoning, as well as clarification questions.

**Metrics.** An answer is typically an entity, a list of entities, or a literal. *Precision@1* (P@1) is an important metric, determining the correctness of the first answer returned to the user, particularly relevant for voice-based interactions. For systems that return ranked answer lists, other proposed metrics are *Hit@k* (with  $k$  being a small number like 5), determining whether the correct answer is present in the top- $k$  results, and *Mean Reciprocal Rank* (MRR), with  $MRR = \frac{1}{rank_i}$ , where  $rank_i$  is the first rank at which a correct answer was found. Additionally, when answers are lists of entities, *precision*, *recall* and *F1 score* (harmonic mean between precision and recall) are used. Results are averaged over all test questions.

### 2.1.4 ConvQA over Text

Early ConvQA systems over textual sources considered the problem as a machine reading comprehension (MRC) task, where the relevant passages or documents are given upfront [Choi et al.(2018), Reddy et al.(2019)]. Most conversational MRC models use the transformer architecture and jointly embed questions and passages [Reddy et al.(2019)].

Instead of knowing relevant passages or documents upfront, [Qu et al.(2020)] proposed open-retrieval ConvQA by including a retrieval stage. For non-conversational open-domain QA, applying retriever-reader models is a popular choice [Chen et al.(2017), Karpukhin et al.(2020), Izacard and Grave(2021)]. These models are also used for the conversational setting [Qu et al.(2020)], typically by appending the conversation to the question. Other methods adjust the learning of the retriever: conversational dense retrieval was proposed, in which a contextualized embedding of the conversa-

tional queries is learned [Yu et al.(2021), Mao et al.(2022), Mao et al.(2024)]. Also, efficient sparse retrieval [Formal et al.(2021)] is adjusted to the conversational setting in order to learn relevant terms based on the question and its conversational context [Hai et al.(2023)].

**Benchmarks.** Two prominent Conversational MRC datasets are CoQA [Reddy et al.(2019)] and QuAC [Choi et al.(2018)]. CoQA contains 8k, QuAC 14k conversations. Both were created by two crowdworkers chatting about a passage (from Wikipedia in the case of QuAC and from seven domains spanning from science exam questions to news and Wikipedia articles in case of CoQA).

QuAC was extended to the open-retrieval setting (OR-QuAC) [Qu et al.(2020)], by creating a collection of over 11 million passages using the whole Wikipedia corpus. Another open-domain conversational dataset is TOPIOCQA [Adlakha et al.(2022)]. The dataset contains around 3.9k conversations, with 13 turns on average and multiple topic shifts per conversation.

In the area of conversational search, the Conversational Assistance Track for the Text Retrieval Challenge (TREC CAsT 2019-2022) [Dalton et al.(2020), Dalton et al.(2022), Dalton et al.(2021), Owoicho et al.(2022)] is notable to mention. It aimed to establish a reusable benchmark for open-domain information-centric conversational dialogs. Dialogs in TREC CAsT are constructed manually and contain manual relevance judgment with passages from corpora, such as MS MARCO [Nguyen et al.(2016)] or TREC Complex Answer Retrieval (CAR) [Dietz et al.(2018)]. The size of this data is rather small (50 test dialogs in the first year, 25 new ones in the following years).

**Metrics.** Answers are typically short phrases or sentences. *F1 score* is the main metric, computed by considering the overlapping words in the prediction and references [Choi et al.(2018), Reddy et al.(2019)]. For the open-domain case, *MRR* and *recall* are used to evaluate the retrieval performance for the retriever and reranker [Qu et al.(2020)]. Further retrieval-oriented metrics are used in [Dalton et al.(2020)], namely *NDCG@k* (Normalized Discounted Cumulative Gain at rank  $k$ ), *MRR@k*, *Recall@k*, and *MAP* (Mean Average Precision).

### 2.1.5 Task-oriented Dialogs

Task-oriented dialog (ToD) systems aim to assist users in accomplishing certain goals like making reservations. Such goal-oriented conversations share some properties with standard ConvQA methods, while being distinct in the following ways. On the one hand, the classical ToD setting is more limited than question answering since no open-ended questions, but questions tailored to fulfill (domain) specific tasks, are answered. Typically, domains, intents and slot representations are pre-defined. On the other hand, these systems fulfill the important conversational property of being *mixed-initiative* [Yang et al.(2004)]: the system *proactively* asks for more information about a certain feature or recommends certain entities that might be interesting to the user, while classical (Conv)QA systems are only driven by the user’s questions.

In the following, we introduce the classical ToD concepts and give a short overview of ToD methods. Task-oriented dialog systems have to solve multiple subtasks. These are exemplified using the introductory sample dialog. Figure 2.2 shows the respective subtasks for the first system turn  $S_1$ , the full dialog from Section 1.1 was:

$U_1$  : *Is The Count of Monte Cristo running in any cinema nearby?*

$S_1$  : Yes, CineStar shows it at 6pm and 8:30pm. Which time do you prefer?

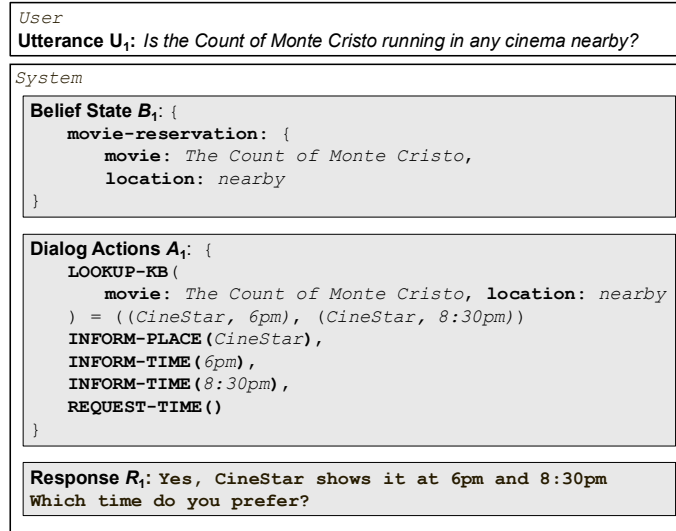


Figure 2.2: An example of a system turn in a task-oriented dialog, with dialog state, action and response (for the first system turn  $S_1$  in the example).

$U_2$ : *8:30 works better for us.*

$S_2$ : *Sure, do you want me to make a reservation? If so, for how many people?*

$U_3$ : *Yes, for three people please.*

$S_3$ : *All done, your reservation number is ABC. Anything else I can help you with?*

### 2.1.5.1 Concepts for Task-oriented Dialogs

**User goal.** A user goal  $G$  is typically defined as a set of constraints (e.g. suitable time in case of the movie screening) and information needs (e.g. the reservation number that needs to be provided to the user). For the movie screening example above, the user goal is to reserve tickets with the following constraints: the movie is *The Count of Monte Cristo*, the cinema is nearby, the preferred time is after 6pm and the number of people is 3. The information need consists of the cinema's name (*CineStar*) and the reservation number (*ABC*), obtained after successful reservation from the system. Whether the user goal is satisfied can usually only be determined at the end of the dialog. The goal is fulfilled for the example above since all constraints are met and all relevant information is provided to the user.

**Dialog state tracking.** This subtask corresponds to understanding the conversational context. The ToD system keeps a structured representation of the current *dialog state*, sometimes also called *belief state*,  $\mathcal{B}$ . It consists of a set of *slot-value* pairs (e.g. slot=*movie*, value=*The Count of Monte Cristo* is one such pair in  $B_1$  in Figure 2.2).

**Dialog action prediction.** This subtask is about selecting one or multiple *dialog actions*, which are semantic representations expressing how the system should react, from a fixed set  $\mathcal{A}$  of available actions, given the current dialog state. Actions can be internal operations (e.g. knowledge base look-ups, like the first in the set  $A_1$  in Figure 2.2: it searches the KB for cinemas showing

the movie *The Count of Monte Cristo* at a nearby location and returning the matched cinemas along the available screening times), or actions can be directed to the user (e.g. actions 2 – 5 from  $A_1$ , where `INFORM-PLACE(.)` and `INFORM-TIME(.)` provides the user with the respective cinema location and screening time, while `REQUEST-TIME(.)` elicits the preferred time from the user).

**Response generation.** This subtask deals with formulating a natural language response  $R$  as output for the user that conveys the information represented by the selected dialog actions (e.g. see  $R_1$  in Figure 2.2).

### 2.1.5.2 Methods for Task-oriented Dialogs

Reinforcement learning methods have been widely used for modeling decisions in task-oriented dialogs [Levin et al.(1998), Young et al.(2010)] (see more details in section 2.4.4). Historically, several models were learned for each of the subtasks [Budzianowski et al.(2018), Rastogi et al.(2020)]. Contemporary ToD systems are based on pre-trained Large language models (LLM) [Hosseini-Asl et al.(2020), Lee(2021), He et al.(2022), Ramachandran et al.(2022), Sun et al.(2023), Snell et al.(2022), Bang et al.(2023)], that are trained in an end-to-end fashion, and generate, token by token, the predicted belief state, dialog actions and the response. [Hosseini-Asl et al.(2020)], one of the first of such models, optimized for all tasks jointly by training a causal language model on all subtasks recasted as a single sequence prediction problem. With increasing LLM capabilities, the focus has shifted to move beyond pre-defined slot-value pairs to automatically learn such information from data [Vukovic et al.(2024)], deal with complex, fuzzy goals [Mo et al.(2024)] and combine chit-chat with goal completion [Chen et al.(2022a), Liu et al.(2023)] and tool use [Shen et al.(2024)].

**Benchmarks.** MultiWOZ [Budzianowski et al.(2018)] is one of the most widely used ToD benchmarks. It contains 10k human-human dialogs over 7 domains and fine-grained annotations for all three subtasks. Larger datasets are the schema-guided dialogue dataset (SGD) dataset [Rastogi et al.(2020)], with around 20k conversations and AirDialogues [Wei et al.(2018)], with 300k goal-oriented conversations. Further datasets that particularly aim for realistic conversations are proposed in [Chen et al.(2021), Chen et al.(2022a)]. Beyond traditional ToD benchmarks, the Amazon Alexa Prize TaskBot Challenge [Gottardi et al.(2022)] combines elements of dialogue systems for task navigation with task-specific and open-domain QA.

**Metrics.** *Slot-based F1* scores are typically used to measure performance for the dialog state tracking and dialog action prediction subtasks. Dialog-level scores are obtained by macro-averaging F1 scores (sometimes also accuracy) for the requested slots over all dialog turns. The fluency of the generated response is measured by calculating (*delexicalized*) *BLEU* scores based on one, or multiple, reference responses, where *delexicalized* means using placeholders such as `LOCATION` instead of concrete entities. For measuring whether the user goal is fulfilled at the end of the dialog, two binary metrics (*INFORM* and *SUCCESS* rates) are proposed in the case of MultiWOZ: The inform rate is 1 if an appropriate entity (that fulfills the user’s constraints) has been predicted by the model. The success rate is 1 only if the inform rate is 1 and if the system provided all attributes requested by the user (e.g. the confirmation number).

Table 2.1 summarizes the most relevant ConvQA and ToD evaluation metrics.

Metric	Definition	Description
<b>F1</b>	$prec = \frac{tp}{tp+fp}, rec = \frac{tp}{tp+fn},$ $2 \cdot \frac{prec \cdot rec}{prec+rec}$	$tp$ : answer tokens in prediction and reference, $fp$ : only in prediction, $fn$ : only in reference
<b>P@1</b>	$A_1^{pred} == A^{gold}$	whether correct answer at rank 1
<b>Hit@k</b>	$A_{1..k}^{pred} == A^{gold}$	whether correct answer among top- $k$ ranks
<b>MRR</b>	$\frac{1}{n} \sum_i^n \frac{1}{rank_i}$	average position of the first correct answer ( $rank_i$ )
<b>NDCG@k</b>	$DCG_k = \sum_{i=1}^k \frac{rel_i}{\log_2(i+1)},$ $nDCG_k = \frac{DCG_k}{IDCG}$	relevance of each result divided by its rank (up to $k$ ) normalized by ideal DCG (IDCG)
<b>BLEU</b>	$BP \cdot e^{\sum_{n=1}^N w_n \log p_n}$	n-gram overlap with references, $p_n$ : n-gram prec., $w_n$ : weight, $BP$ : brevity penalty
<b>INFORM</b>	$e_R \in E_{gold}$	whether entity satisfies user’s constraints (in $E_{gold}$ )
<b>SUCCESS</b>	$e_R \in E_{gold} \cap$ $\forall attr \in \{Attr\}_{req} : attr \in R$	whether $INFORM = 1$ and all requested attributes present in response $R$

Table 2.1: Typical metrics for evaluating ConvQA (upper part) and dialog models (lower part).

## 2.2 Feedback in Question Answering

### 2.2.1 Feedback Categorization

Different types of feedback have been considered in the context of QA. We categorize feedback according to the following dimensions: *source*, *expression*, *timing* and *granularity*:

- **Source:** We can distinguish between *user* and *system* feedback.

User feedback is important to adapt models to users’ needs. User feedback includes user ratings, comments, preferences or cues based on user behavior, such as asking clarification questions or refining previously asked questions.

System feedback can be further divided into *intrinsic* feedback, describing the model’s own feedback (for example, the model confidence), and *extrinsic*, referring to feedback from other systems evaluating the model output (e.g., by using LLMs as judges).

Obtaining system feedback is less effort than harvesting user feedback, while user feedback is a stronger signal. However, both sources might be biased: the preference of one user might not be representative for all kinds of users. LLMs as judges might prefer outputs from models trained on similar data, and therefore may be considered less trustworthy.

- **Expression:** Feedback can be *explicit*, directly stating user satisfaction. For example, clicking a “thumbs up” button, filling out a survey, writing a comment.

*Implicit* feedback refers to feedback that is derived from user behavior or performance-based measures. Examples for implicit signals based on user behavior include question reformulations or task abandoning.

Explicit feedback, such as “great job!”, is a high-confidence, clearly interpretable signal, while implicit feedback based on behavioral cues has a risk of being misinterpreted. For example, users may abandon a conversation for different kinds of reasons. However, obtaining explicit

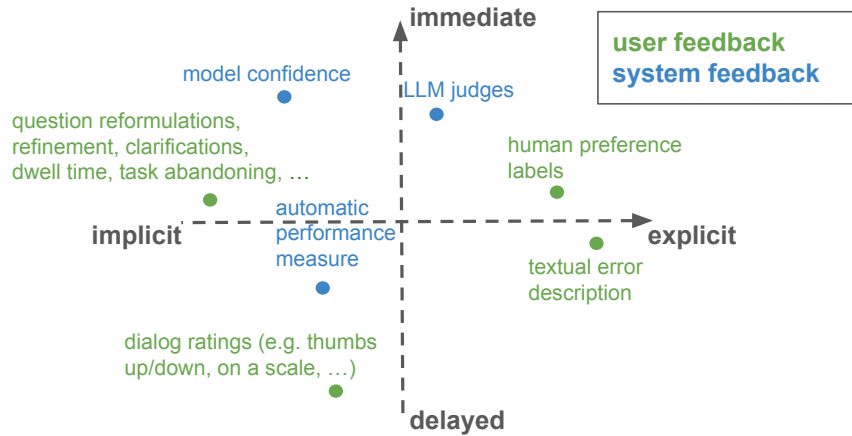


Figure 2.3: Examples of feedback signals categorized according to origin, expression and timing.

feedback is costly and therefore less available. Implicit signals based on user behavior are easier to obtain.

- **Timing:** Feedback can arrive *immediate*, enabling an easy connection between a system action and the outcome, or it can be *delayed*, such as user ratings at the end of a dialog. Some signals could appear both immediately and delayed. For example, feedback from LLM judges may be obtained as immediate signal, rating each system response in isolation, or can be used to score an overall conversation.
- **Granularity:** There are different levels of granularity:
  - binary (positive/negative, single-output, coarse): For example, “thumbs up/down” user feedback, or system feedback judging whether an answer is correct or wrong;
  - scalar/graded (single-output, more detailed): For example, user ratings on a scale (e.g., from 1 – 5), system-based performance metrics, like recall or task completion rates, or the model’s own confidence estimation, measured in terms of perplexity;
  - comparative/ranking-based (multi-output, relative judgment): For example, human pairwise preference (preferring one model output over another) or an LLM that ranks multiple model outputs;
  - qualitative (single- or multi-output, descriptive): For example, user comments describing errors, or an LLM that provides textual description judging the model output.

More fine-grained feedback, such as qualitative feedback, enable better interpretation and may facilitate preventing the respective errors. However, providing binary or comparative feedback is easier for humans.

Figure 2.3 shows example feedback signals, categorized according to the three main dimensions: origin, expression and timing. The granularity dimension is omitted for readability.

### 2.2.2 QA Methods with Feedback

QA methods have incorporated feedback to improve model performance and to better adapt to users. More recently, there is a lot of work in using feedback to align large language models to human values so that they do not create toxic or biased outputs [Kirk et al.(2023)]. This is beyond the use of feedback studied in this thesis, which focuses on improving model performance with feedback.

Earlier works have leveraged explicit user feedback for QA [Abujabal et al.(2018), Kratzwald and Feuerriegel(2019), Zhang et al.(2019), Hancock et al.(2019)]. They use positive feedback to augment their training data. [Campos et al.(2020)] use importance sampling to incorporate binary user feedback, [Gao et al.(2023a)] opted for a three-option user feedback signal (“correct”/“partially correct”/ “wrong”) and train an RL algorithm with it, and [Li et al.(2022c)] collect structured ratings and unstructured natural language explanations as feedback to re-rank answer candidates. Often user feedback is noisy. For example [Wang et al.(2023)] aim to overcome noise in feedback by filtering unreliable users. [Gao et al.(2022)] use supervised data annotations to simulate user feedback. More recently, [Shi et al.(2024)] create LLM preference datasets from real user logs and [Don-Yehiya et al.(2024)] propose to extract feedback that users naturally include when interacting with chat models. They take into account negative feedback reflected in users asking clarification questions, rephrasing their request, or making corrections.

Another line of work considers system-based feedback for improving QA models. Several works consider answering and retrieval performance as feedback signals [Nogueira and Cho(2017), Buck et al.(2018), Zhang et al.(2024), Yang et al.(2023)] [Zhang et al.(2024)]. Most of these use RL for learning from such signals (see Section 2.4.4). In the context of LLMs, [Cui et al.(2024)] create a large-scale AI feedback dataset, where feedback comes from GPT-4, and [Yuan et al.(2024)] propose to learn from the model’s own feedback.

The feedback in this thesis consists of implicit user feedback, as well as performance-based system feedback, often occurring delayed. We consider binary, as well as real-valued and preference-based feedback. More details will be provided in the following chapters.

## 2.3 Reformulations in Question Answering

Reformulations have a long history of being used in Web search to improve the relevance of search results, mainly by augmenting the original query with further relevant terms [Chang et al.(2006), Joachims et al.(2007), Liu and Belkin(2008), Dang and Croft(2010), Xue et al.(2012), Awadallah et al.(2013)]. Bridging the lexical gap between question formulations and the language used in the underlying data source from where answers are retrieved, was a challenging tasks for early QA systems. Therefore, question paraphrases have been created to improve the QA model’s performance by exposing it to different surface forms [Fader et al.(2013), Berant and Liang(2014), Dong et al.(2017)]. Questions have also been reformulated into surface forms easier to understand for the underlying QA [Buck et al.(2018)] or retrieval system [Nogueira and Cho(2017), Das et al.(2019), Yang et al.(2023)].

More recently, questions in their conversational context have been reformulated to benefit the underlying search or ConvQA system [Ren et al.(2018), Mao et al.(2022), Mo et al.(2023), Wu et al.(2022)]. Sometimes the task of re-writing conversational questions into self-sufficient forms is also considered as creating reformulations [Anantha et al.(2021), Yu et al.(2020), Ishii et al.(2022a)].



Figure 2.4: The agent-environment interaction in a Markov Decision Process. The agent is in a state  $s_t$  and decides to take action  $a_t$ . The environment determines its next state  $s_{t+1}$  and gives feedback in form of reward  $r_{t+1}$ , based on the taken action.

Some works also alter the question intent in order to refine previous ill-formed [Liu et al.(2019b), Ponnusamy et al.(2020)] or ambiguous questions [Amplayo et al.(2022)], or in order to transform previously unanswerable questions into related ones that are answerable by the underlying data source [Zhao et al.(2024), Faustini et al.(2023)].

In this thesis, we make use of question reformulations for conversational, incomplete questions, issued by users as well as system-generated ones, in various ways. Our reformulations have different surface forms but express the same question intent.

## 2.4 Reinforcement Learning

Reinforcement Learning (RL) is an important paradigm in Machine Learning. In contrast to supervised learning, there is no fixed labeled dataset which provides for each input the desired output (label). Instead, RL is about learning from interactions: an *agent* explores its *environment* over time in order to achieve a long-term goal. The agent learns from the consequences of its *actions* by receiving *rewards* based on how well it performed, without having access to the correct action at each step. The two most important distinguishing features are decision-making through *trial-and-error* search and learning from *delayed rewards* [Sutton and Barto(2018)]. RL is particularly useful if collecting labeled training data is not feasible or optimal solutions are not known. Another advantage is that rewards can be arbitrary numerical values and the reward function does not need to be differentiable. Also, since models trained in a supervised manner aim to mimic the expert behavior found in the labeled data, they cannot become better than this expert. RL models do not have such upper bounds on performance. Figure 2.4 shows the basic agent-environment interaction.

### 2.4.1 Relevant Concepts

A *Markov Decision Process* (MDP) is a mathematical framework for modeling decision-making under uncertainty and is a core concept in RL used to define interactions with the environment. An MDP is defined as a tuple  $(\mathcal{S}, \mathcal{A}, \delta, \mathcal{R})$ , consisting of the following components:

- **States  $\mathcal{S}$ :** At each timestep  $t$ , the agent receives a representation of the environment’s state  $s_t \in \mathcal{S}$ . States in an MDP have the *Markov property*, which means that a state includes all relevant information from the history so that the agent’s action selection only depends on the

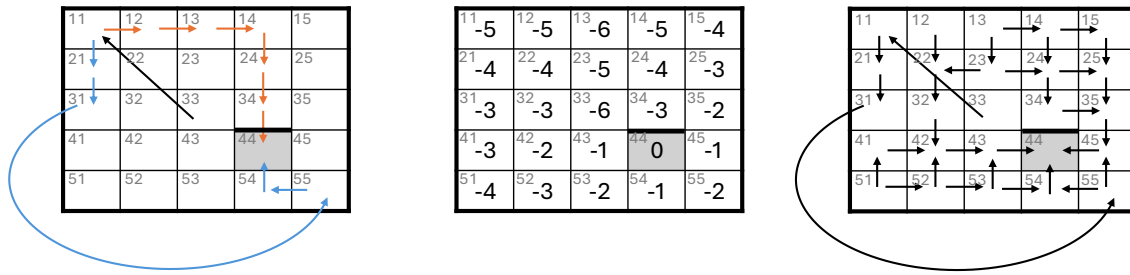


Figure 2.5: Gridworld example, with goal state  $G$ . Two example trajectories are displayed on the left, the optimal value function  $v^*$  is shown in the middle, and the optimal policy  $\pi^*$  is on the right.

current state. A state can also be *partially observable* if not all information in the state is exposed to the agent. In this case, the MDP is called partially-observable (POMDP).

- **Actions  $\mathcal{A}(s_t)$ :** The set of candidate actions at time step  $t$  is based on the current state  $s_t$ . Concretely, the agent has to decide which of the available actions  $a_t \in \mathcal{A}(s_t)$  to take.
- **Transitions:** The transition function  $\delta : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$  defines the probability of reaching state  $s_{t+1}$  from  $s_t$  when taking action  $a_t$ .  $\delta$  fully characterizes the environment's dynamics: the transition probability depends only on the immediately preceding state and action, since the states fulfill the Markov property. In the case of deterministic environments, the transition function can also be directly defined as a mapping between states,  $\delta : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ ;
- **Rewards:**  $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$  is the reward function. The agent receives a numeric value  $r_{t+1}$  for taking action  $a_t$  in state  $s_t$ .

Figure 2.5 provides a simple example where an agent traverses a gridworld in order to reach a goal (state  $s_G = s_{44}$  in the example). In this environment, states are defined as the cells in the gridworld. Available actions are  $\mathcal{A} = \{up, down, left, right, special_{31}, special_{33}\}$ , where the special actions are available only for the states  $s_{31}$  and  $s_{33}$ , respectively. Since in this example the agent should learn to reach the goal in a minimum number of steps, rewards are negative numbers to reflect the cost of moving until the goal is reached. A reward of  $-1$  is assigned for each step, except for moving from  $s_{34}$  to  $s_{44}$ , which is separated by a wall. It is possible for the agent to climb the wall, but with significant effort. Therefore, a reward of  $-10$  is provided:  $\mathcal{R}(s_{34}, down, s_{44}) = -10$ . In this environment, all state transitions are deterministic. For example, action *down* will move the agent one state down with probability 1.0 (e.g.,  $\delta(s_{14}, down) = s_{24}$ ). Actions that would cause the agent to leave the grid, keep the state unchanged (e.g.  $\delta(s_{14}, up) = s_{14}$ ). The transition functions for the special actions are defined as follows:  $\delta(s_{31}, special_{31}) = s_{55}$  and  $\delta(s_{33}, special_{33}) = s_{11}$ , all other actions in these two states leave the state unchanged.

**Episode.** The interaction with the environment results in a trajectory of visited states, actions and rewards,  $s_0, a_0, r_1, s_1, a_1, r_2, s_2, \dots$ . For a finite trajectory, which ends in the terminal state  $s_T$ , this is called an *episode*. In the leftmost gridworld representation in Figure 2.5, two sample episodes (highlighted in blue and orange) are shown, both starting at state  $s_{11}$  and ending in the goal state  $s_{44}$ .

**Cumulative reward.** This is the total reward the agent receives during an interaction (starting from timestep  $t$ ), also called *gain*:  $G_t = r_{t+1} + \gamma r_{t+2} + \dots + \gamma^k r_{t+k+1}$ , where  $\gamma$  is the so-called *discount factor* to balance between intermediate rewards and future rewards (a reward  $k$  steps away is only worth  $\gamma^{k-1}$  what it would be worth if received immediately). When setting  $\gamma = 1.0$  (undiscounted case), the episode in orange in Figure 2.5 has a cumulative expected reward of  $G_0 = -1 + (-1) + (-1) + (-1) + (-1) + (-10) = -15$ , while the episode in blue has  $G_0 = -1 + (-1) + (-1) + (-1) + (-1) = -5$ .

**Policy.** The goal of the agent is to find an *optimal policy*  $\pi^*$ , which is a policy that maximizes the cumulative expected reward. A policy is a function that maps states to actions. A *deterministic* policy maps each state to exactly one action  $\pi(s) = a$ . Usually, the policy is *stochastic* and maps each state to a probability distribution over actions. It defines the probability of taking a specific action  $a$  in state  $s$ , represented as  $\pi(a|s)$ . In our example from Figure 2.5, the final optimal policy guides the agent to reach the goal state in a minimal number of steps, avoiding the high-cost transition from  $s_{34}$  to  $s_{44}$ . The rightmost gridworld representation in Figure 2.5 shows the optimal policy  $\pi^*$  for this example. The arrows indicate the respective optimal action(s) in each state. For example, in state  $s_{43}$  action *right* is the optimal one, directly leading to the goal, while in state  $s_{23}$ , taking actions *left* (followed by the sequence of actions *down, down, right, right*) or *right* (followed by *down, right, down, left* or *right, down, down, left*) would both be optimal.

**Value functions.** The value function  $v(s)$  estimates how good it is for the agent to be in a specific state (= how high the expected future reward from that state is):  $v_\pi(s_t) = \mathbb{E}_\pi[G_t|s_t]$ . This can also be written with the *Bellman equation*:

$$v_\pi(s) = \sum_a \pi(a|s) \cdot \sum_{s',r} \delta(s', r|s, a) \cdot [r + \gamma v_\pi(s)], \forall s \in \mathcal{S} \quad (2.1)$$

This equation expresses the relationship between the value of the current state  $s$  and its successor states  $s'$ . Another variant is the *action-value function*  $Q(s, a)$ , which estimates how good it is to take a specific action  $a$  in state  $s$ :  $Q_\pi(s_t, a_t) = \mathbb{E}_\pi[G_t|s_t, a_t]$ . The *advantage function* is defined as  $\hat{A}(s_t, a_t) = Q(s_t, a_t) - v(s_t)$  and is used to estimate how well the actual taken action is, compared to the other available actions in that state. Learning value functions can be used in order to find better policy, more precisely, finding an optimal value function leads to an optimal policy:  $\pi^*(s_t) = \arg \max_a Q^*(s_t, a)$ . The gridworld representation in the middle of Figure 2.5 displays the optimal value function  $v^*$  for this task. For example,  $v(s_{43}) = -1$  and  $v(s_{23}) = -5$ . The respective action-value functions for state  $s_{43}$  are:  $Q(s_{43}, \textit{right}) = -1$ ,  $Q(s_{43}, \textit{left}) = -2$ ,  $Q(s_{43}, \textit{up}) = -6$ ,  $Q(s_{43}, \textit{down}) = -2$ . The advantage function for  $\hat{A}(s_{43}, \textit{right}) = 0$  and  $\hat{A}(s_{43}, \textit{up}) = -5$ , for example, quantifying that it is much worse to go *up* instead of *right* in this state.

**Learning.** Learning in the context of RL means to iteratively improve a policy and (depending on the used algorithm) the value function based on a (potentially) large number of trajectories. Without any prior knowledge, an agent would typically start with a random policy which is improved through the interactions. During the learning procedure, the agent may choose the current best action in a state, determined by the value function (= *exploitation* of its current knowledge), or may try out a different (potentially better) one (= *exploration*). Exploration is necessary to find better actions,

but too much of it may waste resources and increase learning times, while too much exploitation can prevent the model from learning better strategies. This called the *exploration-exploitation trade-off*.

**Bootstrapping.** Updating value and action-value functions usually involves bootstrapping: instead of waiting for an episode to end, estimates of states and action values are updated based on previously estimated values. For example, with the Bellman equation, we update  $v(s)$  based on the estimates of its immediate successor states  $v(s')$ . This method helps the agent to learn faster and more efficiently, because the values are updated more frequently.

**Model-based vs. model-free algorithms.** Model-based RL algorithms, like Dynamic Programming methods, operate based on a model of the environment and can be used when the dynamics of the environment are known. In the gridworld example from Figure 2.5, the model dynamics defined by  $\delta$ , are given. Therefore, we can apply a model-based DP algorithm, for example *policy iteration*. This algorithm iteratively applies the Bellman equation to calculate the value functions for each state and then improves the current policy by taking the best actions based on the current value function. This way, the optimal policy is learned after only four iterations for the gridworld example in Figure 2.5. However, the model dynamics are often not available in practice. Therefore, model-free RL algorithms, such as *Monte Carlo* methods are used. These do not require such model and learn from *sampling* many episodes instead (trial-and-error learning). Further prominent model-free methods are *Temporal Difference (TD)* methods, that also learn from sampled experience but use bootstrapping to speed up learning. For our gridworld example, more than 100 iterations are needed using a model-free TD algorithm.

**On-policy vs. off-policy methods.** When applying *on-policy* methods, the policy that is used to collect experience, consisting of  $\langle \text{state, action, reward, next state} \rangle$  trajectories, is the same as the one which is iteratively evaluated and improved. When using off-policy methods, there exists two policies: one *behavior policy* that is used to collect experience, and a *target policy* which is the one being evaluated and improved. An advantage of this separation is that the target policy may be deterministic (e.g. greedy), while the behavior policy can be exploratory and sample all possible actions. Also off-policy methods allow to learn from fixed datasets of previously collected experience.

## 2.4.2 Challenges

Reinforcement learning comes with several challenges:

- **Defining suitable rewards.** It is crucial that the designed reward function encourages the agent to actually reach the desired goal. In the gridworld in Figure 2.5, the agent obtained intermediate rewards of  $-1$ , to encourage it to find the shortest path to the goal. However, there are often no intermediate rewards but only a *sparse, delayed* reward obtained when reaching the goal. This makes it challenging for the agent to link actions to outcomes and can slow down learning. To overcome sparseness, it can be beneficial to introduce additional rewards when reaching certain *subgoals*. For example, imagine a slightly modified variant of the gridworld in Figure 2.5, where the final state can only be reached by opening a door which requires the agent to pick up a key first (= subgoal), that is located in one of the other states.

An additional reward could be introduced for picking up the key. If such additional rewards are not designed carefully, the agent may develop some unexpected and undesired behavior that does not lead to the goal but gives high rewards unintentionally (= *reward hacking*).

- **Learning from interaction.** RL methods are not very sample-efficient. When state or action spaces are large, vast amounts of episodes are necessary for learning good policies. Running many interactions on an environment can be time-consuming, costly, and sometimes infeasible. Another issue involves handling the exploration-exploitation trade-off.
- **Stable learning.** RL methods, in particular when using neural networks, can suffer from instability and non-convergence. Some methods have high variance, and performance can greatly depend on initializations. It is also generally harder to tune hyperparameters effectively, compared to supervised learning.

### 2.4.3 RL Methods

In the following, selected RL algorithms that are relevant for this thesis will be presented.

**Value-based methods.** These kind of methods learn value (or action-value) functions to estimate the quality of states and actions. One popular algorithm is *Q-learning*, which is a model-free, temporal-difference (TD) learning algorithm. The estimate of the Q-value at the current timestep  $t$  is updated as follows:

$$\underbrace{Q(s_t, a_t)}_{\text{new Q-value}} \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old Q-value}} + \alpha \underbrace{[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]}_{\text{TD-target}} \quad (2.2)$$

Bootstrapping is used to update the value of the state-action pair  $Q(s_t, a_t)$  with the estimate for its successor state:  $\max_a Q(s_{t+1}, a)$  is the estimated future reward for the next state, which is the maximum Q-value over all possible actions in that state. Q-Learning is off-policy, since a sampling-based policy is used for selecting the next action, which enable exploration, while a greedy policy is used for obtaining the estimate for the maximum Q-value. The immediate reward  $r_{t+1}$  summed with this estimate is called the *TD-target*.

An extension to Q-learning, *Deep Q-Networks (DQN)* [Mnih et al.(2015)], uses deep neural networks to approximate the Q-function to be able to handle environments with large, high-dimensional state and action spaces. Another improvement in DQN is the use of an *experience replay buffer*. It stores past experience consisting of  $\langle \text{state, action, reward, next state} \rangle$  tuples. Minibatches are sampled from this buffer to perform non-correlated updates for more stable learning. Training stability is further increased by introducing a *target network*, which is a copy of the Q-network that is less frequently updated. The objective function is defined as the difference between TD-target and estimated old Q-value (where the difference is also called *TD error*), with network parameters  $\theta$ , and target network parameters  $\theta_{prev}$ :

$$L_t^{DQN}(\theta) = \mathbb{E}_t[(r_{t+1} + \gamma \max_a Q(s_{t+1}, a; \theta_{prev}) - Q(s_t, a_t; \theta))^2] \quad (2.3)$$

Some methods, such as the value-based network in PPO [Schulman et al.(2017)] (see more details below), optimize for the value function  $V(s_t)$ , instead of the Q-values. The general objective function

for value-based methods with target  $\hat{V}_t$  and network parameters  $\theta$ , is defined as:

$$L_t^{VF}(\theta) = \mathbb{E}_t[(\hat{V}_t - V_\theta(s_t))^2] \quad (2.4)$$

**Policy-gradient methods.** These methods directly learn a parametrized policy (typically a neural network), without the need of calculating value functions. This is done by updating the parameters  $\theta$  in the direction that maximizes the expected cumulative reward  $J(\theta) = \mathbb{E}_{s_t \in \mathcal{S}} \mathbb{E}_{a_t \sim \pi_\theta(s_t)}[\mathcal{R}(s_t, a_t, s_{t+1})]$ . One popular policy-gradient method is the *REINFORCE* algorithm [Williams(1992)], which is a model-free, Monte Carlo, on-policy algorithm. The update of its set of parameters  $\theta$  at timestep  $t$  with step size  $\alpha$  is defined as  $\theta_{t+1} \leftarrow \theta_t + \alpha \hat{g}$  with:

$$\nabla J(\theta) = \hat{g} = \mathbb{E}_\pi \left[ G_t \cdot \frac{\nabla \pi(a_t | s_t, \theta)}{\pi(a_t | s_t, \theta)} \right] \quad (2.5)$$

The gradient of the probability of taking action  $a_t$  is scaled by the cumulative reward  $G_t$ . Parameters are updated into the direction of actions with highest rewards, divided by the probability of taking the particular action. This update can suffer from high variance, which can be reduced by introducing a baseline that is subtracted from the reward. Typical baselines are a moving average over past rewards or, more sophisticated, the additionally learned value-function. The objective function is as follows:

$$L_t^{PG}(\theta) = \mathbb{E}_t[(G_t - b_t) \cdot \log \pi_\theta(a_t | s_t)] \quad (2.6)$$

The agent simply samples from the probability distribution  $\pi$  to ensure exploration, whereas value-based methods need to implement sufficient exploration manually. Another advantage of policy-gradient methods is that they can handle large as well as continuous action spaces, which is not possible in Q-learning since the maximum of all action values need to be determined in the update. They also have better convergence properties. However, they often converge to a local maximum instead to a global one, and generally proceed slower, making training more inefficient compared to value-based methods.

**Actor-critic methods.** Actor-critic methods combine policy-gradient and value-based methods. The *actor* controls how the agent behaves (policy-based method), while the *critic* measures how good the taken action is (value-based method). One popular method is the *Proximal Policy Optimization (PPO)* [Schulman et al.(2017)] algorithm. The overall objective consists of multiple parts. The objective function for the policy-gradient part is as follows:

$$L_t^{PG}(\theta) = \mathbb{E}_t \left[ r_t(\theta) \cdot \hat{A}_t(s_t, a_t) - \beta KL[\pi_{ref}(\cdot | s_t), \pi_\theta(\cdot | s_t)] \right] \quad (2.7)$$

with advantage function  $\hat{A}_t(s_t, a_t)$  and the probability ratio  $r_t(\theta) = \frac{\pi_\theta(a_t | s_t)}{\pi_{ref}(a_t | s_t)}$ , describing the probability of taking action  $a_t$  under the current and under some reference policy. KL is the *Kullback-Leibler divergence* between the current policy  $\pi_\theta$  and the reference policy  $\pi_{ref}$ , with an adaptive coefficient  $\beta$ . The reference policy is typically initialized from some initial supervised model. PPO improves training stability (over policy-gradient methods such as REINFORCE): it prevents too large policy updates by constraining the policy to not divert too far from the reference policy.

The overall PPO objective function is as follows:

$$L_t^{PPO}(\theta) = \mathbb{E}_t[L_t^{PG}(\theta) - c_1 L_t^{VF}(\theta) + c_2 H_\pi(s_t)] \quad (2.8)$$

It is a combination of the policy-gradient objective function, the value function  $L_t^{VF}$  (as defined in equation 2.4) and an entropy bonus  $H_\pi$  (with hyperparameters  $c_1, c_2$ ).

**Preference optimization methods.** In the age of large language models, using RL has become popular for aligning language models to human preferences (RLHF) [Ziegler et al.(2019), Stiennon et al.(2020), Ouyang et al.(2022)]. In this case, policy  $\pi$  corresponds to the LLM’s probability distribution over vocabulary tokens: each state contains the tokens generated so far and the next action corresponds to the next predicted token. The reward signal in RLHF comes from preference-based feedback, as opposed to traditional reward signals from environment interactions. Instead of asking to evaluate the goodness of a generation, it is often easier for humans to judge which generation (out of a small set) is better. Therefore, such comparison data  $D$  is collected (usually multiple model outputs  $y$  are sampled for a given input prefix  $x$  and compared) where preferences are typically modeled by the Bradley-Terry (BT) model [Bradley and Terry(1952)]. Under the BT model the preference distribution  $p^*$  can be written as:

$$p^*(y_1 \succ y_2|x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))} \quad (2.9)$$

The goal is to learn to approximate the scoring function  $r^*$  that ranks the preference pairs in the perceived order ( $y_1 \succ y_2$ ). In order to achieve this, a parametrized reward model  $r_{\phi(x,y)}$  is learned by framing the problem as binary classification and optimizing for the following objective, with logistic function  $\sigma$  and with  $y_w$  being preferred over  $y_l$ :

$$L_R(r_\phi, D) = -\mathbb{E}_{(x, y_w, y_l) \sim D} [\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))] \quad (2.10)$$

After that, an RL method, typically the PPO algorithm, is used to update the policy with feedback coming from the reward model.

One method that overcomes the need to train a separate reward model, is *Direct Preference Optimization (DPO)* [Rafailov et al.(2023)]. Instead the reward model is fit implicitly, by modeling the probability of human preference data in terms of the optimal policy rather than the reward model. The DPO objective function is defined as follows (with hyperparameter  $\beta$ ):

$$L_{DPO}(\pi_\theta; \pi_{ref}) = \mathbb{E}_{(x, y_w, y_l) \sim D} \left[ \log \sigma\left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)}\right) \right] \quad (2.11)$$

where  $\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}$  is the reward defined by the LLM implicitly. The function increases the likelihood of preferred generation  $y_w$  and decreases the likelihood of the dispreferred option  $y_l$ . DPO is more stable and computationally lightweight, while at the same time achieving similar performance compared to PPO-based RLHF.

#### 2.4.4 RL in the Context of Question Answering and Dialogs

**RL for multi-hop QA over KGs.** RL has been used to find missing paths in knowledge graphs for KG completion and for answering (multi-hop) questions [Das et al.(2018), Xiong et al.(2017), Lin et al.(2018), Shen et al.(2018), Qiu et al.(2020)]. For these problems, the KG is considered as RL environment, with states consisting of KG nodes (entities) and actions consisting of the paths (relations) connecting these nodes. Rewards are obtained when a desired entity is reached.

Challenges are the sparse rewards, that are only obtained after a potential long KG traversal, and the many spurious paths that may lead to the desired entity by chance via some irrelevant relations.

Due to the large action space, most works apply policy-gradient methods, more precisely the REINFORCE algorithm, for such tasks [Das et al.(2018), Xiong et al.(2017), Lin et al.(2018), Qiu et al.(2020)]. Since REINFORCE is not very sample-efficient, multiple rollouts are performed for each training example in [Das et al.(2018)]. They also add an entropy regularization term for exploring more diverse paths in order to find promising paths faster. To overcome the issue with sparse rewards and spurious path problems, [Lin et al.(2018)] and [Qiu et al.(2020)] introduce reward-shaping to score paths using pre-trained KG embeddings. [Shen et al.(2018)] uses a different strategy for handling reward sparseness: they apply Q-learning in combination with Monte-Carlo Tree Search (MCTS) to make use of the deterministic MDP transitions in the KG environment for selecting the most promising paths.

**RL for question reformulation.** Starting with [Buck et al.(2018)], RL has been applied as means for reformulating a question in a way that works best for the underlying QA or retrieval system [Nogueira and Cho(2017), Li et al.(2022b), Das et al.(2019), Yang et al.(2023)]. RL has also been used to rewrite conversational questions into self-sufficient forms [Chen et al.(2022b), Ishii et al.(2022a)], or for selecting relevant history turns for question expansion [Qiu et al.(2021)]. Also in the context of ConvQA, questions were reformulated for improved retrieval performance [Wu et al.(2022), Jang et al.(2024), Yoon et al.(2025), Zhang et al.(2024)].

Most works use generative models (the more recent ones use LLMs) for reformulating questions. In this case, the policy  $\pi$  corresponds to the probability distribution for the next token prediction. Thus, states correspond to the tokens generated so far (in addition to the conversation history, in the context of ConvQA) and actions are all vocabulary tokens. The environment contains the (potentially blackbox) retrieval/reader modules that provide rewards in form of retrieval or answering performance. In this setting, state and action spaces are very large and reward signals are once again sparse, obtained only for fully generated questions. Works often apply REINFORCE with baseline [Buck et al.(2018), Nogueira and Cho(2017), Li et al.(2022b), Das et al.(2019)] or PPO with KL divergence [Ishii et al.(2022a), Yang et al.(2023), Liu et al.(2019b)]. Earlier works often struggle with degenerated model outputs produced after RL training. Therefore, techniques to increase training stability and preventing such degeneration have been applied. For example, RL is mixed with cross-entropy loss [Ishii et al.(2022a), Wu et al.(2022)] or *Self-Critical Sequence Training (SCST)* [Rennie et al.(2017)] is applied in [Chen et al.(2022b), Wu et al.(2022)], a technique where the reward obtained by greedy generation is compared to the one received by sampling from the word distribution at each generation step. Most recent works which employ LLMs use preference optimization algorithms, such as DPO [Yoon et al.(2025), Zhang et al.(2024)].

**RL for task-oriented dialogs.** Defining dialogs as sequential decision-making and formulating them with MDPs has already been proposed in early works [Levin et al.(1998), Williams and Young(2007), Young et al.(2010)]. In pipeline-based ToD systems, RL has been used to predict the next dialog action the system should take. The RL states comprise the dialog history and the current user utterance, and actions are typically a small pre-defined set, consisting of requests for more information or performing database lookups, etc. The reward is only obtained at the end of dialog based on whether the user’s goal has been fulfilled in the dialog.

For successfully training dialog policies, many interactions with the environment are necessary. One line of work uses on-policy methods that are trained through interacting with simulated users that aim to mimic the behavior of real users in a consistent manner [Liu and Lane(2017), Dhingra et al.(2017)]. Used algorithms comprise REINFORCE, PPO and further actor-critic methods. Another line of work uses off-policy learning, to learn from fixed datasets [Wang et al.(2020), Zhou et al.(2017)]. Many RL works for ToDs explored ways to deal with the sparse reward signal. For example, the ToD model in [Hu et al.(2023)] uses a reward function based on the number of sub-goals that have been accomplished at a given step. Also in complex, realistic settings, user goals are hard to define clearly and it is difficult to evaluate whether a dialog fulfills the goals. Therefore, methods that make use of Inverse Reinforcement Learning (IRL) were proposed, which try to learn the underlying reward function from some human dialog or expert demonstration data [Takanobu et al.(2019), Wang et al.(2020)].

More recently, end-to-end ToD models based on LLMs were introduced. Some works make use of preference data in this setting [Ramachandran et al.(2022), Feng et al.(2023)].

In this chapter, we introduced the relevant background information for this thesis. We introduced relevant ConvQA and ToD concepts and presented related works for ConvQA over KG and text, as well as for ToD systems. We categorized different types of feedback and touched upon how reformulations were leveraged for QA. Lastly, we introduced basic RL concepts and methods relevant for this thesis, as well as work on RL in related areas. In the next chapters, we will present our main contributions.



# Chapter 3

## Learning from Implicit User Feedback via Reformulations

### Contents

---

<b>3.1</b>	<b>Introduction</b>	<b>32</b>
<b>3.2</b>	<b>Model and Architecture</b>	<b>34</b>
3.2.1	CONQUER KG Representation	34
3.2.2	Concepts and Notation	35
3.2.3	System Overview	35
<b>3.3</b>	<b>Context Entity Detection</b>	<b>35</b>
<b>3.4</b>	<b>CONQUER Learning Framework</b>	<b>38</b>
3.4.1	RL Model	38
3.4.2	RL Training	38
3.4.3	Reformulation Prediction	41
<b>3.5</b>	<b>Answer Generation</b>	<b>42</b>
<b>3.6</b>	<b>Benchmark with Reformulations</b>	<b>42</b>
<b>3.7</b>	<b>Experimental Framework</b>	<b>44</b>
3.7.1	Setup	44
3.7.2	CONQUER Configurations	45
3.7.3	Evaluation	45
<b>3.8</b>	<b>Results and Insights</b>	<b>46</b>
3.8.1	Key Findings	46
3.8.2	In-depth Analysis	48
<b>3.9</b>	<b>Related Work</b>	<b>50</b>
<b>3.10</b>	<b>Conclusion</b>	<b>51</b>

---

State-of-the-art methods for ConvQA over knowledge graphs typically learn from crisp question-answer pairs found in popular benchmarks. In reality, however, such training data is hard to come by: users would rarely mark answers explicitly as correct or wrong. This chapter presents a method that takes a step towards a more natural learning paradigm – from noisy and implicit feedback via

question reformulations. A reformulation is likely to be triggered by an incorrect system response, whereas a new follow-up question could be a positive signal on the previous turn’s answer. Our reinforcement learning model, termed CONQUER, can learn from a conversational stream of questions, wrong answers and reformulations. CONQUER’s KG representation and an overview of its components are presented in Section 3.2. Sections 3.3 – 3.5 describe the approach used in CONQUER in detail, particularly how RL is applied. To evaluate CONQUER, we create and release CONVREF, a conversational benchmark with reformulations, introduced in Section 3.6. Experiments show that CONQUER successfully learns from noisy rewards, significantly improving over a state-of-the-art baseline (Section 3.7 and 3.8).

### 3.1 Introduction

**Motivation.** Conversational question answering (ConvQA) has become a convenient and natural mechanism of satisfying information needs that are too complex or exploratory to be formulated in a single shot [Choi et al.(2018), Saha et al.(2018), Reddy et al.(2019), Guo et al.(2018), Qu et al.(2020), Kaiser et al.(2020)]. When the information needs are fact-centric (e.g., about cast of movies, clubs of soccer players, etc.), a suitable data source to retrieve answers from are large knowledge graphs (KG) such as Wikidata [Vrandecic and Krötzsch(2014)]. Figure 3.1 shows a small excerpt of the KG using a simplified graph representation, with red nodes for entities and blue nodes for relations. We address ConvQA over KGs, where system responses are entities.

**Example.** An ideal conversation with five *turns* could be as follows:

*Q*<sub>1</sub>: *When was Avengers: Endgame released in Germany?*

*A*<sub>1</sub>: 24 April 2019

*Q*<sub>2</sub>: *What was the next from Marvel?*

*A*<sub>2</sub>: Spider-Man: Far from Home

*Q*<sub>3</sub>: *Released on?*

*A*<sub>3</sub>: 04 July 2019

*Q*<sub>4</sub>: *So who was Spidey?*

*A*<sub>4</sub>: Tom Holland

*Q*<sub>5</sub>: *And his girlfriend was played by?*

*A*<sub>5</sub>: Zendaya Coleman

Utterances can be colloquial (*Q*<sub>4</sub>) and incomplete (*Q*<sub>2</sub>, *Q*<sub>3</sub>), and inferring the proper context is a challenge (*Q*<sub>5</sub>). Users can provide *feedback* in the form of *question reformulations* [Ponmusamy et al.(2020)]: when an answer is incorrect, users may rephrase the question, hoping for better results. While users never know the correct answer upfront, they may often guess non-relevance when the answer does not match the expected type (director instead of movie) or from additional background knowledge. So, in reality, turn 2 in the conversation above could become expanded into:

*Q*<sub>21</sub>: *What was the next from Marvel?* (**New intent**)

*A*<sub>21</sub>: Stan Lee (**Wrong answer**)

*Q*<sub>22</sub>: *What came next in the series?* (**Reformulation**)

*A*<sub>22</sub>: Marvel Cinematic Universe (**Wrong answer**)

*Q*<sub>23</sub>: *The following movie in the Marvel series?* (**Reformulation**)

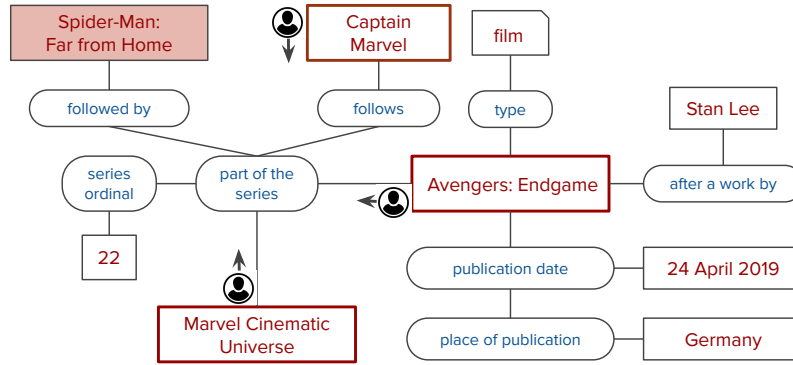


Figure 3.1: KG excerpt from Wikidata required for answering  $Q_1$  and  $Q_2$ . Agents for  $Q_{22}$  (*What came next in the series?*) are shown with possible walk directions. The colored box (**Spider-man: Far from Home**) is the correct answer.

$A_{23}$ : Spider-Man: Far from Home (Correct answer)

$Q_{31}$ : Released on? (New intent)

**Limitations of state-of-the-art.** Work on ConvQA over KGs [Saha et al.(2018), Guo et al.(2018), Christmann et al.(2019), Shen et al.(2019)] do not consider user signals when intermediate utterances lead to unsatisfactory responses, as indicated in the above conversation with reformulations. A few works on QA over KGs have exploited user interactions for online learning [Abujabal et al.(2018), Zhang et al.(2019)], but this is limited to confirming the correctness of answers which can then augment the training data of question-answer pairs. Reformulations as an implicit feedback signal have been leveraged for web search queries [Joachims et al.(2007), Ren et al.(2018)], but that setting is very different from QA over KGs. IR methods rely on observing clicks (and their absence) on ranked lists of documents. This does not carry over to typical QA tasks – especially over voice interfaces with single-entity responses at each turn and no explicitly click-like signal.

**Approach.** We present CONQUER (Conversational Question answering with Reformulations), a new method for learning from implicit user feedback in ConvQA over KGs. CONQUER is based on Reinforcement Learning (RL) and is designed to continuously learn from question reformulations as a cue that the previous system response was unsatisfying.

Given the current (say  $Q_{22}$ ) and the previous utterances ( $Q_1, Q_{21}$ ), CONQUER creates and maintains a set of *context entities* from the KG that are most relevant to the conversation so far. It then positions *RL agents* at each of these context entities, that simultaneously walk over the KG to other entities in their respective neighborhoods. End points of these walks are candidate answers for this turn and are aggregated for producing the final response. Walking directions (see arrows in Figure 3.1 for illustration) are decided by sampling actions from a policy network that takes as input i) encodings of utterances, and ii) KG facts involving the context entities. The policy network is trained via noisy rewards obtained from reformulation likelihoods estimated by a fine-tuned BERT predictor. Experiments on our CONVREF benchmark, that we created from conversations between a system and real users, demonstrate the viability of our proposed learning method CONQUER and its superiority over a state-of-the-art baseline. Our project website, containing the benchmark and a

demo, is publicly accessible at <https://conquer.mpi-inf.mpg.de>. Our code and data is available on GitHub: <https://github.com/magkai/CONQUER>.

**Contributions.** Salient contributions presented in this chapter are:

- A question answering method that can learn from a conversational stream in the *absence of gold answers*;
- A *reinforcement learning* model for QA with rewards based on implicit feedback in the form of question *reformulations*;
- A *reformulation detector* based on BERT that can classify a follow-up utterance as a reformulation or new intent;
- A *benchmark collection augmented with reformulations* for ConvQA over KGs, comprising about 205k reformulations for the 11k conversations.

## 3.2 Model and Architecture

### 3.2.1 CONQUER KG Representation

A KG is typically stored as an RDF database organized into  $\langle S, P, O \rangle$  (subject, predicate, object) triples (facts), where  $S$  is an entity (e.g., `Avengers: Endgame`, `Stan Lee`),  $P$  is a predicate (e.g., `part of series`, `publication date`), and  $O$  is an entity, a type (e.g., `film`, `country`) or a literal (e.g., `26 April 2019`, `22`). See Section 2.1.3.1 in the background chapter for more details. Large KGs like Wikidata also contain  $n$ -ary facts that involve more than two entities. Figure 3.1 shows an example of an  $n$ -ary fact: The path in the graph connecting `Avengers: Endgame` to `part of the series` and over to `Marvel Cinematic Universe` represent a main fact, which is contextualized by the path connecting this main fact to the qualifier predicate node `followed by` and on to the qualifier object node `Spider-man: Far from Home`. The path with `series ordinal` and `22` is another qualifier for the same main fact. In CONQUER, we wish to leverage the entire KG for answering. For that we need to go beyond triples and consider these  $n$ -ary facts.

In the graph representation depicted in Figure 3.1, qualifier predicates are directly connected to their main-fact predicates. However, in this representation, an agent walking from entity to entity (e.g., `Avengers` to `Spider-Man`) misses the context of `Marvel` and would be useless for answering  $Q_2$ . In this case, the main-fact triple (`<Avengers: Endgame, part of series, Marvel Cinematic Universe>`) provides necessary context for making sense of the qualifier (`<followed by, Spider-Man: Far from Home>`). To take  $n$ -ary facts into account during walks by agents, CONQUER creates a modified KG representation where entities are nodes and edges between entities are labeled either by connecting predicates (when a fact has no qualifiers, like `<Avengers: Endgame, after a work by, Stan Lee>`) or by *augmented labels* in cases of facts with qualifiers. The latter scenario is visualized in Figure 3.2. The edge between the main-fact subject `Avengers: Endgame` and the main-fact object `Marvel Cinematic Universe` is augmented by its qualifier information in Figure 3.2 (a). Information from the main fact is also used to augment the connections between the main-fact subject (or object) and qualifier objects, as in Figure 3.2 (b). Connections between qualifier objects are analogously

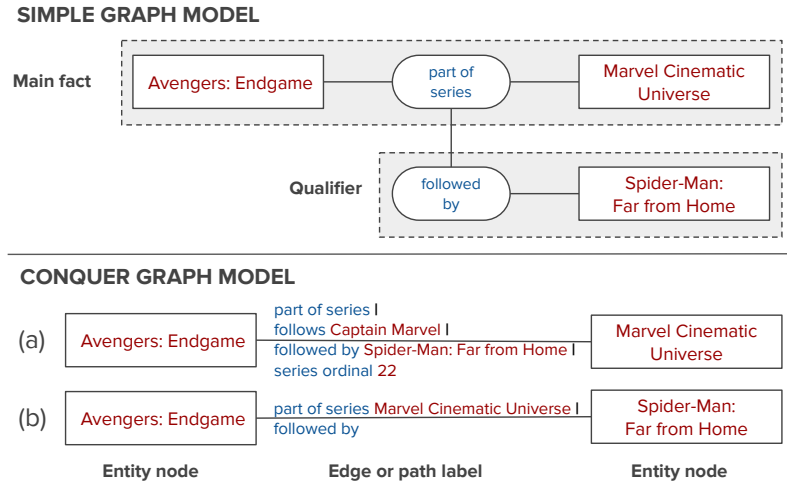


Figure 3.2: CONQUER KG representation for using qualifiers.

augmented by the main fact. These edge labels or *paths* subsequently become *actions* to be chosen by RL agents. The CONQUER graph model is *bidirectional*.

### 3.2.2 Concepts and Notation

A notation overview of the relevant concepts used in this chapter is provided in Table 3.1. The main concepts have already been introduced in the background chapter (see Section 2.1.1 for ConvQA concepts and Section 2.4.1 for relevant RL concepts).

### 3.2.3 System Overview

The workflow of CONQUER is illustrated in Figure 3.3. First, context entities up to the current turn of the conversation are identified. Next, paths from our KG model involving these entities are extracted, and *RL agents* walk along these paths to candidate answers. The paths to walk on (actions by the agent) are decided according to predictions from a *policy network*, which takes as input the conversational context and the KG paths. Aggregating *end points* of walks by the different agents leads to the final answer. Upon observing this answer, the user issues a *follow-up question*. A *reformulation predictor* takes this (original question, follow-up question) sequence as input and outputs a reformulation likelihood. Parameters of the policy network are then updated in an online manner using rewards that are based on this likelihood. Context entities are reset at the end of the conversation, but the policy parameters continue to be updated as more and more conversations take place between the user and the system. Section 3.3 through 3.5 describe CONQUER in detail.

## 3.3 Context Entity Detection

Throughout a conversation  $C$ , we maintain a set of context entities  $E^{ctx}$  that reflect the user’s topical focus and intent. For full-fledged questions this would call for Named Entity Disambiguation (NED), linking entity mentions onto KG nodes [Shen et al.(2015)]. There are many methods and tools for

Notation	Concept
$K$	Knowledge graph
$p \in \mathcal{P}$	Path labels in $K$
$Q, A$	Question and answer
$C$	Conversation
$Q_j$	Question expressing intent $j$
$\langle Q_{jk}   k > 1 \rangle$	Sequence of reformulations for intent $j$
$e^A$	Candidate answer entity
$t$	Turn
$Q_t^{cxt}$	Context questions at turn $t$
$E_t^{cxt}$	Context entities at turn $t$
$h(\cdot)$	Hyperparameters for context entity selection
$\mathbf{Q}_t, \mathbf{Q}_t^{cxt}, \mathbf{E}_t^{cxt}$	Embedding vectors of $Q_t, Q_t^{cxt}, E_t^{cxt}$
$s \in \mathcal{S}$	RL states
$a \in \mathcal{A}_s$	Actions at state $s$
$\mathbf{a}, \mathbf{A}_s$	Embedding vector of $a$ , and matrix of all actions at $s$
$\mathcal{R}(\cdot)$	Reward function
$\boldsymbol{\theta}, \mathbf{W}_1, \mathbf{W}_2 \in \boldsymbol{\theta}$	Parameters of policy network, including weight matrices
$\pi_{\boldsymbol{\theta}}$	Policy parameterized by $\boldsymbol{\theta}$
$J(\boldsymbol{\theta})$	Expected reward with $\boldsymbol{\theta}$
$\alpha$	Step size in REINFORCE update
$H_{\pi}(\cdot, s)$	Entropy regularization term in REINFORCE update
$\beta$	Weight for entropy regularization term

Table 3.1: Notation for salient concepts in CONQUER.

this purpose, including a few that are geared for very short inputs like telegraphic questions [Sawant and Chakrabarti(2013), Piccinno and Ferragina(2014), Li et al.(2020)]. However, none of these can handle contextually incomplete and colloquial utterances that are typical for follow-up questions in a conversation, for example: *What came next from Marvel?* or *Who played his girlfriend?*

For creating and maintaining  $E^{cxt}$ , we apply the following steps. The set  $E^{cxt}$  emanates from the question keywords and is initialized by running an NED tool on the *first question*  $Q_1$ , which is almost always well-formed and complete. Further turns in the conversation incrementally augment  $E^{cxt}$ . Correct answers for questions could qualify for  $E^{cxt}$  and would be strong cues for keeping context. However, they are not considered by CONQUER: the online setting that we tackle does not have any knowledge of ground-truth answers and would thus have to indiscriminately pick up both correct and incorrect answers. Therefore, CONQUER considers only entities derived from user utterances.

Let  $E_{t-1}^{cxt}$  denote the set of context entities up to turn  $t-1$ . Nodes in the neighborhood  $nbd(\cdot)$  of  $E_{t-1}^{cxt}$  form the *candidate context* for turn  $t$  and are subsequently scored for entry into  $E_t^{cxt}$ . In our experiments, we restrict this to 1-hop neighbors, which is usually sufficient to capture all relevant cues. For scoring candidate entities  $n$  for question  $Q_t$  at turn  $t$ , CONQUER computes four measures for each  $n \in nbd(e | e \in E_{t-1}^{cxt})$ :

- **Neighbor overlap:** This is the number of nodes in  $E_{t-1}^{cxt}$  from where  $n$  is reachable in one

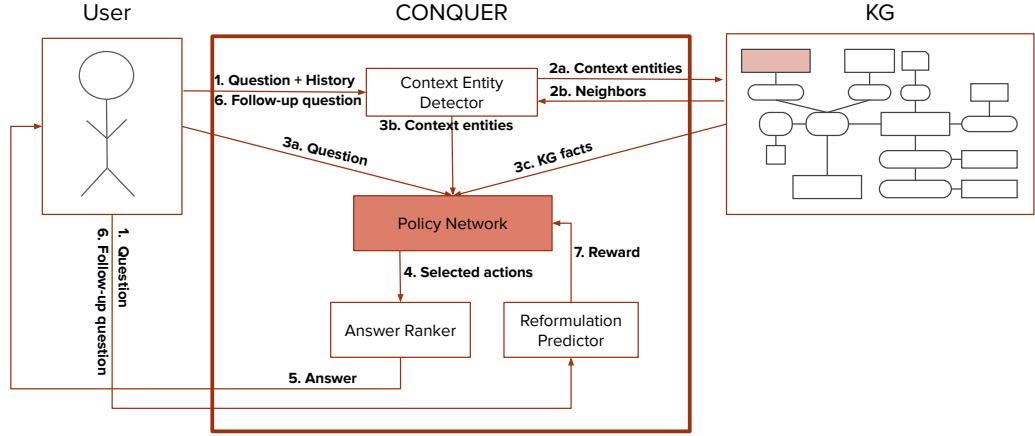


Figure 3.3: Overview of CONQUER, with numbered steps tracing the workflow. Looping through steps 1-7 creates a continuous learning model. Step 6 denotes a follow-up question.

hop, the higher the better. Since this indicates high connectivity to  $E_{t-1}^{cxt}$ , such nodes are potentially good candidates. The number is normalized by the cardinality  $|E_{t-1}^{cxt}|$ , to produce  $overlap(n) \in [0, 1]$ ,

- **Lexical match:** This is the Jaccard overlap between the set of words in the node label of  $n$  and all words in  $Q_t$  (with stopwords excluded):  $match(n) \in [0, 1]$ ,
- **NED score:** Although full-fledged NED would not work well for incomplete and colloquial questions, NED methods can still give useful signals. We run an off-the-shelf tool, which we provide with richer context by concatenating the current with the previous questions as input. We consider its normalized confidence score,  $ned(n) \in [0, 1]$ , but only if the returned entity is in the candidate set  $nbd(e|e \in E_{t-1}^{cxt})$ ; otherwise  $ned(n)$  is zero. This can be thought of as NED restricted to the neighborhood of the current context  $E_t^{cxt}$  as an entity repository,
- **KG prior:** Salient nodes in the KG, as measured by the number of facts they are present in as subject, are indicative of their importance in downstream tasks like QA [Christmann et al.(2019)]. A prior on this KG frequency often helps discriminate obscure nodes from prominent ones. We clip raw frequencies at a factor  $f_{max}$ , and normalize them by  $f_{max}$  to yield  $prior(n) \in [0, 1]$ .

These four scores are linearly combined with hyperparameters  $h_1, \dots, h_4$ , such that  $\sum_{i=1}^4 h_i = 1$ , to compute the *context score*:

$$cxt(n) = h_1 \cdot overlap(n) + h_2 \cdot match(n) + h_3 \cdot ned(n) + h_4 \cdot prior(n) \quad (3.1)$$

If score  $cxt(n)$  is above a specified threshold  $h_{cxt}$ , then  $n$  is inserted into the set of context entities  $E_t^{cxt}$ . Hyperparameters  $h_1, \dots, h_4$  and  $h_{cxt}$  are tuned on a development set. Entities in  $E_t^{cxt}$  are passed on as start points for RL agents to walk from (Section 3.4.2).

## 3.4 CONQUER Learning Framework

### 3.4.1 RL Model

The goal of an RL agent here is to learn to answer conversational questions correctly. The user (issuing the questions) and the KG jointly represent the environment. The agent walks over the knowledge graph  $K$ , where entities are represented as nodes and predicates as path labels (see Section 3.2.1). An agent can only start and end its walk at entity nodes (see Figure 3.2), after traversing a path label. This traversal can be viewed as a Markov Decision Process (MDP), where individual parts  $(\mathcal{S}, \mathcal{A}, \delta, \mathcal{R})$  are defined as follows (adapted from [Das et al.(2018)]):

- **States:** A state  $s \in \mathcal{S}$  is represented by  $s = (Q_t^{cxt}, Q_t, e_t^{cxt})$ , where  $Q_t$  represents the question at turn  $t$  (new intent or reformulation),  $Q_t^{cxt}$  captures a subset of the previous utterances as the (optional) context questions and  $e_t^{cxt} \in E_t^{cxt}$  is one of the context entities for turn  $t$  that serves as the starting point for an agent’s walk.
- **Actions:** The set of actions  $\mathcal{A}_s$  that can be taken in state  $s$  consists of all outgoing paths of the entity node  $e_t^{cxt}$  in  $K$ , so that  $\mathcal{A}_s = \{p | \langle e_t^{cxt}, p, e^A \rangle \in K\}$ . End points of these paths are candidate answers  $e^A$ .
- **Transitions:** The transition function  $\delta$  updates a state to the agent’s destination entity node  $e^A$  along with the follow-up question and (optionally) its context questions;  $\delta : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$  is defined by  $\delta(s, a) = s_{next} = (Q_{t+1}^{cxt}, Q_{t+1}, e^A)$ .
- **Rewards:** The reward depends on the next user utterance  $Q_{t+1}$ . If it expresses a new intent, then reward  $\mathcal{R}(s, a, s_{next}) = 1$ . If  $Q_{t+1}$  has the same intent, then this is a reformulation, making  $\mathcal{R}(s, a, s_{next}) = -1$ .

While we know deterministic transitions inside the KG through nodes and path labels, users’ questions are not known upfront. So we use a *model-free algorithm* that does not require an explicit model of the environment [Sutton and Barto(2018)]. Specifically, we use Monte Carlo methods that rely on explicit trial-and-error *experience*: we learn from sampled state-action-reward sequences from actual or simulated interaction with the environment. Since questions can be arbitrarily formulated on any of several topics, our state space is unbounded in size. Thus, it is not feasible to learn transition probabilities between states. Instead, we use a *parameterized policy* that learns to capture similarities between the question (along with its conversational context) and the KG facts. The parameterized policy is manifested in the weights of a neural network (*policy network*). When a new question arrives, this policy can be applied by an agent to reach an answer entity: this is equivalent to the agent following a path predicted by the policy network.

### 3.4.2 RL Training

Using a policy network has been shown to be more appropriate for KGs than value-based methods, due to the large action space [Xiong et al.(2017)]. To train our network, we apply the policy-gradient algorithm REINFORCE with baseline [Williams(1992)] (see also part on policy-gradient methods in Section 2.4.3 for further details). As baseline, we use the average reward over several training samples for variance reduction. The parameterized policy  $\pi_\theta$  takes information about a state as input and outputs a probability distribution over the available actions in this state. Formally:  $\pi_\theta(s) \mapsto P(\mathcal{A}_s)$ .

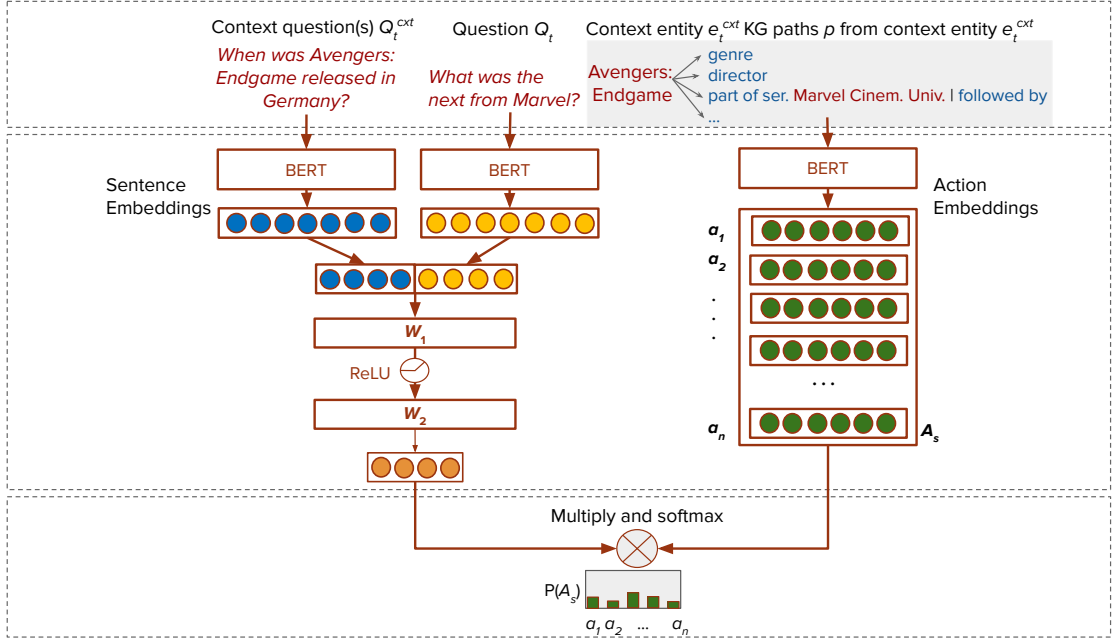


Figure 3.4: Architecture of the policy network in CONQUER.

Figure 3.4 depicts our policy network and Algorithm 1 shows high-level pseudo-code for the policy learning in CONQUER. The network contains a two-layer feed-forward network with non-linear ReLU activation. The policy parameters  $\theta$  consist of the weight matrices  $\mathbf{W}_1$ ,  $\mathbf{W}_2$  of the feed-forward layers. Inputs to the network consist of embeddings of the current question  $Q_t \in \mathbb{R}^d$  (*What was the next from Marvel?*), optionally prepended with some context question embeddings  $Q_t^{ctx} \in \mathbb{R}^d$  (like *When was Avengers: Endgame released in Germany?*). We apply a pre-trained BERT model to obtain these embeddings, by averaging over all hidden layers and over all input tokens (line 6 in Algorithm 1). Context entities  $E_t^{ctx}$  (e.g., *Avengers: Endgame*) are the starting points for an agent’s walk and are identified as explained in Section 3.3. We then retrieve all outgoing paths for these entities from the KG (line 4, Algorithm 1). An action vector  $\mathbf{a} \in \mathcal{A}_s$  consists of the embedding of the respective path  $p$  starting in  $e_t^{ctx}$ ,  $\mathbf{a} \in \mathbb{R}^d$ . These actions are also encoded using BERT. The final embedding matrix  $\mathcal{A}_s \in \mathbb{R}^{|\mathcal{A}_s| \times d}$  consists of the stacked action embeddings (line 7, Algorithm 1). The output of the policy network is the probability distribution  $P(\mathcal{A}_s)$ , that is defined as follows (line 8, Algorithm 1):

$$P(\mathcal{A}_s) = \sigma(\mathcal{A}_s \times (\mathbf{W}_2 \times \text{ReLU}(\mathbf{W}_1 \times [Q_t^{ctx}; Q_t]))) \quad (3.2)$$

where  $\sigma(\cdot)$  is the softmax operator. Then, the final action which the agent will take in this step is sampled from this distribution (line 11, Algorithm 1):

$$a = \mathcal{A}_s^i, i \sim \text{Categorical}(P(\mathcal{A}_s)) \quad (3.3)$$

To update the network’s parameters  $\theta$ , the expected reward  $J(\theta)$  is maximized over each state and the corresponding set of actions:

$$J(\theta) = \mathbb{E}_{s \in \mathcal{S}} \mathbb{E}_{a \sim \pi_\theta(s)} [\mathcal{R}(s, a, s_{next})] \quad (3.4)$$

**Algorithm 1:** Policy learning in CONQUER

---

**Input:** question  $Q_t$ , KG  $K$ , step size ( $\alpha > 0$ ), number of rollouts (*rollouts*), size of update (*batchSize*), entropy weight ( $\beta$ )

**Output:** updated policy parameters  $\theta$

► On initial call: Initialize  $\theta$  randomly

- 1  $Q_t^{cxt} \leftarrow loadContext(Q_t)$
- 2  $E_t^{cxt} = detectContextEntities(Q_t^{cxt}, Q_t)$
- 3 **foreach**  $e_t^{cxt} \in E_t^{cxt}$  **do**
- 4  $\mathcal{P} \leftarrow getKGPaths(e_t^{cxt}, K)$  ► Get KG paths involving context entity
- 5  $s \leftarrow (Q_t^{cxt}, Q_t, e_t^{cxt})$
- 6  $Q_t \leftarrow BERT(Q_t), Q_t^{cxt} \leftarrow BERT(Q_t^{cxt})$  ► Encode questions
- 7  $\mathcal{A}_s \leftarrow stack(BERT(p))$ , where  $p \in \mathcal{P}$
- 8  $P(\mathcal{A}_s) = \sigma(\mathcal{A}_s \times (\mathbf{W}_2 \times ReLU(\mathbf{W}_1 \times [Q_t^{cxt}; Q_t])))$  ► Get action probabilities
- 9  $count \leftarrow 0$
- 10 **while**  $count < rollouts$  **do**
- 11  $i \sim Categorical(P(\mathcal{A}_s))$  ► Sample action
- 12  $a \leftarrow \mathcal{A}_s^i$ , where  $a := p_i$  and  $(e_t^{cxt}, p_i, e^A) \in K$
- 13  $Q_{t+1} \leftarrow getUserFeedback(e^A)$  ► Get feedback in form of next user question
- 14  $Q_{t+1}^{cxt} \leftarrow loadContext(Q_{t+1})$
- 15  $s_{next} \leftarrow (Q_{t+1}^{cxt}, Q_{t+1}, e^A)$
- Determine reward
- 16 **if**  $isReformulation(Q_t, Q_{t+1})$  **then**  $r \leftarrow -1$
- 17 **else**  $r \leftarrow 1$
- 18  $experience.enqueue(s, a, s_{next}, r)$  ► Store experience
- 19  $count \leftarrow count + 1$
- 20 **end**
- 21 **end**
- 22 **if**  $|experience| \geq batchSize$  **then**
- Retrieve *batchSize* entries from experience buffer
- 23  $updateList \leftarrow experience.dequeue(batchSize)$
- 24  $batchUpdate \leftarrow 0, rList \leftarrow getRewards(updateList)$
- 25  $\bar{r} \leftarrow mean(rList), \sigma_r \leftarrow stdDev(rList)$
- Perform batch update
- 26 **foreach**  $(s, a, s_{next}, r) \in updateList$  **do**
- 27  $H_\pi(\cdot, s) \leftarrow -\sum_{a \in \mathcal{A}_s} \pi(a|s) \cdot \log \pi(a|s)$
- 28  $\hat{r} \leftarrow \frac{r - \bar{r}}{\sigma_r}$
- 29  $batchUpdate \leftarrow batchUpdate + \hat{r} \frac{\nabla \pi(a|s, \theta)}{\pi(a|s, \theta)} + \beta H_\pi(\cdot, s)$
- 30 **end**
- 31  $\theta \leftarrow \theta + \alpha \cdot batchUpdate$  ► Do gradient ascent
- 32 **return**  $\theta$

---

For each question in our training set, we do multiple *rollouts*, meaning that the agent samples multiple actions for a given state to estimate the stochastic gradient (the inner expectation in the formula above). Updates to our policy parameters  $\theta$  are performed in batches (lines 26-31, Algorithm 1). Each *experience* of the form  $(s, a, s_{next}, r)$  that the agent has encountered is stored.

A batch of experiences is used for the update, performed as follows:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{\pi} \left[ \hat{r} \cdot \frac{\nabla \pi(a|s, \boldsymbol{\theta})}{\pi(a|s, \boldsymbol{\theta})} + \beta \cdot H_{\pi}(\cdot, s) \right] \quad (3.5)$$

where  $\hat{r}$  is the normalized return and  $\beta$  is a weighting constant for  $H_{\pi}(\cdot, s)$ , that is an entropy regularization term.

We have already introduced the REINFORCE update in the background chapter (section 2.4.3, equation 2.5). For CONQUER, we add a regularization term to the update to ensure better exploration and to prevent the agent from getting stuck in local optima. It is defined as:

$$H_{\pi}(\cdot, s) = - \sum_{a \in \mathcal{A}_s} \pi(a|s) \cdot \log \pi(a|s) \quad (3.6)$$

Parameters  $\boldsymbol{\theta}$  are updated in the direction that increases the probability of taking action  $a$  again when seeing  $s$  next time. The update is inversely proportional to the action probability to not favor frequent actions (see also [Sutton and Barto(2018)], Chapter 13, for more details). REINFORCE is often used with a baseline which is subtracted from the reward to reduce variance. We opted for a simple baseline by normalizing each reward by subtracting the mean and by dividing by the standard deviation of all rewards in the current batch update:  $\hat{r} = \frac{r - \bar{r}}{\sigma_r}$ .

**Remark.** Our policy network learns a similarity-based matching between the user’s information need and KG paths. In our current setting, agents walk only one step and receive rewards based on whether they arrived at the correct entity. This ensures efficiency. Additionally, we opted for this simplified setting for the following reasons: i) CONQUER’s specific KG representation enables us to answer complex questions involving qualifiers within one hop that would require two-hop walking in case of other KG representations, and ii) we have noticed that walking beyond two steps is rarely required for answering natural occurring user requests. Nevertheless, our learning framework allows for multi-step sequential decision-making: By adding a self-loop action, agents could decide on whether they want to stay at the current KG entity (= terminal state), which is then returned as answer to the user or they continue to walk over the KG to the next entity node  $e_{next}$ . Then,  $s_{next} = (Q_t^{cxt}, Q_t, p_{walk}, e_{next})$ , where  $p_{walk}$  is the concatenated paths the agent has traversed so far.

In the following, we describe how we obtain the rewards for CONQUER.

### 3.4.3 Reformulation Prediction

Each answer entity  $e^A$ , reached by an agent after taking the sampled action, is presented to the user. An ideal user, according to our assumption, would ask a follow-up question that is either a reformulation of the same intent (if she thinks the answer is wrong), or an expression of a new intent (if the answer seems correct). This sequence of the original question and the follow-up is then passed on to a reformulation detector to decide whether the two questions express the same intent. We devise such a predictor by fine-tuning a BERT model for sentence pair classification [Devlin et al.(2019)] on a large set of such question pairs. Based on this prediction (reformulation or not), we deduce if the generated answer entity  $e^A$  was correct (no reformulation) or not (reformulation). The agent receives a positive reward (+1) for a correct answer, otherwise a negative one (-1).

Nature of reformulation	Percentage
Words were replaced by synonyms	15%
Expected answer types were added	14%
Coreferences were replaced by topic entity	24%
Whole question was rephrased	71%
Words were reordered	5%
Completed a partially implicit question	20%

Table 3.2: Types of reformulations in CONVREF. Reformulation may belong to multiple categories.

### 3.5 Answer Generation

The learned policy can now be used to generate answers for conversational questions. This happens in two steps: i) selecting actions by individual agents to reach candidate answers (where actions correspond to KG paths and selecting them implies reaching an entity) and ii) ranking the candidates to produce the final answer.

**Selecting actions.** Given a question  $Q_i$  at turn  $t$ , the first step is to extract the context entities  $\{e_t^{cxt}\}$ . Usually there are several of these and, therefore, several starting points for RL walks. Multiple agents traverse the KG in parallel, based on the predictions coming from the trained policy network. Each agent takes the top- $k$  predicted actions ( $k$  is typically small, five in our experiments) from the policy network greedily (no explorations at answering time).

**Ranking answers.** Agents land at end points after following actions predicted for them by the policy network. These are all candidate answers  $\{e^A\}$  that need to be ranked. We interpret the probability scores coming from the network (associated with the predicted action) as a measure of the system’s confidence in the answer and use it as our main ranking criterion. When multiple agents land at the same end point, we use this to boost scores of the respective candidates by adding scores for the individual actions. Candidate answers are then ranked by this final score, and the top-1 entity is shown to the user.

### 3.6 Benchmark with Reformulations

None of the popular QA benchmarks, like WebQuestions [Berant et al.(2013)], ComplexWebQuestions [Talmor and Berant(2018)], QALD [Usbeck et al.(2018)], LC-QuAD [Dubey et al.(2019), Trivedi et al.(2017)], or CSQA [Saha et al.(2018)], contain sessions with question reformulations by real users. The only publicly available benchmark for ConvQA over KGs based on a real user study is CONVQUESTIONS [Christmann et al.(2019)]. Therefore, we used conversation sessions in CONVQUESTIONS as input to our own user study to create our benchmark CONVREF.

**Workflow for user study.** Study participants interacted with a baseline ConvQA system. In this way, we were able to collect real reformulations issued in response to seeing a wrong answer, rather than static paraphrases. To create such a baseline system we trained our randomly initialized policy network with simulated reformulation chains. CONVQUESTIONS comes with one interrogative

<b>Original question:</b> <i>in what location is the movie set?</i> [Movies]
<b>Wrong answer:</b> Doctor Sleep
<b>Reformulation:</b> <i>where does the story of the movie take place?</i>
<b>Original question:</b> <i>which actor played hawkeye?</i> [TV Series]
<b>Wrong answer:</b> M*A*S*H Mania
<b>Reformulation:</b> <i>name of the actor who starred as hawkeye?</i>
<b>Original question:</b> <i>release date album?</i> [Music]
<b>Wrong answer:</b> 01 January 2012
<b>Reformulation:</b> <i>on which day was the album released?</i>
<b>Original question:</b> <i>what's the first one?</i> [Books]
<b>Wrong answer:</b> Agatha Christie
<b>Reformulation:</b> <i>what is the first miss marple book?</i>
<b>Original question:</b> <i>who won in 2014?</i> [Soccer]
<b>Wrong answer:</b> NULL
<b>Reformulation:</b> <i>which country won in 2014?</i>

Table 3.3: Sample reformulations from CONVREF.

paraphrase per question, which could be viewed as a weak proxy for reformulations. A paraphrase was triggered as reformulation in case of a wrong answer during training.

The conversations shown to the users were topically grounded in the 350 seed conversations in CONVQUESTIONS. Since we have paraphrases for each question, we also use the conversations where the original questions are replaced by their paraphrased version. This way we obtain 700 conversations, each with 5 turns. Users were shown follow-up questions in a given conversation interactively, one after the other, along with the answer coming from the baseline QA system. For wrong answers, the user was prompted to reformulate the question up to four times if needed. In this way, users were able to pose reformulations based on previous wrong answers and the conversation history. Note that the baseline QA system often gave wrong answers, as our RL model had not yet undergone full training. This provided us with a challenging stress-test benchmark, where users had to issue many reformulations.

**Participants.** We hired 30 students (Computer Science graduates) for the study. Each participant annotated about 23 conversations. ( $23 \times 30 = 690$ , the remaining ten were randomly assigned to the participants). The total effort required 7 hours per person (including a 1-hour briefing), and each participant was paid 10 Euros per hour (comparable to AMT Master workers). The final session data was sanitized to comply with privacy regulations.

**Final benchmark.** The reformulations from our user study were compiled into the CONVREF benchmark as follows. CONVQUESTIONS generated its original  $11.2k$  conversations by creating question sequences from either a seed question or its paraphrase. Thus, each utterance in the original benchmark can be found among our 700 annotated conversations. So we augment each original utterance by up to 4 reformulations (along the wrong system responses), that we collected. This resulted in a total of  $262k$  turns with about  $205k$  reformulations. We followed CONVQUESTIONS

ratios for the train-dev-test splits, leading to 6.7k training conversations and 2.2k each for dev and test sets. While participants could freely reformulate questions, we noticed different patterns in a random sample of 100 instances (see Table 3.2). Examples of reformulations are shown in Table 3.3. Reformulations had an average length of about 7.6 words, compared to about 6.7 for the initial questions per session. The complete benchmark is available at <https://conquer.mpi-inf.mpg.de>.

## 3.7 Experimental Framework

### 3.7.1 Setup

**KG.** We used the Wikidata NTriples dump from 26 April 2020 with about 12B triples. Triples containing URLs, external IDs, language tags, redundant labels and descriptions were removed, leaving us with about 2B triples. The data was processed according to Section 3.2.1 and loaded into a Neo4j<sup>1</sup> graph database. All data resided in main memory (consuming about 35 GB, including indexes) and was accessed with the Cypher query language.

**Context entities.** We used ELQ [Li et al.(2020)] to obtain NED scores. Frequency clip  $f_{max}$  was set to 100. Parameters  $h_1, h_2, h_3, h_4, h_{cxt}$  were tuned on our development set of 100 manually annotated utterances and set to 0.1, 0.1, 0.7, 0.1, 0.25, respectively (see Section 3.3).

**RL and neural modules.** The code for the RL modules was developed using the TensorFlow Agents<sup>2</sup> library. When the number of KG paths for a context entity exceeded 1000, a thousand paths were randomly sampled owing to memory constraints. All models were trained for 10 epochs, using a batch size of 1000 and 20 rollouts per training sample. All reported experimental figures are averaged over five runs, resulting from differently seeded random initializations of model parameters. We used the Adam optimizer [Kingma and Ba(2015)] with an initial learning rate of 0.001. The weight  $\beta$  for the entropy regularization term is set to 0.1. We used a pre-trained uncased BERT-base model from Hugging Face<sup>3</sup> for obtaining encodings of  $\{Q, Q^{cxt}, A\}$ . To obtain encodings of a sequence, two averages were performed: once over all hidden layers, and then over all input tokens. Dimension  $d = 768$  (from BERT models), and accordingly sizes of the weight matrices were set to  $\mathbf{W}_1 = |input| \times 768$  and  $\mathbf{W}_2 = 768 \times 768$ , where  $|input| = d$  or  $|input| = 2d$  (in case we prepend context questions  $Q_t^{cxt}$ ).

**Reformulation predictor.** To avoid any possibility of leakage from the training to the test data, this classifier was trained only on the CONVREF dev set (as a proxy for an orthogonal source). We fine-tuned the BERT sequence classification model from Hugging Face<sup>4</sup>, with a learning rate of  $3 \times 10^{-5}$  and a batch size of 64 for our classification task. Positive samples were generated by pairing questions within the same intent (= reformulations), while negative sampling was done by pairing across different intents from the *same conversation*. This ensured lexical overlap in negative samples, necessary for more discriminative learning.

<sup>1</sup><https://neo4j.com>

<sup>2</sup><https://www.tensorflow.org/agents>

<sup>3</sup><https://huggingface.co/bert-base-uncased>

<sup>4</sup>[https://huggingface.co/docs/transformers/model\\_doc/bert#transformers.TFBertForSequenceClassification](https://huggingface.co/docs/transformers/model_doc/bert#transformers.TFBertForSequenceClassification)

### 3.7.2 CONQUER Configurations

The CONQUER method has four different setups for training, that are evaluated at answering time. These explore two orthogonal sources of noise and stem from two settings for the user model and two for the reformulation predictor.

**User model:** Although CONVREF is based on a user study, we do not have access to users during model training. Like many other Monte Carlo RL methods, we simulate user behavior for accumulating more interactions that could enrich our training (for example, by performing rollouts). For user simulation, we rely on the reformulations collected in our study. We define ideal and noisy user models as follows.

- **Ideal:** In an ideal user model, we assume that users always behave as expected: reformulating when the generated answer is wrong and only moving on to a new intent when it is correct. We simulate this behavior as follows. For each intent, we have at most 5 different question variants available in CONVREF (1 original question and up to 4 reformulations). If our model gives a wrong answer, we would trigger the next available reformulation for the underlying question intent. If we run out of reformulations, we simply start again with the original question formulation, looping through the sequence of user utterances within the same intent, until the correct answer is reached, or discarding the training intent after a fixed number of tries (5 in our experiments).
- **Noisy:** In the noisy variant, the user is free to move on to a new intent even when the response to the last turn was wrong. This models realistic situations when a user may simply decide to give up on an information need (out of frustration, say). We simulate this behavior as follows. Just like in the ideal case, we trigger available reformulations for the given question intent, one after the other, in case of wrong answers. Once we run out of reformulations for this intent, we would issue the follow-up information need, regardless of having seen a correct response.

**Reformulation predictor:**

- **Ideal:** In an ideal reformulation predictor, we assume that it is known upfront whether a follow-up question is a reformulation or not. We have this information available, since we simulate users, employing the reformulations from CONVREF.
- **Noisy:** In the noisy predictor, we use the BERT-based reformulation detector which may include erroneous predictions.

### 3.7.3 Evaluation

**Benchmarks.** We evaluate CONQUER on the test set of our newly created benchmark with reformulations, CONVREF. We have also collected up to 4 reformulations for each test question. Therefore, we allowed models to have up to 5 answering attempts per question intent, receiving the next available reformulation after a wrong answer. The model’s performance is then measured over distinct intents instead of individual questions/reformulations. Additionally, we evaluate CONQUER on CONVQUESTIONS.

Method	P@1	Hit@5	MRR	RefTriggers	Ref = 0	Ref = 1	Ref = 2	Ref = 3	Ref = 4
CONQUER IdealUser-IdealRef	0.339	0.426	0.376	30058	<b>3225</b>	292	154	70	56
CONQUER IdealUser-NoisyRef	0.338	<b>0.429</b>	0.377	30358	3099	338	170	79	100
CONQUER NoisyUser-IdealRef	<b>0.353</b>	0.428	<b>0.387</b>	<b>29889</b>	3163	403	187	<b>90</b>	<b>116</b>
CONQUER NoisyUser-NoisyRef	0.335	0.417	0.370	30726	2913	<b>425</b>	<b>216</b>	<b>90</b>	104
CONVEX	0.225	0.257	0.241	34861	1980	278	200	24	35

Table 3.4: Main results on answering performance over the CONVREF test set. All CONQUER variants outperformed CONVEX with statistical significance, and required less reformulations than CONVEX to provide the correct answer.

CONQUER/Baseline	Movies	TV Series	Music	Books	Soccer
IdealUser-IdealRef	0.320	0.316	0.281	<b>0.449</b>	<b>0.329</b>
IdealUser-NoisyRef	0.344	0.340	0.303	0.425	0.308
NoisyUser-IdealRef	<b>0.368</b>	<b>0.367</b>	<b>0.324</b>	0.413	<b>0.329</b>
NoisyUser-NoisyRef	0.327	0.296	0.300	0.381	0.327
CONVEX	0.274	0.188	0.195	0.224	0.244

Table 3.5: P@1 results across topical domains.

**Baseline.** We use the CONVEX system [Christmann et al.(2019)] as the state-of-the-art ConvQA baseline in our experiments. CONVEX detects answers to conversational utterances over KGs in a two-stage process based on judicious graph expansion: it first detects so-called frontier nodes that define the context at a given turn. Then, it finds high-scoring candidate answers in the vicinity of the frontier nodes. Hyperparameters of CONVEX were tuned on CONVREF for fair comparison.

**Metrics.** Answers form a ranked list, where the number of correct answers is usually one, but sometimes two or three. We use three standard metrics for evaluating QA performance: i) precision at the first rank (P@1), ii) answer presence in the top-5 results (Hit@5) and iii) mean reciprocal rank (MRR). Additionally, we count the total number of reformulations triggered by a model to arrive at correct answers (max. 4 reformulations per intent), “*RefTriggers*”. The smaller this number is, the better. We also count the number of intents that have been answered without triggering any reformulation (“*Ref=0*”), triggering one (“*Ref=1*”), up to four reformulations (“*Ref=4*”). The more intents answered earlier, the better. We use the standard metrics of i) precision, ii) recall and iii) F1-score for evaluating context entity detection quality. These measures are also used for assessing reformulation prediction performance, where the output is one of two classes: reformulation or not. Gold labels are available from CONVREF.

## 3.8 Results and Insights

### 3.8.1 Key Findings

Tables 3.4 and 3.5 show our main results on the CONVREF test set. As explained earlier, P@1, Hit@5 and MRR are measured over distinct intents, not utterances. For example, even when an intent is satisfied only at the third reformulation, we deem  $P@1 = 1$  (and 0 when the correct answer

Method	P@1	Hit@5	MRR
CONQUER	<b>0.263</b>	<b>0.343</b>	<b>0.298</b>
CONVEX	0.184	0.219	0.200

Table 3.6: Performance on CONVQUESTIONS.

is not found after five attempts). Statistical significance tests are performed wherever applicable: we used the McNemar’s test for binary metrics (P@1, Hit@5) and the  $t$ -test for real-valued ones (MRR, F1). Tests were unpaired when there are unequal numbers of utterances handled in each case due to unequal numbers of reformulations triggered and paired in other standard cases. 1-sided tests were performed for checking for superiority (for baselines) or inferiority (for ablation analyses). In all cases, null hypotheses were rejected when  $p \leq 0.05$ . Best values in table columns are marked in **bold**, wherever applicable.

**CONQUER is robust to noisy models.** We did not observe any major differences among the four configurations of CONQUER. While the “*IdealUser-IdealRef*” model is able to answer more questions correctly at the first attempt (“*Ref = 0*”), “*NoisyUser-IdealRef*” requires least reformulations in total (“*RefTriggers*”). Interestingly, some of the noisy versions (“*IdealUser-NoisyRef*” and “*NoisyUser-IdealRef*”) even achieve the absolute best numbers on the metrics, indicating that a certain amount of noise and non-deterministic user behavior may in fact help the agent to generalize better to unseen conversations. Note that the variants with ideal models are not to be interpreted as potential upper bounds for QA performance: while “*IdealUser*” represents a model of systematic user behavior, “*IdealRef*” rules out one source of model error.

**CONQUER outperforms CONVEX.** All variants of CONQUER were significantly better than the baseline CONVEX on the three metrics ( $p < 0.05$ , 1-sided tests), showing that CONQUER successfully learns from the implicit feedback from question reformulations. CONVEX on the other hand, as well as any of the existing (conversational) KG-QA systems [Saha et al.(2018), Guo et al.(2018), Shen et al.(2019)], cannot learn from incorrect answers and indirect signals such as reformulations. Note that CONVEX was evaluated in the same way as CONQUER for fair comparison: it had 5 attempts to answer each question intent, receiving reformulations after a failed attempt. Additionally, CONQUER can also be applied in the standard setting where  $\langle$ question, gold answer $\rangle$  instances are available, in the following way: CONQUER receives a positive reward (+1) for returning an answer that matches with the ground truth answer, and a negative reward (−1) for returning wrong answers. When trained on the original CONVQUESTIONS benchmark, that contains gold answers but lacks reformulations, CONQUER again outperforms CONVEX (with an increase of 8% in P@1, see Table 3.6).

**CONQUER needs fewer reformulations.** In Table 3.4, “*RefTriggers*” shows the number of reformulations needed to arrive at a correct answer (or reaching the maximum of 5 turns). We observe that CONQUER triggers substantially fewer reformulations ( $\simeq 30k$ ) than CONVEX ( $\simeq 34k$ ). This confirms an intuitive hypothesis that when a model learns to answer better, it also satisfies an intent faster (less turns needed per intent). Zooming into this statistic (“*Ref = 0, 1, 2, ...*”), we observe that CONQUER answers a bulk of the intents without needing any reformulation, a testimony to

All	No Overlap	No Match	No NED	No prior
<b>0.731</b>	0.726	0.728	0.684	0.718

Table 3.7: Ablation study for context entity detection (F1).

Context model	P@1	Hit@5	MRR
Curr. ques. + Cxt. ent.	<b>0.294</b>	<b>0.407</b>	<b>0.346</b>
Curr. ques. + Cxt. ent. + First ques.	0.254	0.370	0.305
Curr. ques. + Cxt. ent. + First ques. + Prev. ques.	0.257	0.370	0.307
Curr. ques. + Cxt. ent. + First refs. + Prev. refs.	0.262	0.382	0.316

Table 3.8: Effect of context models on answering performance.

successful training ( $\simeq 3k$  in comparison to  $\simeq 2k$  for CONVEX). The numbers quickly taper off with subsequent turns in a conversation, but remain higher than the baseline. CONVEX relies on a context graph that is iteratively expanded over turns; this often becomes unwieldy at deeper turns. Unlike CONVEX, we found CONQUER’s performance to be relatively stable even for higher intent depths (P@1 = 0.237, 0.397, 0.336, 0.319, 0.385 for intents 1 through 5, respectively).

**CONQUER works well across domains.** P@1 results for the five topical domains in CONVQUESTIONS are shown in Table 3.5. We note that the good performance of CONQUER is not just by skewed success on one or two favorable domains (while being significantly better than CONVEX for each topic), but holds for all five of them (books being slightly better than the others).

### 3.8.2 In-depth Analysis

Having shown the across-the-board superiority of CONQUER over CONVEX, we now scrutinize the various components and design choices that make up the proposed architecture. All analysis experiments are reported on the CONVREF dev set, and the realistic “*NoisyUser-NoisyRef*” CONQUER variant is used by default.

**All features vital for context entity detection.** We first perform an ablation experiment (Table 3.7) on the features responsible for identifying context entities. Observing F1-scores averaged over questions, it is clear that all four factors contribute to accurately identifying context entities (no NED as well as no prior scores resulted in statistically significant drops). It is interesting to understand the trade-off here: a high precision indicates a handful of accurate entities that may not create sufficient scope for the agents to learn meaningful paths. On the other hand, a high recall could admit a lot of context entities from where the correct answer may be reachable, but via spurious paths. The F1-score is thus a reliable indicator for the quality of a particular tuple of hyperparameters.

**Context entities effective in history modeling.** After examining features for  $E^{cxt}$ , let us take a quick look at the effect of  $Q^{cxt}$ . We tried several variants and show the best four (including the

Method	Fine-tuned BERT			Fine-tuned RoBERTa		
	Prec	Rec	F1	Prec	Rec	F1
New intent	0.986	0.944	0.965	0.988	0.924	0.955
Reformulations	0.810	0.948	0.873	0.760	0.956	0.847

Table 3.9: Classification performance for reformulations.

Method	P@1	Hit@5	MRR
Path	<b>0.294</b>	0.407	<b>0.346</b>
Path + Answer entity	0.275	0.394	0.329
Context entity + Path	0.293	<b>0.408</b>	<b>0.346</b>
Context entity + Path + Answer entity	0.273	0.398	0.328

Table 3.10: Design choices of actions for the policy network.

one used in CONQUER) in Table 3.8. While the CONQUER architecture is meant to have scope for incorporating various parts of a conversation, we found that explicitly encoding previous questions significantly degraded answering quality (the first row, where  $Q^{cxt} = \phi$ , works significantly better than all other options). “*refs.*” indicate that embeddings of reformulations for that intent were averaged. Without “*refs.*”, only the first question in that intent is used. Results indicate that context entities from the KG suffice to create a satisfactory representation of the conversation history. Note that these  $E_t^{cxt}$  are derived not just from the current turn, but are carried over from previous ones. Nevertheless, we believe that there is scope for using  $Q^{cxt}$  better: history modeling for ConvQA is an open research topic [Qu et al.(2019a), Qu et al.(2019b), Gupta and Sharma(2021), Qiu et al.(2021)], and reformulations introduce new challenges.

**Reformulation predictor works well.** A crucial component of the success of CONQUER’s “*NoisyRef*” variants is the reformulation detector. Due to its importance, we explored several options like fine-tuned BERT [Devlin et al.(2019)] and fine-tuned RoBERTa [Liu et al.(2019a)] models to perform this classification. RoBERTa produced slightly poorer performance than BERT, which was really effective (Table 3.9). Prediction of new intents is observed to be slightly easier (higher numbers) due to expected lower levels of lexical overlaps.

**Path label preferable as actions.** When defining what constitutes an action for an agent, we have the option of appending the answer entity  $e^A$  or the context entity  $e^{cxt}$  to the KG path (world knowledge in BERT-like encodings often helps in directly finding good answers). We found, unlike similar applications in KG reasoning [Das et al.(2017), Lin et al.(2018), Qiu et al.(2020)], excluding  $e^A$  actually worked significantly better for us (Table 3.10, row 1 vs. 2). This can be attributed to the low semantic similarity of answer nodes with the question, that acts as a confounding factor. Including  $e^{cxt}$  does not change the performance (row 1 vs. 3). The reason is that an agent selects actions starting at one specific start point ( $e^{cxt}$ ): all of these paths thus share the embedding for this start point, resulting in an indistinguishable performance. The last row corresponds to matching the question to the entire KG fact, which again did not work so well due to the potentially low semantic

similarity of the answer entity  $e^A$  with the question.

**Error analysis points to future work.** We analyzed 100 random samples where CONQUER produced a wrong answer ( $P@1 = 0$ ). We found them to be comprised of: 17% ranking errors (correct answer in top-5 but not at top-1), 23% action selection errors (context entity correct but path wrong), 30% context entity detection errors (including 3% empty set), 23% not in the KG (derived quantitative answers), and 7% wrong gold labels.

**Answer ranking robust to minor variations.** Our answer ranking (Section 3.5) uses cumulative prediction scores (scores from multiple agents added), with  $P@1 = 0.294$ . We explored variants where we used prediction scores with ties broken by majority voting since an answer is more likely if more agents land on it ( $P@1 = 0.291$ ), majority voting with ties broken with higher prediction scores ( $P@1 = 0.273$ ), and taking the candidate with the highest prediction score without majority voting ( $P@1 = 0.294$ ). Most variants were nearly equivalent to each other, showing *robustness of the learnt policy*.

**Runtimes.** The policy network of CONQUER takes about 10.88 ms to produce an answer, as averaged over test questions in CONVREF. The maximal answering time was 1.14 s.

### 3.9 Related Work

**QA over KGs.** KG-QA has a long history [Yahya et al.(2012), Berant et al.(2013), Unger et al.(2014), Roy and Anand(2020)], evolving from addressing simple questions via templates [Bast and Haussmann(2015), Abujabal et al.(2017)] and neural methods [Yih et al.(2015), Huang et al.(2019b)], to more challenging settings of complex [Lu et al.(2019), Bhutani et al.(2019)], heterogeneous [Oguz et al.(2020), Sun et al.(2019)] and conversational QA [Shen et al.(2019), Müller et al.(2019), Christmann et al.(2019)] (see Section 2.1.3 for more detail on related ConvQA over KG work). However, these methods do not consider question reformulations in conversational sessions and solely learn from question-answer training pairs.

**Reformulations in search and QA.** In the parallel field on text-QA, question reformulation or rewriting has been pursued as conversational question completion [Vakulenko et al.(2021), Anantha et al.(2021), Yu et al.(2020), Ren et al.(2018), Xu et al.(2020)]. In our work, we revive the more traditional sense of reformulations [Chang et al.(2006), Awadallah et al.(2013), Dang and Croft(2010), Jansen et al.(2009)], where users pose queries in a different way when system responses are unsatisfactory. Several works on search and QA apply RL to automatically generate or retrieve reformulations that would proactively result in the best system response [Buck et al.(2018), Nogueira and Cho(2017), Das et al.(2019), Ponnusamy et al.(2020)]. In contrast, CONQUER learns from free-form user-generated reformulations. Question paraphrases can be considered as proxies of reformulations, without considering system responses. Paraphrases have been leveraged in a number of ways in QA [Berant and Liang(2014), Dong et al.(2017), Fader et al.(2013)]. However, such models ignore information about sequences of user-system interactions in real conversations.

**User feedback in QA.** Many existing methods leverage explicit, positive user feedback as a source of additional training data [Abujabal et al.(2018), Kratzwald and Feuerriegel(2019), Zhang

et al.(2019), Hancock et al.(2019), Campos et al.(2020)] (see Section 2.2 for further details). Such explicit feedback is hard to obtain at scale, as it incurs a substantial burden on the user. In contrast, CONQUER is based on the more realistic setting of implicit feedback from reformulations, which do not intrude on the user’s natural behavior.

## 3.10 Conclusion

This chapter introduces the first framework for learning continuously from a stream of utterances and reformulations in conversational question answering over knowledge graphs. Our RL-based method CONQUER successfully leverages implicit negative feedback when users reformulate previously failed questions. User reformulations are a non-intrusive feedback, which becomes part of the natural conversation flow and keeps the user effort minimal. However, CONQUER relies on the assumption that users would provide reformulations each time, while in a realistic scenario users might be less patient if the system repeatedly fails to provide the correct answer. We explored noisy user behavior to some extent (moving on to new intent even though answer was wrong): Experiments with a benchmark based on a user study showed that CONQUER is robust to noisy user models and reformulation predictors, and all variants significantly outperformed the baseline system CONVEX [Christmann et al.(2019)]. This is only the first step towards a QA setting for learning without static QA pairs. Future work could further investigate realistic user behavior.



## Chapter 4

# Learning from Data Augmentation Guided by Performance-based Feedback

### Contents

---

<b>4.1</b>	<b>Introduction</b>	<b>54</b>
<b>4.2</b>	<b>The REIGN Framework</b>	<b>56</b>
<b>4.3</b>	<b>Reformulation Category Selector</b>	<b>58</b>
4.3.1	Reformulation Taxonomy	58
4.3.2	RCS Model Training	59
4.3.3	RCS Model Usage	61
<b>4.4</b>	<b>Reformulation Generator</b>	<b>61</b>
4.4.1	RG Model Training	61
4.4.2	RG Model Usage	63
<b>4.5</b>	<b>Conversational Question Answering</b>	<b>63</b>
4.5.1	ConvQA Model Training	63
4.5.2	ConvQA Model Usage	65
<b>4.6</b>	<b>Experimental Setup</b>	<b>65</b>
<b>4.7</b>	<b>Results and Insights</b>	<b>66</b>
4.7.1	Key Findings	66
4.7.2	In-depth Analysis	68
<b>4.8</b>	<b>Related Work</b>	<b>70</b>
<b>4.9</b>	<b>Conclusion</b>	<b>71</b>

---

In the previous chapter, we leveraged user reformulations for learning. While this resulted in valuable implicit feedback, obtaining user reformulations at scale can be costly. Therefore, this chapter looks into harvesting system-generated reformulations to train more robust ConvQA models.

Models for ConvQA over KGs are usually trained and tested on benchmarks of gold QA pairs. This implies that training is limited to surface forms seen in the respective datasets, and evaluation is

on a small set of held-out questions. Through our framework REIGN, we take several steps to remedy this restricted learning setup. Sections 4.2 – 4.5 describe the REIGN components in detail. REIGN systematically generates reformulations for conversational questions, which is a challenging problem, given the incomplete nature of such questions. It guides ConvQA models towards better performance by feeding them only those reformulations that help improve answering quality. REIGN is based on a deep reinforcement learning approach. Experiments are presented in Sections 4.6 and 4.7. For a rigorous evaluation of robustness for trained models, we use and release large numbers of diverse reformulations generated by prompting ChatGPT for benchmark test sets (resulting in 20x increase in size). Our findings show that ConvQA models with robust training via reformulations significantly outperform those with standard training from gold QA pairs only.

## 4.1 Introduction

**Motivation.** Answering questions about entities, powered by curated knowledge graphs (KGs) at the backend, is a vital component of Web search [Bast and Haussmann(2015), Linjordet and Balog(2022), Roy and Anand(2021), Yih et al.(2015)]. Nowadays, users’ information needs are increasingly being expressed as a conversation, in a sequence of questions and answers  $\langle Q_t, A_t \rangle$ , over turns  $\{t\}$  [Zamani et al.(2023), Dalton et al.(2022), Owoicho et al.(2023)]:

*Q*<sub>1</sub>: *What’s the 2022 LOTR TV series called?*  
*A*<sub>1</sub>: *The Rings of Power (TROP)*  
*Q*<sub>2</sub>: *TROP airing on?*  
*A*<sub>2</sub>: *Amazon Prime Video*  
*Q*<sub>3</sub>: *Which actor plays Isildur in the series?*  
*A*<sub>3</sub>: *Maxim Baldry*  
*Q*<sub>4</sub>: *And who in the Jackson trilogy?*  
*A*<sub>4</sub>: *Harry Sinclair*  
*Q*<sub>5</sub>: *When did the series start? ...*

A conversation over a KG contains a set of entities (“*The Lord of the Rings: The Rings of Power*”, “*Amazon Prime Video*”), their relationships (“*aired on*”), and types (“*TV series, video streaming service*”). ConvQA has high contemporary interest [Kacupaj et al.(2022), Christmann et al.(2023), Qu et al.(2020), Li et al.(2022a), Jeong et al.(2023), Jain and Lapata(2024)], spurred on to a big extent by systems like ChatGPT that support a conversational interface.

**Limitations of state-of-the-art.** We quantify *robustness* in QA in terms of the number of distinct question formulations of a given intent that a QA model can answer correctly: the higher this number, the more robust the model. Methods for conversational question answering (ConvQA) over KGs are usually trained and evaluated on benchmarks of gold-standard (question, answer) pairs [Saha et al.(2018), Shen et al.(2019), Guo et al.(2018), Christmann et al.(2022b)]. Such a paradigm limits *robust learning* by being restricted to question formulations roughly seen during training time. One approach in QA to demonstrate generalizability is to train and evaluate models on multiple benchmarks [Kacupaj et al.(2022), Marion et al.(2021), Lan and Jiang(2021)]. This only addresses the problem partially: the training and evaluation are still limited to surface forms seen in any of the benchmarks. A particular aspect of existing benchmarks, that is attributable to their

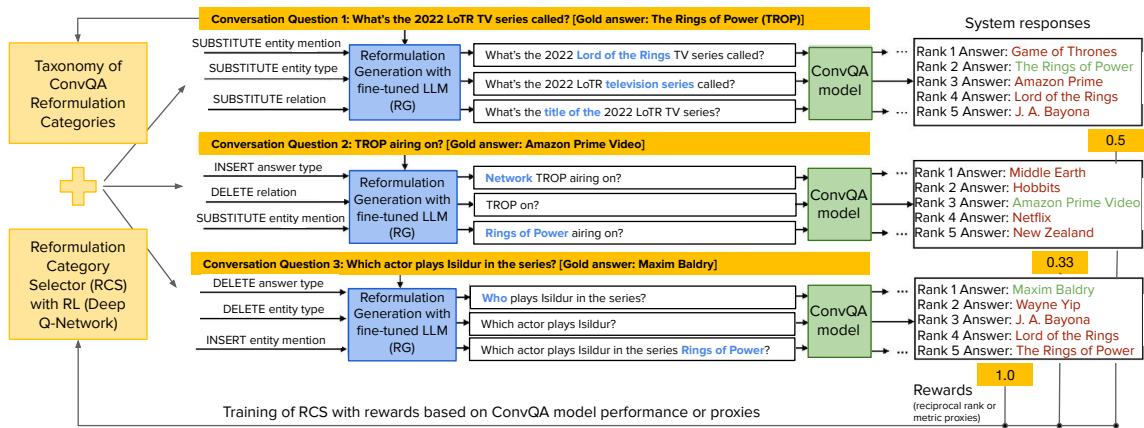


Figure 4.1: Performance-guided reformulation generation in REIGN, illustrated through our running example conversation.

construction choices via graph sampling [Saha et al.(2018)] or crowdsourcing guidelines [Christmann et al.(2019), Christmann et al.(2022b)], is that they often do not contain sloppy question formulations that could be asked by real users in the wild.

In the example conversation,  $Q_4$  is phrased in a very casual way, asking for Isildur’s actor in the LOTR movie trilogy (Peter Jackson directed the LOTR movies). With this difficult input, the QA system may give a wrong response. A seemingly natural approach to counter such effects would be to have the QA system automatically reformulate the question into a more complete version [Vakulenko et al.(2021), Raposo et al.(2022), Yu et al.(2020), Chen et al.(2022b), Anantha et al.(2021)], such as *Which actor played the role of Isildur in the Lord of the Rings movie trilogy directed by Peter Jackson?* – this kind of run-time question rewriting to a complete natural language form in a deployed system may sometimes work, but adds inference-time overhead and may not always improve performance [Ishii et al.(2022b)].

**Approach.** We take a different route: instead of reformulating a conversational question at inference time, we strengthen the *training* of the ConvQA model by exposing it upfront to a larger variety of intent-preserving surface forms for the same training sample. Examples of such syntactic variations representing the same question semantics are in Figure 4.1, for  $Q_1 - Q_3$  (original questions in orange boxes, perturbed segments in reformulations in blue). With this more diverse training data, the ConvQA model learns to cope better with different syntactic formulations.

Our reformulations are created from first principles. We propose a *taxonomy* of reformulation categories for ConvQA, that systematically manipulates parts of a given conversational question, based on string edit operations. We generate noisy supervision data for each category to fine-tune an LLM, our reformulation generator (RG). New lexico-syntactic forms in reformulations originate via the use of a rich set of aliases in KGs, and the world knowledge in LLMs.

Given that our generated instances are noisy, it is unlikely that, for a given question, all categories of reformulations would improve the ConvQA model’s performance. As a result, for each question, we would like to judiciously select *a few of these* that are most beneficial. So we pass generated reformulations to the QA model we wish to improve, and obtain ranked answer lists as responses

– shown in boxes with green (correct) and red (incorrect) answers in the right half of Figure 4.1. The model’s answer performance *metrics* (or proxies) are used as rewards (in orange) to train a Reformulation Category Selector (RCS) with Deep Q-Networks [Mnih et al.(2015)], a form of RL that approximates *value functions*. The trained RCS is then used as a means for *model-specific data augmentation*: it selects only the top- $k$  reformulations that would be used as additional training data for the QA model for maximum performance improvement. Instances of such question-specific categories are in Figure 4.1 (left half). This entire framework, termed REIGN, (REInforced reformulation GeNeration) is the main contribution presented in this chapter.

**Evaluation.** To assess the benefits of REIGN, we perform experiments against two state-of-the-art baselines: CONQUER [Kaiser et al.(2021)] based on reinforcement learning, and EXPLAINNN [Christmann et al.(2023)] based on graph neural networks. Note that REIGN operates by model-aware training on top of these baselines. For test data, we leverage the generative ability of ChatGPT (GPT-3.5) as a proxy to obtain human-like reformulations at scale: each original question is augmented with 20 distinct reformulations. This data is accessible at our project website <https://reign.mpi-inf.mpg.de> and our code is available on GitHub: <https://github.com/magkai/REIGN>.

**Contributions.** This chapter calls for more robust training and evaluation of ConvQA models, our salient contributions being:

- A novel *taxonomy* of question reformulations for ConvQA over KGs, based on string edit distance;
- A reinforcement learning model with *Deep Q-Networks*, that selects helpful reformulations of conversational questions guided towards better QA performance;
- About 335k *conversational question reformulations* of test cases in two ConvQA benchmarks, suitable for rigorous evaluation of future models;
- The REIGN *framework* with reusable components that judiciously augments benchmark training tailored to specific ConvQA models.

Table 4.1 gives an overview of the salient notation used in this chapter. The relevant concepts have either been introduced in the background chapter (see Section 2.1.1 for ConvQA concepts and 2.4.1 for RL concepts) or will be introduced next (see Section 4.2 for REIGN-specific concepts).

## 4.2 The REIGN Framework

An overview of the workflow in the proposed REIGN architecture is depicted in Figure 4.2. The pipeline consists of three trainable models, where the first two are our contributions:

- A **reformulation category selector (RCS) model**, that takes a question  $Q_t$  as input, and produces a reformulation category  $RC_t^i$  for transforming  $Q_t$ , as output;
- A **reformulation generator (RG) model**, that takes some  $Q_t$  and  $RC_t^i$  as input, and produces a reformulation  $Q_t^i$  of  $Q_t$  according to  $RC_t^i$ , as output;

Notation	Concept
$C, t \in \{1, 2, \dots\}$	Conversation, conversational turn
$Q = \langle q_1 \dots q_n \rangle, A$	Question and its tokens, Answer
$Q_t, A_t^{gold}$	Question at turn $t$ , its ground truth answer
$Q_t^i$	Reformulation of $Q_t$ based on ref. category $i$
$\langle A_t^i \rangle$	Predicted ranked answers for reformulations $Q_t^i$
$e, p \in K$	entity, predicate in KG
$e_t^{cxt} \in E_t^{cxt}$	context entities at turn $t$
$\langle RC^i \rangle$	RCS reformulation categories
$\langle RC_t^i \rangle_k$	top- $k$ predicted ref. categories for $Q_t$
$s \in S$	RCS states
$a \in \mathcal{A}$	RCS actions
$\mathcal{R}(\cdot)$	RCS reward function
$\Phi(\langle q_1 \dots q_n \rangle)$	Function to map $\langle q_i \rangle$ to state space
$M(s, \mathcal{A})$	Action masking vector
$Q(s, a)$	Q-value for $a$ in $s$
$Q^*(s, a)$	Optimal Q-value
$\pi$	RCS policy
$\alpha$	Step size in Q-Learning
$\gamma$	Discount factor in Q-Learning
$\mathbf{W}_1, \mathbf{W}_2 \in \theta$	Parameters of RCS Deep Q-Network
$Prob(\cdot)$	Probability
$\tau$	Boltzmann temperature for action sampling

Table 4.1: Notation for salient concepts in REIGN.

- An **external ConvQA model**, that takes some  $Q_t$  as input, and produces a ranked list of answers  $\langle A_t \rangle$  as output.

Reinforcement Learning (RL) is used to train the RCS model (Deep Q-Networks [Mnih et al.(2015)] in this work), with the goal of learning to select the most suitable transformation categories given a specific question, using existing QA performance metrics or suitable alternatives as reward signals. The categories come from our novel reformulation taxonomy. The RG model is trained with (distantly) supervised learning, using a language model (BART in our case [Lewis et al.(2020a)]), fine-tuned with questions, paired with a specific category, and the resulting reformulation in the form  $\langle (Q_t, RC_t^i); Q_t^i \rangle$ . This is distant supervision in the sense that the reformulations used for fine-tuning are generated in a noisy manner using rules following our taxonomy, and are not human reformulations. The ConvQA model used could be trained with supervised learning [Christmann et al.(2023), Kacupaj et al.(2022), Shen et al.(2019)] or RL [Kaiser et al.(2021)], according to its original training paradigm. In Figure 4.2, the original model ConvQA<sub>orig</sub> is trained with  $\langle Q_t, A_t^{gold} \rangle$  pairs in a ConvQA benchmark, while the more robust model ConvQA<sub>robust</sub> is trained on additional QA pairs where the reformulations  $\langle Q_t^i \rangle$  for a specific  $Q_t$  are also paired with the original gold answer  $A_t^{gold}$ . Note that in this work, we reuse ConvQA benchmark development sets for training the RCS model, fine-tuning the RG model, adjusting hyperparameters for all models, and selecting best REIGN configurations. We intentionally avoid large-scale learning of RCS and RG on train sets to

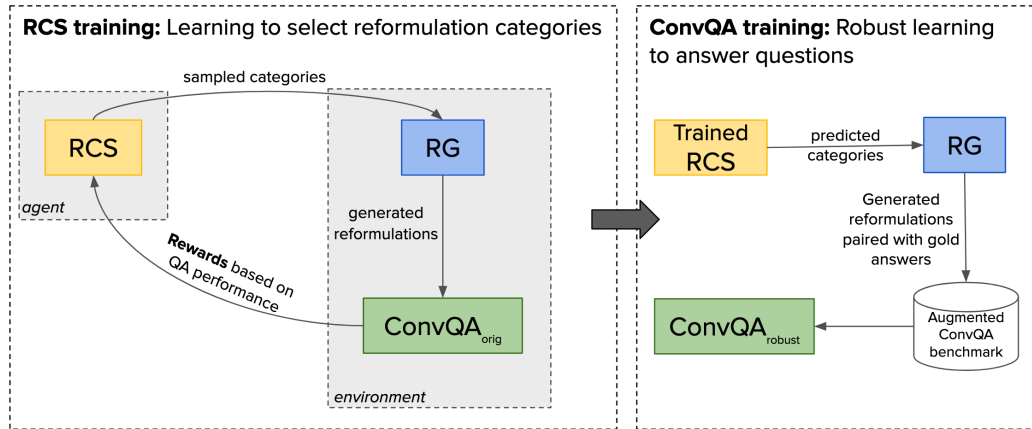


Figure 4.2: REIGN workflow: Training of Reformulation Category Selection with RL (including supervised RG training), followed by ConvQA training with data augmented with top reformulations.

stress-test generalizability of REIGN components: only the QA model is learned from the full train set. Further, any kind of leakage to test sets is thereby precluded for all models. Each component will now be described in more detail.

## 4.3 Reformulation Category Selector

### 4.3.1 Reformulation Taxonomy

**Categories.** We propose a taxonomy of reformulations, a topic that has mostly been treated as monolithic in past work [Buck et al.(2018), Hermjakob et al.(2002), Kaiser et al.(2021)]. To begin with, observe that a reformulation of a conversational question is a *modification* of its basic *parts*. Thus, a systematic generation of reformulations involves an understanding of these parts and meaningful modifications. For (Conv)QA over KGs, these basic question components comprise mentions of one or more entities, their types, predicates, and expected answer types. In analogy with string edit operations, our modifications include insertion, deletion and substitution. Transposition could be another basic operation, but we do not consider that in this work as reordering question phrases has little effect on several retrieval models. Viewing these three operations and the four parts of a question as operands, we obtain a taxonomy as shown in Figure 4.3, where reformulation categories are leaf nodes (marked in orange). Examples are “*INSERT entity-type*”, “*SUBSTITUTE relation*”, and “*DELETE relation*”. Note that we require our reformulations to be *intent-preserving*: this imposes constraints on what we can insert or substitute in the original question. We cannot, for example, replace an entity or relation by a different one – that would disturb the semantics of the conversation as a whole.

**Phenomena.** As shown with dashed boxes in Figure 4.3, our taxonomy subsumes several classes of conversational phenomena:

- Insertions *complete* the question to a more intent-explicit form;
- Deletions cause *ellipses* in context;

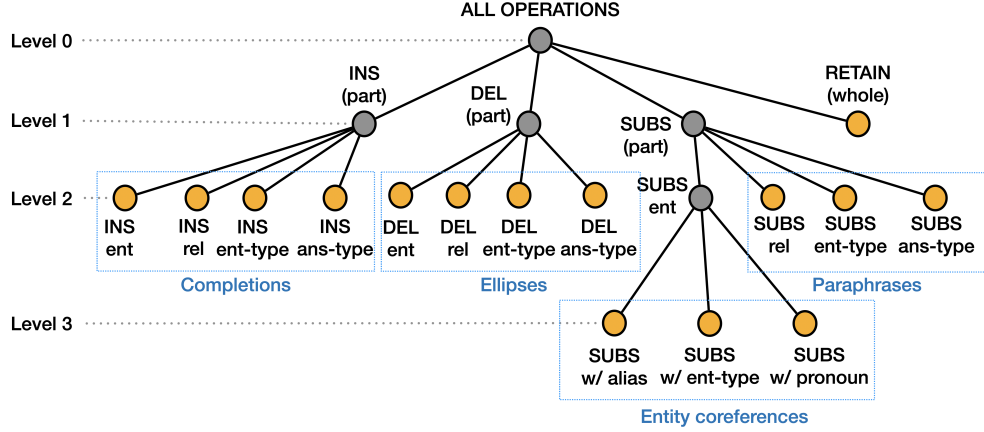


Figure 4.3: Taxonomy of reformulation categories. Legend: part = question-part; INS = Insert, DEL = Delete, SUBS = Substitute; ent = entity mention, rel = relation, ent-type = entity type mention, ans-type = answer type mention; w/ = with.

- Substitutions create *paraphrases*;
- Substituting entity mentions specifically leads to *coreferencing*.

The last operation can be sub-divided into three categories as per the case of substitution with a pronoun (“TROP”  $\mapsto$  “it”), or with its type (“TROP”  $\mapsto$  “the series”), or with an alias (“TROP”  $\mapsto$  “Rings of Power”). A special case in our taxonomy is the operation “RETAIN whole question”, where the question is left as such: it can be considered as a degenerate reformulation. Finally, we have **15 reformulation categories**,  $\langle RC^i \rangle$ , corresponding to these leaf nodes.

### 4.3.2 RCS Model Training

**Overall idea.** Given an input question and the taxonomy, we would like the Reformulation Category Selector (RCS) model to suggest some categories so that training with reformulations that belong to these categories would lead to better QA performance *metrics*. This, in turn, means that we would like the RCS to estimate *values* that correspond as much as possible to such metrics. This motivates us to use Deep Q-Networks (DQN) [Mnih et al.(2015)] (also see Section 2.4.3), a reinforcement learning approach that learns a value function approximator, where we use QA metrics as rewards. The estimate of the value of a (state, action) pair is in turn used to infer the policy for predicting or sampling actions. This is in contrast to the relatively more popular choice of learning policy gradients that directly model action probabilities given an input state (for example, the REINFORCE algorithm [Williams(1992)]).

Concretely, we employ DQNs to train an agent (the RCS model) to select actions ( $a \in \mathcal{A}$ ) (reformulation categories) given a current state ( $s \in \mathcal{S}$ ) (the input question  $Q_t$ ). The agent interacts with an environment (the reformulation generator and the ConvQA model, described later). This environment provides the next state  $s' \in \mathcal{S}$  (the generated reformulation  $Q_t^i$ ) and a reward  $r = \mathcal{R}(\cdot)$  (QA performance metric) to the agent. This is a Markov Decision Process (MDP) comprising states, actions, the transition function, and rewards  $(\mathcal{S}, \mathcal{A}, \delta, \mathcal{R})$ , with individual parts defined next. Algorithm 2 shows the precise steps of applying Deep Q-Learning in the RCS model.

- **States:** A state  $s \in \mathcal{S}$  is defined by a conversational question, represented by its encoding with function  $\Phi$ :  $s = \Phi(Q_t)$  (lines 3, 7 in Algorithm 2); We use BERT [Devlin et al.(2019)] embeddings in our experiments (averaged over each question token, and over all hidden layers).
- **Actions:** The set of actions  $\mathcal{A} = [1, \dots, 15]$  corresponds to the 15 reformulation categories  $\langle RC^i \rangle$  from our taxonomy. Note that every action (category) may not be available at every state. For instance, when a question does not have any mention of an entity type, it is not meaningful to apply the actions of deletion or substitution of an entity type. Therefore, we use *action masking* as follows to allow only *valid actions* to be chosen given the current state (when this information is available). A masking vector  $M(s, \mathcal{A})$  that has ones at indices corresponding to valid actions, and zeros elsewhere, is element-wise multiplied with the vector containing learned probabilities of actions at a given state (line 4 in Algorithm 2).
- **Transitions:** The transition function  $\delta$  is deterministic and updates  $s \in \mathcal{S}$  by applying one of the actions  $a \in \mathcal{A}$ , resulting in a new state  $s' \in \mathcal{S}$  that corresponds to the encoding of the reformulation  $Q_t^a$  (based on reformulation category  $RC_t^a$ ). The RG model then produces the actual reformulation (line 6 in Algorithm 2).
- **Rewards:** The reward function  $\mathcal{R}(\cdot)$  models the quality of the chosen action, and guides the agent towards its goal, which here is an improved answering performance. When a selected category leads to a reformulation on which the ConvQA model obtains *better performance than the original question*, the agent should get a high reward, and vice versa (line 8 in Algorithm 2 calls the ConvQA model to calculate this reward). Thus, an obvious choice here is to use any desired QA performance metric as the reward. We use the reciprocal rank (RR) [Voorhees(1999)] metric in this work, that is the reciprocal of the first rank at which a gold answer is found. We use it for the following reasons: (i) we have binary relevance of response entities, either correct (1) or incorrect (0); (ii) we deal with factoid QA, and there are usually only a few correct answers (typically between one and three for the benchmarks used). Formally, this reward, based on *reciprocal rank difference*, is computed as:

$$\mathcal{R}(\cdot) = \text{ReciprocalRank}(\langle A_t^i \rangle) - \text{ReciprocalRank}(\langle A_t \rangle) \quad (4.1)$$

where  $\langle A_t \rangle$  and  $\langle A_t^i \rangle$  are the ranked lists of responses by the ConvQA model to  $Q_t$  and  $Q_t^i$ , respectively. If the correct answer was not found in the top few positions (five in our experiments), then we set the reciprocal rank value to  $-1$ . The range of this reward then lies in the closed interval  $[-2, +2]$ . Since this nicely corresponds to a symmetric positive reward and punishment in respective cases of success and failure, we do not perform any further reward normalization. Note, however, that our framework can be used with any metric of choice: the use of RL removes the dependency on the metric being differentiable.

**Algorithm.** We use Deep Q-Networks (DQN) as our RL algorithm, which is a *model-free, value-based method* that learns to predict so-called Q-values  $\mathcal{Q}(s, a)$  for each state-action pair to quantify the usefulness of taking action  $a$  in state  $s$  under a policy  $\pi$  [Mnih et al.(2015)]. The policy is a function mapping states to actions based on the Q-values. Q-values are predicted via a neural Q-network with trainable parameters  $\theta$  (a two-layer feed-forward network in our case):

$$\mathcal{Q}_\theta(s, \langle a \rangle) = M(s, \mathcal{A}) \circ (\mathbf{W}_2 \times \text{ReLU}(\mathbf{W}_1 \times s)) \quad (4.2)$$

where  $\mathcal{Q}_\theta(s, \langle a \rangle)$  is a function returning a vector of size  $|\mathcal{A}|$  and stores the obtained values for every action  $a \in \mathcal{A}$  given some  $s \in \mathcal{S}$ ;  $\mathbf{s} \in \mathbb{R}^{d \times 1}$ ;  $d$  is the size of the input encoding vector;  $\mathbf{W}_1 \in \mathbb{R}^{h \times d}$ ,  $\mathbf{W}_2 \in \mathbb{R}^{|\mathcal{A}| \times h}$  are the weight matrices;  $M(s, \mathcal{A}) \in \mathbb{R}^{|\mathcal{A}| \times h}$  is the action mask; and hidden size  $h$  of embedding vectors is a tunable hyperparameter. ReLU is the non-linear activation function, and  $\odot$  denotes element-wise multiplication between two vectors.

The parameters  $\theta$  of the DQN ( $\mathbf{W}_1, \mathbf{W}_2$ ) are randomly initialized and updated batch-wise (lines 13 – 17 in Algorithm 2). A batch consists of a set of *experiences*: each experience is a tuple of the form  $(s, a, s', r)$ . The loss function is the mean-squared error between the temporal difference target  $(r + \gamma \cdot \max_{a'} \mathcal{Q}_{\theta_{prev}}(s', a'))$ , and the current Q-values of each state-action pair in the batch, with  $\theta_{prev}$  being the DQN’s parameters from the previous step [Mnih et al.(2013)] (see Section 2.4.3, Equation 2.2 and Equation 2.4 and line 17, Algorithm 2).

During training, the agent needs to explore different actions in each state via sampling. Here, we sample from a Boltzmann distribution to enable such exploration (line 5, Algorithm 2). A Boltzmann distribution is parameterized by a temperature  $\tau$  that we can use to conveniently control the degree of exploration:

$$\text{Prob}(a^{sample}, \tau) = \frac{e^{\mathcal{Q}_\theta(s, a^{sample})/\tau}}{\sum_{a \in \mathcal{A}} e^{\mathcal{Q}_\theta(s, a)/\tau}} \quad (4.3)$$

A  $\tau$ -value close to zero means taking the best action (with highest reward at this point) greedily more often, whereas larger values ( $\tau$  is unbounded) make the actual Q-values less relevant and the policy more like a random one.

### 4.3.3 RCS Model Usage

We train the RCS on the development set of a ConvQA benchmark, and apply it on the questions in the training set. At *RCS inference time*, the agent follows a greedy policy  $\pi$  with respect to Q-values, and typically chooses an action  $a^{greedy}$  in a state  $s$  as below:

$$a^{greedy} = \pi_s = \arg \max_{a \in \mathcal{A}} \mathcal{Q}_\theta(s, a) \quad (4.4)$$

In our case, we take the top- $k$  predicted reformulation categories  $\langle RC_t^i \rangle_k$  from the RCS (instead of the top-1 from the arg max) for each training question  $Q_t$ . The RG then picks these up to actually generate the new question variants  $\langle Q_t^i \rangle_k$ .

## 4.4 Reformulation Generator

### 4.4.1 RG Model Training

**Basic setup.** The reformulation generator (RG) is implemented by fine-tuning a pre-trained LLM for sequence generation (BART [Lewis et al.(2020a)] in our case). The concatenation of the conversation history  $\langle Q_1 A_1 \dots Q_{t-1} A_{t-1} \rangle$ , the current question  $Q_t$ , and a special reformulation category tag (“rc1”, “rc2”, ... “rc15”: for each of our 15 categories,  $\langle RC^i \rangle$ ) constitute the input, and the category-specific reformulation is the output.

**Noisy data for fine-tuning.** We generate fine-tuning data for the BART model through *distant supervision*. This is a noisy process, but the alternative of strong supervision would entail the use of

**Algorithm 2:** Deep Q-Learning in RCS model

---

**Input:** Sequence of conversational questions  $\langle Q \rangle$ , step size ( $\alpha > 0$ ), discount factor ( $\gamma > 0$ ), Boltzmann temperature ( $\tau > 0$ ), size of update (*batchSize*), initial DQN parameters  $\theta$

**Output:** Updated DQN parameters  $\theta$

```

1 experience  $\leftarrow \langle \rangle$  ► Initialize experience queue
2 foreach  $Q_t \in \langle Q \rangle$  do
3    $s \leftarrow \Phi(Q_t)$  ► Encode question
4    $Q_\theta(s, \mathcal{A}) = M(s, \mathcal{A}) \circ (\mathbf{W}_2 \times \text{ReLU}(\mathbf{W}_1 \times s))$  ► Get Q-values for actions in s
5    $a^{sample} \sim \text{Prob}(\mathcal{A}, \tau)$  ► Use Eq. 4.3
6    $Q_t^{a^{sample}} \leftarrow \text{RG}(\langle Q_1 \dots A_{t-1} Q_t RC_t^{a^{sample}} \rangle)$  ► Invoke RG
7    $s' \leftarrow \Phi(Q_t^{a^{sample}})$  ► Encode reformulation
8    $r \leftarrow \mathcal{R}(\text{ConvQA}(Q_t), a_{sample}, \text{ConvQA}(Q_t^{a^{sample}}))$  ► Invoke ConvQA for reward
9   experience.enqueue( $s, a^{sample}, s', r$ ) ► Store experience
10  if  $|experience| \geq batchSize$  then
11    ► Retrieve batchSize elements from buffer
12    batch  $\leftarrow experience.dequeue(batchSize)$ 
13     $q^{batch} \leftarrow \langle \rangle$   $q^{targets} \leftarrow \langle \rangle$ 
14    foreach  $(s, a, s', r) \in batch$  do
15       $q^{batch}.enqueue(Q_\theta(s, a))$ 
16       $q^{targets}.enqueue(r + \gamma \cdot \max_{a'} Q_\theta(s', a'))$  ► Calculate and store TD difference
17    end
18     $\theta \leftarrow \theta - \alpha \nabla \frac{1}{batchSize} \cdot \sum_{i=0}^{batchSize} (q_i^{targets} - q_i^{batch})^2$  ► Do gradient descent
19 end
20 return  $\theta$ 

```

---

human-generated reformulations. This would be expensive to obtain at scale. Also benchmarks, like the previously presented CONVREF benchmark [Kaiser et al.(2021)] (see section 3.6), lack category labels. Further, given a conversational question, an average crowdworker is not likely to be able to come up with several diverse and distinct reformulations for each category.

Specifically, we adopt the following strategy. First, we need to find mentions of entities, types, and predicates in the question. We use TAGME [Ferragina and Scaiella(2010)] for entity mention detection and the Wikidata Search API<sup>1</sup> for linking them to the KG. The CLOCQ API [Christmann et al.(2022a)] is used to obtain type and alias information. We use aliases in curated KGs for synonyms of entities and predicates, a rich and precise, yet relatively under-explored resource. We annotate entity type mentions by searching the question before and after any entity mention in the case of question types, and after question words in the case of answer types. To obtain predicate annotations, we extract KG paths connecting the linked question entities and answer entities, or between question entities (e.g. in case of yes/no questions) using the CLOCQ API. The similarity of each path with respect to the question is scored with SBERT [Reimers and Gurevych(2019)] and the predicate on the top-scoring path is assumed to be the intended predicate. We only keep the top 100 predicates found this way to reduce noise. Predicate mentions (relations) are extracted by searching for verbs, the predicate label in the KG, or its alias(es) in the question. If not found,

<sup>1</sup>[https://www.mediawiki.org/wiki/API:Main\\_page](https://www.mediawiki.org/wiki/API:Main_page)

we remove mentions of the entities and types, and question words from the question, and treat the remainder as the predicate mention.

Once we have mentions and their disambiguations, we can apply our transformations from the taxonomy (Figure 4.3) on input questions. Figure 4.4 shows the effects of these transformations.

For deletion, the mention is simply removed from the question token sequence. Two special conditions are enforced in order to keep the conversational question intent-preserving: i) We only delete context entities  $e_{t-1}^{ctx}$  (coming from the conversation history) and not newly introduced entities from the current turn  $t$  (deleting these would potentially make the question under-specified). For example, in *Who played Isildur in LOTR?*, we can delete the context entity “LOTR”, already mentioned in the history, but not the newly introduced entity “Isildur”. ii) relations are only deleted if they also appeared in the previous question. For example, we can only change *Who played the lead role in Oppenheimer?*  $\mapsto$  *What about Oppenheimer?* if the previous question was *Who played the lead role in Barbie?*

For substitution, the main decision to make is the source of alternative surface forms for the linked KG items. Substitution happens in-place: the source mention is replaced by the target mention from the KG alias list in its corresponding position in the question. Each unique alias results in a unique transformation possibility. Pronoun replacements for human entities are performed by looking up their gender in the KG.

For insertions, the main concern is the position of insertion in the question: (i) mentions of answer types and relations are inserted just after the *wh*- question word; (ii) mentions of entity types are inserted just before the respective entity; and (iii) entity mentions are inserted at the end of the question (for simplicity).

The strategy above entails that some categories will have an extremely large number of training cases (like substitutions, as we often have 5+ synonyms for an entity or predicate) while others would have relatively lower volumes (like relation deletions). To increase the number of data points for sparse categories, we adopt a *back-and-forth strategy*: it is easy to add an entity mention to a question when none are detected (like *Directed by...?*  $\mapsto$  *LOTR part 1 directed by?*), and then the *reverse* of this rule would give us a sample of entity deletion. Finally, to remove class imbalance during BART fine-tuning, we sample an equal number of data points for each category.

#### 4.4.2 RG Model Usage

The BART model is fine-tuned on distantly supervised data generated with the ConvQA dev set. At RG inference time, it is applied on the train set where a question and a category from the RCS are already available.

## 4.5 Conversational Question Answering

### 4.5.1 ConvQA Model Training

A ConvQA model is trained on sequences of  $\langle Q_t, A_t^{gold} \rangle$  pairs. In the original training mode, QA pairs are directly used from the benchmark train sets. This original or initial QA model is used to collect rewards for the RCS in one pass over the dev set (as mentioned earlier, the QA dev set is used to train the RCS model). After the trained RCS and RG models select and generate the reformulations

### Completions

$$\begin{aligned}
\langle q_1 \dots q_n \rangle &\xrightarrow{\text{INS ent}} \langle q_1 \dots q_n \ e_{t-1}^{cxt} \rangle \\
\langle q_1 \dots q_n \rangle &\xrightarrow[\text{mention}(p) \in Q_{t-1}]{\text{INS rel}} \langle q_1 \dots \text{mention}(p) \dots q_{n+1} \rangle \\
\langle q_1 \dots \text{mention}(e_t^{cxt}) \dots q_n \rangle &\xrightarrow{\text{INS ent-type}} \langle q_1 \dots \text{type}(e_t^{cxt}) \ \text{mention}(e_t) \dots q_{n+1} \rangle \\
\langle q_1 \dots q_n \rangle &\xrightarrow{\text{INS ans-type}} \langle q_1 \ \text{type}(A) \dots q_{n+1} \rangle
\end{aligned}$$

### Ellipses

$$\begin{aligned}
\langle q_1 \dots \text{mention}(e_{t-1}^{cxt}) \dots q_n \rangle &\xrightarrow{\text{DEL ent}} \langle q_1 \dots q_{n-1} \rangle \\
\langle q_1 \dots \text{mention}(p) \dots q_n \rangle &\xrightarrow[\text{mention}(p) \in Q_{t-1}]{\text{DEL rel}} \langle q_1 \dots q_{n-1} \rangle \\
\langle q_1 \dots \text{mention}(\text{type}(e_t^{cxt})) \dots q_n \rangle &\xrightarrow{\text{DEL ent-type}} \langle q_1 \dots q_{n-1} \rangle \\
\langle q_1 \ \text{type}(A) \dots q_n \rangle &\xrightarrow{\text{DEL ans-type}} \langle q_1 \dots q_{n-1} \rangle
\end{aligned}$$

### Paraphrases

$$\begin{aligned}
\langle q_1 \dots \text{mention}(p) \dots q_n \rangle &\xrightarrow{\text{SUBS rel}} \langle q_1 \dots \text{mention}(\text{alias}(p)) \dots q_n \rangle \\
\langle q_1 \dots \text{mention}(\text{type}(e_t^{cxt})) \dots q_n \rangle &\xrightarrow{\text{SUBS ent-type}} \langle q_1 \dots \text{alias}(\text{type}(e_t^{cxt})) \dots q_n \rangle \\
\langle q_1 \dots \text{mention}(\text{type}(A)) \dots q_n \rangle &\xrightarrow{\text{SUBS ans-type}} \langle q_1 \dots \text{alias}(\text{type}(A)) \dots q_n \rangle
\end{aligned}$$

### Entity coreferences

$$\begin{aligned}
\langle q_1 \dots \text{mention}(e_t^{cxt}) \dots q_n \rangle &\xrightarrow{\text{SUBS ent w/ alias}} \langle q_1 \dots \text{alias}(e_t^{cxt}) \dots q_n \rangle \\
\langle q_1 \dots \text{mention}(e_{t-1}^{cxt}) \dots q_n \rangle &\xrightarrow{\text{SUBS ent w/ ent-type}} \langle q_1 \dots \text{type}(e_{t-1}^{cxt}) \dots q_n \rangle \\
\langle q_1 \dots \text{mention}(e_{t-1}^{cxt}) \dots q_n \rangle &\xrightarrow{\text{SUBS ent w/ pronoun}} \langle q_1 \dots \text{pronoun}(e_{t-1}^{cxt}) \dots q_n \rangle
\end{aligned}$$

### Retain original question

$$\langle q_1 \dots q_n \rangle \xrightarrow{\text{RETAIN whole}} \langle q_1 \dots q_n \rangle$$

Figure 4.4: Transformations applied to questions annotated in the CONVMIX benchmark.

for each training question  $\langle Q_t^i \rangle_k$ , these reformulations are paired with the corresponding gold answer  $A_t^{gold}$  of the original training question  $Q_t$ . These new  $\langle Q_t^i, A_t^{gold} \rangle$  pairs are added to the benchmark,

and now the ConvQA model is trained on this augmented resource. This model is expected to be more robust than the original model (original and robust models are marked ConvQA<sub>orig</sub> and ConvQA<sub>robust</sub> in Figure 4.2, respectively).

### 4.5.2 ConvQA Model Usage

The trained ConvQA model is directly applied to the questions in test sets at *answering time* to produce ranked lists of entities. The RCS and RG models are only applied for improving the ConvQA models training but are not used at answering time.

## 4.6 Experimental Setup

**Benchmarks.** We use two ConvQA benchmarks: CONVMIX [Christmann et al.(2022b)] (more recent) and CONVQUESTIONS [Christmann et al.(2019)] (more popular). These contain realistic questions from crowdworkers. Conversations in CONVQUESTIONS are generated by permuting questions from a seed set of 700 conversations: we used only the train set for this seed (420 conversations) for training ConvQA models, to decouple the effect of data augmentation inherent in the CONVQUESTIONS benchmark. Data statistics about these benchmarks can be found in Table 4.2.

For robust evaluation, we created large test sets in the following way. We obtained 20 reformulations from ChatGPT (gpt-3.5-turbo model) for each test question in these benchmarks, with ten each from two different settings: (i) one asked for reformulations with access to the full conversation history (previous questions and gold answers), and (ii) the other only with the current question. Examples are shown in Table 4.8, at the end of this chapter. All GPT-generated reformulations are available at our website <https://reign.mpi-inf.mpg.de>. We did not find any duplicates in the ChatGPT reformulations. For prompting GPT, we tried a few alternatives. We saw that few-shot examples did not have a noticeable effect on the generations. Thus, we used the following zero-shot prompt (the ‘History’ line is omitted for generating the variants without history) and set the temperature value to zero, for obtaining deterministic behavior (as far as possible):

*Reformulate the ‘Question’ 10 times in a short, informal way. Assume third person singular if not obvious from the question.*  
 ‘History’: {Conversation history}  
 ‘Question’: {Question}  
 ‘Reformulation’:

The second sentence was used to avoid generations like *Your place of birth?* instead of the correct *His ...?* or *Her ...?*.

**Baselines.** ConvQA models belong to two families, one based on *history modeling*, and the other on *question completion* (see Section 2.1.2 in the background chapter). We choose one open-source system from each family for KG-QA: we use our previously presented model from Chapter 3, CONQUER [Kaiser et al.(2021)] (history modeling with context entities, with RL) and EXPLAIGNN [Christmann et al.(2023)] (completion to an intent-explicit structured representation, with Graph Neural Networks). EXPLAIGNN was built for heterogeneous sources, and we use the KG-only model, in line with our setting. Default configurations were used for both systems.

Benchmark	Train	Dev	Test	GPT-Test
CONVMIX [Christmann et al.(2022b)]	8.4k (1680)	2.8k (560)	4.8k (760)	100.8k (760)
CONVQUESTIONS [Christmann et al.(2019)]	33.6k (6720)	11.2k (2240)	11.2k (2240)	235.2k (2240)

Table 4.2: Benchmark sizes as #questions (#conversations). Reformulations are also counted as individual questions to be answered. Questions for the GPT-Test sets subsume the original test questions.

**Metrics.** All methods produce ranked lists of entities with binary relevance. We thus use the following three metrics: Precision@1 (P@1), Mean Reciprocal Rank (MRR), and whether a correct answer is in the top-5 (Hit@5). We also define a new metric *Robust*, that computes, for each question, the number of reformulations correctly answerable by a ConvQA model, averaged over the number of test intents. The *Robust* measure lies between 0 and the number of reformulations per question including the original formulation (hence 21 in our case). The higher this value, the more robust the model. Statistical significance (\*) is conducted via McNemar’s test better suited for binary variables (P@1 and Hit@5) than t-test (real-valued), and 2-tailed paired *t*-test otherwise (MRR, Robust), with  $p < 0.05$ .

**Initializing REIGN.** We use Wikidata as our KG: all models use the dump from 31-01-2022. We use BART (<https://bit.ly/3N9WPVj>, for the RG model), and BERT (<https://bit.ly/3NkKRsd>, for state encoding in the RCS model) implementations from Hugging Face. As history input to BART, we use only the first and the previous turn of the conversation [Qu et al.(2019a), Vakulenko et al.(2021)]. Hyperparameters for the Deep Q-Network in the RCS were tuned on the CONVMIX dev set:  $d = 768$ , hidden size  $h = 128$ , Boltzmann temperature  $\tau = 0.3$ , discount factor  $\gamma = 1.0$  (no decay for future rewards), learning rate =  $10^{-5}$ , batch size = 10, and epochs = 5. The RG was trained for 3 epochs, with learning rate =  $3 \times 10^{-5}$ , and batch size = 10. Two thousand examples from each reformulation category were used for fine-tuning BART. Both RCS and RG models are only trained on CONVMIX and applied *zero-shot* to CONVQUESTIONS. Five reformulation categories were selected by RCS for every question ( $k = 5$ ). A single GPU (NVIDIA Quadro RTX 8000, 48 GB GDDR6) was used to train and evaluate all models. The TensorFlow Agents library (<https://www.tensorflow.org/agents>) is used for the RL components.

## 4.7 Results and Insights

### 4.7.1 Key Findings

**REIGN results in robust training.** The four methods CONQUER, CONQUER + REIGN (CONQUER coupled with REIGN), EXPLAIGNN, and EXPLAIGNN + REIGN (EXPLAIGNN coupled with REIGN) are evaluated on the two benchmarks CONVMIX (Table 4.3) and CONVQUESTIONS (Table 4.4). A clear observation is that methods interfaced with REIGN systematically outperform the original ConvQA models, on all test sets and metrics. While numbers are reported on the original test sets for completeness, results become much more significant on the GPT-test sets, with  $p$ -values of the order of  $10^{-80}$  (recall that these values are averaged over  $\simeq 100k$ - $200k$  cases, Table 4.2). Importantly, versions

Benchmark →	CONVMIX Test			GPT-CONVMIX Test			
Method ↓	P@1	MRR	Hit@5	P@1	MRR	Hit@5	Robust
CONQUER	0.218	0.272	0.337	0.173	0.224	0.287	6.531
CONQUER + REIGN	<b>0.245*</b>	<b>0.292*</b>	<b>0.346*</b>	<b>0.190*</b>	<b>0.236*</b>	<b>0.289*</b>	<b>7.035*</b>
EXPLAIGNN	0.370	0.438	0.526	0.278	0.346	0.433	10.983
EXPLAIGNN + REIGN	<b>0.384*</b>	<b>0.446*</b>	<b>0.531</b>	<b>0.295*</b>	<b>0.361*</b>	<b>0.448*</b>	<b>11.130*</b>

Table 4.3: Main results comparing REIGN-enhanced ConvQA models with their standalone versions on CONVMIX and the GPT-augmented test set, which is 20x larger. The higher value per column per QA model is in bold.

Benchmark →	CONVQUESTIONS Test			GPT-CONVQUESTIONS Test			
Method ↓	P@1	MRR	Hit@5	P@1	MRR	Hit@5	Robust
CONQUER	0.236	0.287	0.360	0.197	0.245	0.304	6.447
CONQUER + REIGN	<b>0.238</b>	<b>0.290*</b>	<b>0.371*</b>	<b>0.202*</b>	<b>0.252*</b>	<b>0.310*</b>	<b>7.224*</b>
EXPLAIGNN	0.271	0.355	0.466	0.219	0.290	0.382	8.400
EXPLAIGNN + REIGN	<b>0.318*</b>	<b>0.411*</b>	<b>0.529*</b>	<b>0.226*</b>	<b>0.302*</b>	<b>0.402*</b>	<b>8.925*</b>

Table 4.4: Main results where REIGN is applied zero-shot on CONVQUESTIONS (and its GPT-augmented version of the test set). The higher value per column per QA model is in bold.

with REIGN score systematically higher on the *Robust* metric, showing that the improved models are capable of handling more lexical and syntactic variations on average (differences higher for larger GPT-sets). ConvQA with these benchmarks and GPT reformulations are challenging: the values are far less than 21 (see Section 4.6 for bounds of the robustness metric). We also computed the number of unique *intents* that *newly become answerable* ( $P@1 = 1$  for at least one question or one of its reformulations) with REIGN: this is 115 (CONVMIX-GPT-set) and 407 (CONVQUESTIONS-GPT-set) for CONQUER and 33 (CONVMIX-GPT-set) and 284 (CONVQUESTIONS-GPT-set) for EXPLAIGNN, showing that our robust training can put more unique information needs within reach of the ConvQA model. Reformulations generated by REIGN are shown in Table 4.9, at the end of this chapter. On average, original questions, REIGN, and GPT-reformulations, are 5.9, 7.5, and 7.2 words long.

**REIGN components are generalizable.** Results on the CONVQUESTIONS benchmark showcase successful zero-shot application of REIGN modules. Given that the CONVQUESTIONS test sets are much larger than the ones for CONVMIX (see Table 4.2), improved results over the original QA modules show that our RCS and RG modules, individually, are immune to idiosyncrasies in specific datasets.

**Benefits of REIGN hold over domains and turns.** We report drill-down results over five domains and individual conversation turns in Tables 4.5 and 4.6. We show that the benefits provided by reinforced reformulation generation are not limited to specific domains, or shallow conversation turns only.

Method ↓ / Domain →	Books	Movies	Music	TV series	Soccer
CONQUER	0.227	0.175	0.159	0.141	0.163
CONQUER + REIGN	<b>0.239*</b>	<b>0.200*</b>	<b>0.167*</b>	<b>0.160*</b>	<b>0.184*</b>
EXPLAIGNN	0.298	<b>0.287*</b>	0.265	0.274	0.265
EXPLAIGNN + REIGN	<b>0.333*</b>	0.283	<b>0.301*</b>	<b>0.281*</b>	<b>0.275*</b>

Table 4.5: Domain-wise P@1 results on GPT-CONVMIX testset.

Method ↓ / Turn →	1	2	3	4	5	6-10
CONQUER	0.205	0.193	0.177	0.184	0.160	0.133
CONQUER + REIGN	<b>0.210*</b>	<b>0.214*</b>	<b>0.194*</b>	<b>0.204*</b>	<b>0.184*</b>	<b>0.147*</b>
EXPLAIGNN	0.333	0.297	0.286	0.292	0.277	0.205
EXPLAIGNN + REIGN	<b>0.350*</b>	<b>0.318*</b>	<b>0.311*</b>	<b>0.305*</b>	<b>0.291*</b>	<b>0.216*</b>

Table 4.6: Turn-wise P@1 results on GPT-CONVMIX testset.

### 4.7.2 In-depth Analysis

In Table 4.7, we report in-depth analyses of the moving parts in REIGN, using CONQUER on the CONVMIX-GPT-set. Trends with EXPLAIGNN and CONVQUESTIONS are similar. We do not use this table for making design choices – rather, we expose large-scale effects of sub-optimal configurations: hence the choice of a  $\simeq 100\text{k}$ -GPT set instead of the typically small dev set.

**RCS with DQN is vital.** First and foremost, we show that selecting reformulations with our DQN is necessary, and simply taking *all* noisy reformulations does not serve as a sledgehammer for performance improvement even at three times the number of data points used (Row 1 vs. Row 4). This makes a solid case for judicious augmentation. Using all reformulations does lead to higher answer recall as seen through the Hit@5 value, but at the cost of precise ranking. Using top-5 reformulations is a sweet spot for deploying the RCS (Row 1 vs. 2 and 3). Using higher numbers drastically increases the training time and often produces degenerate reformulations. Contrast against a *random* choice of categories inside the RCS is a natural experiment, and we find this to be sub-optimal (see P@1 in Row 5). Another stronger baseline is to *sample*  $k = 5$  categories according to the Q-value distribution: this again falls short of a top- $k$  prediction (Row 6).

**The whole taxonomy matters.** It may appear that using only insertion or substitution operations from the taxonomy may suffice for robust learning, but we find that considering all categories jointly (Row 1) is superior to using only individual “meta”-categories (INS, DEL SUBS in Rows 7 – 9). Figure 4.5 shows the union of the top-5 frequent predictions from our RCS DQN for the two benchmarks. Insertion of question entity types and expected answer types are generally useful for disambiguation, and substituting relations with aliases naturally makes the system more robust to predicate paraphrasing. The original question was retained 10 – 20% of the time.

**Question rewriting is not enough.** As discussed in Section 4.1, reformulating a conversational question into a more complete form at *answering time* is a prevalent approach in ConvQA. As such, comparison with such rewriting or completion approaches is out of scope, as we focus on more ro-

Row	Configuration	P@1	MRR	Hit@5	#Data points
1	RCS (DQN, top-5) + RG (BART) [Full]	<b>0.190</b>	<b>0.236</b>	0.289	43.6k
2	RCS (DQN, top-3) + RG (BART)	0.184	0.231	0.288	30.5k
3	RCS (DQN, top-1) + RG (BART)	0.178	0.228	0.288	15.9k
4	No RCS (All cats) + RG (BART)	0.188	0.234	0.292	126k
5	No RCS (Random cats) + RG (BART)	0.182	0.232	<b>0.293</b>	42k
6	No RCS (Sample cats) + RG (BART)	0.185	0.231	0.287	41.9k
7	No RCS (INS part) + RG (BART)	0.183	0.230	0.288	42k
8	No RCS (DEL part) + RG (BART)	0.172	0.218	0.273	42k
9	No RCS (SUBS part) + RG (BART)	0.183	0.228	0.282	58.8k
10	No RCS + No RG (Question completion)	0.175	0.224	0.284	15.1k
11	No RCS + No RG (Question rewriting)	0.180	0.230	0.291	15.1k

Table 4.7: Large-scale effects of design choices in REIGN (with CONQUER on GPT-CONVMIX).

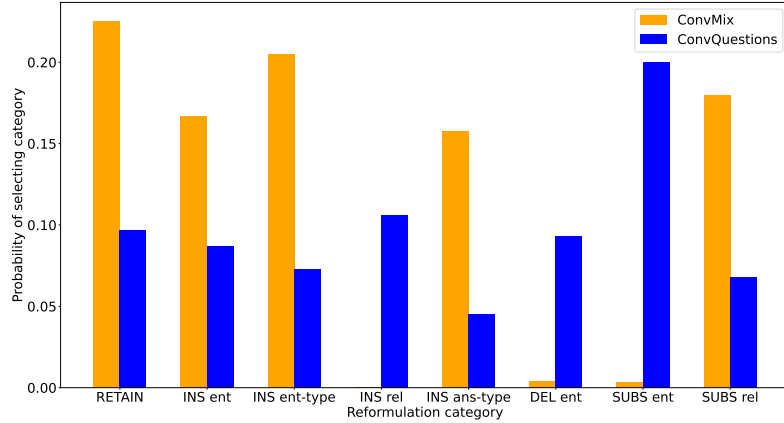


Figure 4.5: Common category predictions by the RCS DQN.

but *training*. Nevertheless, we explore the natural possibility of using completed forms of questions during training, as opposed to a set of noisy reformulations. The CONVMIX benchmark contains intent-explicit questions by the original crowdworkers who generated the conversations, and can thus be treated as gold standard completions. Additionally, we use T5 as in [Lin et al.(2021)] for question rewriting. In both cases, the rewritten questions are directly used to train the respective QA model (without the REIGN-specific RCS and RG components). We found that this falls short of our proposed method (Row 1 vs. Row 10 and Row 11). Interestingly, corroborating findings from the BART experiments, noisy rewrites with T5 outperform human completions. Note that completion or rewriting entails *one* longer version of the question (hence  $\simeq$  15k data points): we find that generating a *small set* of potentially incomplete variants with REIGN improves performance.

**Intrinsic rewards also work well.** Our DQN uses differences in reciprocal ranks, computed from gold answers in benchmarks, as *extrinsic* rewards. A natural question is what happens in cases where such relevance assessments are not available. We thus explored an alternative of an *intrinsic* reward [Liu et al.(2022), Chen et al.(2022b)] computed as the *difference* in the ConvQA model’s prob-

*abilities* of its top-1 answer for the reformulation and the original question [Liu et al.(2022), Chen et al.(2022b)] (this is analogous to blind relevance feedback). On a positive note, this resulted in comparable performance on the CONVMIX dev set (0.270 P@1 for extrinsic vs. 0.269 for intrinsic).

**Manual error analysis.** The authors analyzed 10 reformulations from each category for both BART reformulations and the original fine-tuning data ( $15 \times 10 \times 2 = 300$  in all), to look for potential issues. There were only minor problems detected for both scenarios. The concerns with BART were as follows: unintelligible intent (4 cases), hallucinations (5), wrong category applied (13), information removed unintentionally (15), transformation possible but not made (13), unsuitable entity or type added (4), and information already in the question was added again (5). The concerns with the initial noisy data can sometimes be traced back to incorrect processing of the benchmarks (Section 4.4.1), like wrong predicate (4 cases) and addition of information already present due to incorrect markup (4). Other errors include: intent changed (6), unsuitable types inserted (7 cases), and changes possible but not made (2).

**GPT cannot replace the REIGN pipeline.** It is a common trend nowadays to use LLMs like GPT at multiple points in pipelines. We thus checked whether the same ChatGPT model that generated our test set could actually replace the whole REIGN pipeline by directly generating reformulations for training questions and feeding them into the ConvQA models for training. Importantly, we found that this underperforms REIGN, considering five reformulations for each alternative on the original CONVMIX dev set (evaluation of GPT reformulations on GPT test sets could result in undesirable biases): 0.270 P@1 for REIGN vs. 0.261 for GPT (CONQUER), and 0.423 for REIGN vs. 0.405 for GPT (EXPLAIGNN). Note that the GPT reformulations are *model-agnostic*: this shows that reformulations generated with *model-aware performance feedback* is indeed a better choice for robust training.

## 4.8 Related Work

**Conversational question answering.** Section 2.1 has already provided an overview of related work on ConvQA. REIGN is not a QA model by itself, but can improve the performance of any given ConvQA system: we demonstrate this by choosing one method from each family of approaches (history modeling and question completion) in our experiments [Kaiser et al.(2021), Christmann et al.(2023)].

**Robustness in QA.** Improving the robustness or generalizability of ConvQA models has not been explored in-depth: work has mostly been limited to specific benchmarks of choice [Saha et al.(2018), Guo et al.(2018), Shen et al.(2019), Christmann et al.(2019), Christmann et al.(2022b)]. Implicitly, authors have tried to prove robust behavior by the use of multiple benchmarks [Kacupaj et al.(2022), Marion et al.(2021), Lan and Jiang(2021)], or zero-shot application of models to new benchmarks [Christmann et al.(2023)]. *Data augmentation*, given one or more benchmarks, is one of the prominent approaches for increasing model robustness in QA [Bartolo et al.(2021), Puri et al.(2020), Liu et al.(2020), Sachan and Xing(2018), Baheti et al.(2020), Shen et al.(2022), Yang et al.(2019)]. Our work stands out as model-specific data augmentation, a philosophy for effective training by trying to fill “gaps” in a specific model’s learned behavior, instead of feeding in a very

large volume of noisy data to all models. Some recent works in QA over text investigate model robustness by perturbing input passages [Gekhman et al.(2023), Neeman et al.(2023)], while we tap into question reformulations as a perturbation on the question-side.

**Reformulations in search and QA.** Work on question or query reformulations in search [Li et al.(2022b), Xue et al.(2012), Ponnusamy et al.(2020), Nogueira and Cho(2017)], QA [Tomuro(2003), Liu and Belkin(2008), Hermjakob et al.(2002), Buck et al.(2018), Kaiser et al.(2021)], and recommenders [Zhang et al.(2022)], can be broadly positioned in a  $2 \times 2$  space according to the definition of a reformulation:

- rephrasing (of the same intent) [Kaiser et al.(2021), Dong et al.(2017), Fader et al.(2013), Tomuro and Lytinen(2001)] *versus* refinement (into a variation of the previous intent) [Liu and Belkin(2008), Zhang et al.(2022), Nogueira and Cho(2017), Liu et al.(2019b), Kiesel et al.(2021)]; and
- using for better training [Li et al.(2022b), Ponnusamy et al.(2020)] *versus* using for better inference [Buck et al.(2018), Xue et al.(2012), Nogueira and Cho(2017), Das et al.(2019)].

This work falls into the *rephrasing-for-training* quadrant, viewing reformulations as rephrased user utterances for the current question in a conversation, and leveraging these for training a more robust model. Early work on automatic acquisition of query reformulation patterns [Xue et al.(2012), Tomuro(2003), Liu and Belkin(2008), Tomuro and Lytinen(2001)], or on *paraphrasing* for improving model robustness [Abujabal et al.(2018), Fader et al.(2014), Fader et al.(2013), Dong et al.(2017), Berant and Liang(2014), Abujabal et al.(2019), Gan and Ng(2019)], did not account for answers from previous turns, and more generally, did not address the specific difficulty of incomplete and ad-hoc user utterances in conversations.

## 4.9 Conclusion

This chapter presented a method to make conversational question answering models more robust with generated reformulations guided towards better QA performance. A reformulation taxonomy for conversational questions was proposed that breaks down questions into discrete parts, inspired by KG components, such as entities, predicates and types. This way, questions can be modified systematically. REIGN can be perceived as a model-specific data augmentation: The proposed RL framework judiciously learns to pick the most suitable reformulation variants for enhanced training of the underlying ConvQA model, as opposed to brute-force data augmentation with all possible reformulations. Experiments with two state-of-the-art ConvQA methods demonstrate the benefits of the REIGN: ConvQA models trained with reformulations coming from REIGN achieve higher answering performance and are more robust to different question variations. Also, REIGN components are reusable and can be applied without re-training on new datasets.

---

<b>[Books] History:</b>	<i>How many Pulitzer Prizes has John Updike won? 2.</i>
<b>Question:</b>	<i>Which was the first book to win him the award?</i>
<b>Reformulation 1:</b>	<i>What book earned John Updike his first Pulitzer Prize?</i>
<b>Reformulation 2:</b>	<i>What was the author's first book to win a Pulitzer?</i>
<b>Reformulation 3:</b>	<i>Title of John Updike's first Pulitzer Prize-winning book?</i>

---

<b>[Movies] History:</b>	<i>Which year was the Hobbit An unexpected journey released? 2012.</i>
<b>Question:</b>	<i>What is the book based on?</i>
<b>Reformulation 1:</b>	<i>What's the book about?</i>
<b>Reformulation 2:</b>	<i>What's the book's topic?</i>
<b>Reformulation 3:</b>	<i>What's the book's subject?</i>

---

<b>[Music] History:</b>	<i>Which singer sang the number Single Ladies? Beyonce. What is the year of its release? 2008. Who is her spouse? Jay-Z . What is his date of birth? 4 December 1969.</i>
<b>Question:</b>	<i>Was Kanye West a composer of the song?</i>
<b>Reformulation 1:</b>	<i>Did Kanye West contribute to the lyrics of the song?</i>
<b>Reformulation 2:</b>	<i>Did Kanye West perform the song with Beyonce?</i>
<b>Reformulation 3:</b>	<i>Was Kanye West featured in the song?</i>

---

<b>[TV series] History:</b>	<i>What is the release year of the TV series See? 2019.</i>
<b>Question:</b>	<i>created by?</i>
<b>Reformulation 1:</b>	<i>Who's responsible for it?</i>
<b>Reformulation 2:</b>	<i>Who's the mastermind?</i>
<b>Reformulation 3:</b>	<i>Who's the author?</i>

---

<b>[Soccer] History:</b>	<i>Pele scored how many goals in international play? 77. Has he scored the most goals? No.</i>
<b>Question:</b>	<i>Did Messi beat his goal total?</i>
<b>Reformulation 1:</b>	<i>Did Messi surpass Pele's international goal record?</i>
<b>Reformulation 2:</b>	<i>Has Messi scored more international goals than Pele?</i>
<b>Reformulation 3:</b>	<i>Did Messi break Pele's goal-scoring record?</i>

---

Table 4.8: Examples of GPT reformulations for test set questions.

---

**[Books] History:** *Which book won the 2017 Pulitzer Prize for Fiction? The Underground Railroad. subject of the book? Slavery in the United States. publisher of the novel? Doubleday.*

**Question:** *author of the fiction?*

---

**Reformulation 1:** *creator of the fiction?* [SUBS rel]

**Reformulation 2:** *Which individual is author of the fiction?* [INS ans-type]

**Reformulation 3:** *author of the fiction The Underground Railroad?* [INS ent]

---

**[Movies] History:** *Who was the director of The Lord of the Rings? Peter Jackson. Who played Frodo Baggins?*

**Question:** *Who played Frodo Baggins?*

---

**Reformulation 1:** *Who Frodo Baggins?* [DEL rel]

**Reformulation 2:** *Who portrayed Frodo Baggins?* [SUBS rel]

**Reformulation 3:** *Who played Frodo Baggins in it?* [SUBS pronoun]

---

**[Music] History:** -

**Question:** *Formation year of the band U2?*

---

**Reformulation 1:** *Formation year of the rock band U2?* [SUBS ent-type]

**Reformulation 2:** *Which year is Formation year of the band U2?* [INS ans-type]

**Reformulation 3:** *Formation year of U2?* [DEL ent-type]

---

**[TV series] History:** *Who played as Marty in Ozark series? Jason Bateman. and Wendy Byrde? Laura Linney. who is the director of the series? Jason Bateman. How many episodes are in the series? 30.*

**Question:** *production company of the series?*

---

**Reformulation 1:** *production company of the series **television series?*** [INS ent-type]

**Reformulation 2:** *production company of the series Ozark?* [INS ent]

**Reformulation 3:** *production house of the series?* [SUBS rel]

---

**[Soccer] History:** *What is the full name of footballer Neymar? Neymar da Silva Santos Junior. Birthplace of Neymar? Brazil . When was he born? 5 February 1992.*

**Question:** *Which club does he play now?*

---

**Reformulation 1:** *Which club does he play now association football player?* [INS ent-type]

**Reformulation 2:** *Which club does he play now Neymar?* [INS ent]

**Reformulation 3:** *Which Football team does he play now?* [SUBS ans-type]

---

Table 4.9: Examples of REIGN-generated reformulations along with their respective reformulation categories, used for training (one noisy transformation is highlighted in red).



# Chapter 5

## Learning From Preferences based on Delayed System Feedback

### Contents

---

<b>5.1 Learning from Preferences for Conversational QA . . . . .</b>	<b>76</b>
5.1.1 Introduction . . . . .	76
5.1.2 The PRAISE Method . . . . .	77
5.1.3 Experimental Setup . . . . .	82
5.1.4 Results and Insights . . . . .	83
5.1.5 Related Work . . . . .	85
5.1.6 Conclusion . . . . .	86
<b>5.2 Learning from Preferences for Task-oriented Dialogs . . . . .</b>	<b>90</b>
5.2.1 Introduction . . . . .	90
5.2.2 The SUIT Method . . . . .	91
5.2.3 Experimental Setup . . . . .	94
5.2.4 Results and Insights . . . . .	95
5.2.5 Related Work . . . . .	97
5.2.6 Conclusion . . . . .	98

---

In the previous chapter, we improved the conversational context understanding capabilities of ConvQA models by enhancing their training data with a subset of most beneficial question reformulations. We used the QA performance as feedback signal for this selection. In this chapter, we perceive conversational context understanding as one, out of several, subtasks in our conversational pipeline that we are going to improve with final performance feedback. More precisely, we present two methods that learn from preference data for intermediate subtasks using the model’s own generations and task-level supervision signals. The first method, named PRAISE (Section 5.1), is an LLM-based pipeline for ConvQA with retrieval augmented generation (RAG). While the previously presented methods in this thesis (CONQUER and REIGN) focused on ConvQA over KGs, PRAISE operates over both, KG and textual sources. More precisely, in PRAISE, we train LLMs for each of the following subtasks: i) conversational context understanding, ii) retrieval and filtering of relevant

information and iii) final answer generation. As labeled training data for individual subtasks is unavailable in practice, PRAISE learns from its own generations using the final answering performance as feedback signal without human intervention and treats intermediate information, like relevant evidence, as weakly labeled data. We apply Direct Preference Optimization [Rafailov et al.(2023)] by contrasting successful and unsuccessful samples for each subtask. Section 5.1.2 describes in detail how each subtask is handled in PRAISE. The experiments in Section 5.1.4 show the effectiveness of this training paradigm: PRAISE shows improvements per subtask and achieves new state-of-the-art performance on a popular ConvQA benchmark.

The second method that will be presented, named SUIT (Section 5.2), is an LLM-based end-to-end task-oriented dialog model. Task-oriented dialog (ToD) systems also have to solve multiple subgoals to accomplish user goals, with feedback often obtained solely at the end of the dialog. Section 5.2.2 provides the details of our method SUIT, which is trained iteratively. Dialogs are sampled from the current model and subgoals that contribute to dialog success are determined using distant supervision and counterfactual learning to obtain high quality training samples for the next iteration. In our experiments in Section 5.2.4, we show how this data improves supervised fine-tuning and preference learning results over multiple iterations. SUIT reaches new state-of-the-art performance on a popular ToD benchmark.

## 5.1 Learning from Preferences for Conversational QA

### 5.1.1 Introduction

**Motivation.** An example conversation is given below:

*Q*<sub>1</sub>: *By which band was The Wall recorded?*  
*A*<sub>1</sub>: *Pink Floyd*  
*Q*<sub>2</sub>: *Where was the band formed?*  
*A*<sub>2</sub>: *London*  
*Q*<sub>3</sub>: *Who joined to replace Sid?*  
*A*<sub>3</sub>: ...

The third turn’s question *Q*<sub>3</sub> : *Who joined to replace Sid?* can be answered only by contextualizing it with the first question and its answer *A*<sub>1</sub> : *Pink Floyd*. Large Language models (LLMs) have strong capabilities to contextualize the user’s utterances [Lewis et al.(2020b)], and are a key part of modern ConvQA systems. However, LLMs are prone to hallucinations, giving non-factual answers or missing tangible provenance. To mitigate these risks, LLM-based QA systems are usually coupled with Retrieval Augmented Generation (RAG) [Gao et al.(2023b)], to provide the LLM with retrieved evidence. These considerations lead to a three-stage pipeline: i) question understanding and contextualization, ii) evidence retrieval and filtering/ranking, iii) answer generation from best evidence.

This constitutes a bottleneck in training data. In realistic settings, often only final answers are available as ground truth, but there is no ground truth for question interpretation (like reformulated/completed questions) and no gold-standard evidence for the retrieval stage. To overcome this problem and improve the performance of each subtask in the pipeline, we propose a sampling-based approach: we derive weakly-labeled intermediate training data, by solely observing final answers.

**Limitations of state-of-the-art.** Methods for ConvQA [Zamani et al.(2023)] contextualize questions with cues from prior turns, by reformulating questions, or by augmenting questions with selected pieces from the conversation. All methods are driven by training data (or LLM fine-tuning data). There are two main limitations: i) relying on intermediate supervision signals [Jeong et al.(2023), Mao et al.(2024)] - for evidence retrieval in RAG architectures, there is no cost-efficient way of compiling ground-truth data - and/or ii) relying on human feedback [Mo et al.(2023), Kaiser et al.(2021), Vakulenko et al.(2021)], which is costly to collect and often far from perfect. In contrast, PRAISE learns from final answering feedback in an automatic way.

**Approach.** We introduce PRAISE (Preference-based Learning with Retrieval Augmented Iterative SEquence generation for ConvQA), a pipeline architecture, consisting of question understanding (QU), evidence retrieval and filtering (ERF) and answer generation (AG). Our QU model generates question reformulations so as to benefit the subsequent retrieval and answer generation. The retrieval often yields long lists of potentially relevant evidence: feeding all these into the AG stage would incur high computational costs, if possible at all, given the LLM’s limited context length. Therefore, the ERF stage of PRAISE contains a judiciously designed LLM-based evidence filtering technique, efficiently operating with subsets of evidence pieces. Finally, the AG stage is trained to give faithful answers using tangible evidence containing answer and question cues. We perform parameter-efficient fine-tuning of a Llama-3-8B-Instruct model [Touvron et al.(2023)]. For each subtask, PRAISE samples generations from an initial model and learns from pairing successful and unsuccessful generations using Direct Preference Optimization (DPO) [Rafailov et al.(2023)]. The intuition is that earlier tasks are optimized for later tasks via delayed feedback signals without human intervention.

**Contributions.** This work makes the following contributions:

- The PRAISE architecture for the ConvQA pipeline, where each stage can be optimized by learning with *weakly-labeled intermediate data*;
- A *sampling-based* technique for automatically deriving *preference data* from *delayed* feedback based on final answers;
- Experiments demonstrating the superior performance of PRAISE on a popular ConvQA benchmark.

Our project website is available at <https://praise.mpi-inf.mpg.de>, our code and data is on GitHub: <https://github.com/magkai/PRAISE>.

### 5.1.2 The PRAISE Method

Figure 5.1 gives an overview of the PRAISE architecture, along with an example conversation, and Table 5.1 lists the salient notation. First, the QU step rewrites the question into a self-sufficient form. Next, a large list of potentially relevant evidence is obtained by the retrieval component, which is an external module that is not in focus of this work. The number of evidence pieces are then reduced to the top- $k$  most relevant ones in the evidence filtering step (ERF), before the final answer is generated (AG). The available training data consists of  $\langle \text{question}, \text{answer} \rangle$  pairs; so we

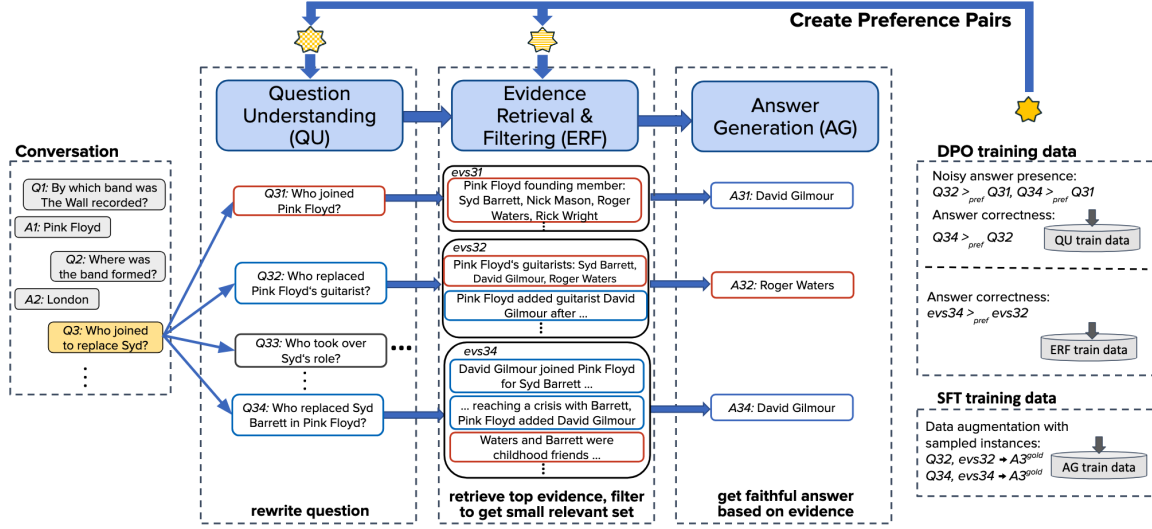


Figure 5.1: Overview and illustration of PRAISE (preferred/correct outputs are in blue, incorrect/uninformative outputs in red).

do not have direct supervision signals for the QU and ERF stages. The three stages are explained in more detail in the following, including how to obtain training data and train an LLM for each stage. Additionally, Algorithm 3 summarizes the individual steps in training the PRAISE pipeline. The prompts used for each model are shown in Figure 5.2 and 5.3 at the end of this section.

### 5.1.2.1 Question Understanding (QU)

This stage transforms the current user question  $Q_i$  into a more complete reformulation  $Q_{ij}$ , by considering question-answer pairs  $\langle Q_\nu, A_\nu \rangle_{\nu \in \{0..i-1\}}$  from the conversation history:

$$Q_{ij} \leftarrow \mathbf{QU}_{\text{PRAISE}}(Q_i, \langle Q_\nu, A_\nu \rangle_{\nu \in \{0..i-1\}}) \quad (5.1)$$

**Initialization.** We use an LLM in few-shot mode ( $QU_{few}$ ), showing it five examples on how to rewrite questions. The model is prompted to generate various surface forms for the same question.

**Training.** For QU training, we create a dataset  $\mathcal{D}_{QU_{sample}}$  based on the initial model’s generations ( $QU_{few}$ ) as follows. We do not rely on any human-rewrites for this task. Our goal is to optimize the questions for retrieval and answering performance. For each original question, we sample a small set  $r$  of different reformulations ( $r = 5$  in our experiments; line 1 in Algorithm 3); each of these is then fed into evidence retrieval and answer generation (line 2 in Algorithm 3). The quality of the generated formulations is determined by the later stages as follows. A sample  $Q_{ij}$  is preferred if it fulfills the constraints:

1.  $Q_{ij}$  results in retrieving evidence that mentions question entities and contains correct answers (answer presence);
2.  $Q_{ij}$  leads to finding the right answer for this question (answer correctness).

Notation	Concept
$Q_i, A_i$	question with intent $i$ , corresponding answer
$A_{im}$	sampled answer $m$ for question $Q_i$
$Q_{ij}$	reformulation $j$ for $Q_i$
$\langle Q_{ij} \rangle_r$	reformulation set of size $r$
$\langle Q_{ij} \rangle^+, \langle Q_{il} \rangle^-$	preferred, dispreferred sets of reformulations
$evs_{il} = \langle eid : etext \rangle_k$	evidence set $l$ for $Q_i$ , with $k$ identifier-text pairs
$\langle eid_{il} \rangle^+, \langle eid_{im} \rangle^-$	preferred, dispreferred sets of evidence ids
$\pi, \pi_{ref}$	current policy, reference policy
$\beta$	hyperparameter, controlling the KL-divergence penalty

Table 5.1: Notation for salient concepts in PRAISE.

Data for fine-tuning the QU model is based on these two constraints. Since we lack gold evidence data, we rely on the following noisy estimate: evidence pieces that contain both question and answer entities are considered as positive samples. All generated question formulations from  $\mathcal{D}_{QUsample}$  that both constraints fulfill are used for supervised fine-tuning (SFT), resulting in model  $QU_{SFT}$  (line 4, Algorithm 3). This model is trained using a standard negative log-likelihood loss:

$$L_{SFT}(\pi_\theta) = -\mathbb{E}_{(x,y) \sim \mathcal{D}} \left[ \sum_{t=0}^T \log \pi_\theta(y_t | y_{t-1..0}, x) \right] \quad (5.2)$$

with  $y$  corresponding to reformulations  $\langle Q_{ij} \rangle^+$  that fulfill both constraints and  $x = x_{QUfew}$  being the prompt, followed by the concatenation of the conversation history and the current question that should be reformulated;  $\mathcal{D} = \mathcal{D}_{QUsample}$  and  $\pi_\theta = QU_{init}$ .

Subsequently, we construct preference pairs for further training with Direct Preference Optimization (line 3 and 5, Algorithm 3), resulting in  $(QU_{DPO})$ :

- (i)  $Q_{ij} >_{pref} Q_{ik}$  if  $Q_{ij}$  fulfills (1) but not  $Q_{ik}$
- (ii)  $Q_{ij} >_{pref} Q_{il}$  if both fulfill (1) and only  $Q_{ij}$  fulfills (2)

The DPO loss has been introduced in Section 2.4.3, Equation 2.11.  $y_w$  corresponds to a preferred reformulation, such as  $Q_{ij}$ , while  $y_l$  resembles a dispreferred one, such as  $Q_{ik}$  and  $Q_{il}$ . The policy  $\pi_\theta$  corresponds to our  $QU_{PRAISE}$  model and  $\pi_{ref} = QU_{SFT}$ .

In Figure 5.1, the rewritten questions  $Q_{32}$  : *Who replaced Pink Floyd’s guitarist?* and  $Q_{34}$  : *Who replaced Syd Barrett in Pink Floyd?* both fulfill (1) by retrieving relevant evidence, but only  $Q_{34}$  results in answering the question correctly and faithfully - fulfilling both (1) and (2). Question  $Q_{31}$  results in a correct answer, but this answer is not faithful given the retrieved evidence (*David Gilmour* is not mentioned in the retrieved set of evidence  $evs_{31}$ , first evidence in the ERF column in Figure 5.1), and therefore not considered as a positive sample.

**Inference.** The final  $QU_{PRAISE}$  model uses greedy generation to obtain a question reformulation.

**Algorithm 3:** PRAISE Training Approach

---

**Input:**  $\mathcal{D}_{train}$ , PRAISE *init* with  $QU_{few}$ ,  $ERF_{retrieval}$ ,  $AG_{SFT}$  (trained with  $\mathcal{D}_{train}$ ),  
 $\text{train}(\text{model}, \text{data}) \in \{SFT, DPO\}$

**Output:** PRAISE *full*

.....

► Create  $QU_{PRAISE}$  with the following steps:

- 1  $\langle Q_{ij} \rangle_r \sim QU_{few}(Q_i, \langle Q_\nu, A_\nu \rangle_{\nu \in \{0..i-1\}})$  ► sample reformulations
- 2  $\mathcal{D}_{QUsample} \leftarrow \langle Q_{ij}, \langle etext_{ij} \rangle_k, A_{ij} \rangle$  ► retrieve evidence and get answers per reformulation
- 3  $\langle Q_{ij} \rangle^+ \leftarrow \langle Q_{ij}, \langle etext_{ij} \rangle_k, A_{ij} \rangle$ ,  $\langle Q_{il} \rangle^- \leftarrow \langle Q_{il}, \langle etext_{il} \rangle_k, A_{il} \rangle$  ► according to QU conditions  
 (i) and (ii) in 5.1.2.1
- 4  $QU_{SFT} \leftarrow \text{train}(QU_{few}, \langle Q_{ij} \rangle^+)$
- 5  $QU_{DPO} \leftarrow \text{train}(QU_{few}, \langle Q_{ij}^+, Q_{il}^- \rangle)$

.....

► Create  $ERF_{PRAISE}$  with the following steps:

- 6  $ERF_{SFT} \leftarrow \text{train}(ERF_{init}, \langle eid : etext \rangle_k^{noisy+})$  ► SFT with noisy evs
- 7  $\langle \langle eids_{il} \rangle_k \rangle_e \sim ERF_{SFT}(Q_{ij}, \langle eid : etext \rangle_s)$  ► sample  $e$  eid sets of size  $k$  per ref
- 8  $\mathcal{D}_{ERFsample} \leftarrow \langle Q_{ij}, \langle etext_{il} \rangle_k, A_{il} \rangle$  ► generate answers for each  $\langle \text{ref}, \text{evs} \rangle$  pair
- 9  $\langle eids_{il} \rangle^+ \leftarrow \langle Q_{ij}, \langle etext_{il} \rangle_k, A_{il} \rangle$ ,  $\langle eids_{im} \rangle^- \leftarrow \langle Q_{ij}, \langle etext_{im} \rangle_k, A_{im} \rangle$  ► based on answer  
 correctness
- 10  $ERF_{DPO} \leftarrow \text{train}(ERF_{SFT}, \langle eids_{il}^+, eids_{im}^- \rangle)$

.....

► Create  $AG_{PRAISE}$  with the following step:

- 11  $AG_{SFTsample} \leftarrow \text{train}(AG_{SFT}, \mathcal{D}_{QUsample} \cup \mathcal{D}_{ERFsample})$

.....

- 12  $QU_{PRAISE} \leftarrow QU_{DPO}$ ,  $ERF_{PRAISE} \leftarrow ERF_{DPO}$ ,  $AG_{PRAISE} \leftarrow AG_{SFTsample}$
- 13 PRAISE *full*  $\leftarrow QU_{PRAISE} + ERF_{PRAISE} + AG_{PRAISE}$
- 14 return PRAISE *full*

---

**5.1.2.2 Evidence Retrieval And Filtering (ERF)**

This step identifies relevant pieces of evidence, tapping into knowledge graphs (KG), text corpora and web tables.

**Initialization.** The initial component for this step consists of the retrieval part only ( $ERF_{retrieval}$ ). Our approach is agnostic to the underlying retrieval method: we make use of off-the-shelf tools, to obtain a large number of top- $n$  evidence ranked by BM25 scores. The emphasis here is on high recall; obtaining high precision is left for the subsequent LLM-based filtering, as explained next.

**Training.** Each top- $n$  evidence is assigned a random id. We feed the rewritten question along with id-content pairs ( $\langle eid : etext \rangle$ ) into the LLM for filtering. The objective of the filtering model is to output a small subset of top- $k$  ( $k \ll n$ ) ids,  $\langle eids_{il} \rangle$ , corresponding to the most relevant evidence  $\langle evs_{il} \rangle$  with  $|\langle evs_{il} \rangle| \leq k$  (so as to obtain a tractable context length for the underlying answering LLM and to keep the token costs as low as possible):

$$\langle eids_{il} \rangle_k \leftarrow \mathbf{ERF}_{PRAISE}(Q_{ij}, \langle eid : etext \rangle_s) \quad (5.3)$$

We start by training the filtering model using SFT ( $ERF_{SFT}$ ), by considering evidence that contains

both question and answer entities as a positive sample.

In the example of Figure 5.1, the first and second evidence in set  $evs_{32}$  and set  $evs_{34}$ , respectively, are considered as positive samples, containing the entities *Pink Floyd*, *Syd Barrett* and *David Gilmour*. This is a weakly supervised technique, as we may face irrelevant snippets that merely contain the right entities by accident without being helpful for answering. In Figure 5.1, the first evidence from  $evs_{32}$  is such a spurious case. The  $ERF_{SFT}$  has to predict this noisy ground truth ( $\langle eid : etext \rangle_k^{noisy+}$ ) among a subset of retrieved evidence (line 6 in Algorithm 3). We include a set of challenging distractors which are evidence that have close semantic similarity to the question but do not contain the correct answers. The model is then trained using Equation 5.2. The effectiveness of the initial supervised model for the downstream answering task is limited by two important factors: (i) The lack of actual ground truth and (ii) the predicted set containing a mixture of relevant and irrelevant evidence, where it is not clear how the irrelevant ones affect the answering.

For high precision, the supervised ERF model is therefore enhanced by preference learning as follows. We sample  $e$  evidence sets per question from  $ERF_{SFT}$  ( $e = 10$  in our experiments). These pairs of question and evidence are passed to the AG stage to obtain answers, resulting in  $\mathcal{D}_{ERFsample}$  (line 7-8, Algorithm 3). Now we can exploit the available ground truth for answers as a distant signal: evidence set  $evs_{il}$  is preferred over set  $evs_{im}$  ( $evs_{il} >_{pref} evs_{im}$ ) if for the same question,  $evs_{il}$  in the input leads the AG model to the correct answer, while  $evs_{im}$  does not. In Figure 5.1, evidence set  $evs_{34}$  is preferred over set  $evs_{32}$ , since  $evs_{34}$  leads to the correct answer, but not  $evs_{32}$ . We use the preference pairs as training data for DPO (using Equation 2.11 from Section 2.4.3 and line 10, Algorithm 3), resulting in an enhanced model  $ERF_{DPO}$ .

**Inference.** For tractability, we split the top- $n$  retrieval results (which could be 500 or more) into smaller, manageable chunks of size  $s$ , and let the final  $ERF_{PRAISE}$  model identify relevant evidence ids for each chunk. The union over the per-chunk outputs form the evidence that is given to the AG stage. If less than  $k$  evidence pieces are returned, we expand the set with more top-ranked evidence from the original BM25 scoring.

### 5.1.2.3 Answer Generation (AG)

This step generates answers  $A_{im}$  for a question formulation  $Q_{ij}$ , based on the filtered evidence content  $\langle etext \rangle_k \in \langle evs_{il} \rangle_k$ :

$$A_{im} \leftarrow \mathbf{AG}_{PRAISE}(Q_{ij}, \langle etext \rangle_k) \quad (5.4)$$

In the example of Figure 5.1, both  $A_{31}$  and  $A_{34}$  are correct (*David Gilmour*), but only  $A_{34}$  is faithful with respect to the provided evidence set.

**Initialization.** We initialize the AG model by fine-tuning an LLM with the available (question, answer) pairs from the benchmark training data ( $\mathcal{D}_{train}$ ), resulting in  $AG_{SFT}$ . We use  $QU_{few}$  and standard retrieval (without filtering) to create the input for the AG stage.

**Training.** We improve the initial model by SFT using the sampled data from the previous stages (based on Equation 5.2 with  $\mathcal{D} = \mathcal{D}_{QUsample} \cup \mathcal{D}_{ERFsample}$ , line 11 in Algorithm 3). This way, the model learns from a larger variety of different surface forms and becomes more robust with respect

to the order in which relevant and irrelevant information appears in the evidence set.

**Inference.** We take the greedy model output along with multiple sampled outputs to obtain an answer list for better comparison with other (ranking-based) methods.

### 5.1.3 Experimental Setup

**Benchmark.** We conduct our experiments on ConvMix [Christmann et al.(2022b)], a popular ConvQA benchmark over heterogeneous sources (from Wikidata and Wikipedia), with realistic questions over 5 different domains - books, movies, tv-series, music, soccer - coming from crowdworkers, with crisp sets of entities as answers.

**Baselines.** We study the following methods: **CONVINSE** [Christmann et al.(2022b)], a method that casts questions into intent-explicit structured representations and uses fusion-in-decoder to generate answers; **EXPLAIGNN** [Christmann et al.(2023)], which uses graph neural networks for reasoning over evidence and generating answers; **Mistral-7B + Graph + Memory** [Jain and Lapata(2024)], which uses a graph structured representation with graph embeddings directly injected into an LLM (we compare with their best model); **UniHGKR** [Min et al.(2025)], with self-supervised pre-training, embedding alignment, and retriever fine-tuning (we compare to its 7B model). We also compare against end-to-end LLMs, specifically **Mistral-7B-Instruct** [Jiang et al.(2023)], in zero-shot mode and fine-tuned (reported numbers are taken from [Jain and Lapata(2024)]); **Llama-3-8B-Instruct** [Touvron et al.(2023)], in zero-shot and few-shot mode, and fine-tuned with benchmark QA-pairs, with the entire conversation history in the input but without RAG.

**Metrics.** We report Precision@1 (P@1), Mean Reciprocal Rank (MRR), and correct-answer-in-top-5 (Hit@5). Retrieval and evidence filtering performance is measured by answer presence (AP) in the top- $k$  selected evidence ( $k = 500$  and  $k = 50$ , respectively).

**Implementation Details.** We use Llama-3-8B-Instruct [Touvron et al.(2023)] as underlying LLM for PRAISE and perform parameter-efficient fine-tuning, more precisely, low-rank adaptation (LoRA) [Hu et al.(2021)], which fine-tunes low-rank adapter weights, while keeping the rest of the model frozen. This way, only ca. 12% of our model’s parameters are updated per subtask. We swap between the trained adapters at inference time, using the one for the current subtask, respectively.<sup>1</sup> The initial pipeline for PRAISE consists of i)  $QU_{init}$ : a Llama-3-8B-Instruct in few-shot mode (with 5 examples and prompted to generate 5 question formulations), ii)  $ERF_{init}$ : evidence retrieval without filtering, (retrieving  $n = 500$  evidence snippets and taking the top  $k = 50$  evidence based on BM25 scores), and iii)  $AG_{init}$ : a Llama-3-8B-Instruct model fine-tuned on benchmark QA-pairs for answer generation. We adopt techniques from [Christmann et al.(2022b)], specifically, using the CLOCQ tool [Christmann et al.(2022a)] for fact retrieval from the Wikidata KG and retrieving text snippets and table rows from Wikipedia. We perform beam search multinomial sampling with beam size=10 (allowing to combine the advantages of beam search with sampling), for ERF training and AG inference. The QU and ERF models are fine-tuned with SFT first, followed by DPO training for 1 epoch each. Further hyperparameters were set as follows: learning rate =  $1e - 6$ , maximal input length

<sup>1</sup>[https://huggingface.co/docs/peft/main/en/conceptual\\_guides/adapter](https://huggingface.co/docs/peft/main/en/conceptual_guides/adapter)

Method ↓ Metrics →	P@1	Hit@5	MRR
CONVINSE	0.342	0.386	0.365
EXPLAIGNN	0.406	0.561	0.471
Mistral-7B + Graph + Memory	0.445	0.512	-
UniHGKR-7B	0.465	0.562	0.514
Mistral-7B-Instruct (zero-shot)	0.292	0.346	-
Mistral-7B-Instruct (fine-tuned)	0.350	0.400	-
Llama-3-8B-Instruct (zero-shot)	0.403	0.554	0.455
Llama-3-8B-Instruct (few-shot)	0.461	0.612	0.511
Llama-3-8B-Instruct (fine-tuned)	0.483	0.640	0.536
PRAISE	<b>0.620</b>	<b>0.746</b>	<b>0.665</b>

Table 5.2: Main results comparing PRAISE to end-to-end LLMs and other competitors on the CONVMIX test set.

Method ↓ Metrics →	P@1	Hit@5	MRR	AP@500	AP@50
$QU_{init} + ERF_{init} + AG_{init}$	0.495	0.642	0.543	0.734	0.486
$QU_{PRAISE} + ERF_{init} + AG_{init}$	0.528	0.683	0.579	<b>0.767</b>	0.536
$QU_{init} + ERF_{PRAISE} + AG_{init}$	0.559	0.693	0.616	0.734	0.725
$QU_{init} + ERF_{init} + AG_{PRAISE}$	0.546	0.678	0.591	0.734	0.486
$QU_{PRAISE} + ERF_{PRAISE} + AG_{init}$	0.582	0.712	0.627	<b>0.767</b>	<b>0.758</b>
$QU_{PRAISE} + ERF_{init} + AG_{PRAISE}$	0.580	0.708	0.624	<b>0.767</b>	0.536
$QU_{init} + ERF_{PRAISE} + AG_{PRAISE}$	0.587	0.723	0.635	0.734	0.725
PRAISE $_{full}(QU + ERF + AG)$	<b>0.620</b>	<b>0.746</b>	<b>0.665</b>	<b>0.767</b>	<b>0.758</b>

Table 5.3: Effect of different pipeline components in PRAISE.

= 512, maximal target length = 75, batch size = 1, LoRA rank  $r = 64$ , and for DPO  $\beta = 0.1$ , controlling the KL-divergence penalty. All experiments are run on a single GPU (NVIDIA Tesla A100, 80 GB HBM2e). More details can be found in our code base (<https://github.com/magkai/PRAISE>).

## 5.1.4 Results and Insights

### 5.1.4.1 Key Findings

**PRAISE achieves best performance.** Table 5.2 compares the performance of PRAISE to the baselines on the CONVMIX test set. PRAISE outperforms all baselines by a substantial margin, gaining 15.5 percentage points in P@1 over the second-best method (UniHGKR-7B - a model of similar size), with similarly strong improvements on the other metrics. PRAISE also outperforms the end-to-end LLMs, showing the benefits of including RAG and a multi-stage pipeline.

**All pipeline stages contribute.** Table 5.3 shows the effects of different components in PRAISE, by comparing the full PRAISE model to its variants, where one or two of its components – QU, ERF and AG – are replaced with the respective initial models. The variants with one PRAISE stage and

Domain →	Books		Movies		TV series		Music		Soccer	
Method ↓	P@1	AP@50	P@1	AP@50	P@1	AP@50	P@1	AP@50	P@1	AP@50
PRAISE <i>init</i>	0.491	0.477	0.530	0.438	0.506	0.500	0.505	0.510	0.443	0.510
PRAISE <i>full</i>	<b>0.650</b>	<b>0.814</b>	<b>0.634</b>	<b>0.721</b>	<b>0.614</b>	<b>0.716</b>	<b>0.630</b>	<b>0.779</b>	<b>0.572</b>	<b>0.760</b>

Table 5.4: Domain-wise for PRAISE and the initial pipeline ( $QU_{init} + ERF_{init} + AG_{init}$ ).

Turn →	1		2		3		4		5		6-10	
Method ↓ /	P@1	AP@50	P@1	AP@50	P@1	AP@50	P@1	AP@50	P@1	AP@50	P@1	AP@50
PRAISE <i>init</i>	0.684	0.554	0.586	0.572	0.496	0.521	0.453	0.459	0.429	0.462	0.364	0.382
PRAISE <i>full</i>	<b>0.743</b>	<b>0.829</b>	<b>0.664</b>	<b>0.820</b>	<b>0.662</b>	<b>0.805</b>	<b>0.595</b>	<b>0.742</b>	<b>0.574</b>	<b>0.718</b>	<b>0.515</b>	<b>0.663</b>

Table 5.5: Turn-wise results for PRAISE and the initial pipeline ( $QU_{init} + ERF_{init} + AG_{init}$ ).

two initial models show that each of the three components alone already improves performance. This trend is further enhanced with two components by PRAISE, and the full PRAISE outperforms all other configurations by a notable margin.

#### 5.1.4.2 In-depth Analysis

**PRAISE achieves high answer presence.** Table 5.3 gives insights on answer presence. In the retrieval step, top-500 evidence snippets are obtained, resulting in an answer presence of ca. 73%. When using the reformulated questions by  $QU_{PRAISE}$ , AP is increased by 3%. For filtering, when we simply use BM25 scores to obtain top-50 evidence, the AP drops sharply: down to 49% for the initial model. In contrast, the ERF of PRAISE retains high answer presence at 75.8%, when reducing the initially retrieved top-500 down to 50 evidence pieces. Another interesting observation is that the intermediate  $ERF_{SFT}$  model has a much lower answer presence (ca. 51%) than our  $ERF_{PRAISE}$  model. This underlines that the DPO-based fine-tuning with feedback from the AG model is crucial to optimize the preceding ERF stage. Overall, this reconfirms our intuition that optimizing the subtasks of the pipeline, with training samples automatically derived from delayed feedback via question-answer pairs, is beneficial.

**Preference optimization is effective.** Instead of performing preference learning, we could also directly use the generations that successfully lead to the correct answer for further supervised learning (replacing both,  $QU_{PRAISE}$  and  $ERF_{PRAISE}$ , with  $QU_{SFT}$  and  $ERF_{SFT}$  respectively, in the pipeline). This results in a performance decrease ( $P@1 = 0.585$  instead of 0.620 for PRAISE), demonstrating that using DPO after an initial supervised fine-tuning step is more effective.

**PRAISE creates fluent reformulations.** While the original questions are relatively short – ca. 5.6 words on average, the reformulations generated by  $QU_{init}$  have an average length of 9.3 words, and the ones by  $QU_{PRAISE}$  go up to 10.1. Our method successfully contextualizes the questions with cues from previous turns in the conversation, while keeping the questions natural and fluent. This is confirmed by analyzing 200 reformulations manually. Table 5.6 at the end of this chapter shows some exemplary conversations from CONVMIX, along outputs generated by the three stages in PRAISE.

**Performance improvements across domains and turns.** Table 5.4 shows domain-wise results and Table 5.5 the turn-wise results for PRAISE, in comparison to the initial pipeline ( $QU_{init} + ERF_{init} + AG_{init}$ ) on the CONVMIX testset. The performance of PRAISE is robust across domains, consistently improving upon the initial pipeline (with up to 16 percentage points increase in  $P@1$  and 34 percentage points increase in  $AP@50$ ). The answer presence of PRAISE is highest for books (81%) and lowest for movies and tv-series (72%). Nonetheless, PRAISE achieves high precision for all of these three domains (in the range of 61 to 65%). We attribute this to the rich signals from the underlying LLM’s parametric memory (from its original pre-training). Thus, PRAISE incorporates informative RAG without losing the benefits from LLM memory.

As to be expected, turn-wise performance is highest for the first few turns and drops with increasing turn depth (for the initial pipeline as well as for the final PRAISE model). Yet, PRAISE outperforms the initial pipeline consistently over turns (with up to 15 percentage points difference in  $P@1$  and up to 28 percentage points difference in  $AP@50$  for turns 6 – 10). Also, the initial pipeline does not answer questions faithfully, indicated by a  $P@1$  score higher than the answer presence.

**Runtime analysis and memory consumption.** PRAISE requires around 5.4 s (seconds) on average to answer a conversational question. This time is spent as follows:  $QU_{PRAISE}$  takes around 0.5 s,  $ERF_{PRAISE}$  takes 4.6 s (with 1.3 s spent for evidence retrieval and 3.3 s for evidence filtering), and  $AG_{PRAISE}$  requires 0.34 s. The time for  $ERF_{PRAISE}$  can be decreased further by either retrieving less evidence (currently 500) or splitting into larger chunks for evidence filtering (currently, we split into 10 chunks with 50 evidence pieces each, resulting in 10 calls to the filtering model). Instead of loading three LLMs (one for each subtask) into memory, we swap our trained task-specific adapters, keeping only one LLM in memory. This way, PRAISE requires around 34 GB of GPU memory.

### 5.1.5 Related Work

**Conversational Context Understanding.** There are two paradigms for conversational context understanding - question completion and history modeling (see background, Section 2.1.2). We opt for rewriting the question into a (potentially more complete) version, since we use LLMs in our pipeline, which benefit from fluent input. Many earlier works required human rewrites (e.g., [Vakulenko et al.(2021), Anantha et al.(2021)]), which is costly to collect and not necessarily optimal for retrieval. Recent works aim to improve rewriting for retrieval (e.g., [Mo et al.(2023), Zhang et al.(2024), Yoon et al.(2025), Ma et al.(2023)]), whereas PRAISE optimizes for retrieval as well as answering performance.

**RAG for ConvQA.** The RAG paradigm has recently received great attention as an enhancement of LLM-based generation including question answering to mitigate the risk of hallucination, and to make QA (more) faithful [Guu et al.(2020), Lewis et al.(2020b), Gao et al.(2023b)]. Many ConvQA methods make use of LLMs, dense or sparse neural retrieval and various kinds of representation learning (e.g., [Christmann et al.(2023), Hai et al.(2023), Jain and Lapata(2024), Jeong et al.(2023), Mao et al.(2024), Mo et al.(2023), Tran et al.(2024)]).

**Learning from Automatic Feedback for QA.** Reinforcement learning has been used as a means to improve QA models with answering performance as feedback (e.g., [Buck et al.(2018), Kaiser

et al.(2024b), Yang et al.(2023)]). In the context of LLMs, Reinforcement Learning from Human Feedback (RLHF) [Ziegler et al.(2019), Ouyang et al.(2022)] has been proposed to align LLMs to human preferences. Direct Preference Optimization (DPO) [Rafailov et al.(2023)], and variants, were introduced as less complex and more stable options. Since collecting human preference is costly, an alternative is to use automatic feedback (e.g., [Yuan et al.(2024), Peng et al.(2023), Zhang et al.(2024)], also see Section 2.2.2 in the background chapter for further details).

### 5.1.6 Conclusion

This work presents PRAISE, an LLM-based pipeline for ConvQA, where earlier tasks are improved with feedback from later tasks. To achieve these enhancements, we use DPO with training data derived from contrastive pairs of successful and unsuccessful samples. This method is particularly suitable for domains in which the task can be divided into subtasks, and when there is delayed, task-level feedback but only sparse or no intermediate annotations available. The next part shows how we apply techniques from PRAISE to such a domain, namely task-oriented dialogs, where we use dialog-level feedback for inferring relevance of intermediate subtasks.

---

**Conversation [books]:** *What number of Pulitzers has John Updike won? 2; What book won the author the award first? Rabbit Is Rich; name of the other? Rabbit at Rest; publication year of the book?*

---

**QU (rewritten question):** *What is the publication year of the book "Rabbit at Rest" by John Updike?*

**ERF (filtered evidence):** [*id-150*: John Updike, award received, National Book Critics Circle Award, point in time, 1990, for work, Rabbit at Rest], [*id-391*: Rabbit at Rest, Publication date, 1990], [*id-127*: Rabbit at Rest is a 1990 novel by John Updike] ...

**AG (generated answer):** *1990*

---

**Conversation: [movies]:** *What was Jack Nicholson's last film of the 1990s? As Good as It Gets; Who directed?*

---

**QU (rewritten question):** *Who directed the film "As Good as It Gets" starring Jack Nicholson?*

**ERF (filtered evidence):** [*id-389*: As Good as It Gets is a 1997 American romantic comedy film directed by James L. Brooks from a screenplay he co-wrote with Mark Andrus.], [*id-307*: James L. Brooks and Jack Nicholson, doing what they do best, combine smart dialogue and flawless acting...], [*id-187*: As Good as It Gets, director, James L. Brooks] ...

**AG (generated answer):** *James L. Brooks*

---

**Conversation: [TV series]:** *Who created the iron fist tv series? Scott Buck; The composer of the series?*

---

**QU (rewritten question):** *Composer of the Iron Fist TV series?*

**ERF (filtered evidence):** [*id-140*: Iron Fist, composer, Trevor Morris], [*id-38*: Iron Fist, Composers, Trevor Morris , Robert Lydecker], [*id-493*: "In late October 2016, Trevor Morris was revealed to be composing the music for the first season.] ...

**AG (generated answer):** *Trevor Morris*

---

**Conversation: [music]:** *What year was Inna born? 1986; What country is she from? Romania; Her first album? Hot; How many singles were released?*

---

**QU (rewritten question):** *How many singles were released from Inna's debut album Hot?*

**ERF (filtered evidence):** [*id-4*: To support Hot, five singles were released, which according to Libertatea sold a total of 500,000 copies ...], [*id-418*: It featured several other successful singles in Europe, including "Amazing" (2009).], [*id-111*: Inna's debut single, "Hot", was sent to Romanian radio stations in August 2008.] ...

**AG (generated answer):** *5*

---

**Conversation [soccer]:** *On which date was the soccer player Neymar born? 5 February 1992; Complete name? Neymar da Silva Santos Júnior; How tall? 5 ft 9 in; Position at which he plays?*

---

**QU (rewritten question):** *What position does Neymar play in soccer?*

**ERF (filtered evidence):** [*id-231*: ... Neymar was voted the Best player and Best Forward], [*id-296*: Neymar, Personal information, Position(s), Left winger, attacking midfielder, forward], [*id-206*: List of international goals scored by Neymar, Neymar is a Brazilian professional footballer who plays as a forward.]

**AG (generated answer):** *Left winger, forward*

---

Table 5.6: Examples of outputs generated by the three stages in PRAISE (three evidence pieces are shown due to space limitations, irrelevant evidence indicated in red).

**QU Prompt:** *You are given a conversation and the current user question. Your task is to reformulate the question into a self-sufficient form, that clearly conveys the question intent, by using information from the conversation history. Just create the standalone question without commentary and without answering the question.*

*'History':* {Conversation history}

*'Question':* {Question}

*'Reformulation':*

**ERF Prompt:** *You are given a question and a set of references, each with a unique identifier. Your task is to judge which references provide relevant information for answering the given question. You should output the corresponding ids of the most relevant references, nothing else.*

*'Question':* {Question}

*'Context':* {<evidence id, evidence text>}

*'Ids':*

**AG Prompt:** *You are given a question and references which may or may not help answer the question. Please answer the question in as few words as possible by using the information provided in the references that is relevant in answering the question.*

*'Question':* {Question}

*'Context':* {<evidence id, evidence text>}

*'Answer':*

Figure 5.2: Prompts for question understanding, evidence filtering and answer generation in PRAISE.

**QU Fewshot Prompt:** *You are given a conversation and the current user question. Your task is to reformulate the question into a self-sufficient form, that clearly conveys the question intent, by using information from the conversation history. Just create the standalone question without commentary and without answering the question. Please learn from the examples below.*

*'History':* who is the author of book The Velveteen Rabbit? Margery Williams; ...

Genre? children's novel;

*'Question':* country of origin

*'Reformulations':*

1. Which is the country of origin of book The Velveteen Rabbit?
2. Where is the book The Velveteen Rabbit from?
3. Where is Margery Williams from?
4. Country of origin The Velveteen Rabbit?
5. Place where The Velveteen Rabbit novel was created?

*'History':* For which country team did Paul Pogba play? France;

Where was he born? Lagny-sur-Marne;

*'Question':* position played on team?

*'Reformulations':*

1. At which position did Paul Pogba play on team?
2. What is Paul Pogba's position in the France national team?
3. Paul Pogba position on team France
4. Which position on the field Paul Pogba
5. What role does Paul Pogba occupy in the french team?

*'History':* how many seasons of Mindhunter series? 2;

who is the director? David Fincher;

*'Question':* How many episodes?

*'Reformulations':*

1. How many episodes are in the Mindhunter series?
2. Number of episodes Mindhunter series
3. Total amount of episodes of Mindhunter
4. How many episodes are in David Fincher's Mindhunter series?
5. Episode count for Mindhunter

*'History':* -

*'Question':* Who played as Mark Zuckerberg in The Social Network?

*'Reformulations':*

1. Who played as Mark Zuckerberg in The Social Network?
2. The Social Network actor Mark Zuckerberg
3. Cast member for Mark Zuckerberg in the movie The Social Network
4. Who acted as Mark Zuckerberg in The Social network movie
5. Who starred as Mark Zuckerberg in The Social Network?

*'History':* U2 had how many band members? 5; which year did it formed? 1976;

what about the band drummer? Larry Mullen Jr.

*'Question':* who was their bass guitarist?

*'Reformulations':*

1. Who was the bass guitarist of the U2 band?
2. Which U2 band member played the guitar?
3. Bass guitar player U2 band
4. U2 bass guitarist
5. Who played the guitar in the U2 band formed in 1976

*'History':* {Conversation history}

*'Question':* {Question}

*'Reformulations':*

Figure 5.3: Prompt for few-shot question understanding with 5 in-context examples in PRAISE.

## 5.2 Learning from Preferences for Task-oriented Dialogs

### 5.2.1 Introduction

**Motivation.** Task-oriented dialog (ToD) systems converse with users to assist them in various tasks, such as booking restaurants, querying weather forecasts and resolving customer service issues. A task-oriented sample conversation for making a hotel reservation (with user requests  $U_i$  and system responses, consisting of tuples  $\langle B_i, A_i, R_i \rangle$ ):

**User Goal:** *informable: hotel area:north, price range:moderate*  
*requestable: booking reference, hotel address*  
 $U_1$ : *I need a hotel in the north of the city with a moderate price.*  
 $B_1$ : *hotel area:north, price range:moderate*  
 $A_1$ : *INFORM-COUNT( $\cdot$ ), RECOMMEND-HOTEL( $\cdot$ )*  
 $R_1$ : *There are 10 moderately priced hotels in the north. I would recommend the abc hotel which has a 4 star rating.*  
 $U_2$ : *Are there any sights nearby?*  
 $B_2$ : *hotel area:north, price range:moderate*  
 $A_2$ : *INFORM-SIGHTS( $\cdot$ )*  
 $R_2$ : *No, sights are located in the center only.*  
 $U_3$ : *Can you book it now and give me the address and reference number please?*  
 $B_3$ : *hotel area:north, price range:moderate*  
 $A_3$ : *REQUEST-COUNT( $\cdot$ ), REQUEST-DURATION( $\cdot$ )*  
 $R_3$ : *How many people and how long will you stay?*  
 $U_4$ : *We are 4 and will stay for 2 nights, starting from next Sunday*  
 $B_4$ : *hotel area:north, price range:moderate people:4, duration:2, start:Sunday*  
 $A_4$ : *INFORM-REFERENCE( $\cdot$ ), INFORM-ADDRESS( $\cdot$ )*  
 $R_4$ : *Booking was successful. Your reference number is AB12 and the hotel is located at king street. Anything else I can help you with?*

In the example, the user’s goal is defined as a set of constraints (*informable slots*, e.g. price range) and information needs (*requestable slots*, e.g. hotel address) that should be satisfied at the end of the dialog. For accomplishing such goals, ToD systems must be able to solve multiple subproblems (also see Section 2.1.5): (1) *dialog state tracking* (DST) (belief states  $B_i$  with slot-value pairs, e.g. for *area* or *price range* in the example), (2) dialog action prediction (actions  $A_i$  like *REQUEST-COUNT( $\cdot$ )*, *INFORM-ADDRESS( $\cdot$ )* in the example), (3) response generation (e.g. asking about the length of the stay and the number of people in the example,  $R_i$ ).

**Limitations of state-of-the-art.** Specialized approaches focus on solving specific problems, e.g., amongst others, [Lee et al.(2021)], AG-DST [Tian et al.(2021)], D3ST [Zhao et al.(2022)] focus on DST, LarL [Zhao et al.(2019)], TCUP [Vlastelica et al.(2023)] concentrate on response generation. End-to-end systems try to leverage complex models to solve all problems jointly. Modern ToD systems in that category are based on pre-trained Large Language Models (LLMs) and cast dialog state, action and response generation into sequence prediction problems. SimpleTOD [Hosseini-Asl et al.(2020)] was the first approach which successfully applied this paradigm, by training a causal language model using Supervised Fine-Tuning (SFT). A challenge for ToD systems is the fact that

Notation	Concept
$D_i \in \mathcal{D}_x$	dialog $i$ in training data at iteration $x$
$\mathcal{D}_c, \mathcal{D}_s$	set of generated candidate dialogs, successful dialogs
$\mathcal{D}_G \in g(\mathcal{D}_c)$	set of candidate dialogs sharing same user goal
$U_{it}, S_{it}$	user utterance, system response for turn $t$ in dialog $D_i$
$C_{it}, B_{it}$	conversational context, belief state for turn $t$
$T_{it} = [A_{it}, R_{it}]$	joint prediction of dialog action and response

Table 5.7: Notation for salient concepts in SUIIT.

ultimate success with respect to the user’s goal is observed at the end of the dialog. While Reinforcement Learning (RL) approaches [Zhao et al.(2019), Lubis et al.(2020), Vlastelica et al.(2023)] optimize for such sparse rewards, most LLM-based systems neglect these signals and only optimize next turn predictions.

**Approach.** We introduce SUIIT (= SUBgoal-aware ITERative Training), an end-to-end ToD system based on LLMs, which contrary to prior work learns from dialog-level success signals. Due to the sparseness of these signals, it is unclear which turns, states, actions and responses contribute to the overall success of the dialog. For instance, the second turn in the previous sample conversation is irrelevant for the success of the dialog. The user’s goal does not depend on the availability of sights in close vicinity to the hotel, contrary to the respective price range, which will affect the system’s success. We aim to identify these important subparts, which we call *subgoals*, from multiple dialog variants, generated by an LLM. A naive approach would consider all generations from successful dialogs for SFT, or pair all generations from successful with unsuccessful dialogs for preference learning algorithms, such as RLHF [Ouyang et al.(2022)] or Direct Preference Optimization (DPO) [Rafailov et al.(2023)]. However, these naive approaches cannot distinguish between subgoals that are relevant for the final goal from those that are not. We employ distant supervision and counterfactual learning to identify these subgoals that play a major role in dialog success. Our method is then iteratively improved using these subgoals as further high-quality training data.

**Contributions.** This work makes the following contributions:

- An *iterative* training procedure for LLM-based task-oriented dialog models with *automatically derived* training data;
- A *distant supervision* approach that uses *counterfactual learning* for finding relevant *subgoals* based on dialog-level feedback;
- Experiments demonstrating the superior performance of SUIIT on a popular ToD benchmark.

### 5.2.2 The SUIIT Method

Figure 5.4 provides an overview of the SUIIT approach and Table 5.7 lists the salient notation. As input we rely on a ToD dataset  $\mathcal{D}$ , where each dialog is associated with a user goal. This goal describes the user’s information need and a set of constraints, that should be fulfilled at the end

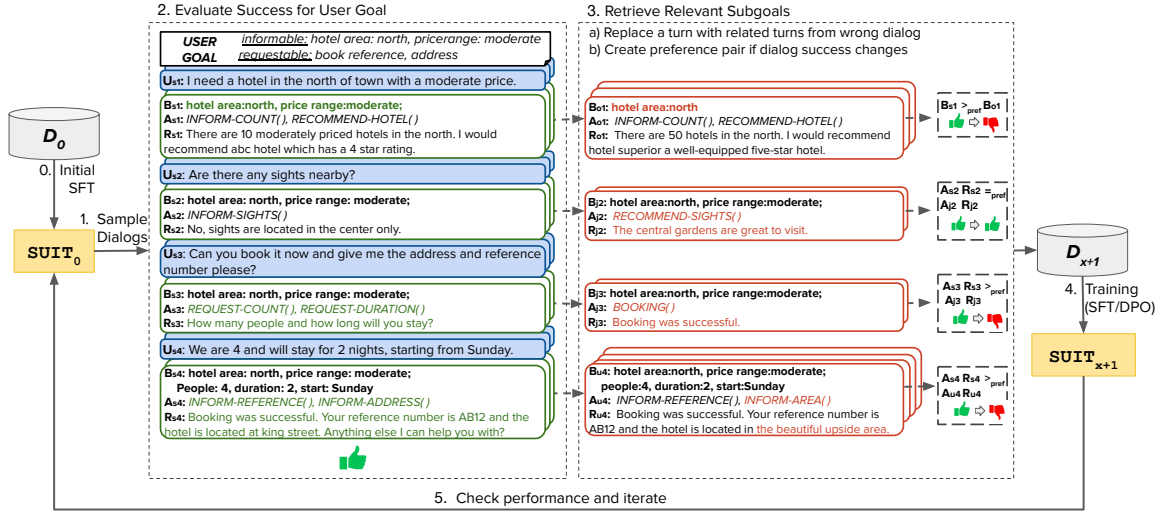


Figure 5.4: Overview of training procedure in SUIT. Subgoals ( $B_{st}$  or  $A_{st}/R_{st}$ ) in sampled dialogs that are evaluated as successful ( $D_s$ ), are replaced with respective parts from wrong dialogs ( $D_{o,j,u}$ ) to determine its relevance for dialog success; relevant subgoals are added as further training data.

of the dialog. First, an initial LLM is trained on  $\mathcal{D}$  with supervised fine-tuning, using turn-level supervision (Step 0). Then, we sample from this model to create dialog variants for each user goal in the training data (Step 1). Next, we determine dialog success for the newly created dialogs (Step 2). For each successful dialog, we apply distant supervision to identify subgoals that contribute to the ultimate success of the dialog, by comparing them with generations coming from unsuccessful dialogs (more details in Section 5.2.2.2). The subgoals considered relevant comprise further training samples (Step 3). The SUIT approach can apply SFT or preference learning based on a dataset which pairs subgoals with negative examples from unsuccessful dialogs (Step 4). This procedure is repeatable by sampling from the newly obtained model once again (Step 5). Using this effective training paradigm, SUIT improves SOTA performance. Compared to prior end-to-end ToD systems, SUIT is not based on model customization and does not require feedback from reward models or annotators. Any off-the-shelf LLM can be plugged into SUIT, which makes it simple to set up and use. For preference learning, we apply DPO due to its efficiency, low complexity and stable training.

### 5.2.2.1 Initial LLM-based ToD Model

For each turn index  $t$  in a dialog  $D_i$ , we are given a dialog context  $C_{it} = [U_{i0}, S_{i0}, \dots, S_{it-1}, U_{it}]$  consisting of the current  $U_{it}$ , and previous user utterances  $U_{ij_{j < t}}$ , as well as system turns  $S_{ij_{j < t}}$ . The goal is to train an initial LLM ( $SUIT_0$ ) for generating the system turn  $S_{it}$ , which contains belief states  $B_{it}$ , system actions  $A_{it}$ , and a response  $R_{it}$ . We split the problem into two prediction tasks:

1. We predict the belief state  $B_{it} = SUIT_0(C_{it})$ ;
2. Actions and responses are jointly generated  $T_{it} = SUIT_0([C_{it}, B_{it}])$ , where  $T_{it} = [A_{it}, R_{it}]$ ;

We prompt the model twice and introduce special tokens  $[C]$ ,  $[U]$ ,  $[S]$ ,  $[B]$ ,  $[A]$ ,  $[R]$  to indicate dialog context, user utterances, system responses, belief states and actions accordingly (see prompts in

**Algorithm 4:** SUIT Training Approach

---

**Input:** dialogs  $\mathcal{D}_x = \mathcal{D}$ ,  $x = 0$ , user goal partitioning over dialogs  $g(\mathcal{D})$ , initial model  $\text{SUIT}_x$ ,  $\text{train} \in \{\text{DPO}, \text{SFT}\}$ , success criteria  $\text{succ}$

**Output:** iteratively improved model  $\text{SUIT}_x$

```

1 repeat
2    $\mathcal{D}_c \leftarrow \bigcup_{D_i \in \mathcal{D}} \{[U_{i0}, \dots, S'_{it}, \dots, S'_{i|D_i}|] \mid S'_{it} = [B_{it}, T_{it}] \sim \text{SUIT}_x\}$  (1)
3    $\mathcal{D}_{x+1} \leftarrow \emptyset$ 
4   foreach dialog set with same goal  $\mathcal{D}_G \in g(\mathcal{D}_c)$  do
5     foreach dialog  $D_s \in \mathcal{D}_G$ :  $\text{succ}(D_s)$  (2) do
6        $\mathcal{D}_{x+1} \leftarrow \mathcal{D}_{x+1} \cup \{(C_{st}, S'_{st}) \mid$  (3)
7          $\exists D_o \in \mathcal{D}_G, S'_{ot} \in D_o : \neg \text{succ}(D_o) \wedge$ 
8          $\neg \text{succ}([U_{s0}, \dots, S'_{ot}, \dots, S'_{s|D_s}|])\}$ 
9    $\text{SUIT}_{x+1} \leftarrow \text{train}(\text{SUIT}_x, \mathcal{D}_{x+1})$  (4)
10   $x \leftarrow x + 1$ 
11 until  $\text{SUIT}_x$  does not improve (5)
12 return  $\text{SUIT}_x$ 

```

---

Figure 5.5 at the end of this chapter). The model is trained to minimize the negative log-likelihood over the training dataset. We provide ground truth belief for action and response prediction at training time. During inference, SUIT generates belief states conditioned on input dialog contexts. Actions and responses are then predicted conditioned on the context and this generated belief.

### 5.2.2.2 Subgoal Candidate Generation

We split each training dialog  $D_i$  into all possible dialog contexts  $C_{it}$  and sample a *set* of  $k$  states  $\mathcal{B}_{it}$ , and per state, another  $k$  actions  $\mathcal{A}_{it}$  and responses  $\mathcal{R}_{it}$  from a given SUIT model<sup>2</sup>:

$$\mathcal{B}_{it} = \bigcup_{B \sim \text{SUIT}(C_{it})}^k B, \mathcal{T}_{it} = \bigcup_{B \in \mathcal{B}_{it}, T \sim \text{SUIT}(C_{it}, B)}^k T \quad (5.5)$$

Algorithm 4 shows the precise steps of the SUIT training loop. We obtain  $k^2 + 1$  (including the greedy model output) candidate dialogs  $\mathcal{D}_c$  by replacing the turns  $S_{it}$  with samples  $S'_{it}$  (Step (1) in Algorithm 4 and Figure 5.4).

### 5.2.2.3 Distant Supervision for Subgoal Detection

We only consider successful dialogs as source of potential training data (Step (2) in Algorithm 4 and Figure 5.4). To determine dialog success, we use the evaluation function from [Nekvinda and Dušek(2021)] by verifying if both INFORM and SUCCESS metrics are fulfilled after the last turn. More precisely, a dialog is considered successful if the last offered entity satisfies the user’s goal constraints and the system mentioned all *requestable* slots defined in the user’s goal in its response. In Figure 5.4 and Algorithm 4, dialog  $D_s$  is successful. For each successful dialog, we search for unsuccessful dialogs sharing the same user goal (Step (3) in Algorithm 4, and  $D_{j,o,u}$  in Figure 5.4).

<sup>2</sup>We enforce that the samples contain greedy generations

If found, we go over the successful dialogs turn-wise and apply a counterfactual learning strategy: we replace a state  $B_{st}$  (or action/response,  $A_{st}/R_{st}$ ) with the respective state (action/response) in the unsuccessful one. After each replacement, we once again evaluate the modified dialog. If the dialog is now unsuccessful, the replaced subgoal was indeed crucial for making it successful. If the dialog is still successful, we cannot make any judgment, since the replacement from the unsuccessful dialog might be correct (there can be correct subgoals in unsuccessful dialogs). Therefore, we sample replacements from different unsuccessful dialogs to see whether the evaluation changes. Please note, that we only make *one* replacement at a time, while the other turns of the successful dialog remain unchanged. State replacements are done separately, while actions and responses are replaced jointly. Replacements for a respective turn  $t$  come from another dialog (with same user goal) at the same turn level  $t$ . This makes sense in our setup, since samples share the same ground truth dialog context. Nevertheless, our method is robust to different dialog flows. For high variations in dialog flow, one could additionally apply a similarity based scoring to find the most suitable turn for replacement first. Replacing the state at turn 1 in Figure 5.4, as well as replacing action/responses at turn 3 and 4, each change the evaluation of dialog  $s$  from successful to unsuccessful, whereas the replacement with  $A_{j2}/R_{j2}$  results in no change. Therefore,  $B_{s1}$ ,  $A_{s3}$ ,  $R_{s3}$ ,  $A_{s3}$ ,  $R_{s3}$  are considered as relevant subgoals and will be used for training, while  $A_{s2}$  and  $R_{s2}$  are *not* used as training data, since no replacement was found that has changed the evaluation of the dialog. This procedure creates a small, high-quality training set,  $\mathcal{D}_{x+1}$  (Step (4) in Algorithm 4 and Figure 5.4), which is dense in samples that are critical to the final dialog success. For SFT, we use the selected subgoals, while for DPO, we take the selected subgoals as preferred samples (like  $B_{s1}$ ) and the replacements, which made the dialog change from successful to unsuccessful, as dispreferred samples (like  $B_{o1}$ ). In summary, SUIT’s iterative training approach consists of the following steps:

- (1) Given an LLM  $\text{SUIT}_x$ , generate more dialog variations  $\mathcal{D}_c$  by sampling;
- (2) Evaluate *success* for all dialogs using the evaluation function from [Nekvinda and Dušek(2021)];
- (3) Identify relevant subgoals by replacing successful ones using distant supervision;
- (4) Apply SFT or DPO to derive model  $\text{SUIT}_{x+1}$ ;
- (5) Assess model performance, repeat or stop;

### 5.2.3 Experimental Setup

**Benchmark.** We use MultiWOZ 2.2 [Zang et al.(2020)], which is a popular ToD benchmark. It contains 10k human-human dialogs over 7 domains. For pre-processing we follow the standard approach for this dataset<sup>3</sup>. Table 5.8 shows some statistics about it.

**Baselines.** We compare with state-of-the-art end-to-end systems from the MultiWOZ leaderboard. MARS [Sun et al.(2023)] uses a contrastive loss to differentiate dialog contexts with the same states from dissimilar ones. KRLS [Yu et al.(2023)] applies RL with a special reward, sensitive to important keywords. DIACTOD [Wu et al.(2023)] uses an action encoder to perform nearest neighbor search over latent representations of ground truth actions.

<sup>3</sup>[https://github.com/Tomiinek/MultiWOZ\\_Evaluation/tree/master](https://github.com/Tomiinek/MultiWOZ_Evaluation/tree/master)

	# Goals	#Turns	Avg. #Turns/Dialog
train	8437	56776	6.70
dev	1000	7374	7.37
test	1000	7372	7.37

Table 5.8: Data statistics for the MultiWOZ benchmark (version 2.2.)

**Metrics.** We follow the standardized evaluation from [Nekvinda and Dušek(2021)] to allow for a better comparability. A delexicalized BLEU score measures response coherence, while INFORM and SUCCESS rates express how much a user’s goal is fulfilled at the end of the dialog. It is common to assess the overall performance with a COMBINED score =  $BLEU + \frac{INFORM+SUCCESS}{2}$ .

**Implementation Details.** We use an encoder-decoder Flan-T5 large model from Hugging Face<sup>4</sup> in our experiments. It is trained for one epoch per iteration. We verbalize states and actions to be more suitable for generative models. After initial model training (SUIT<sub>0</sub>), SUIT models are trained for up to two rounds of SFT/DPO. As stopping criteria we use the COMBINED score. For efficiency, we sample half of the user goals in the training data per iteration for creating new dialogs. We use  $k = 2$  for sampling these new dialogs and additionally take the greedy generation (resulting in  $k^2 + 1$  different dialogs).

For supervised learning, the batch size was set to 2 and learning rate to  $5e - 5$ . The maximal input length of 512 was used, the maximal target length was 256 and for generation, beam search with a beam size of 5 was used. We use default parameters when sampling from the model (top- $k$  was set to 0). For DPO training, we set hyperparameters as follows:  $\beta = 0.1$ , batch size = 2, learning rate =  $1e - 6$ , warmup steps = 150, maximal input length = 512, and target length = 256.

## 5.2.4 Results and Insights

### 5.2.4.1 Key Findings

Table 5.9 shows SUIT models, trained for up to two iterations with SFT/DPO. It compares their performances with competitors and baselines, including models trained for one round with all successful dialogs instead of using relevant subgoals only.

**SUIT improves SOTA performance.** Our initial model reaches the highest BLEU score (19.94). While further iterations decrease this metric slightly, they reach the highest INFORM (90.0), SUCCESS (87.1), and COMBINED (105.02) scores and surpass all end-to-end competitors on the MultiWOZ leaderboard. The decreasing BLEU score is unavoidable. Due to sampling, the generations may deviate from the fixed reference responses. The responses, however, are still accurate and fluent. Both training paradigms (DPO/SFT) show improvements. In fact, best results are reached with a combination (DPO-SFT).

**SUIT identifies the most helpful training data.** Training with all successful dialogs, even though comprising a much larger set (31586 vs. 4838 for SUIT<sub>1</sub> and 2493/2166 for SUIT<sub>2</sub>), performs worse than training with relevant subgoals only. This shows the quality degradation of training

<sup>4</sup><https://huggingface.co/google/flan-t5-large>

Method	BLEU	INFORM	SUCCESS	COMBINED	#samples
MARS	19.90	88.9	78.0	103.4	-
KRLS	19.00	89.2	80.3	103.8	-
DIACTOD	17.50	89.5	84.2	104.4	-
SUIT <sub>0</sub> (initial SFT)	<b>19.94</b>	80.4	72.5	96.39	-
+ all subgoals (SFT)	19.50	87.0	79.4	102.70	31586
+ all subgoals (DPO)	17.79	86.9	80.6	101.54	31586
SUIT <sub>1</sub> (SFT)	17.75	89.8	84.0	104.65	4838
SUIT <sub>1</sub> (DPO)	17.44	88.5	82.7	103.04	4838
SUIT <sub>2</sub> (SFT-SFT)	15.11	89.7	85.9	102.91	2493
SUIT <sub>2</sub> (SFT-DPO)	17.17	89.5	84.4	104.12	2493
SUIT <sub>2</sub> (DPO-SFT)	16.47	<b>90.0</b>	<b>87.1</b>	<b>105.02</b>	2166
SUIT <sub>2</sub> (DPO-DPO)	16.92	88.8	84.4	103.52	2166

Table 5.9: SUIT results compared to other SOTA systems on MultiWOZ 2.2. SUIT<sub>0</sub> denotes the initial SUIT model, SUIT<sub>1</sub>/SUIT<sub>2</sub> the model trained for one/two iterations; the used training method per iteration in (·), either SFT or DPO.

Model ↓ / Domain →	INFORM					SUCCESS				
	train	attraction	restaurant	taxi	hotel	train	attraction	restaurant	taxi	hotel
SUIT <sub>0</sub>	94.5	84.3	88.3	100.0	83.0	78.8	68.9	75.3	72.8	74.9
SUIT <sub>1</sub> (SFT)	92.1	97.5	96.3	100.0	89.8	86.7	85.4	87.2	86.7	83.5
SUIT <sub>2</sub> (DPO-SFT)	92.3	97.2	96.1	100.0	90.4	86.7	89.4	89.2	89.7	84.0

Table 5.10: Domain-wise results for INFORM and SUCCESS metrics of best SUIT model per iteration.

data, when not selecting relevant subgoals carefully and confirms the benefits of SUIT’s approach (Section 5.2.2.2).

**Sampling new dialogs greatly improves the initial model SUIT<sub>0</sub>.** The INFORM metric improves from 80.4 to 89.8 and SUCCESS from 72.5 to 84.0 after training for one iteration. A second iteration further improves SUCCESS from 84.0 up to 87.1. After the second iteration, the COMBINED score is not increasing any further, thus we stop the training process.

#### 5.2.4.2 In-depth Analysis

**Performance improvements of SUIT across domains.** Table 5.10 shows the domain-wise results for INFORM and SUCCESS of our best SUIT models (per iteration). SUIT performs well on all domains. While results for the train domain are already quite high for SUIT<sub>0</sub>, results of the other domains are improved a lot over iterations. Especially, there are large jumps for the attraction domain (INFORM from 84.3 to 97 and SUCCESS from 68.9 to 89.4).

**Analysis of SUIT’s subgoal selection.** Table 5.11 shows some statistics on sampling new dialogs in SUIT. For each iteration, half of the user goals (4218) in the train set are sampled. Then, 5 new

sampled from	# Goals	# Dialogs		# Successful Dialogs per Goal					
		# suc.	# unsuc.	0	1	2	3	4	5
SUIT <sub>0</sub>	4218	7720	13370	1510	590	599	593	477	449
SUIT <sub>1</sub> (SFT)	4218	11983	9107	1212	212	248	407	641	1498
SUIT <sub>1</sub> (DPO)	4218	11831	9259	1278	242	236	356	481	1625

Table 5.11: Sampling statistics for SUIT models.

sampled from	# Goals	# Subgoal Samples	
		states	actions/responses
SUIT <sub>0</sub>	2238	836	4002
SUIT <sub>1</sub> (SFT)	1461	617	1876
SUIT <sub>1</sub> (DPO)	1278	746	1420

Table 5.12: Subgoal training data statistics for SUIT models.

dialogs are created for each sampled user goal, resulting in 21090 newly generated dialogs in total (see amount of dialogs evaluated as succ/unsucc in the table). On the right-hand side of Table 5.11, the number of user goals, for which there were 0/5 successful dialogs, 1/5 successful dialog, ..., up to 5 out of 5 successful dialogs, is shown. Entries in 1-4, which contain at least one successful and one unsuccessful dialog, are the candidates used in SUIT, since our approach requires comparison between successful and unsuccessful dialogs for the same user goal.

Table 5.12 shows statistics about the new subgoal-based training data obtained by our approach. The number of different user goals present in the new train data corresponds to goals where we could determine relevant subgoals with our method (taken from the candidate pool of 1 – 4 successful dialogs per goal from Table 5.11). This number is split into subgoals representing states and subgoals representing action/response generations. We also found that the most relevant turns (turns containing relevant subgoals) are turns 2-5 and that the top-5 relevant dialog actions are: INFORM-NAME( $\cdot$ ), INFORM-REFERENCE( $\cdot$ ), INFORM-PRICE( $\cdot$ ), INFORM-AREA( $\cdot$ ), INFORM-PHONE( $\cdot$ ).

Table 5.13, at the end of this chapter, shows excerpts from sampled dialogs where SUIT detected relevant subgoals, along with the negative example used for DPO. Highlighted in green, one can see the relevant information from a subgoal that contributes in reaching dialog success, whereas in red the information from the unsuccessful dialog is shown, that changed the evaluation of the successful dialog when used as a replacement.

### 5.2.5 Related Work

**End-to-end ToD Systems.** Traditional ToD systems consists of multiple task-specific models. SimpleTOD [Hosseini-Asl et al.(2020)], the first end-to-end LM-based ToD model, optimizes all sub-tasks jointly using causal language modeling. Further works uses specialized losses (MT-TOD [Lee(2021)], Mars [Sun et al.(2023)]) or focus on special encoders (DiactTOD [Wu et al.(2023)]) or learn adapters for the individual tasks (TOATOD [Bang et al.(2023)]). KRLS [Yu et al.(2023)], CASPI [Ramachandran et al.(2022)], CALM [Snell et al.(2022)] and RewardNet [Feng et al.(2023)]

apply RL with special reward functions. GALAXY [He et al.(2022)] applies semi-supervised learning and in [Steindl et al.(2024)] a data augmentation approach by mixing existing dialogs is proposed, whereas our sampling-based approach additionally enriches lexical variety and carefully selects the most helpful training data using distant supervision.

**Preference Learning for ToD Systems.** Reinforcement Learning from Human Feedback (RLHF) has become a popular paradigm to improve LLM-based models [Kaufmann et al.(2025)]. Prior work on task-oriented dialogs used preferences for learning reward functions and then apply classical RL methods to update the model, using the learned reward [Sugiyama et al.(2012), Ramachandran et al.(2022), Feng et al.(2023), Liu et al.(2025)]. In this work, we contrast successful and unsuccessful subgoals and use Direct Preference Optimization [Rafailov et al.(2023)] for improving our model, without the need of an explicit reward model and without feedback from users or other systems.

**Learning from Dialog-level Feedback.** Learning local rewards from global dialog-session based rewards was explored in [Snell et al.(2022), Hu et al.(2023), Lee et al.(2025)]. Different strategies are applied to learn from the sparse dialog-level signals: [Hu et al.(2023)] takes future dialog actions into account to estimate the value of the current one, [Snell et al.(2022)] performs task relabeling to not only learn from successful but also unsuccessful dialogs, and [Lee et al.(2025)] uses LLMs to decompose global, session-level feedback. In contrast, SUI uses distant supervision and counterfactual learning to determine relevance of intermediate turns.

## 5.2.6 Conclusion

SUI is an iterative training approach for ToD systems, which couples sampling to derive new dialogs, with distant supervision and counterfactual learning to determine subgoals that impact the final dialog success. This coupling enables SUI models to improve dialog success metrics and advance the state-of-the-art on a popular ToD benchmark.

The SUI method uses sampling to create a variety of surface forms that are candidates for further training and are scored using delayed, performance-based feedback, just like in PRAISE. Yet, the main focus in PRAISE was to create weakly-labeled training data for intermediate subtasks for which no supervision data is available. SUI’s counterfactual learning approach rather has the purpose of filtering/refining training data over multiple iterations: only the sampled generations of subgoals that matter for dialog success are retained.

---

*Domain: Hotel*

**Context:** **User:** Hello! Can you tell me about places to stay in the north area of town? I'll be on a business trip, so I do need free wifi. **System:** I have several options that meet your needs. I would recommend the Avalon, unless you need free parking.

**User:** I do not need parking, is the Avalon moderately priced?

---

**Positive Action:** *booking hotel inform NAME; inform PRICE;*

**Negative Action:** *booking hotel inform PRICE; inform AREA; inform COUNT;*

**Positive Response:** *NAME is PRICE. would you like me to book it for you?*

**Negative Response:** *very good! it is in the AREA and has COUNT stars. would you like me to book it for you? missing PRICE*

---

*Domain: Attraction*

**Context:** **User:** Hi, I am planning my Cambridge trip and could use some help with a particular attraction. **System:** I sure can, what is the name of the attraction you are seeking information about?

**User:** I'm looking for tenpin I want the address and entrance fee.

---

**Positive Action:** *attraction inform ADDRESS; inform PRICE; inform NAME; inform POST; general*

**Negative Action:** *attraction inform AREA; inform PRICE; inform NAME; general*

**Positive Response:** *NAME is a nearby attraction, admission is PRICE, and they are located at ADDRESS, postcode POST. is there anything else?*

**Negative Response:** *NAME is in AREA and it's PRICE to enter. can i help you with anything else? missing ADDRESS*

---

*Domain: Train*

**Context:** ... **User:** Okay, sounds good. Also I need to get a train from Liverpool street to Cambridge. **System:** What day would you like to make this trip? Do you have any other specifications?

**User:** The train should leave after 12:45 and should leave on friday. for 2 people please book.

---

**Positive State:** *train departure: london liverpool street; destination: cambridge;*

**Negative State:** *train departure: cambridge; destination: london liverpool street; swapped departure and destination*

---

*Domain: Restaurant*

**Context:** []

**User:** I'm looking for a restaurant with Mediterranean food.

---

**Positive Action:** *booking restaurant inform AREA; inform COUNT; inform FOOD; inform NAME; inform PRICE;*

**Negative Action:** *restaurant inform COUNT;*

**Positive Response:** *i have COUNT PRICE FOOD restaurants in the AREA. NAME and NAME. would you like me to book one for you?*

**Negative Response:** *there are COUNT. for booking do you have a preferred information is there a specific time of the day or time of day you would like to arrive by? noisy/less concrete*

---

*Domain: Taxi*

**Context:** **User:** I am traveling to Cambridge and excited about seeing location attractions. Could you help me find a place to go, like a college? **System:** Certainly. I have many available. There is corpus christi college, fore example ... **User:** I'm also looking for a hotel in the centre. ... **System:** Your booking at university arms hotel was successful with a reference number of SIHTVV32. Is there anything else I can do for you today?

---

**User:** I need a taxi please between the 2 places. I want to leave the attraction by 2:30.

**Positive State:** taxi departure: corpus christi; destination: university arms hotel; leave is 02:30; hotel area: centre; bookday: tuesday; bookstay: 1; name: university arms hotel; attraction type: college;

**Negative State:** hotel area: centre; bookday: tuesday; bookstay: 1; name: university arms hotel; stars: 4; attraction type: college;

**Postive Action:** taxi inform PHONE; inform TYPE;

**Negative Action:** taxi request PLACE;

**Postive Response:** booking completed! booked car type: TYPE contact number: PHONE

**Negative Response:** sure, i can provide you a car if you like. where are you departing from? info from context not considered

---

Table 5.13: Dialog examples with positive/negative subgoals.

#### Belief state prediction:

*Sentence:* [C] [U] {user} [U] [S] {system} [S] [U] {user} [U] ... [U] {user} [U] [C]

*What can be extracted from this sentence?*

#### Joint action and response prediction:

*Get the response for this dialog:*

[C] [U]{user}[U] [S] {system} [S] [U] {user} [U] ... [U] {user} [U] [C] [B] {beliefstate} [B]

Figure 5.5: SUI: Prompts for belief state and action/response predictions.

# Chapter 6

## Conclusion and Outlook

### Contents

---

<b>6.1</b>	<b>Summary</b>	<b>101</b>
<b>6.2</b>	<b>Discussion</b>	<b>103</b>
6.2.1	Lessons Learned	103
6.2.2	Limitations	107
<b>6.3</b>	<b>Outlook</b>	<b>109</b>

---

### 6.1 Summary

This thesis explores methods that are able to learn from implicit and delayed feedback using reinforcement learning for effectively training conversational QA and dialog models. Table 6.2 provides a summary of the contributions.

In Chapter 3, we presented CONQUER, a ConvQA model that learns from implicit user feedback via reformulations. CONQUER is based on the assumption that a user is likely to reformulate their question after receiving a supposedly wrong system response. We train a classifier to detect if two questions are reformulations or express different intents. Based on this signal, a reward in the RL setting is defined as being positive in case of a new intent, and as negative in case of a reformulation. We detect conversational context entities coming from the current question and the conversational history and link them to the knowledge graph. These are starting points for RL agents, that predict which KG paths to take to arrive at an answer, trained using the REINFORCE algorithm. This way, a similarity function between the user’s request and relevant KG paths is learned. In experiments with simulated users, CONQUER successfully learns from rewards based on reformulations and is robust to errors from the reformulation predictor and to unexpected user behavior. We also release CONVREF, a ConvQA benchmark enhanced with reformulations collected in a user study.

In Chapter 4, we presented REIGN, an RL-based framework that selects the most helpful question reformulations as additional training data for an underlying ConvQA model, which can be perceived as model-specific data augmentation. In contrast to CONQUER, reformulations in REIGN are system-generated. We propose a reformulation taxonomy for ConvQA based on string edit distance that defines basic operations that can be applied to modify conversational questions in a systematic

CONQUER (Chapter 3)	<ul style="list-style-type: none"> <li>• A question answering method that can learn from a conversational stream in the absence of gold answers;</li> <li>• An RL model for QA with rewards based on implicit feedback in the form of question reformulations;</li> <li>• A reformulation detector that can classify a follow-up utterance as a reformulation or new intent;</li> <li>• A benchmark augmented with reformulations for ConvQA over KGs, with about 205k reformulations for the 11k conversations.</li> </ul>
REIGN (Chapter 4)	<ul style="list-style-type: none"> <li>• A novel taxonomy of question reformulations for ConvQA over KGs, based on string edit distance;</li> <li>• An RL model with Deep Q-Networks, that selects reformulations of conversational questions guided towards better QA performance;</li> <li>• About 335k conversational question reformulations of test cases in two ConvQA benchmarks, suitable for robust model evaluation;</li> <li>• The REIGN framework with reusable components that judiciously augments benchmark training tailored to specific ConvQA models.</li> </ul>
PRAISE (Chapter 5.1)	<ul style="list-style-type: none"> <li>• The PRAISE architecture for the ConvQA pipeline, with each stage optimized by learning with weakly-labeled intermediate data;</li> <li>• A sampling-based technique for automatically deriving preference data from delayed feedback based on final answers;</li> <li>• Experiments demonstrating the superior performance of PRAISE on a popular ConvQA benchmark.</li> </ul>
SUIT (Chapter 5.2)	<ul style="list-style-type: none"> <li>• An iterative training procedure for LLM-based task-oriented dialog models with automatically derived training data;</li> <li>• A distant supervision approach that uses counterfactual learning for finding relevant subgoals based on dialog-level feedback;</li> <li>• Experiments demonstrating the superior performance of SUIT on a popular ToD benchmark.</li> </ul>

Table 6.2: Summary of Contributions.

way. The REIGN framework consists of two models. The first model, the reformulation category selector (RCS), is trained using Deep Q-Learning. It determines based on the input question (along the conversation history) and the given ConvQA model, which reformulation category to apply. Reformulation categories correspond to actions and the QA performance is used as reward signal. The second model, the reformulation generator (RG), generates the reformulation based on the

selected category. It is trained using noisy distantly supervised data. Finally, the ConvQA model is trained using benchmark training data augmented with top selected reformulations from REIGN. Experiments show performance improvements for ConvQA models equipped with REIGN compared to their standalone versions. We evaluate robustness to different surface forms by creating a large-scale set of reformulations for test questions.

In Chapter 5, we presented two methods for learning from delayed feedback for intermediate subtasks. In Section 5.1, we presented PRAISE, a pipeline-based ConvQA model that consists of fine-tuned LLMs for the subtasks of question understanding, evidence filtering and answer generation. We sample from an initial model and compare each generation with respect to the resulting QA performance. Samples are preferred that contain the correct answer, generated based on relevant evidence. This way, weakly-labeled data for question understanding and evidence filtering can be inferred. Direct Preference Optimization is applied, to learn from contrasting preferred and dispreferred generations. Experiments show that PRAISE reaches new state-of-the-art performance and greatly improves upon the initial pipeline, by reaching improvements for each subtask.

In Section 5.2, we presented SUIT, an iterative training approach for LLM-based end-to-end trained task-oriented dialog models. We also sample from an initial model that was trained using supervised learning and create multiple dialogs per user goal. The model is improved further, by training with subgoals that are crucial for making the dialog successful. These relevant subgoals are obtained with a counterfactual approach, by swapping parts from successful dialogs with unsuccessful ones, and determining whether this affects the dialog-level success metrics. We then either perform supervised fine-tuning with relevant subgoals or DPO with contrastive pairs. Experiments show that SUIT reaches new state-of-the-art performance and improves upon the initial model with respect to the dialog success metrics.

## 6.2 Discussion

In the following, we reflect on lessons learned in this thesis, and point out limitations.

### 6.2.1 Lessons Learned

#### 6.2.1.1 Learning from Implicit and Delayed Feedback

In this thesis, we proposed methods that learn from implicit and delayed feedback. We have demonstrated that these signals can be used effectively for inferring training data for ConvQA and increasing model performance and robustness. Table 6.4 and Figure 6.1 summarize the different types of feedback used in this thesis, along with the goals that we aimed to accomplish by learning from them.

**CONQUER’s learning paradigm is broadly applicable** (see Chapter 3). CONQUER’s learning approach shares certain properties with the RLHF (Reinforcement Learning from Human Feedback) [Ouyang et al.(2022)] paradigm, in which LLM output is scored with a reward function trained with human preference judgments. The CONQUER model also receives rewards for its outputs, where the reformulation predictor serves as a reward model. In CONQUER, the answer correctness is assessed implicitly, based on potential reformulations. The reformulation predictor requires training data consisting of ⟨question, reformulation⟩ and ⟨question, follow-up question⟩ pairs, while in the traditional RLHF setting, humans have to rate model outputs explicitly to train a reward model.

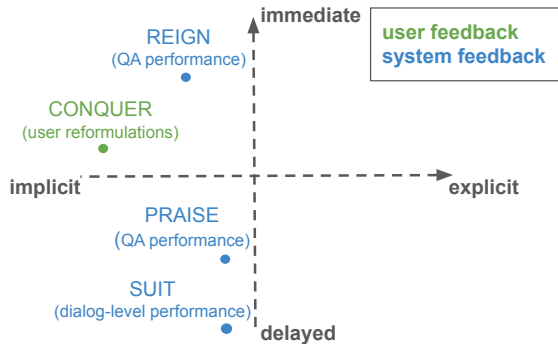


Figure 6.1: Thesis projects categorized according to the main feedback dimensions: source, expression and timing.

Model	Feedback	Feedback Categories	Goal
CONQUER	question reformulations	user, implicit, immediate (multi-step: delayed), binary	learning in the absence of gold-labeled annotations
REIGN	QA performance	system (extrinsic/intrinsic), implicit, immediate, scalar	estimating reformulation quality for data augmentation
PRAISE	QA performance	system, implicit, delayed, comparative	obtaining training data for intermediate subtasks
SUIT	dialog-level performance	system, implicit, delayed, comparative	determining relevant subgoals

Table 6.4: Summary of different types of feedback used in this thesis, and what to learn from them.

Relying on user reformulations as feedback signals has the advantage that it does not put any additional burden on the user, and such reformulations are part of the natural conversation flow.

**Implicit performance-based feedback allows finding helpful reformulations in REIGN** (see Chapter 4). The REIGN framework selects reformulations for which the initial ConvQA model achieves high answering performance as further training points. This is an implicit feedback, since achieving good performance with the initial ConvQA model does not directly mean that the reformulation is a valuable data point for further training. In initial experiments, we also considered the opposite case: selecting challenging reformulations for which the ConvQA model cannot reach the correct answer, and also mixing easy and challenging cases and applying curriculum learning [Bengio et al.(2009)] to gradually increase difficulty. However, using the reformulations for which the ConvQA model can reach an answer worked best in practice. One reason might be that the challenging ones are often noisy: they might lack relevant information, or sometimes reformulation categories have not been applied correctly. Therefore, the entity linking may already fail, making it difficult for the later stages in the QA pipeline to reach an answer, even once the ConvQA model is improved with further training data. Especially these reformulations are helpful, for which the ConvQA model

could not find a correct answer for the original question, but for the reformulation. Therefore, our reward favors reformulations that achieve better performance than the original question.

**Suitable question formulations and relevant evidence is learned implicitly in PRAISE** (see Chapter 5.1). Delayed feedback enables optimizing earlier stages in the pipeline for the later stages. Question reformulations are tailored to high recall retrieval and good answering performance. For evidence filtering, we only receive feedback for sets of evidence. We can have a mix of truly relevant and irrelevant evidence snippets per set. Obtaining QA feedback for each evidence in isolation would be too costly. Correct evidence will appear in more successful than unsuccessful sets, entailing that we implicitly learn relevant evidence (see retained answer presence in top- $k$  filtered set in Section 5.1.4).

**Delayed feedback is successfully attributed to subgoals in SUIT** (see Chapter 5.2). Swapping turns from successful dialogs with the ones from unsuccessful dialogs allows us to determine changes in dialog success. With this counterfactual approach, subgoals that contribute to the overall dialog success are detected. Focusing on training with these improves performance (see Section 5.2.4).

### 6.2.1.2 RL for Learning from Feedback

Reinforcement learning is the method to use for sequential decision-making problems with delayed feedback, while supervised learning would treat each decision in isolation and cannot consider long-term effects. Therefore, we decided to adopt the RL paradigm for our multi-step problems with delayed feedback. Table 6.6 provides a summary of the RL representations used in this thesis.

**Rewards based on task performance metrics allow for cost-efficient scoring of sampled experience.** Our rewards are based on QA performance in REIGN and PRAISE, and based on dialog success in SUIT. Since we do not evaluate new question intents, but reformulations of question intents from ConvQA benchmarks, we can make use of ground truth answers for reward calculation. We also do not define new user goals for task-oriented dialogs, and can automatically evaluate whether a dialog fulfills a specific user goal.

For CONQUER, obtaining rewards is more expensive, since we rely on user reformulations. However, our trained reformulation predictor is able to provide rewards to our model in an easy manner.

With our defined reward functions, we are able to evaluate environment interactions cost-efficiently and enable our RL agents to explore many different samples, whether these are different KG paths, reformulations, generated evidence identifiers or generated dialog turns.

**Ensuring stable RL training is challenging.** In initial experiments, we tried to learn arbitrary question modifications in REIGN and experienced model degradation. The model overfitted to the sparse reward signal and learned spurious correlations. We introduced reformulation categories that form a discrete set of actions. This way, we decreased the size of the search space. This was crucial for enabling stable training in REIGN. For stable training in CONQUER, we applied variance reduction by subtracting a baseline and performed multiple rollouts to get a good estimate of the gradient. In PRAISE and SUIT, KL regularization was applied in order to prevent the policy from changing too drastically, and thus, ensuring stability.

Another important factor for RL training is the exploration of the environment. We performed

Model	States	Actions	Rewards	Method
CONQUER	conversation history, question, context entity	KG path containing context entity	-1: reformulation, +1: new intent	policy-gradient algorithm REINFORCE
REIGN	conversation history, question	15 reformulation categories	difference in reciprocal rank of reformulation and original question	value-based Deep Q-Learning algorithm
PRAISE	conversation history, tokens generated so far	all tokens in LLM vocabulary	P@1 comparisons between samples	Direct Preference Optimization
SUIT	conversation history, tokens generated so far	all tokens in LLM vocabulary	dialog SUCCESS metric	Direct Preference Optimization

Table 6.6: Summary of RL representations used in this thesis.

different strategies, such as entropy regularization in CONQUER and different sampling strategies in REIGN and PRAISE, to achieve this. In PRAISE and SUIT, an important choice was also the number of training data per subtask and per iteration (for SUIT).

More technical details can be found in the respective chapters (see Section 3.7 for CONQUER, Section 4.6 for REIGN, Section 5.1.3 for PRAISE, and Section 5.2.3 for SUIT).

### 6.2.1.3 Training Data Selection in the Presence of Noise

In this thesis, we have seen that in the presence of noise, it is often undesirable to add large amounts of data: instead we should carefully select data for further training.

The REIGN framework selects a subset of reformulations most helpful for the underlying QA model. In experiments, we show that we obtain worse performance if the training data is augmented with all generated reformulation variants (see Table 4.7 in Section 4.7). Not all categories result in meaningful reformulations for a given question. The reformulation generator in REIGN was trained with distant supervision from rule-based annotations, resulting in some noisy generations.

The SUIT method learns which subgoals truly matter for the dialog success. Training with all turns from successful dialogs results in worse performance even though much more training data is used (see Table 5.9 in Section 5.2.4). Training with irrelevant turns may make the model more robust to different surface variations, but does not increase dialog success, since irrelevant turns add noise. Additionally, the approach filters out noise coming from wrong ground truth in the dataset.

### 6.2.1.4 Conversational Context Understanding

**Conversational context entities convey relevant information.** In CONQUER, we introduced the notion of conversational context entities, which are entities from the conversational history that are relevant for the current question. Since traditional NED tools do not work well for the conversational setting [Joko et al.(2021)], we also took the KG neighborhood into account for successfully

linking context entities to the KG (see Section 3.3). Adding additional context information to the current question by appending previous turns from the history had no large effect for CONQUER, indicating that these entities convey most relevant information about the user’s information need in QA over KGs. Adding context entities to the current question was also one of the most popular transformations selected in REIGN (see Figure 4.5 in Section 4.7.2).

**Adapting conversational question reformulations to the QA model is better than resembling human-style questions.** In REIGN we selected reformulation categories based on the individual question intent as well as the underlying QA model. This showed better performance compared to other selection choices, such as making questions more complete using insertion operations (see Line 7, Table 4.7). Question completion and question rewriting approaches, trained based on human-written reformulations, also showed worse performance (see Lines 10 – 11, Table 4.7).

In PRAISE, we also did not rely on human rewrites: The generated question reformulations are optimized for retrieval and QA performance. Question reformulations in PRAISE achieve higher performance than reformulations from LLMs in few-shot mode (see Table 5.3 in Section 5.1.4).

## 6.2.2 Limitations

### 6.2.2.1 Online Feedback

Ideally, we would like to collect feedback in an online manner and continuously improve the model. However, such data is not available and running a long-term, large-scale user study, spanning several months, would be very costly and is out of scope for this thesis. Therefore, we opted for user simulations and offline feedback in the thesis.

**User simulation in CONQUER.** We simulated user behavior in CONQUER, by triggering follow-up questions and reformulations based on previously collected data in a user study (see Section 3.7). Real user behavior, in particular with respect to the willingness to reformulate questions after receiving wrong answers multiple times, might differ from the synthesized user behavior in our study. Future work could investigate user reformulation behavior for ConvQA further, or incorporate findings from reformulation behavior in Web search. Other implicit signals, such as users abandoning the conversation after too many system mistakes, could be considered as well.

**No interleaved training of model components in REIGN.** In REIGN we evaluated reformulation quality based on the performance of a frozen ConvQA model (initialized with benchmark training data). Ideally, we would interleave the training of the reformulation category selector and the ConvQA model to take into account the improved ConvQA model performance for reward calculation. This way, we can take into account if category choices change over time. We opted for a separate training for better efficiency.

**DPO is an offline method.** In Chapter 5, we collected and scored model outputs, then we improved model components further with the newly obtained training data. Future work could apply DPO with online feedback [Guo et al.(2024)], which allows continuous model training and avoids the distribution shift that arise from training with offline data.

### 6.2.2.2 Entity-centric View

This thesis has an entity-centric view of the nature of conversational questions and answers. We follow the assumption that realistic user needs revolve around one or a few topic entities which are explored along related entities throughout a conversation (see introductory examples in Figure 1.1 and 1.2 and Section 2.1.1). ToD systems also focus on finding entities, such as a specific hotel or restaurant, that satisfy all user constraints in a dialog [Budzianowski et al.(2018)]. The ConvQA datasets used in this thesis share this assumption [Christmann et al.(2019), Christmann et al.(2022b)]. Answers consists of crisp entities, list of entities, or numeric quantities. This affects the choice of metrics used for evaluation, such as P@1 and MRR. CONQUER and REIGN both operate on KGs, for which entity-based answers are very common. CONQUER would also work on different answer granularities when meaningful graph structure is available. When answers are retrieved from text or when LLMs are used for answer generation as in PRAISE, we map answers to the closest corresponding KG entities, to allow for a canonical evaluation. However, this mapping might lead to some inaccuracies, in particular when comparing with answers generated by LLMs that typically output sentences or short phrases.

Future work could apply the presented methods to domain-specific data, such as biomedical or financial data for which entities play an important role. It would also be interesting to explore approaches on less entity-centric datasets with long-form answers. In this case, it is more challenging to derive feedback based on answer correctness, since answers are defined less clearly. For example, considering non-entity-centric conversations evolving around advice or opinion-seeking topics, such as planning a holiday. Such conversations would be more open-ended than the task-oriented dialogs we considered in this thesis. However, they would still require factoid information such as available sights of interest, available tours, distances between places, etc. Measuring dialog success in terms of user satisfaction is also more challenging than in the traditional ToD setup.

### 6.2.2.3 Training Overhead

The proposed approaches in this thesis focus on improving models at *training* time: We were able to train a ConvQA model without gold-labeled data (see CONQUER, Chapter 3) and developed different methods to improve upon initial models by inferring additional weakly-labeled training data (see REIGN, PRAISE, SUIT, Chapters 4-5). We took several steps for making our approaches efficient. For example, by applying parameter-efficient fine-tuning to update only a small amount of LLM parameters for each subtask in PRAISE (see Section 5.1.3). Nevertheless, the proposed methods come with additional training costs:

- Training RL models requires substantial interaction data (see CONQUER and REIGN);
- Multiple models in REIGN are trained sequentially, due to their dependencies: First, the RG model is trained and then applied during RL training of the RCS model. Then, both RG and RCS models are applied to augment the training data of the ConvQA model, which is then trained with the enhanced data.
- In PRAISE and SUIT, we create initial models, sample from them to construct further training data, and then use this data to improve the models further.

Note that there is no overhead at inference time, the methods are relatively fast (for example, see runtimes of CONQUER in Section 3.8.2 and for PRAISE in Section 5.1.4).

## 6.3 Outlook

**Beyond *conversational* context.** This thesis focuses on conversations with incomplete follow-up questions that can only be understood in their conversational context. Nowadays, LLMs can keep track of conversational context fairly well [Zhao et al.(2023)]. There are further forms of challenging context understanding, though. One challenge would be to enable long-term interactions, spanning multiple sessions. Conversational systems would require dedicated memory to remember crucial information from previous sessions that might be helpful for later interactions. For example, relevant information could contain information frequently searched by the user. Further forms of personalization would be possible by storing user profiles with individual preferences, interests, skill levels or certain important personal events. Harvesting the situational context of the interaction, such as the user’s location, the device and current time, could also be beneficial to best adapt the model’s output to the user’s current situation. For example, on phone screens, output should be very concise. When driving to work, content such as news, weather, traffic situation should be provided, whereas on weekends latest sport news and information about movies, etc. are more desirable.

**Combining different types of feedback.** This thesis focuses on implicit and delayed feedback mechanisms. In CONQUER, we explored implicit user feedback, while in the rest of the thesis we considered feedback based on end-to-end performance metrics. Future work could combine different types of feedback:

- User feedback is crucial to adapt systems to user needs. Other implicit forms, such as interaction frequency and duration, or more explicit cues, such as user satisfaction expressed with thumbs up/down or “thank you” messages;
- Further implicit feedback that enable automatic evaluations based on system performance, such as task-completion rate or speed;
- Other system-based feedback from the environment, for example, based on interactions with other LLMs or agents;
- The model’s intrinsic feedback, for example, based on information gain, or curiosity-driven rewards [Pathak et al.(2017)] based on prediction errors, in order to focus on shortcomings of the current model.

These signals greatly vary in availability, frequency, range, expressiveness and noise. Combining them in meaningful ways is a challenging task.

**Reinforcement Learning for LLMs.** Reinforcement learning has been applied in this thesis to learn from different types of feedback. RL has also been successfully used in the context of LLMs, mainly as a further round of fine-tuning to align its outputs with human preferences [Ouyang et al.(2022)] and to improve model capabilities. However, model outputs are typically optimized for

the next turn only. Looking at each turn in isolation may lead to suboptimal conversations. Ideally, we want to optimize for long-term interaction, for example similar to the work in [Wu et al.(2025)].

Another challenge that arises when evaluating entire dialogs is appropriate credit assignment of delayed rewards [Harutyunyan et al.(2019)]. In SUT, we aimed to identify turns that contributed to the final dialog success. However, we did not distinguish between the turns that were considered relevant. In reality, some user goals might be more important than others and some turns might fulfill multiple goals. Therefore, not all turns should be treated equally. Performing effective credit assignment of final rewards to previous steps where not all steps contributed equally, is challenging but also beneficial so that the model learns to focus on what matters.

This thesis investigated applying reinforcement learning to conversational question answering for enabling models to learn from implicit feedback. It opens up new avenues for research on learning from feedback for continuous model improvement.

# List of Figures

1.1	Conversational QA Example . . . . .	2
1.2	Task-oriented Dialog Example . . . . .	2
2.1	Example of a Knowledge Graph Snippet . . . . .	13
2.2	Example of a System Turn in a Task-oriented Dialog . . . . .	16
2.3	Categorization of Exemplary Feedback Signals . . . . .	19
2.4	Agent-Environment Interaction in a Markov Decision Process . . . . .	21
2.5	Gridworld Example with Optimal Value and Optimal Policy Functions . . . . .	22
3.1	CONQUER: KG Excerpt from Wikidata . . . . .	33
3.2	CONQUER: KG Representation with Qualifiers . . . . .	35
3.3	CONQUER: Workflow Overview . . . . .	37
3.4	CONQUER: Architecture of Policy Network . . . . .	39
4.1	REIGN: Performance-guided Reformulation Generation . . . . .	55
4.2	REIGN: Workflow Overview . . . . .	58
4.3	REIGN: Taxonomy of Reformulation Categories . . . . .	59
4.4	REIGN: Question Transformations . . . . .	64
4.5	REIGN: Category Predictions by Deep Q-Network . . . . .	69
5.1	PRAISE: Workflow Overview . . . . .	78
5.2	PRAISE: Prompts for QU, ERF and AG . . . . .	88
5.3	PRAISE: Prompt for Few-shot QU . . . . .	89
5.4	SUIT: Workflow Overview . . . . .	92
5.5	SUIT: Prompts . . . . .	100
6.1	Thesis Projects in Feedback Dimensions . . . . .	104



# List of Tables

2.1	Evaluation Metrics for ConvQA and Task-oriented Dialogs . . . . .	18
3.1	CONQUER: Notation for Salient Concepts . . . . .	36
3.2	CONQUER: Types of Reformulations in CONVREF . . . . .	42
3.3	CONQUER: Sample Reformulations from CONVREF . . . . .	43
3.4	CONQUER: Main Results . . . . .	46
3.5	CONQUER: Domain-wise Results . . . . .	46
3.6	CONQUER: Performance on CONVQUESTIONS . . . . .	47
3.7	CONQUER: Ablation for Context Entity Detection . . . . .	48
3.8	CONQUER: Effect of Context Models . . . . .	48
3.9	CONQUER: Classification Performance for Reformulations . . . . .	49
3.10	CONQUER: Action Choices for Policy Learning . . . . .	49
4.1	REIGN: Notation for Salient Concepts . . . . .	57
4.2	REIGN: Benchmark Statistics . . . . .	66
4.3	REIGN: Main Results on CONVMIX . . . . .	67
4.4	REIGN: Main Results on CONVQUESTIONS . . . . .	67
4.5	REIGN: Domain-wise Results . . . . .	68
4.6	REIGN: Turn-wise Results . . . . .	68
4.7	REIGN: Effects of Design Choices . . . . .	69
4.8	REIGN: Examples of GPT-generated Reformulations . . . . .	72
4.9	REIGN: Examples of REIGN-generated Reformulations . . . . .	73
5.1	PRAISE: Notation for Salient Concepts . . . . .	79
5.2	PRAISE: Main Results . . . . .	83
5.3	PRAISE: Effect of Different Pipeline Components . . . . .	83
5.4	PRAISE: Domain-wise Results . . . . .	84
5.5	PRAISE: Turn-wise Results . . . . .	84
5.6	PRAISE: Example Outputs per Stage . . . . .	87
5.7	SUIT: Notation for Salient Concepts . . . . .	91
5.8	SUIT: Benchmark Statistics . . . . .	95
5.9	SUIT: Main Results . . . . .	96
5.10	SUIT: Domain-wise Results . . . . .	96
5.11	SUIT: Sampling Statistics . . . . .	97

5.12	SUIT: Subgoal Training Data Statistics . . . . .	97
5.13	SUIT: Example Dialogs with Subgoals . . . . .	100
6.2	Summary of Contributions . . . . .	102
6.4	Summary of Types of Feedback . . . . .	104
6.6	Summary of RL Representations . . . . .	106

# List of Algorithms

1	CONQUER: Policy Learning . . . . .	40
2	REIGN: Deep Q-Learning . . . . .	62
3	PRAISE: Training Approach . . . . .	80
4	SUIT: Training Approach . . . . .	93



# Bibliography

- [Abbasiantaeb et al.(2024)] Zahra Abbasiantaeb, Yifei Yuan, Evangelos Kanoulas, and Mohammad Aliannejadi. 2024. Let the LLMs Talk: Simulating Human-to-Human Conversational QA via Zero-Shot LLM-to-LLM Interactions. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*. ACM, 8–17. <https://doi.org/10.1145/3616855.3635856>
- [Abujabal et al.(2017)] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2017. QUINT: Interpretable Question Answering over Knowledge Bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017 - System Demonstrations*. Association for Computational Linguistics, 61–66. <https://doi.org/10.18653/V1/D17-2011>
- [Abujabal et al.(2018)] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2018. Never-Ending Learning for Open-Domain Question Answering over Knowledge Bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. ACM, 1053–1062. <https://doi.org/10.1145/3178876.3186004>
- [Abujabal et al.(2019)] Abdalghani Abujabal, Rishiraj Saha Roy, Mohamed Yahya, and Gerhard Weikum. 2019. ComQA: A Community-sourced Dataset for Complex Factoid Question Answering with Paraphrase Clusters. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. 307–317. <https://doi.org/10.18653/V1/N19-1027>
- [Adlakha et al.(2022)] Vaibhav Adlakha, Shehzaad Dhuliawala, Kaheer Suleman, Harm de Vries, and Siva Reddy. 2022. TopiOCQA: Open-domain Conversational Question Answering with Topic Switching. *Trans. Assoc. Comput. Linguistics* 10 (2022), 468–483. [https://doi.org/10.1162/TACL\\_A\\_00471](https://doi.org/10.1162/TACL_A_00471)
- [Amplayo et al.(2022)] Reinald Kim Amplayo, Kellie Webster, Michael Collins, Dipanjan Das, and Shashi Narayan. 2022. Query Refinement Prompts for Closed-Book Long-Form Question Answering. *CoRR* abs/2210.17525 (2022). <https://doi.org/10.48550/ARXIV.2210.17525> arXiv:2210.17525
- [Anantha et al.(2021)] Raviteja Anantha, Svitlana Vakulenko, Zhucheng Tu, Shayne Longpre, Stephen Pulman, and Srinivas Chappidi. 2021. Open-Domain Question Answering Goes Con-

- versational via Question Rewriting. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*. Association for Computational Linguistics, 520–534. <https://doi.org/10.18653/V1/2021.NAACL-MAIN.44>
- [Auer et al.(2007)] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007 (Lecture Notes in Computer Science, Vol. 4825)*. Springer, 722–735. [https://doi.org/10.1007/978-3-540-76298-0\\_52](https://doi.org/10.1007/978-3-540-76298-0_52)
- [Awadallah et al.(2013)] Ahmed Hassan Awadallah, Xiaolin Shi, Nick Craswell, and Bill Ramsey. 2013. Beyond clicks: query reformulation as a predictor of search satisfaction. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*. ACM, 2019–2028. <https://doi.org/10.1145/2505515.2505682>
- [Baheti et al.(2020)] Ashutosh Baheti, Alan Ritter, and Kevin Small. 2020. Fluent Response Generation for Conversational Question Answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 191–207. <https://doi.org/10.18653/V1/2020.ACL-MAIN.19>
- [Bang et al.(2023)] Namoo Bang, Jeehyun Lee, and Myoung-Wan Koo. 2023. Task-Optimized Adapters for an End-to-End Task-Oriented Dialogue System. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 7355–7369. <https://doi.org/10.18653/V1/2023.FINDINGS-ACL.464>
- [Bartolo et al.(2021)] Max Bartolo, Tristan Thrush, Robin Jia, Sebastian Riedel, Pontus Stenetorp, and Douwe Kiela. 2021. Improving Question Answering Model Robustness with Synthetic Adversarial Data Generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 8830–8848. <https://doi.org/10.18653/V1/2021.EMNLP-MAIN.696>
- [Bast and Haussmann(2015)] Hannah Bast and Elmar Haussmann. 2015. More Accurate Question Answering on Freebase. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. ACM, 1431–1440. <https://doi.org/10.1145/2806416.2806472>
- [Bengio et al.(2009)] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009 (ACM International Conference Proceeding Series, Vol. 382)*. ACM, 41–48. <https://doi.org/10.1145/1553374.1553380>
- [Berant et al.(2013)] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on*

- Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA*. ACL, 1533–1544. <https://doi.org/10.18653/V1/D13-1160>
- [Berant and Liang(2014)] Jonathan Berant and Percy Liang. 2014. Semantic Parsing via Paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*. The Association for Computer Linguistics, 1415–1425. <https://doi.org/10.3115/V1/P14-1133>
- [Bhutani et al.(2019)] Nikita Bhutani, Xinyi Zheng, and H. V. Jagadish. 2019. Learning to Answer Complex Questions over Knowledge Bases with Query Composition. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 739–748. <https://doi.org/10.1145/3357384.3358033>
- [Bradley and Terry(1952)] Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika* 39, 3/4 (1952), 324–345.
- [Buck et al.(2018)] Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2018. Ask the Right Questions: Active Question Reformulation with Reinforcement Learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=S1CChZ-CZ>
- [Budzianowski et al.(2018)] Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. MultiWOZ - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 5016–5026. <https://aclanthology.org/D18-1547/>
- [Campos et al.(2020)] Jon Ander Campos, Kyunghyun Cho, Arantxa Otegi, Aitor Soroa, Eneko Agirre, and Gorka Azkune. 2020. Improving Conversational Question Answering Systems after Deployment using Feedback-Weighted Learning. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*. International Committee on Computational Linguistics, 2561–2571. <https://doi.org/10.18653/V1/2020.COLING-MAIN.230>
- [Chang et al.(2006)] Youjin Chang, Iadh Ounis, and Minkoo Kim. 2006. Query reformulation using automatically generated query concepts from a document space. *Inf. Process. Manag.* 42, 2 (2006), 453–468. <https://doi.org/10.1016/J.IPM.2005.03.025>
- [Chen et al.(2021)] Derek Chen, Howard Chen, Yi Yang, Alexander Lin, and Zhou Yu. 2021. Action-Based Conversations Dataset: A Corpus for Building More In-Depth Task-Oriented Dialogue Systems. In *Proceedings of the 2021 Conference of the North American Chapter*

- of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021.* Association for Computational Linguistics, 3002–3017. <https://doi.org/10.18653/V1/2021.NAACL-MAIN.239>
- [Chen et al.(2017)] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers.* Association for Computational Linguistics, 1870–1879. <https://doi.org/10.18653/V1/P17-1171>
- [Chen et al.(2020)] Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2020. GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020.* ijcai.org, 1230–1236. <https://doi.org/10.24963/IJCAI.2020/171>
- [Chen et al.(2022a)] Zhiyu Chen, Bing Liu, Seungwhan Moon, Chinnadhurai Sankar, Paul A. Crook, and William Yang Wang. 2022a. KETOD: Knowledge-Enriched Task-Oriented Dialogue. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022.* Association for Computational Linguistics, 2581–2593. <https://doi.org/10.18653/V1/2022.FINDINGS-NAACL.197>
- [Chen et al.(2022b)] Zhiyu Chen, Jie Zhao, Anjie Fang, Besnik Fetahu, Oleg Rokhlenko, and Shervin Malmasi. 2022b. Reinforced Question Rewriting for Conversational Question Answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: EMNLP 2022 - Industry Track, Abu Dhabi, UAE, December 7 - 11, 2022.* Association for Computational Linguistics, 357–370. <https://doi.org/10.18653/V1/2022.EMNLP-INDUSTRY.36>
- [Choi et al.(2018)] Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question Answering in Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018.* Association for Computational Linguistics, 2174–2184. <https://doi.org/10.18653/V1/D18-1241>
- [Christmann et al.(2019)] Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look before you Hop: Conversational Question Answering over Knowledge Graphs Using Judicious Context Expansion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019.* ACM, 729–738. <https://doi.org/10.1145/3357384.3358016>
- [Christmann et al.(2022a)] Philipp Christmann, Rishiraj Saha Roy, and Gerhard Weikum. 2022a. Beyond NED: Fast and Effective Search Space Reduction for Complex Question Answering over Knowledge Bases. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022.* ACM, 172–180. <https://doi.org/10.1145/3488560.3498488>
- [Christmann et al.(2022b)] Philipp Christmann, Rishiraj Saha Roy, and Gerhard Weikum. 2022b. Conversational Question Answering on Heterogeneous Sources. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval,*

- Madrid, Spain, July 11 - 15, 2022*. ACM, 144–154. <https://doi.org/10.1145/3477495.3531815>
- [Christmann et al.(2023)] Philipp Christmann, Rishiraj Saha Roy, and Gerhard Weikum. 2023. Explainable Conversational Question Answering over Heterogeneous Sources via Iterative Graph Neural Networks. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 643–653. <https://doi.org/10.1145/3539618.3591682>
- [Cui et al.(2024)] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. ULTRAFEEDBACK: Boosting Language Models with Scaled AI Feedback. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. <https://openreview.net/forum?id=B0orDpKHj>
- [Dalton et al.(2022)] Jeffrey Dalton, Sophie Fischer, Paul Owoicho, Filip Radlinski, Federico Rossetto, Johanne R. Trippas, and Hamed Zamani. 2022. Conversational Information Seeking: Theory and Application. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 3455–3458. <https://doi.org/10.1145/3477495.3532678>
- [Dalton et al.(2020)] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2020. TREC CAsT 2019: The Conversational Assistance Track Overview. *CoRR* abs/2003.13624 (2020). arXiv:2003.13624 <https://arxiv.org/abs/2003.13624>
- [Dalton et al.(2021)] Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2021. TREC CAsT 2021: The Conversational Assistance Track Overview. In *Proceedings of the Thirtieth Text REtrieval Conference, TREC 2021, online, November 15-19, 2021 (NIST Special Publication, Vol. 500-335)*. National Institute of Standards and Technology (NIST). <https://trec.nist.gov/pubs/trec30/papers/Overview-CAsT.pdf>
- [Dang and Croft(2010)] Van Dang and W. Bruce Croft. 2010. Query reformulation using anchor text. In *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010*. ACM, 41–50. <https://doi.org/10.1145/1718487.1718493>
- [Das et al.(2019)] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step Retriever-Reader Interaction for Scalable Open-domain Question Answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. <https://openreview.net/forum?id=HkfPSh05K7>
- [Das et al.(2018)] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. <https://openreview.net/forum?id=Syg-YfWCW>

- [Das et al.(2017)] Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question Answering on Knowledge Bases and Text using Universal Schema and Memory Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*. Association for Computational Linguistics, 358–365. <https://doi.org/10.18653/V1/P17-2057>
- [Deng et al.(2022)] Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. 2022. PACIFIC: Towards Proactive Conversational Question Answering over Tabular and Textual Data in Finance. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 6970–6984. <https://doi.org/10.18653/V1/2022.EMNLP-MAIN.469>
- [Devlin et al.(2019)] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 4171–4186. <https://doi.org/10.18653/V1/N19-1423>
- [Dhingra et al.(2017)] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, 484–495. <https://doi.org/10.18653/V1/P17-1045>
- [Dietz et al.(2018)] Laura Dietz, Ben Gamari, Jeff Dalton, and Nick Craswell. 2018. TREC Complex Answer Retrieval Overview. In *Proceedings of the Twenty-Seventh Text REtrieval Conference, TREC 2018, Gaithersburg, Maryland, USA, November 14-16, 2018 (NIST Special Publication, Vol. 500-331)*. National Institute of Standards and Technology (NIST). <https://trec.nist.gov/pubs/trec27/papers/Overview-CAR.pdf>
- [Don-Yehiya et al.(2024)] Shachar Don-Yehiya, Leshem Choshen, and Omri Abend. 2024. Naturally occurring feedback is common, extractable and useful. *CoRR* abs/2407.10944 (2024). arXiv:2407.10944 <https://arxiv.org/abs/2407.10944>
- [Dong et al.(2017)] Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Association for Computational Linguistics, 875–886. <https://doi.org/10.18653/V1/D17-1091>
- [Dubey et al.(2019)] Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. LC-QuAD 2.0: A Large Dataset for Complex Question Answering over Wikidata and DBpedia. In *The Semantic Web - ISWC 2019 - 18th International Semantic*

- Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 11779)*. Springer, 69–78. [https://doi.org/10.1007/978-3-030-30796-7\\_5](https://doi.org/10.1007/978-3-030-30796-7_5)
- [Elgohary et al.(2019)] Ahmed Elgohary, Denis Peskov, and Jordan L. Boyd-Graber. 2019. Can You Unpack That? Learning to Rewrite Questions-in-Context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 5917–5923. <https://doi.org/10.18653/V1/D19-1605>
- [Fader et al.(2013)] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-Driven Learning for Open Question Answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*. The Association for Computer Linguistics, 1608–1618. <https://aclanthology.org/P13-1158/>
- [Fader et al.(2014)] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. ACM, 1156–1165. <https://doi.org/10.1145/2623330.2623677>
- [Faustini et al.(2023)] Pedro Faustini, Zhiyu Chen, Besnik Fetahu, Oleg Rokhlenko, and Shervin Malmasi. 2023. Answering Unanswered Questions through Semantic Reformulations in Spoken QA. In *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 729–743. <https://doi.org/10.18653/V1/2023.ACL-INDUSTRY.70>
- [Feng et al.(2023)] Yihao Feng, Shentao Yang, Shujian Zhang, Jianguo Zhang, Caiming Xiong, Mingyuan Zhou, and Huan Wang. 2023. Fantastic Rewards and How to Tame Them: A Case Study on Reward Learning for Task-oriented Dialogue Systems. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. <https://openreview.net/forum?id=086pmarAris>
- [Ferragina and Scaiella(2010)] Paolo Ferragina and Ugo Scaiella. 2010. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*. ACM, 1625–1628. <https://doi.org/10.1145/1871437.1871689>
- [Formal et al.(2021)] Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. 2021. SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 2288–2292. <https://doi.org/10.1145/3404835.3463098>
- [Galkin et al.(2020)] Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. 2020. Message Passing for Hyper-Relational Knowledge Graphs. In *Proceedings*

- of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020. Association for Computational Linguistics, 7346–7359. <https://doi.org/10.18653/V1/2020.EMNLP-MAIN.596>
- [Gan and Ng(2019)] Wee Chung Gan and Hwee Tou Ng. 2019. Improving the Robustness of Question Answering Systems to Question Paraphrasing. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 6065–6075. <https://doi.org/10.18653/V1/P19-1610>
- [Gao et al.(2023a)] Ge Gao, Hung-Ting Chen, Yoav Artzi, and Eunsol Choi. 2023a. Continually Improving Extractive QA via Human Feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 406–423. <https://doi.org/10.18653/V1/2023.EMNLP-MAIN.27>
- [Gao et al.(2022)] Ge Gao, Eunsol Choi, and Yoav Artzi. 2022. Simulating Bandit Learning from User Feedback for Extractive Question Answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 5167–5179. <https://doi.org/10.18653/V1/2022.ACL-LONG.355>
- [Gao et al.(2023b)] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023b. Retrieval-Augmented Generation for Large Language Models: A Survey. *CoRR* abs/2312.10997 (2023). <https://doi.org/10.48550/ARXIV.2312.10997> arXiv:2312.10997
- [Gekhman et al.(2023)] Zorik Gekhman, Nadav Oved, Orgad Keller, Idan Szpektor, and Roi Reichart. 2023. On the Robustness of Dialogue History Representation in Conversational Question Answering: A Comprehensive Study and a New Prompt-based Method. *Trans. Assoc. Comput. Linguistics* 11 (2023), 351–366. [https://doi.org/10.1162/TACL\\_A\\_00549](https://doi.org/10.1162/TACL_A_00549)
- [Gottardi et al.(2022)] Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Somani, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tur, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. 2022. Alexa, Let’s Work Together: Introducing the First Alexa Prize TaskBot Challenge on Conversational Task Assistance. *CoRR* abs/2209.06321 (2022). <https://doi.org/10.48550/ARXIV.2209.06321> arXiv:2209.06321
- [Guo et al.(2018)] Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2018. Dialog-to-Action: Conversational Question Answering Over a Large-Scale Knowledge Base. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. 2946–2955. <https://proceedings.neurips.cc/paper/2018/hash/d63fbf8c3173730f82b150c5ef38b8ff-Abstract.html>

- [Guo et al.(2024)] Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Ramé, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. 2024. Direct Language Model Alignment from Online AI Feedback. *CoRR* abs/2402.04792 (2024). <https://doi.org/10.48550/ARXIV.2402.04792> arXiv:2402.04792
- [Gupta and Sharma(2021)] Somil Gupta and Neeraj Sharma. 2021. Role of Attentive History Selection in Conversational Information Seeking. *CoRR* abs/2102.03749 (2021). arXiv:2102.03749 <https://arxiv.org/abs/2102.03749>
- [Guu et al.(2020)] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval Augmented Language Model Pre-Training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 3929–3938. <http://proceedings.mlr.press/v119/guu20a.html>
- [Hai et al.(2023)] Nam Le Hai, Thomas Gerald, Thibault Formal, Jian-Yun Nie, Benjamin Piwowarski, and Laure Soulier. 2023. CoSPLADE: Contextualizing SPLADE for Conversational Information Retrieval. In *Advances in Information Retrieval - 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2-6, 2023, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 13980)*. Springer, 537–552. [https://doi.org/10.1007/978-3-031-28244-7\\_34](https://doi.org/10.1007/978-3-031-28244-7_34)
- [Hancock et al.(2019)] Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazaré, and Jason Weston. 2019. Learning from Dialogue after Deployment: Feed Yourself, Chatbot!. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 3667–3684. <https://doi.org/10.18653/V1/P19-1358>
- [Harutyunyan et al.(2019)] Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Rémi Munos. 2019. Hindsight Credit Assignment. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. 12467–12476. <https://proceedings.neurips.cc/paper/2019/hash/195f15384c2a79cedf293e4a847ce85c-Abstract.html>
- [He et al.(2022)] Wanwei He, Yinpei Dai, Yinhe Zheng, Yuchuan Wu, Zheng Cao, Dermot Liu, Peng Jiang, Min Yang, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2022. GALAXY: A Generative Pre-trained Model for Task-Oriented Dialog with Semi-supervised Learning and Explicit Policy Injection. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 10749–10757. <https://doi.org/10.1609/AAAI.V36I10.21320>
- [Hermjakob et al.(2002)] Ulf Hermjakob, Abdessamad Echihabi, and Daniel Marcu. 2002. Natural Language Based Reformulation Resource and Wide Exploitation for Question Answering. In

- Proceedings of The Eleventh Text REtrieval Conference, TREC 2002, Gaithersburg, Maryland, USA, November 19-22, 2002 (NIST Special Publication, Vol. 500-251)*. National Institute of Standards and Technology (NIST). <http://trec.nist.gov/pubs/trec11/papers/usc.hermjakob.pdf>
- [Hernández et al.(2015)] Daniel Hernández, Aidan Hogan, and Markus Krötzsch. 2015. Reifying RDF: What Works Well With Wikidata?. In *Proceedings of the 11th International Workshop on Scalable Semantic Web Knowledge Base Systems co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 11, 2015 (CEUR Workshop Proceedings, Vol. 1457)*. CEUR-WS.org, 32–47. [https://ceur-ws.org/Vol-1457/SSWS2015\\_paper3.pdf](https://ceur-ws.org/Vol-1457/SSWS2015_paper3.pdf)
- [Hosseini-Asl et al.(2020)] Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A Simple Language Model for Task-Oriented Dialogue. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/e946209592563be0f01c844ab2170f0c-Abstract.html>
- [Hu et al.(2021)] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *ArXiv* (2021).
- [Hu et al.(2023)] Zhiyuan Hu, Yue Feng, Yang Deng, Zekun Li, See-Kiong Ng, Anh Tuan Luu, and Bryan Hooi. 2023. Enhancing Large Language Model Induced Task-Oriented Dialogue Systems Through Look-Forward Motivated Goals. *CoRR* abs/2309.08949 (2023). <https://doi.org/10.48550/ARXIV.2309.08949> arXiv:2309.08949
- [Hua et al.(2013)] Xian-Sheng Hua, Linjun Yang, Jingdong Wang, Jing Wang, Ming Ye, Kuansan Wang, Yong Rui, and Jin Li. 2013. Clickage: towards bridging semantic and intent gaps via mining click logs of search engines. In *ACM Multimedia Conference, MM '13, Barcelona, Spain, October 21-25, 2013*. ACM, 243–252. <https://doi.org/10.1145/2502081.2502283>
- [Huang et al.(2019a)] Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. 2019a. FlowQA: Grasping Flow in History for Conversational Machine Comprehension. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. <https://openreview.net/forum?id=ByftGnR9KX>
- [Huang et al.(2019b)] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019b. Knowledge Graph Embedding Based Question Answering. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*. ACM, 105–113. <https://doi.org/10.1145/3289600.3290956>
- [Ishii et al.(2022a)] Etsuko Ishii, Bryan Wilie, Yan Xu, Samuel Cahyawijaya, and Pascale Fung. 2022a. Integrating Question Rewrites in Conversational Question Answering: A Reinforcement Learning Approach. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 55–66. <https://doi.org/10.18653/V1/2022.ACL-SRW.6>

- [Ishii et al.(2022b)] Etsuko Ishii, Yan Xu, Samuel Cahyawijaya, and Bryan Wilie. 2022b. Can Question Rewriting Help Conversational Question Answering?. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP, Insights@ACL 2022, Dublin, Ireland, May 26, 2022*. Association for Computational Linguistics, 94–99. <https://doi.org/10.18653/V1/2022.INSIGHTS-1.13>
- [Iyyer et al.(2017)] Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based Neural Structured Learning for Sequential Question Answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, 1821–1831. <https://doi.org/10.18653/V1/P17-1167>
- [Izacard and Grave(2021)] Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*. Association for Computational Linguistics, 874–880. <https://doi.org/10.18653/V1/2021.EACL-MAIN.74>
- [Jain and Lapata(2024)] Parag Jain and Mirella Lapata. 2024. Integrating Large Language Models with Graph-based Reasoning for Conversational Question Answering. *CoRR* abs/2407.09506 (2024). <https://doi.org/10.48550/ARXIV.2407.09506> arXiv:2407.09506
- [Jang et al.(2024)] Yunah Jang, Kang-il Lee, Hyunkyung Bae, Hwanhee Lee, and Kyomin Jung. 2024. IterCQR: Iterative Conversational Query Reformulation with Retrieval Guidance. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*. Association for Computational Linguistics, 8121–8138. <https://doi.org/10.18653/V1/2024.NAACL-LONG.449>
- [Jansen et al.(2009)] Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. 2009. Patterns of query reformulation during Web searching. *J. Assoc. Inf. Sci. Technol.* 60, 7 (2009), 1358–1371. <https://doi.org/10.1002/ASI.21071>
- [Jeong et al.(2023)] Soyeong Jeong, Jinheon Baek, Sung Ju Hwang, and Jong Park. 2023. Phrase Retrieval for Open Domain Conversational Question Answering with Conversational Dependency Modeling via Contrastive Learning. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 6019–6031. <https://doi.org/10.18653/V1/2023.FINDINGS-ACL.374>
- [Jiang et al.(2023)] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth ee Lacroix, and William El Sayed. 2023. Mistral 7B. *CoRR* abs/2310.06825 (2023). <https://doi.org/10.48550/ARXIV.2310.06825> arXiv:2310.06825
- [Joachims et al.(2007)] Thorsten Joachims, Laura A. Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the accuracy of implicit feedback from clicks and

- query reformulations in Web search. *ACM Trans. Inf. Syst.* 25, 2 (2007), 7. <https://doi.org/10.1145/1229179.1229181>
- [Joko et al.(2021)] Hideaki Joko, Faegheh Hasibi, Krisztian Balog, and Arjen P. de Vries. 2021. Conversational Entity Linking: Problem Definition and Datasets. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 2390–2397. <https://doi.org/10.1145/3404835.3463258>
- [Kacupaj et al.(2021)] Endri Kacupaj, Joan Plepi, Kuldeep Singh, Harsh Thakkar, Jens Lehmann, and Maria Maleshkova. 2021. Conversational Question Answering over Knowledge Graphs with Transformer and Graph Attention Networks. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*. Association for Computational Linguistics, 850–862. <https://doi.org/10.18653/V1/2021.EACL-MAIN.72>
- [Kacupaj et al.(2022)] Endri Kacupaj, Kuldeep Singh, Maria Maleshkova, and Jens Lehmann. 2022. Contrastive Representation Learning for Conversational Question Answering over Knowledge Graphs. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*. ACM, 925–934. <https://doi.org/10.1145/3511808.3557267>
- [Kaiser et al.(2024a)] Magdalena Kaiser, Patrick Ernst, and György Szarvas. 2024a. Learning from Relevant Subgoals in Successful Dialogs using Iterative Training for Task-oriented Dialog Systems. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*. Association for Computational Linguistics, 6236–6246. <https://doi.org/10.18653/V1/2024.FINDINGS-EMNLP.362>
- [Kaiser et al.(2020)] Magdalena Kaiser, Rishiraj Saha Roy, and Gerhard Weikum. 2020. Conversational Question Answering over Passages by Leveraging Word Proximity Networks. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 2129–2132. <https://doi.org/10.1145/3397271.3401399>
- [Kaiser et al.(2021)] Magdalena Kaiser, Rishiraj Saha Roy, and Gerhard Weikum. 2021. Reinforcement Learning from Reformulations in Conversational Question Answering over Knowledge Graphs. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 459–469. <https://doi.org/10.1145/3404835.3462859>
- [Kaiser et al.(2024b)] Magdalena Kaiser, Rishiraj Saha Roy, and Gerhard Weikum. 2024b. Robust Training for Conversational Question Answering Models with Reinforced Reformulation Generation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining, WSDM 2024, Merida, Mexico, March 4-8, 2024*. ACM, 322–331. <https://doi.org/10.1145/3616855.3635822>
- [Kaiser and Weikum(2025)] Magdalena Kaiser and Gerhard Weikum. 2025. Preference-based Learning with Retrieval Augmented Generation for Conversational Question Answering. In *Compan-*

- ion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*. ACM, 1053–1057. <https://doi.org/10.1145/3701716.3715544>
- [Karpukhin et al.(2020)] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 6769–6781. <https://doi.org/10.18653/V1/2020.EMNLP-MAIN.550>
- [Kaufmann et al.(2025)] Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. 2025. A Survey of Reinforcement Learning from Human Feedback. *Trans. Mach. Learn. Res.* 2025 (2025). <https://openreview.net/forum?id=f70kIurx4b>
- [Ke et al.(2022)] Xirui Ke, Jing Zhang, Xin Lv, Yiqi Xu, Shulin Cao, Cuiping Li, Hong Chen, and Juanzi Li. 2022. Knowledge-augmented Self-training of A Question Rewriter for Conversational Knowledge Base Question Answering. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 1844–1856. <https://doi.org/10.18653/V1/2022.FINDINGS-EMNLP.133>
- [Kiesel et al.(2021)] Johannes Kiesel, Xiaoni Cai, Roxanne El Baff, Benno Stein, and Matthias Hagen. 2021. Toward Conversational Query Reformulation. In *Proceedings of the Second International Conference on Design of Experimental Search & Information REtrieval Systems, Padova, Italy, September 15-18, 2021 (CEUR Workshop Proceedings, Vol. 2950)*. CEUR-WS.org, 91–101. <https://ceur-ws.org/Vol-2950/paper-12.pdf>
- [Kim et al.(2021)] Gangwoo Kim, Hyunjae Kim, Jungsoo Park, and Jaewoo Kang. 2021. Learn to Resolve Conversational Dependency: A Consistency Training Framework for Conversational Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 6130–6141. <https://doi.org/10.18653/V1/2021.ACL-LONG.478>
- [Kingma and Ba(2015)] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. <http://arxiv.org/abs/1412.6980>
- [Kirk et al.(2023)] Hannah Kirk, Andrew M. Bean, Bertie Vidgen, Paul Röttger, and Scott Hale. 2023. The Past, Present and Better Future of Feedback Learning in Large Language Models for Subjective Human Preferences and Values. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 2409–2430. <https://doi.org/10.18653/V1/2023.EMNLP-MAIN.148>

- [Kratzwald and Feuerriegel(2019)] Bernhard Kratzwald and Stefan Feuerriegel. 2019. Learning from On-Line User Feedback in Neural Question Answering on the Web. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. ACM, 906–916. <https://doi.org/10.1145/3308558.3313661>
- [Lan and Jiang(2021)] Yunshi Lan and Jing Jiang. 2021. Modeling Transitions of Focal Entities for Conversational Knowledge Base Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 3288–3297. <https://doi.org/10.18653/v1/2021.acl-long.255>
- [Lee et al.(2021)] Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue State Tracking with a Language Model using Schema-Driven Prompting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 4937–4949. <https://doi.org/10.18653/v1/2021.emnlp-main.404>
- [Lee et al.(2025)] Dong Won Lee, Hae Won Park, Cynthia Breazeal, and Louis-Philippe Morency. 2025. Aligning Dialogue Agents with Global Feedback via Large Language Model Reward Decomposition. *CoRR* abs/2505.15922 (2025). <https://doi.org/10.48550/ARXIV.2505.15922> arXiv:2505.15922
- [Lee(2021)] Yohan Lee. 2021. Improving End-to-End Task-Oriented Dialog System with A Simple Auxiliary Task. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*. Association for Computational Linguistics, 1296–1303. <https://doi.org/10.18653/v1/2021.findings-emnlp.112>
- [Leeka et al.(2016)] Jyoti Leeka, Srikanta Bedathur, Debajyoti Bera, and Medha Atre. 2016. *Quark-X*: An Efficient Top-K Processing Framework for RDF Quad Stores. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*. ACM, 831–840. <https://doi.org/10.1145/2983323.2983727>
- [Levin et al.(1998)] Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1998. Using Markov decision process for learning dialogue strategies. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98, Seattle, Washington, USA, May 12-15, 1998*. IEEE, 201–204. <https://doi.org/10.1109/ICASSP.1998.674402>
- [Lewis et al.(2020a)] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 7871–7880. <https://doi.org/10.18653/v1/2020.acl-main.703>

- [Lewis et al.(2020b)] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html>
- [Li et al.(2020)] Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient One-Pass End-to-End Entity Linking for Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 6433–6441. <https://doi.org/10.18653/V1/2020.EMNLP-MAIN.522>
- [Li et al.(2022b)] Xiangsheng Li, Jiaxin Mao, Weizhi Ma, Zhijing Wu, Yiqun Liu, Min Zhang, Shaoping Ma, Zhaowei Wang, and Xiuqiang He. 2022b. A Cooperative Neural Information Retrieval Pipeline with Knowledge Enhanced Automatic Query Reformulation. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*. ACM, 553–561. <https://doi.org/10.1145/3488560.3498516>
- [Li et al.(2022a)] Yongqi Li, Wenjie Li, and Liqiang Nie. 2022a. MMCoQA: Conversational Question Answering over Text, Tables, and Images. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 4220–4231. <https://doi.org/10.18653/V1/2022.ACL-LONG.290>
- [Li et al.(2022c)] Zichao Li, Prakhar Sharma, Xing Han Lù, Jackie Chi Kit Cheung, and Siva Reddy. 2022c. Using Interactive Feedback to Improve the Accuracy and Explainability of Question Answering Systems Post-Deployment. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 926–937. <https://doi.org/10.18653/V1/2022.FINDINGS-ACL.75>
- [Lin et al.(2021)] Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2021. Multi-Stage Conversational Passage Retrieval: An Approach to Fusing Term Importance Estimation and Neural Query Rewriting. *ACM Trans. Inf. Syst.* 39, 4 (2021), 48:1–48:29. <https://doi.org/10.1145/3446426>
- [Lin et al.(2018)] Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-Hop Knowledge Graph Reasoning with Reward Shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 3243–3253. <https://doi.org/10.18653/V1/D18-1362>
- [Linjordet and Balog(2022)] Trond Linjordet and Krisztian Balog. 2022. Would You Ask it that Way?: Measuring and Improving Question Naturalness for Knowledge Graph Question Answering. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and*

- Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 3090–3098. <https://doi.org/10.1145/3477495.3531739>
- [Liu and Lane(2017)] Bing Liu and Ian R. Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*. IEEE, 482–489. <https://doi.org/10.1109/ASRU.2017.8268975>
- [Liu et al.(2020)] Dayiheng Liu, Yeyun Gong, Jie Fu, Yu Yan, Jiusheng Chen, Jiancheng Lv, Nan Duan, and Ming Zhou. 2020. Tell Me How to Ask Again: Question Data Augmentation with Controllable Rewriting in Continuous Space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 5798–5810. <https://doi.org/10.18653/V1/2020.EMNLP-MAIN.467>
- [Liu et al.(2022)] Jiacheng Liu, Skyler Hallinan, Ximing Lu, Pengfei He, Sean Welleck, Hannaneh Hajishirzi, and Yejin Choi. 2022. Rainier: Reinforced Knowledge Introspector for Common-sense Question Answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 8938–8958. <https://doi.org/10.18653/V1/2022.EMNLP-MAIN.611>
- [Liu et al.(2025)] Xuanqing Liu, Luyang Kong, Wei Niu, Afshin Khashei, Belinda Zeng, Steve Johnson, Jon Jay, Davor Golac, and Matt Pope. 2025. Learning LLM Preference over Intra-Dialogue Pairs: A Framework for Utterance-level Understandings. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 3: Industry Track, Albuquerque, New Mexico, USA, April 30, 2025*. Association for Computational Linguistics, 86–98. <https://doi.org/10.18653/V1/2025.NAACL-INDUSTRY.8>
- [Liu and Belkin(2008)] Ying-Hsang Liu and Nicholas J. Belkin. 2008. Query reformulation, search performance, and term suggestion devices in question-answering tasks. In *Proceedings of the 2nd International Conference on Information Interaction in Context, IiX 2008, London, UK, October 14-17, 2008 (ACM International Conference Proceeding Series, Vol. 348)*. ACM, 21–26. <https://doi.org/10.1145/1414694.1414702>
- [Liu et al.(2019a)] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 <http://arxiv.org/abs/1907.11692>
- [Liu et al.(2023)] Ye Liu, Stefan Ultes, Wolfgang Minker, and Wolfgang Maier. 2023. Unified Conversational Models with System-Initiated Transitions between Chit-Chat and Task-Oriented Dialogues. In *Proceedings of the 5th International Conference on Conversational User Interfaces, CUI 2023, Eindhoven, The Netherlands, July 19-21, 2023*. ACM, 33:1–33:9. <https://doi.org/10.1145/3571884.3597125>

- [Liu et al.(2019b)] Ye Liu, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip S. Yu. 2019b. Generative Question Refinement with Deep Reinforcement Learning in Retrieval-based QA System. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 1643–1652. <https://doi.org/10.1145/3357384.3358046>
- [Lu et al.(2019)] Xiaolu Lu, Soumajit Pramanik, Rishiraj Saha Roy, Abdalghani Abujabal, Yafang Wang, and Gerhard Weikum. 2019. Answering Complex Questions by Joining Multi-Document Evidence with Quasi Knowledge Graphs. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. ACM, 105–114. <https://doi.org/10.1145/3331184.3331252>
- [Lubis et al.(2020)] Nurul Lubis, Christian Geishauer, Michael Heck, Hsien-Chin Lin, Marco Moresi, Carel van Niekerk, and Milica Gasic. 2020. LAVA: Latent Action Spaces via Variational Auto-encoding for Dialogue Policy Optimization. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*. International Committee on Computational Linguistics, 465–479. <https://doi.org/10.18653/V1/2020.COLING-MAIN.41>
- [Ma et al.(2023)] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query Rewriting in Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 5303–5315. <https://doi.org/10.18653/V1/2023.EMNLP-MAIN.322>
- [Mao et al.(2024)] Kelong Mao, Chenlong Deng, Haonan Chen, Fengran Mo, Zheng Liu, Tetuya Sakai, and Zhicheng Dou. 2024. ChatRetriever: Adapting Large Language Models for Generalized and Robust Conversational Dense Retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*. Association for Computational Linguistics, 1227–1240. <https://doi.org/10.18653/V1/2024.EMNLP-MAIN.71>
- [Mao et al.(2023)] Kelong Mao, Zhicheng Dou, Fengran Mo, Jiewen Hou, Haonan Chen, and Hongjin Qian. 2023. Large Language Models Know Your Contextual Search Intent: A Prompting Framework for Conversational Search. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 1211–1225. <https://doi.org/10.18653/V1/2023.FINDINGS-EMNLP.86>
- [Mao et al.(2022)] Kelong Mao, Zhicheng Dou, Hongjin Qian, Fengran Mo, Xiaohua Cheng, and Zhao Cao. 2022. ConvTrans: Transforming Web Search Sessions for Conversational Dense Retrieval. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*. Association for Computational Linguistics, 2935–2946. <https://doi.org/10.18653/V1/2022.EMNLP-MAIN.190>
- [Marion et al.(2021)] Pierre Marion, Pawel Krzysztof Nowak, and Francesco Piccinno. 2021. Structured Context and High-Coverage Grammar for Conversational Question Answering over

- Knowledge Graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 8813–8829. <https://doi.org/10.18653/V1/2021.EMNLP-MAIN.695>
- [Min et al.(2025)] Dehai Min, Zhiyang Xu, Guilin Qi, Lifu Huang, and Chenyu You. 2025. UniHGKR: Unified Instruction-aware Heterogeneous Knowledge Retrievers. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*. Association for Computational Linguistics, 4577–4594. <https://doi.org/10.18653/V1/2025.NAACL-LONG.234>
- [Mnih et al.(2013)] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR* abs/1312.5602 (2013). arXiv:1312.5602 <http://arxiv.org/abs/1312.5602>
- [Mnih et al.(2015)] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharmashan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nat.* 518, 7540 (2015), 529–533. <https://doi.org/10.1038/NATURE14236>
- [Mo et al.(2023)] Fengran Mo, Kelong Mao, Yutao Zhu, Yihong Wu, Kaiyu Huang, and Jian-Yun Nie. 2023. ConvGQR: Generative Query Reformulation for Conversational Search. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 4998–5012. <https://doi.org/10.18653/V1/2023.ACL-LONG.274>
- [Mo et al.(2024)] Lingbo Mo, Shun Jiang, Akash Maharaj, Bernard Hishamunda, and Yunyao Li. 2024. HierTOD: A Task-Oriented Dialogue System Driven by Hierarchical Goals. *CoRR* abs/2411.07152 (2024). <https://doi.org/10.48550/ARXIV.2411.07152> arXiv:2411.07152
- [Müller et al.(2019)] Thomas Müller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering Conversational Questions on Structured Data without Logical Forms. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 5901–5909. <https://doi.org/10.18653/V1/D19-1603>
- [Neeman et al.(2023)] Ella Neeman, Roei Aharoni, Or Honovich, Leshem Choshen, Idan Szpektor, and Omri Abend. 2023. DisentQA: Disentangling Parametric and Contextual Knowledge with Counterfactual Question Answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 10056–10070. <https://doi.org/10.18653/V1/2023.ACL-LONG.559>

- [Nekvinda and Dušek(2021)] Tomáš Nekvinda and Ondřej Dušek. 2021. Shades of BLEU, Flavours of Success: The Case of MultiWOZ. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*. Association for Computational Linguistics, Online, 34–46. <https://doi.org/10.18653/v1/2021.gem-1.4>
- [Nguyen et al.(2016)] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated Machine Reading Comprehension Dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016 (CEUR Workshop Proceedings, Vol. 1773)*. CEUR-WS.org. [https://ceur-ws.org/Vol-1773/CoCoNIPS\\_2016\\_paper9.pdf](https://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf)
- [Nguyen et al.(2014)] Vinh Nguyen, Olivier Bodenreider, and Amit P. Sheth. 2014. Don’t like RDF reification?: making statements about statements using singleton property. In *23rd International World Wide Web Conference, WWW ’14, Seoul, Republic of Korea, April 7-11, 2014*. ACM, 759–770. <https://doi.org/10.1145/2566486.2567973>
- [Nogueira and Cho(2017)] Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-Oriented Query Reformulation with Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Association for Computational Linguistics, 574–583. <https://doi.org/10.18653/v1/D17-1061>
- [Oguz et al.(2020)] Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2020. Unified Open-Domain Question Answering with Structured and Unstructured Knowledge. *CoRR* abs/2012.14610 (2020). arXiv:2012.14610 <https://arxiv.org/abs/2012.14610>
- [Ouyang et al.(2022)] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. [http://papers.nips.cc/paper\\_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html)
- [Owoicho et al.(2022)] Paul Owoicho, Jeff Dalton, Mohammad Aliannejadi, Leif Azzopardi, Johanne R. Trippas, and Svitlana Vakulenko. 2022. TREC CAsT 2022: Going Beyond User Ask and System Retrieve with Initiative and Response Generation. In *Proceedings of the Thirty-First Text REtrieval Conference, TREC 2022, online, November 15-19, 2022 (NIST Special Publication, Vol. 500-338)*. National Institute of Standards and Technology (NIST). [https://trec.nist.gov/pubs/trec31/papers/Overview\\_cast.pdf](https://trec.nist.gov/pubs/trec31/papers/Overview_cast.pdf)
- [Owoicho et al.(2023)] Paul Owoicho, Ivan Sekulic, Mohammad Aliannejadi, Jeffrey Dalton, and Fabio Crestani. 2023. Exploiting Simulated User Feedback for Conversational Search: Ranking,

- Rewriting, and Beyond. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 632–642. <https://doi.org/10.1145/3539618.3591683>
- [Pathak et al.(2017)] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. 2017. Curiosity-driven Exploration by Self-supervised Prediction. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research, Vol. 70)*. PMLR, 2778–2787. <http://proceedings.mlr.press/v70/pathak17a.html>
- [Peng et al.(2023)] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, and Jianfeng Gao. 2023. Check Your Facts and Try Again: Improving Large Language Models with External Knowledge and Automated Feedback. *CoRR* abs/2302.12813 (2023). <https://doi.org/10.48550/ARXIV.2302.12813> arXiv:2302.12813
- [Piccinno and Ferragina(2014)] Francesco Piccinno and Paolo Ferragina. 2014. From TagME to WAT: a new entity annotator. In *ERD'14, Proceedings of the First ACM International Workshop on Entity Recognition & Disambiguation, July 11, 2014, Gold Coast, Queensland, Australia*. ACM, 55–62. <https://doi.org/10.1145/2633211.2634350>
- [Ponnusamy et al.(2020)] Pragaash Ponnusamy, Alireza Roshan Ghias, Chenlei Guo, and Ruhi Sarikaya. 2020. Feedback-Based Self-Learning in Large-Scale Conversational AI Agents. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 13180–13187. <https://doi.org/10.1609/AAAI.V34I08.7022>
- [Puri et al.(2020)] Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training Question Answering Models From Synthetic Data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 5811–5826. <https://doi.org/10.18653/V1/2020.EMNLP-MAIN.468>
- [Qiu et al.(2021)] Minghui Qiu, Xinjing Huang, Cen Chen, Feng Ji, Chen Qu, Wei Wei, Jun Huang, and Yin Zhang. 2021. Reinforced History Backtracking for Conversational Question Answering. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 13718–13726. <https://doi.org/10.1609/AAAI.V35I15.17617>
- [Qiu et al.(2020)] Yunqi Qiu, Yuanzhuo Wang, Xiaolong Jin, and Kun Zhang. 2020. Stepwise Reasoning for Multi-Relation Question Answering over Knowledge Graph with Weak Supervision. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*. ACM, 474–482. <https://doi.org/10.1145/3336191.3371812>

- [Qu et al.(2020)] Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W. Bruce Croft, and Mohit Iyyer. 2020. Open-Retrieval Conversational Question Answering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 539–548. <https://doi.org/10.1145/3397271.3401110>
- [Qu et al.(2019a)] Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019a. BERT with History Answer Embedding for Conversational Question Answering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*. ACM, 1133–1136. <https://doi.org/10.1145/3331184.3331341>
- [Qu et al.(2019b)] Chen Qu, Liu Yang, Minghui Qiu, Yongfeng Zhang, Cen Chen, W. Bruce Croft, and Mohit Iyyer. 2019b. Attentive History Selection for Conversational Question Answering. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 1391–1400. <https://doi.org/10.1145/3357384.3357905>
- [Rafailov et al.(2023)] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*. [http://papers.nips.cc/paper\\_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html)
- [Ramachandran et al.(2022)] Govardana Sachithanandam Ramachandran, Kazuma Hashimoto, and Caiming Xiong. 2022. [CASPI] Causal-aware Safe Policy Improvement for Task-oriented Dialogue. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 92–102. <https://doi.org/10.18653/V1/2022.ACL-LONG.8>
- [Raposo et al.(2022)] Gonalo Raposo, Rui Ribeiro, Bruno Martins, and Lusa Coheur. 2022. Question Rewriting? Assessing Its Importance for Conversational Question Answering. In *Advances in Information Retrieval - 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10-14, 2022, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 13186)*. Springer, 199–206. [https://doi.org/10.1007/978-3-030-99739-7\\_23](https://doi.org/10.1007/978-3-030-99739-7_23)
- [Rastogi et al.(2020)] Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards Scalable Multi-Domain Conversational Agents: The Schema-Guided Dialogue Dataset. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 8689–8696. <https://doi.org/10.1609/AAAI.V34I05.6394>
- [Reddy et al.(2019)] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Trans. Assoc. Comput. Linguistics* 7 (2019), 249–266. [https://doi.org/10.1162/TACL\\_A\\_00266](https://doi.org/10.1162/TACL_A_00266)

- [Reimers and Gurevych(2019)] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 3980–3990. <https://doi.org/10.18653/V1/D19-1410>
- [Ren et al.(2018)] Gary Ren, Xiaochuan Ni, Manish Malik, and Qifa Ke. 2018. Conversational Query Understanding Using Sequence to Sequence Modeling. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*. ACM, 1715–1724. <https://doi.org/10.1145/3178876.3186083>
- [Rennie et al.(2017)] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-Critical Sequence Training for Image Captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 1179–1195. <https://doi.org/10.1109/CVPR.2017.131>
- [Roy and Anand(2020)] Rishiraj Saha Roy and Avishek Anand. 2020. Question Answering over Curated and Open Web Sources. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 2432–2435. <https://doi.org/10.1145/3397271.3401421>
- [Roy and Anand(2021)] Rishiraj Saha Roy and Avishek Anand. 2021. *Question Answering for the Curated Web: Tasks and Methods in QA over Knowledge Bases and Text Collections*. Morgan & Claypool Publishers. <https://doi.org/10.2200/S0113ED1V01Y202109ICR076>
- [Sachan and Xing(2018)] Mrinmaya Sachan and Eric P. Xing. 2018. Self-Training for Jointly Learning to Ask and Answer Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, 629–640. <https://doi.org/10.18653/V1/N18-1058>
- [Saha et al.(2018)] Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex Sequential Question Answering: Towards Learning to Converse Over Linked Question Answer Pairs with a Knowledge Graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press, 705–713. <https://doi.org/10.1609/AAAI.V32I1.11332>
- [Sawant and Chakrabarti(2013)] Uma Sawant and Soumen Chakrabarti. 2013. Learning joint query interpretation and response ranking. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. International World Wide Web Conferences Steering Committee / ACM, 1099–1110. <https://doi.org/10.1145/2488388.2488484>

- [Schulman et al.(2017)] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 <http://arxiv.org/abs/1707.06347>
- [Shen et al.(2019)] Tao Shen, Xiubo Geng, Tao Qin, Daya Guo, Duyu Tang, Nan Duan, Guodong Long, and Daxin Jiang. 2019. Multi-Task Learning for Conversational Question Answering over a Large-Scale Knowledge Base. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 2442–2451. <https://doi.org/10.18653/V1/D19-1248>
- [Shen et al.(2015)] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Trans. Knowl. Data Eng.* 27, 2 (2015), 443–460. <https://doi.org/10.1109/TKDE.2014.2327028>
- [Shen et al.(2022)] Xiaoyu Shen, Gianni Barlacchi, Marco Del Tredici, Weiwei Cheng, Bill Byrne, and Adrià Gispert. 2022. Product Answer Generation from Heterogeneous Sources: A New Benchmark and Best Practices. In *Proceedings of the Fifth Workshop on e-Commerce and NLP (ECNLP 5), Dublin, Ireland, May 2022*. Association for Computational Linguistics, 99–110. <https://doi.org/10.18653/v1/2022.ecnlp-1.13>
- [Shen et al.(2018)] Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao. 2018. M-Walk: Learning to Walk over Graphs using Monte Carlo Tree Search. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. 6787–6798. <https://proceedings.neurips.cc/paper/2018/hash/c6f798b844366ccd65d99bc7f31e0e02-Abstract.html>
- [Shen et al.(2024)] Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. 2024. TaskBench: Benchmarking Large Language Models for Task Automation. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*. [http://papers.nips.cc/paper\\_files/paper/2024/hash/085185ea97db31ae6dcac7497616fd3e-Abstract-Datasets\\_and\\_Benchmarks\\_Track.html](http://papers.nips.cc/paper_files/paper/2024/hash/085185ea97db31ae6dcac7497616fd3e-Abstract-Datasets_and_Benchmarks_Track.html)
- [Shi et al.(2024)] Taiwei Shi, Zhuoer Wang, Longqi Yang, Ying-Chun Lin, Zexue He, Mengting Wan, Pei Zhou, Sujay Kumar Jauhar, Xiaofeng Xu, Xia Song, and Jennifer Neville. 2024. WildFeedback: Aligning LLMs With In-situ User Interactions And Feedback. In *NeurIPS 2024 Workshop on Behavioral Machine Learning*. <https://openreview.net/forum?id=07QCozT1pi>
- [Snell et al.(2022)] Charlie Snell, Sherry Yang, Justin Fu, Yi Su, and Sergey Levine. 2022. Context-Aware Language Modeling for Goal-Oriented Dialogue Systems. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*. Association for Computational Linguistics, 2351–2366. <https://doi.org/10.18653/V1/2022.FINDINGS-NAACL.181>

- [Steindl et al.(2024)] Sebastian Steindl, Ulrich Schäfer, and Bernd Ludwig. 2024. Counterfactual Dialog Mixing as Data Augmentation for Task-Oriented Dialog Systems. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*. ELRA and ICCL, 4078–4087. <https://aclanthology.org/2024.lrec-main.363>
- [Stiennon et al.(2020)] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. <https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html>
- [Suchanek et al.(2007)] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. ACM, 697–706. <https://doi.org/10.1145/1242572.1242667>
- [Sugiyama et al.(2012)] Hiroaki Sugiyama, Toyomi Meguro, and Yasuhiro Minami. 2012. Preference-learning based Inverse Reinforcement Learning for Dialog Control. In *13th Annual Conference of the International Speech Communication Association, INTERSPEECH 2012, Portland, Oregon, USA, September 9-13, 2012*. ISCA, 222–225. <https://doi.org/10.21437/INTERSPEECH.2012-72>
- [Sun et al.(2023)] Haipeng Sun, Junwei Bao, Youzheng Wu, and Xiaodong He. 2023. Mars: Modeling Context & State Representations with Contrastive Learning for End-to-End Task-Oriented Dialog. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, 11139–11160. <https://doi.org/10.18653/V1/2023.FINDINGS-ACL.708>
- [Sun et al.(2019)] Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. 2019. PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 2380–2390. <https://doi.org/10.18653/V1/D19-1242>
- [Sutton and Barto(2018)] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning - an introduction, 2nd Edition*. MIT Press. <http://www.incompleteideas.net/book/the-book-2nd.html>
- [Takanobu et al.(2019)] Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided Dialog Policy Learning: Reward Estimation for Multi-Domain Task-Oriented Dialog. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 100–110. <https://doi.org/10.18653/V1/D19-1010>

- [Talmor and Berant(2018)] Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, 641–651. <https://doi.org/10.18653/v1/N18-1059>
- [Tian et al.(2021)] Xin Tian, Liankai Huang, Yingzhan Lin, Siqi Bao, Huang He, Yunyi Yang, Hua Wu, Fan Wang, and Shuqi Sun. 2021. Amendable Generation for Dialogue State Tracking. In *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI, Online, Nov 2021*. Association for Computational Linguistics, 80–92. <https://doi.org/10.18653/v1/2021.nlp4convai-1.8>
- [Tomuro(2003)] Noriko Tomuro. 2003. Interrogative reformulation patterns and acquisition of question paraphrases. In *Proceedings of the second international workshop on Paraphrasing, July 2003*. Association for Computational Linguistics, 33–40. <https://aclanthology.org/W03-1605.pdf>
- [Tomuro and Lytinen(2001)] Noriko Tomuro and Steven L Lytinen. 2001. Selecting features for paraphrasing question sentences. In *Proceedings of the Workshop on Automatic Paraphrasing at Natural Language Processing Pacific Rim Symposium (NLPRS)*. 55–62. <https://condor.depaul.edu/~ntomuro/papers/paraphrase.pdf>
- [Touvron et al.(2023)] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *CoRR* abs/2302.13971 (2023). <https://doi.org/10.48550/ARXIV.2302.13971> arXiv:2302.13971
- [Tran et al.(2024)] Hai Dang Tran, Andrew Yates, and Gerhard Weikum. 2024. Conversational Search with Tail Entities. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24-28, 2024, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 14609)*. Springer, 303–317. [https://doi.org/10.1007/978-3-031-56060-6\\_20](https://doi.org/10.1007/978-3-031-56060-6_20)
- [Trivedi et al.(2017)] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 10588)*. Springer, 210–218. [https://doi.org/10.1007/978-3-319-68204-4\\_22](https://doi.org/10.1007/978-3-319-68204-4_22)
- [Unger et al.(2014)] Christina Unger, André Freitas, and Philipp Cimiano. 2014. An Introduction to Question Answering over Linked Data. In *Reasoning Web. Reasoning on the Web in the Big Data Era - 10th International Summer School 2014, Athens, Greece, September 8-13, 2014. Proceedings (Lecture Notes in Computer Science, Vol. 8714)*. Springer, 100–140. [https://doi.org/10.1007/978-3-319-10587-1\\_2](https://doi.org/10.1007/978-3-319-10587-1_2)

- [Usbeck et al.(2018)] Ricardo Usbeck, Ria Hari Gusmita, Axel-Cyrille Ngonga Ngomo, and Muhammad Saleem. 2018. 9th Challenge on Question Answering over Linked Data (QALD-9) (invited paper). In *Joint proceedings of the 4th Workshop on Semantic Deep Learning (SemDeep-4) and NLIWoD4: Natural Language Interfaces for the Web of Data (NLIWOD-4) and 9th Question Answering over Linked Data challenge (QALD-9) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, California, United States of America, October 8th - 9th, 2018 (CEUR Workshop Proceedings, Vol. 2241)*. CEUR-WS.org, 58–64. <https://ceur-ws.org/Vol-2241/paper-06.pdf>
- [Vakulenko et al.(2021)] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2021. Question Rewriting for Conversational Question Answering. In *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining, Virtual Event, Israel, March 8-12, 2021*. ACM, 355–363. <https://doi.org/10.1145/3437963.3441748>
- [Vlastelica et al.(2023)] Marin Vlastelica, Patrick Ernst, and Gyuri Szarvas. 2023. Taming Continuous Posteriors for Latent Variational Dialogue Policies. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*. AAAI Press, 13673–13681. <https://doi.org/10.1609/AAAI.V37I11.26602>
- [Voorhees(1999)] Ellen M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999 (NIST Special Publication, Vol. 500-246)*. National Institute of Standards and Technology (NIST). [http://trec.nist.gov/pubs/trec8/papers/qa\\_report.pdf](http://trec.nist.gov/pubs/trec8/papers/qa_report.pdf)
- [Voskarides et al.(2020)] Nikos Voskarides, Dan Li, Pengjie Ren, Evangelos Kanoulas, and Maarten de Rijke. 2020. Query Resolution for Conversational Search with Limited Supervision. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 921–930. <https://doi.org/10.1145/3397271.3401130>
- [Vrandeic and Krötzsch(2014)] Denny Vrandeic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85. <https://doi.org/10.1145/2629489>
- [Vukovic et al.(2024)] Renato Vukovic, David Arps, Carel van Niekerk, Benjamin Matthias Ruppik, Hsien-Chin Lin, Michael Heck, and Milica Gasic. 2024. Dialogue Ontology Relation Extraction via Constrained Chain-of-Thought Decoding. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2024, Kyoto, Japan, September 18 - 20, 2024*. Association for Computational Linguistics, 370–384. <https://doi.org/10.18653/V1/2024.SIGDIAL-1.33>
- [Wang et al.(2020)] Huimin Wang, Baolin Peng, and Kam-Fai Wong. 2020. Learning Efficient Dialogue Policy from Demonstrations through Shaping. In *Proceedings of the 58th Annual Meeting*

- of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020.* Association for Computational Linguistics, 6355–6365. <https://doi.org/10.18653/V1/2020.ACL-MAIN.566>
- [Wang et al.(2023)] Liang Wang, Ivano Lauriola, and Alessandro Moschitti. 2023. Accurate Training of Web-based Question Answering Systems with Feedback from Ranked Users. In *Proceedings of the The 61st Annual Meeting of the Association for Computational Linguistics: Industry Track, ACL 2023, Toronto, Canada, July 9-14, 2023.* Association for Computational Linguistics, 660–667. <https://doi.org/10.18653/V1/2023.ACL-INDUSTRY.63>
- [Wei et al.(2018)] Wei Wei, Quoc V. Le, Andrew M. Dai, and Jia Li. 2018. AirDialogue: An Environment for Goal-Oriented Dialogue Research. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018.* Association for Computational Linguistics, 3844–3854. <https://doi.org/10.18653/V1/D18-1419>
- [Williams and Young(2007)] Jason D. Williams and Steve J. Young. 2007. Partially observable Markov decision processes for spoken dialog systems. *Comput. Speech Lang.* 21, 2 (2007), 393–422. <https://doi.org/10.1016/J.CSL.2006.06.008>
- [Williams(1992)] Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 8 (1992), 229–256. <https://doi.org/10.1007/BF00992696>
- [Wu et al.(2023)] Qingyang Wu, James Gung, Raphael Shu, and Yi Zhang. 2023. DiactTOD: Learning Generalizable Latent Dialogue Acts for Controllable Task-Oriented Dialogue Systems. In *Proceedings of the 24th Meeting of the Special Interest Group on Discourse and Dialogue, SIGDIAL 2023, Prague, Czechia, September 11 - 15, 2023.* Association for Computational Linguistics, 255–267. <https://doi.org/10.18653/V1/2023.SIGDIAL-1.24>
- [Wu et al.(2025)] Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. 2025. CollabLLM: From Passive Responders to Active Collaborators. In *Proceedings of the Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, Canada, July 13 - 19, 2025.* OpenReview.net. <https://openreview.net/forum?id=DmH4HHVb3y>
- [Wu et al.(2022)] Zeqiu Wu, Yi Luan, Hannah Rashkin, David Reitter, Hannaneh Hajishirzi, Mari Ostendorf, and Gaurav Singh Tomar. 2022. CONQRR: Conversational Query Rewriting for Retrieval with Reinforcement Learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022.* Association for Computational Linguistics, 10000–10014. <https://doi.org/10.18653/V1/2022.EMNLP-MAIN.679>
- [Xiong et al.(2017)] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017.* Association for Computational Linguistics, 564–573. <https://doi.org/10.18653/V1/D17-1060>

- [Xu et al.(2020)] Zihan Xu, Jiangang Zhu, Ling Geng, Yang Yang, Bojia Lin, and Daxin Jiang. 2020. Learning to Generate Reformulation Actions for Scalable Conversational Query Understanding. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 2269–2272. <https://doi.org/10.1145/3340531.3412112>
- [Xue et al.(2012)] Xiaobing Xue, Yu Tao, Daxin Jiang, and Hang Li. 2012. Automatically Mining Question Reformulation Patterns from Search Log Data. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 2: Short Papers*. The Association for Computer Linguistics, 187–192. <https://aclanthology.org/P12-2037/>
- [Yahya et al.(2012)] Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, Maya Ramanath, Volker Tresp, and Gerhard Weikum. 2012. Natural Language Questions for the Web of Data. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*. ACL, 379–390. <https://aclanthology.org/D12-1035/>
- [Yang et al.(2004)] Fan Yang, Peter A. Heeman, and Kristy Hollingshead. 2004. Towards understanding mixed-initiative in task-oriented dialogues. In *8th International Conference on Spoken Language Processing, INTERSPEECH-ICSLP 2004, Jeju Island, Korea, October 4-8, 2004*. ISCA, 217–220. <https://doi.org/10.21437/INTERSPEECH.2004-122>
- [Yang et al.(2023)] Haoyan Yang, Zhitao Li, Yong Zhang, Jianzong Wang, Ning Cheng, Ming Li, and Jing Xiao. 2023. PRCA: Fitting Black-Box Large Language Models for Retrieval Question Answering via Pluggable Reward-Driven Contextual Adapter. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 5364–5375. <https://doi.org/10.18653/v1/2023.emnlp-main.326>
- [Yang et al.(2019)] Wei Yang, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. Data Augmentation for BERT Fine-Tuning in Open-Domain Question Answering. *CoRR* abs/1904.06652 (2019). arXiv:1904.06652 <http://arxiv.org/abs/1904.06652>
- [Ye et al.(2023)] Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. 2023. Enhancing Conversational Search: Large Language Model-Aided Informative Query Rewriting. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 5985–6006. <https://doi.org/10.18653/v1/2023.findings-emnlp.398>
- [Yih et al.(2015)] Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 1321–1331. <https://doi.org/10.3115/v1/P15-1128>

- [Yoon et al.(2025)] Chanwoong Yoon, Gangwoo Kim, Byeongguk Jeon, Sungdong Kim, Yohan Jo, and Jaewoo Kang. 2025. Ask Optimal Questions: Aligning Large Language Models with Retriever’s Preference in Conversation. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*. Association for Computational Linguistics, 5899–5921. <https://doi.org/10.18653/V1/2025.FINDINGS-NAACL.328>
- [Young et al.(2010)] Steve J. Young, Milica Gasic, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The Hidden Information State model: A practical framework for POMDP-based spoken dialogue management. *Comput. Speech Lang.* 24, 2 (2010), 150–174. <https://doi.org/10.1016/J.CSL.2009.04.001>
- [Yu et al.(2020)] Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul N. Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-Shot Generative Conversational Query Rewriting. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*. ACM, 1933–1936. <https://doi.org/10.1145/3397271.3401323>
- [Yu et al.(2021)] Shi Yu, Zhenghao Liu, Chenyan Xiong, Tao Feng, and Zhiyuan Liu. 2021. Few-Shot Conversational Dense Retrieval. In *SIGIR ’21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 829–838. <https://doi.org/10.1145/3404835.3462856>
- [Yu et al.(2023)] Xiao Yu, Qingyang Wu, Kun Qian, and Zhou Yu. 2023. KRLS: Improving End-to-End Response Generation in Task Oriented Dialog with Reinforced Keywords Learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*. Association for Computational Linguistics, 12338–12358. <https://doi.org/10.18653/V1/2023.EMNLP-MAIN.759>
- [Yuan et al.(2024)] Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-Rewarding Language Models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net. <https://openreview.net/forum?id=0NphYcmgua>
- [Zamani et al.(2023)] Hamed Zamani, Johanne R. Trippas, Jeff Dalton, and Filip Radlinski. 2023. Conversational Information Seeking. *Found. Trends Inf. Retr.* 17, 3-4 (2023), 244–456. <https://doi.org/10.1561/15000000081>
- [Zang et al.(2020)] Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2: A Dialogue Dataset with Additional Annotation Corrections and State Tracking Baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI, ACL 2020*. 109–117. <https://aclanthology.org/2020.nlp4convai-1.pdf#page=119>
- [Zhang et al.(2022)] Shuo Zhang, Mu-Chun Wang, and Krisztian Balog. 2022. Analyzing and Simulating User Utterance Reformulation in Conversational Recommender Systems. In *SIGIR ’22: The 45th International ACM SIGIR Conference on Research and Development*

- in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 133–143. <https://doi.org/10.1145/3477495.3531936>
- [Zhang et al.(2024)] Tianhua Zhang, Kun Li, Hongyin Luo, Xixin Wu, James R. Glass, and Helen Meng. 2024. Adaptive Query Rewriting: Aligning Rewriters through Marginal Probability of Conversational Answers. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*. Association for Computational Linguistics, 13444–13461. <https://doi.org/10.18653/V1/2024.EMNLP-MAIN.746>
- [Zhang et al.(2019)] Xinbo Zhang, Lei Zou, and Sen Hu. 2019. An Interactive Mechanism to Improve Question Answering Systems via Feedback. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 1381–1390. <https://doi.org/10.1145/3357384.3358059>
- [Zhao et al.(2022)] Jeffrey Zhao, Raghav Gupta, Yuan Cao, Dian Yu, Mingqiu Wang, Harrison Lee, Abhinav Rastogi, Izhak Shafran, and Yonghui Wu. 2022. Description-Driven Task-Oriented Dialog Modeling. *CoRR* abs/2201.08904 (2022). arXiv:2201.08904 <https://arxiv.org/abs/2201.08904>
- [Zhao et al.(2019)] Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking Action Spaces for Reinforcement Learning in End-to-end Dialog Agents with Latent Variable Models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 1208–1218. <https://doi.org/10.18653/V1/N19-1123>
- [Zhao et al.(2024)] Wenting Zhao, Ge Gao, Claire Cardie, and Alexander M. Rush. 2024. I Could’ve Asked That: Reformulating Unanswerable Questions. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*. Association for Computational Linguistics, 4207–4220. <https://doi.org/10.18653/V1/2024.EMNLP-MAIN.242>
- [Zhao et al.(2023)] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *CoRR* abs/2303.18223 (2023). <https://doi.org/10.48550/ARXIV.2303.18223> arXiv:2303.18223
- [Zhou et al.(2017)] Li Zhou, Kevin Small, Oleg Rokhlenko, and Charles Elkan. 2017. End-to-End Offline Goal-Oriented Dialog Policy Learning via Policy Gradient. *CoRR* abs/1712.02838 (2017). arXiv:1712.02838 <http://arxiv.org/abs/1712.02838>
- [Ziegler et al.(2019)] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. Fine-Tuning Language Models from Human Preferences. *CoRR* abs/1909.08593 (2019). arXiv:1909.08593 <http://arxiv.org/abs/1909.08593>