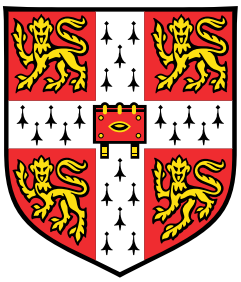
Utilizing Large Language Models for Question Answering in Task-Oriented Dialogues



Abigail Sticha

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy in Machine Learning and Machine Intelligence

Hughes Hall August 2023

Declaration

I, Abigail Sticha of Hughes Hall, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose. The word count, excluding declarations, bibliography, photographs, and diagrams, but including tables, footnotes, figure captions, and appendices, is 14,995 words.

All software used in this thesis was written in Python. The MultiDoc2Dial dataset, baseline model, and shared task evaluations were taken and replicated from the MultiDoc2Dial codebase[1] (Feng et al., 2021). The Chat Completion API[2] and LlamaIndex codebase[3] (Liu, 2022) were used during experimentation, while the UniEval codebase was used for automatic evaluation[4] (Zhong et al., 2022). We clone LlamaIndex version 0.7.9 and modify this for the implementation of all proposed experiments built upon the LlamaIndex framework. The remaining software was written from scratch using standard Python packages.

Abigail Sticha
August 2023

[1] https://github.com/IBM/multidoc2dial
[2] https://platform.openai.com/docs/guides/gpt/chat-completions-api
[3] https://github.com/jerryjliu/llama_index
[4] https://github.com/maszhongming/UniEval

Acknowledgements

I would like to thank my supervisors, Dr. Norbert Braunschweiler and Dr. Kate Knill, for their countless suggestions, guidance, and support throughout the project. I would also like to thank Dr. Rama Doddipatla for his helpful suggestions, and to Toshiba for providing me with the needed computing resources to complete this project.

I want to express my sincere appreciation to DeepMind and the Cambridge Trust for their generous support, which has made it possible for me to study at the University of Cambridge. Their funding for this course has been instrumental in shaping my educational journey.

Above all, I would like to thank my friends and family from home for supporting me even from miles away, as well as my peers at Cambridge, for their support and cherished moments, turning this chapter of my life into a remarkable journey.

Abstract

Task-oriented dialogue systems, such as assistant chatbots and conversational AI systems like ChatGPT, have gained prominence for their personalized question-answering capabilities, often utilizing large language models (LLMs) as knowledge bases. However, these systems face limitations when knowledge outside their intrinsic scope is required. This thesis aims to address these limitations by designing more faithful and useful systems that can accurately respond to users based on external information. This task is approached using the MultiDoc2Dial dataset, which consists of dialogues in different social services domains between an information-seeking user and an information-giving agent who grounds responses by accessing information from multiple documents.

Our study begins by replicating the MultiDoc2Dial baseline which employs the Retrieval Augmented Generation (RAG) framework to augment the response generating BART-LLM with external databases. Despite RAG's proficiency, there has been a significant shift in approach towards utilising up-scaled LLMs that leverage few-shot learning, rather than fine-tuning. Hence, we explore these few-shot techniques to improve upon the baseline.

Before experimentation can be carried out, we show that similarity-based metrics conventionally used for this task, such as F1, can no longer be used to guide system development as they fail to differentiate hallucinations and inadequately measure quality in up-scaled LLMs due to their increased verbosity. Thus, we introduce an improved evaluation methodology combining automatic, LLM self-assessment, and human evaluations.

Guided by this revised evaluation, we present two novel systems surpassing the baseline in accuracy, linguistic quality, and faithfulness. The first system employs an innovative reranking technique using LLMs to rank document relevance without the need for fine-tuning. The second system builds upon the ReAct framework by incorporating a self-reflection mechanism, ensuring answers are grounded in retrieved content. Overall, our efforts advance few-shot prompting as a way to learn to condition on external evidence, making our strategy adaptable to any pre-trained LLM.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In recent years, task-oriented dialogue systems, such as Apple Siri, customer support chatbots, and more recently, conversational AI platforms such as ChatGPT, have surged in popularity due to their ability to provide efficient and personalised assistance to users for specific tasks such as question-answering. However, these systems are often limited by the knowledge they possess, and users may ask questions that require additional information from sources outside the system. Integrating this additional knowledge from other sources, such as web-pages, formatted documents, or plain text, has become crucial to enhancing the faithfulness and usefulness of task-oriented dialogue systems, especially in specialized domains like medicine, social services, or law.

To illustrate, imagine a scenario where a lawyer needs information about a case and the details are stored within their internal files. In this situation, relying solely on ChatGPT or similar tools would be insufficient. Instead, a system augmented with the internal files that could output **well formatted** and **factually correct** responses **grounded** in information from their files is needed. Despite the undeniable necessity of augmenting task-oriented dialogues with external knowledge, this field of research is still emerging and poses significant challenges, such as poor retrieval due to unstructured documents and the tendency to generate factually wrong or unsupported responses (hallucinations). Our project aims to address these challenges to advance the faithfulness and usefulness of task-oriented dialogue systems.

As this is a relatively new and evolving field, it is important to define related terminology in the scope of our work. We aim to achieve a more faithful system where improving faithfulness requires reducing hallucinations and increasing groundedness. We adopt the definition of hallucination similar to that used in Dziri et al. (2022) and Honovich et al. (2021) where an agent response is hallucinated if the factual information contained in the response is not

supported by any snippet of retrieved information from the external database. Groundedness conceptualizes the opposite of hallucination as a response is considered grounded if all factual information is supported in the retrieved information. In addition to increasing faithfulness, we also aim to increase the usefulness of the system which involves ensuring the accuracy of the retrieved information and increasing linguistic qualities of the responses, such as coherence and naturalness.

Previous research has predominantly approached this task as either purely a question-answering task, neglecting the integration of dialogue history, or a machine reading comprehension task based on a single given document. Our work takes these approaches a step further, addressing a more real-world scenario by utilizing the MultiDoc2Dial dataset (Feng et al., 2021) which contains goal-oriented *dialogues* grounded in *multiple documents*. Figure 1.1 shows a sample dialogue with 4 segments on the left that are grounded in three different documents from the MultiDoc2Dial dataset. This figure highlights the complexities of adopting this more robust approach. Notably, at each turn the system must retrieve the correct associated document, taking into account shifts between documents within the dialogue, and then use this document along with context from the dialogue's history to accurately respond to the user.

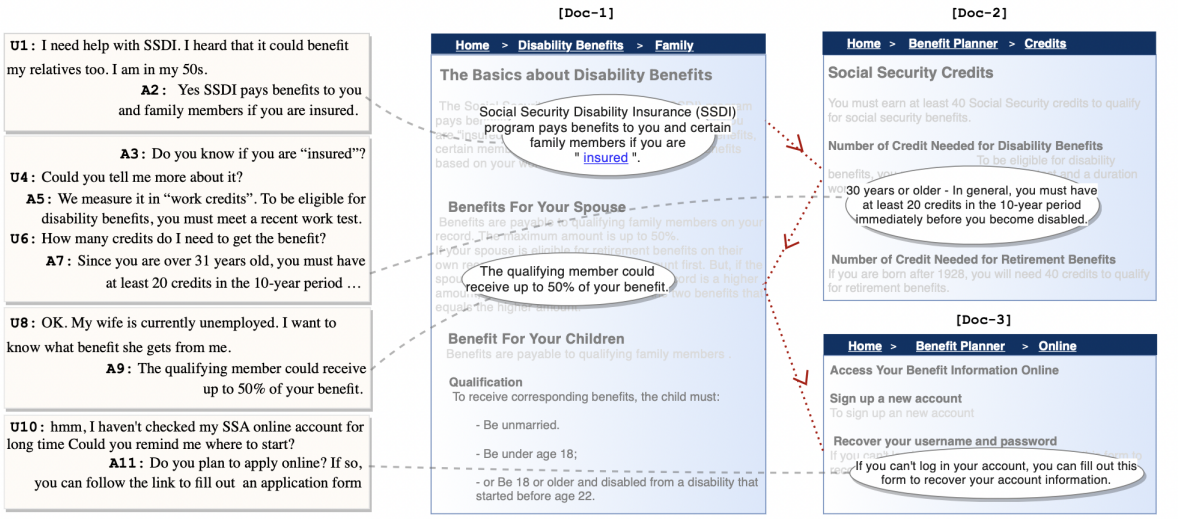


Figure 1.1 A sample goal-oriented dialogue (left) that is grounded in three relevant documents (center, right) from the MultiDoc2Dial dataset taken from Feng et al. (2021). The dialogue contains four segments where all turns within a segment are grounded in the same document. Dashed blue lines denote the document grounding of each segment, while dashed red lines signify transitions between documents.

Different techniques have been used for this task, but recently large language models (LLMs) have been shown to perform strongly in the domain. For example, the baseline model released with MultiDoc2Dial implements the Retrieval Augmented Generation (RAG)

framework (Lewis et al., 2020), which uses a document retriever (Karpukhin et al., 2020) with BART, a well-known LLM, as a response generator. While RAG has been a dominant paradigm for augmenting LLMs with knowledge, fine-tuning is necessary to ensure faithfulness to retrieved information, which can be costly and impractical for larger LLMs. Fortunately, the potential for new LLM approaches to this task has emerged with the release of LLMs with even more parameters (Brown et al., 2020; Chowdhery et al., 2022; Hoffmann et al., 2022) that have shown the unique ability to perform well on downstream tasks without fine-tuning. This project aims to establish techniques to adapt these large-scale LLMs, specifically GPT-3, and the commonly associated approaches such as few-shot prompting (Brown et al., 2020), Chain-of-Thought reasoning (Wei et al., 2022), and tool use (Schick et al., 2023; Yao et al., 2022) to advance the faithfulness of existing systems. Although these approaches have found success in other contexts, their application in dialogue systems remains relatively unexplored.

Finally, while new approaches associated with GPT-3 present potential for furthering research in task-oriented dialogue, their use requires a complete restructuring of the evaluation framework for this task. Past research relied on metrics that measured word overlap (Banerjee and Lavie, 2005; Lin, 2004; Post, 2018; Rajpurkar et al., 2016) which have become less meaningful with the increased verbosity and human-like nature of GPT-based responses and also fail to evaluate the common hallucination issue in LLMs. Therefore, we set out to define a robust and meaningful evaluation framework that provides insight into the faithfulness, factual correctness and linguistic quality of proposed systems which will guide our system development.

1.2 Contributions

We summarize our main contributions as follows:

1. A comprehensive review of approaches and terminology used for augmenting LLMs with external data which is necessary for a field that is evolving by the day.

2. An improved retrieval mechanism implementing a reranking system that involves no fine-tuning and outperforms a fine-tuned DPR retriever (Karpukhin et al., 2020).

3. An improved algorithmic approach built on ReAct (Yao et al., 2022), which has yet to be extended to dialogue systems within published work, which demonstrate **higher retrieval accuracy**, more **favourable linguistic qualities**, and **fewer hallucinations** than existing systems.

4. A robust and meaningful evaluation framework that measures three separate facets: **linguistic quality** (i.e. naturalness, coherence, understandability, and engagingness), **faithfulness**, and **factual correctness**.

5. An implementation of an LLM self-evaluation framework that leverages GPT-4 and thoughtfully designed few-shot demonstration which adds to the emerging and growing literature on LLMs as evaluators.

6. A carefully designed, comprehensive, and publicly available human evaluation framework that evaluates our two best systems and draws a comparison to existing systems. This evaluation can be used by future researchers leveraging the MultiDoc2Dial corpus.

1.3 Overview

The rest of the thesis is structured as follows:

In **Chapter 2** we introduce key technical concepts relating to LLMs and outline recent research for the task of augmenting task-oriented dialogue systems with external knowledge.

Chapter 3 frames the task, justifying our choice of the dataset, describing the dataset in more detail, and addressing our restructured approach for evaluating proposed systems.

We provide an overview of different response generation systems that we will explore in **Chapter 4**.

Chapter 5 reports the results from experimentation, highlighting our improved approaches, and includes the results of our LLM self-evaluation and human evaluation frameworks.

To conclude, **Chapter 6** summarizes the most important outcomes of the project and provides direction for future work.

Chapter 2

Fundamental Large Language Model Concepts

In this section, we contextualize this thesis by outlining foundational concepts, namely transformer models and prompt engineering approaches, and then perform a comprehensive review of the relevant literature on augmenting LLMs with external knowledge.

2.1 Transformer models

Transformers, a revolutionary neural network architecture, are fundamental to our work. Introduced in the paper "Attention Is All You Need" by Vaswani et al. (2017), transformers are designed to handle sequential data, such as language, by employing a novel attention mechanism. Unlike traditional recurrent neural networks (RNNs) that process input sequentially, transformers use self-attention to capture relationships between all tokens in the input sequence simultaneously. The attention function is defined as

$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

where Q, K, and V are matrices representing a query, key, and value respectively, and contain the vector representation of each word in the sequence. Put simply, the weights that will be assigned to V are calculated as being to what extent every word in a sequence Q is influenced by all the other words K of the sequence. The division by the root of d_k, the length of the sequence, ensures that long sequences don't push the results towards too small of gradients. This approach allows transformers to consider long-range dependencies efficiently, enabling them to capture context and meaning effectively across sentences and paragraphs.

Transformers consist of encoder and decoder layers, shown in Figure 2.1, both of which leverage the attention mechanism making them suitable for sequence-to-sequence tasks, including text generation. The encoder uses the self-attention mechanism to process the input sequence, $\mathbf{x}$, and generate a context-aware representation. The decoder takes this representation and produces the output sequence, aligning input and output effectively using another attention mechanism. The model is auto-regressive, meaning that it consumes the previously generated symbols as additional input when generating the next symbol.

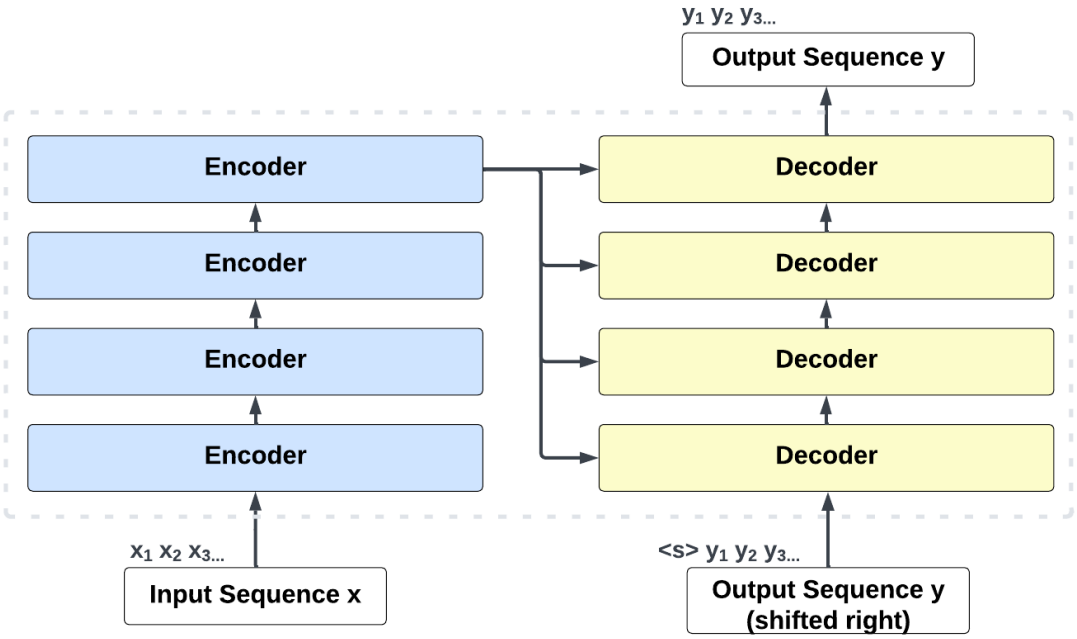


Figure 2.1 Encoder-Decoder structure of Transformers.

Due to their ability to parallelize computations, effectively process long sequences, and leverage self-supervised training techniques, transformers have become the foundation for state-of-the-art LLMs. There are four transformer-based LLMs that are particularly relevant to our work: BERT, BART, GPT-3, and GPT-4. We describe these models in the following subsections while highlighting key similarities and differences between them.

2.1.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a transformer model introduced by Devlin et al. (2018). This model is an *encoder-only* transformer pre-trained on data from Wikipedia and Google's BooksCorpus in an unsupervised manner using masked

language modeling and next-sentence prediction objectives. Because it is comprised of strictly encoder blocks, the model outputs a continuous contextualized embedding corresponding to each input token. The model can be fine-tuned by adding a single additional output layer and using labeled text to fine-tune for downstream NLP tasks. $\text{BERT}_{\text{base}}$ contains 110 million parameters and $\text{BERT}_{\text{large}}$ contains 340 million parameters, but this is the smallest model out of the four transformer models that we utilize in our work.

2.1.2 BART

BART (Bidirectional and Auto-Regressive Transformers) was introduced in Lewis et al. (2019) and is another model that is particularly effective when fine-tuned for downstream NLP tasks, especially text generation.

Unlike BERT's encoder-only design, BART's architecture is a sequence-to-sequence model, featuring both encoder and decoder components, enabling it to generate text. During training, BART undergoes a unique two-step process. First, the text is corrupted with a noising function. This builds off the token masking approach in BERT by applying additional techniques such as sentence reconstruction and shuffled tokens. Next, a sequence-to-sequence model is learned to reconstruct the original text. This approach generalizes the original word masking and next sentence prediction objectives in BERT by forcing the model to reason more about overall sentence length and make longer-range transformations to the input. In addition to differences in architecture, BART differs from BERT in terms of size as $\text{BART}_{\text{base}}$ contains 139 million parameters and $\text{BART}_{\text{large}}$ contains around 400 million parameters.

2.1.3 GPT-3 and GPT-4

GPT (Generative Pre-trained Transformer) models are a series of large-scale, transformer models developed by OpenAI. These models are *decoder-only* models, pre-trained in a self-supervised fashion to perform next-token prediction. GPT-3 has 175 billion parameters which is over 400 times the size of $\text{BART}_{\text{large}}$, while GPT-4 has 70 trillion parameters. Although GPT-4 is larger and therefore more powerful than GPT-3, we will primarily be using GPT-3, specifically gpt-3.5-turbo on the OpenAI API[1], for experimentation and only use GPT-4, gpt-4 on the OpenAI API[2], for our LLM self-evaluation framework (Section 3.3.3). We primarily use GPT-3 as it is both a capable and cost-effective GPT model, whereas GPT-4 has much higher costs to users.

[1] https://platform.openai.com/docs/models/gpt-3-5
[2] https://platform.openai.com/docs/models/gpt-4

In addition to the difference in architecture and size from BERT and BART, GPT-3 and GPT-4 are trained on an extremely large corpus of English text data extracted from millions of web pages and then fine-tuned using Reinforcement Learning from Human Feedback (RLHF) which improves truthfulness and reduces toxic output generation (Ouyang et al., 2022). This increase in size and use of human-annotated data allows these models to output highly fluent and human-like responses. Additionally, these features equip the models with unprecedented generalization abilities, meaning that GPT-3 and GPT-4 can perform well given several or no examples at run-time even when no fine-tuning is performed. Although BERT and BART have some ability to generalize to unseen examples, their primary strength lies in fine-tuning for specific tasks with more labeled examples. This groundbreaking generalization behavior is called zero-shot and few-shot learning and is the foundation for prompt engineering approaches explained in the following section.

2.2 Prompt Engineering

In reviewing the transformers pertinent to this project, we observe that GPT-3 and GPT-4 are much larger than BERT and BART models and therefore have capabilities that go beyond the fine-tuning approach popular with BERT and BART. Although GPT-3 can be fine-tuned to improve performance, this approach is limited by computational resources and unavailable model weights (Hu et al., 2021). Prompt Engineering directly addresses these limitations.

Prompt Engineering refers to methods for communicating with LLMs to elicit desired behavior without updating the model weights (Weng, 2023) and is a defining new paradigm for leveraging these models. Although GPT-3 may have popularized these prompt engineering techniques (Brown et al., 2020), other models that fit in this LLM category, such as PaLM with 540 billion parameters (Chowdhery et al., 2022) and Chinchilla with 70 billion parameters (Hoffmann et al., 2022), are also capable of these generalization techniques. We chose GPT-3 over these other models due to its accessibility, popularity among researchers, and ease of use.

Two important prompt engineering techniques that will be fundamental in the existing and proposed GPT-3 systems are zero-shot/few-shot prompting and Chain-of-Thought reasoning.

2.2.1 Zero-Shot and Few-Shot Learning

One of the most powerful features of GPT-3 is that it can perform new tasks that it has never been trained on by showing it a set of high-quality demonstrations, each consisting of both input and desired output, on the target task. This is called few-shot learning, popularized by Brown et al. (2020), which helps LLMs transfer to new tasks via inference alone by

conditioning on concatenation of demonstrations and test input, without any gradient updates. The demonstrations consist of *k* text examples. When k=0 this is called zero-shot learning and consists of simply feeding the task text to the model and asking for results. Zero-shot learning can still achieve state-of-the-art results when used with GPT-3 in many tasks, but few-shot learning often leads to better performance. However, few-shot learning comes at the cost of more token consumption and may hit the context length limit when input and output text are long (Weng, 2023). Since our task requires both the dialogue history and relevant context information as inputs to the LLM, we will primarily be restricted to the zero-shot setting due to the large number of input tokens necessary even without in-context examples.

2.2.2 Chain-of-Thought Reasoning

Chain-of-thought (CoT) reasoning (Wei et al., 2022) builds on zero/few-shot learning by using these methods to guide an LLM to generate a sequence of short sentences describing reasoning logic step by step to eventually lead to the final answer. The few-shot CoT approach is illustrated in Figure 2.2 (right) and is compared to standard few-shot prompting (left). Zero-shot CoT prompting techniques involve adding a natural language statement to the prompt to illicit reasoning, such as 'Let's think step by step' (Kojima et al., 2022). CoT demonstrates performance gains over standard prompting when applied to complicated reasoning tasks with a model of 100 billion parameters or more (Wei et al., 2022). Since augmenting task-oriented dialogue systems with external knowledge is considered a complicated reasoning task, CoT will prove to be a useful technique for proposed systems.

2.3 Augmented Large Language Models

Now that we have explored transformers and prompt engineering techniques as the building blocks for existing and proposed systems, we can shift to reviewing recent literature on incorporating external knowledge into task-oriented dialogue systems.

In our review of transformer models, we observed how LLMs are pre-trained on extensive amounts of data. This allows these models to learn in-depth knowledge from the data (Petroni et al., 2019) and act as parameterized implicit knowledge bases (Roberts et al., 2020), but research has shown that these models cannot access new or private knowledge, fail to provide insight into their predictions, and often generate plausible looking statements that are factually incorrect (Creswell et al., 2022; Maynez et al., 2020; Roller et al., 2020; Welleck et al., 2019; Ye and Durrett, 2022). These incorrect responses are even more dangerous due to the fluency and realistic nature of LLM responses, as they can convince the user of non-factual statements.

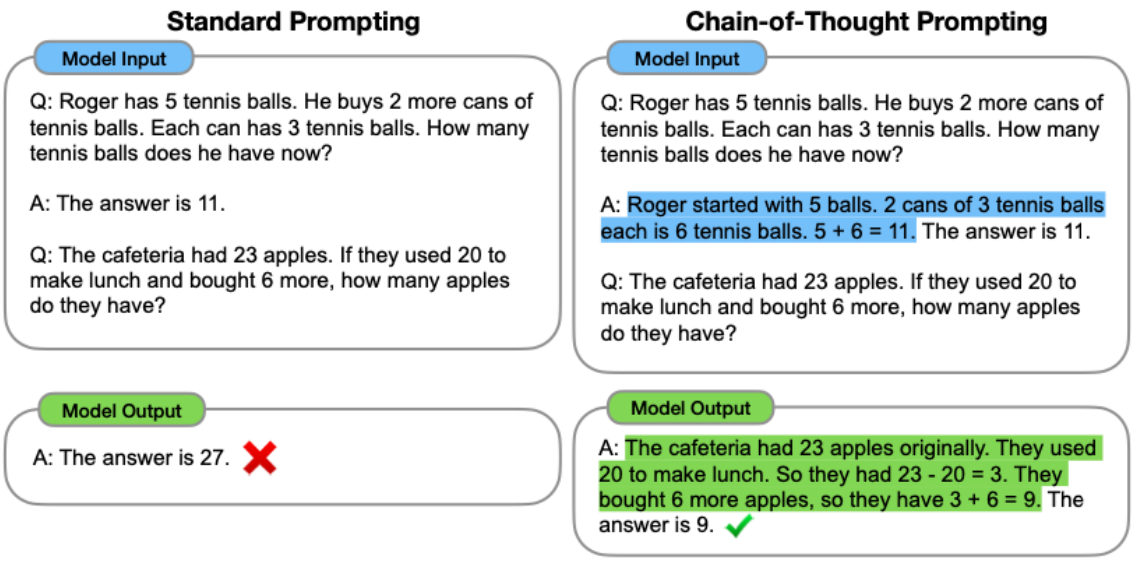


Figure 2.2 A comparison of standard prompting techniques (left) with Chain-of-Thought prompting technique (highlighted on right). The figure is taken from Wei et al. (2022).

A promising solution to these issues is designing systems where generative models learn to interact with the external world in retrieving knowledge, augmenting their own generation abilities (Mialon et al., 2023). Such techniques include the Retrieval Augmented Generation (RAG) and newer GPT-based approaches built on few-shot learning.

2.3.1 Retrieval Augmented Generation Framework

Retrieval Augmented Generation (RAG), an end-to-end fine-tuned architecture that leverages BERT and BART, has been the the dominant paradigm for augmenting LLMs with external knowledge since it was proposed in Lewis et al. (2020). This model uses a retriever-generator architecture, an extension of retriever-reader architecture (Guu et al., 2020; Lee et al., 2019). RAG combines pre-trained parametric (LLM) and non-parametric memory (external knowledge index) together for language generation and trains the system in an end-to-end manner. The retriever aims to retrieve the top-n most relevant document passages given the query. Then, the generator takes these top-n document passages along with the query as inputs to generate the agent response.

This architecture has been primarily studied in the context of open-domain question answering (QA) but has been shown to generalize to QA in task-oriented dialogue (Feng et al., 2021; Shuster et al., 2021). To adapt this framework for our task, we construct the query so that it is the concatenation of the dialogue history and current user turn. An illustration of this

framework is shown in Figure 2.3 and the retrieval, generation, training, and decoding steps are explained further below.

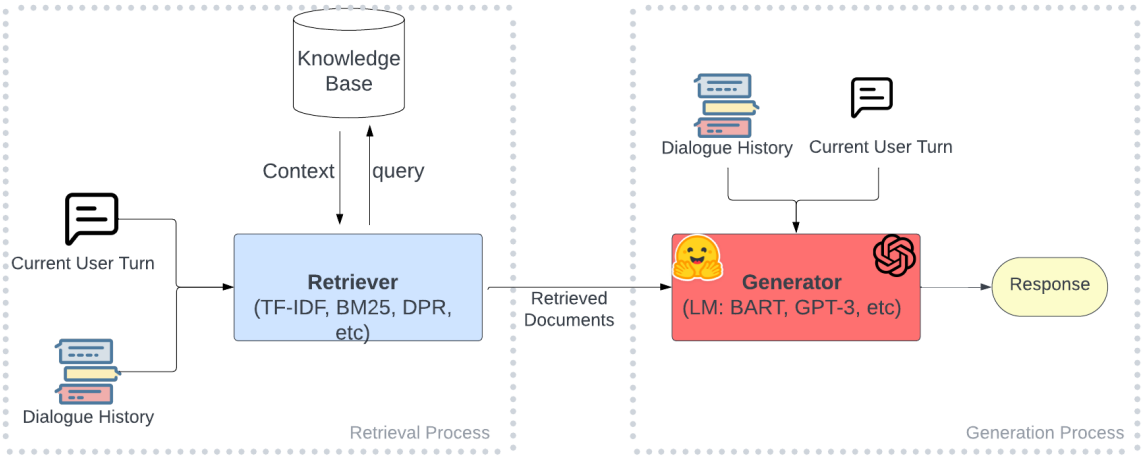


Figure 2.3 Retriever-Generator Framework combines a document retrieval system with a general LLM. In the case of task-oriented dialogue QA, the query is a concatenation of the dialogue history and current user turn.

Retriever

Within the RAG paradigm, the retriever, $p_\eta(z|x)$, has parameters η and returns a (top-K truncated) distribution over text passages, z, given a query, x.

There are several different retrievers that can be implemented. Traditional retrievers include TF-IDF or BM25 (Robertson et al., 2009) which match keywords efficiently with an inverted index. For these retrievers, x and z are both represented in high-dimensional, sparse vectors. These sparse retrievers suffer from the weakness that they cannot encode semantic knowledge. To alleviate this problem, Karpukhin et al. (2020) proposes a Dense Passage Retrieval (DPR) that uses dense vector representations for x and z based on contextual embeddings and shows that this method significantly outperforms BM25. Consequently, this is the retriever used in RAG.

DPR leverages a dual-encoder architecture (Bromley et al., 1993) comprised of a BERT-based question encoder to create a dense representation of the query, $\mathbf{q}(x)$, and another BERT-based context encoder for the document passages representation, $\mathbf{d}(z)$:

$$\mathbf{d}(z) = \text{BERT}_d(z) \tag{2.2}$$

$$\mathbf{q}(x) = \text{BERT}_q(x) \tag{2.3}$$

In order to retrieve k-top passages a notion of similarity must be defined. Lewis et al. (2020) defines the similarity between the query and document passage by using the exponential of the dot product between vectors:

$$p_\eta(z|x) \propto \exp(\mathbf{d}(z)^T \mathbf{q}(x)) \tag{2.4}$$

The document encoder is used before run-time to compute an embedding for each document passage and then a single MIPS index is built using FAISS (Johnson et al., 2019). At run time, the query encoder encodes the query and the k passages of which vectors are the closest to the query vector are retrieved.

Generator

Within the RAG framework $\text{BART}_{\text{large}}$ is used as the generator. BART generates a current token based on a context of the previous $i-1$ tokens, $y_{1:i-1}$, the original input, x, and the top-K retrieved passages, z:

$$p_\theta(y_i|x, z, y_{1:i-1}) \tag{2.5}$$

Training and Decoding

Finally, to train the generator along with the previously defined retriever in an end-to-end manner, the retrieved documents are treated as latent variables. There are two methods for marginalizing over the latent documents, RAG-Sequence uses the same document passage to predict each target token while RAG-token can predict each target token based on a different document passages. RAG-token is more appropriate for the dialogue context as agent responses might need to incorporate context from multiple passages of a document. Formally RAG-token is defined by:

$$p_{\text{RAG-Token}}(y|x) \approx \prod_i^N \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z|x) p_\theta(y_i|x, z, y_{1:i-1}) \tag{2.6}$$

During training, there is no direct supervision on what passage should be retrieved. Rather, for each input/output pair (x_j, y_j) the model is trained by minimizing the negative marginal log-likelihood of each target, $\sum_j -\log p(y_j|x_j)$, using stochastic gradient descent with Adam. During training the query encoder and BART are updated while the document encoder (and index) are held constant.

At test time we must approximate arg $\max_y p(y|x)$. Since RAG-token can be seen as a standard, auto-regressive seq2seq generator with transition probability

$$p'_\theta(y_i|x, y_{1:i-1}) = \sum_{z \in \text{top-}k(p(\cdot|x))} p_\eta(z_i|x) p_\theta(y_i|x, z_i, y_{1:i-1}) \tag{2.7}$$

to decode, we can plug $p'_\theta(y_i|x, y_{1:i-1})$ into a standard beam decoder.

2.3.2 GPT-3 Approaches: Use of Zero-Shot Learning and Tools

The RAG architecture and similar models (Borgeaud et al., 2022; Guu et al., 2020; Izacard and Grave, 2020) have shown strong performance in downstream, knowledge-intensive tasks, such as QA in task-oriented dialogue, but this model must be fine-tuned in order to remain faithful to retrieved information. Because of this, there is an emerging field of research that has shifted away from this end-to-end trained and fine-tuned architecture, to approaches that leverage the few-shot capabilities of larger scale LLMs to augment LLMs with knowledge that is not necessarily stored in their weights (Mialon et al., 2023; Zhao et al., 2023a). These newer approaches still follow the retriever generator framework illustrated in Figure 2.3, namely, relevant information is retrieved and then used for response generation.

Augmentation with Zero-shot Prompting

The first, and most simple, GPT-based approach that we will explore in our work is leveraging GPT-3 as a zero-shot learner by engineering our prompt to contain both the query (dialogue history and current user turn) as well as retrieved information output from a separate retrieval process, such as DPR. This process can be used to 'freeze' the retrieval process and isolate the effects of implementing different generators (i.e. BART vs. GPT-3).

Use of Tools

Another approach to augmenting GPT-3 with external knowledge is using tools, such as calculators and search engines, which are external modules typically invoked by a rule or special token and have their output integrated into the LLM's context (Mialon et al., 2023).

Research on the use of tools for augmenting LLMs is split into two sub-categories. The first approach is to train LLMs to decide when to call tools. LLMs of this type include Toolformer (Schick et al., 2023), WebGPT (Nakano et al., 2021), Atlas (Izacard et al., 2022), LlaMDA (Thoppilan et al., 2022), and BlenderBot (Shuster et al., 2022b). The second approach, and the approach that we will build upon in this thesis, is leveraging CoT reasoning to decide when to

call tools and subsequently incorporate the retrieved knowledge. This line of work is growing in importance because as LLMs continue to evolve and advance, researchers need to formulate methods that can be adaptably utilized with emerging models, instead of consistently training or fine-tuning novel architectures.

There are several works that have recently combined the use of CoT and tools (He et al., 2022; Trivedi et al., 2022; Zhao et al., 2023b), but the most popular is ReAct (Yao et al., 2022). Importantly, all cited works that leverage this approach apply this technique to either open-domain QA or fact-verification tasks. However, applying these models to the more complex domain of knowledge-grounded dialogue remains unexplored in published work, and is only beginning to be explored in open-source codebase development. We chose ReAct to build upon due to its popularity among researchers and integration into chat engines in open-source codebases[3].

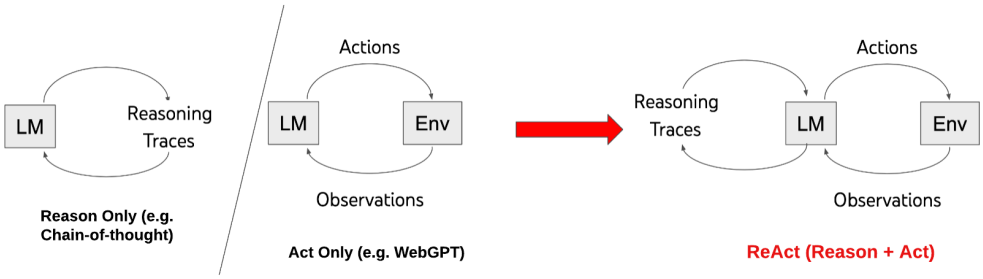


Figure 2.4 Comparison of Chain-of-Thought and Agent frameworks (left) to the ReAct framework (right). The original diagram can be found on the blog post released with Yao et al. (2022).

ReAct integrates CoT prompting with tool use to create a model that can both Reason and Act (ReAct) (Figure 2.4). The core concept of this approach is that LLMs use reasoning to integrate various tools for complex tasks. Tools, in turn, draw from external sources, enhancing reasoning and preventing failures, creating a mutually beneficial relationship (Mialon et al., 2023). ReAct uses the same few-shot learning technique that was used in the original CoT paper (Wei et al., 2022), where a frozen LLM is prompted with 3-6 few-shot in-context examples to generate both domain-specific actions and free-form language thoughts to solve the task. A snippet of a few-shot example from the paper is shown in Figure 2.5. ReAct is equipped with three sets of tools, search[entity] returns the first 5 sentences from the entity's wiki page, lookup[string] returns the next sentence in the wiki page containing string, and finish[answer] outputs the final answer. We will slightly modify this approach to use QueryEngine[query] which will use a document retriever to retrieve the top-k relevant documents to a query.

[3]https://gpt-index.readthedocs.io/en/latest/examples/chat_engine/chat_engine_react.html

```
ReAct_FewShot = (
    '''
    Question: What is the elevation range for the area that the eastern sector of the
    Colorado orogeny extends into?
    Thought 1: I need to search Colorado orogeny, find the area that the eastern sector
    of the Colorado orogeny extends into, then find the elevation range of the
    area.
    Action 1: Search[Colorado orogeny]
    Observation 1: The Colorado orogeny was an episode of mountain building (an orogeny) in
    Colorado and surrounding areas.
    Thought 2: It does not mention the eastern sector. So I need to look up eastern
    sector.
    Action 2: Lookup[eastern sector]
    Observation 2: (Result 1 / 1) The eastern sector extends into the High Plains and is called
    the Central Plains orogeny.
    Thought 3: The eastern sector of Colorado orogeny extends into the High Plains. So I
    need to search High Plains and find its elevation range.
    Action 3: Search[High Plains]
    Observation 3: High Plains refers to one of two distinct land regions
    Thought 4: I need to instead search High Plains (United States).
    Action 4: Search[High Plains (United States)]
    Observation 4: The High Plains are a subregion of the Great Plains. From east to west, the
    High Plains rise in elevation from around 1,800 to 7,000 ft (550 to 2,130
    m).[3]
    Thought 5: High Plains rise in elevation from around 1,800 to 7,000 ft, so the answer
    is 1,800 to 7,000 ft.
    Action 5 Finish[1,800 to 7,000 ft]
    ...

    '''
)
```

Figure 2.5 A snippet of a few-shot demonstration used for the original ReAct implementation in Yao et al. (2022)

In this Background section, we explored recent work, such as RAG and ReAct, that aims to advance the quality of task-oriented dialogue systems for QA. However, even augmented LLMs exhibit hallucinations (Ye and Durrett, 2022). These hallucinations may be even more dangerous because LLM predictions based on tools may look more trustworthy and authoritative, when in fact many of them still may be incorrect or unsupported by retrieved information (Mialon et al., 2023). Therefore, it is crucial to advance the field of LLM faithfulness, accuracy, and quality within the augmented LLM framework.

Chapter 3

Framing the Task: Dataset and Evaluation Framework

In order to design more useful and faithful systems, we must first precisely define the task for which these systems are intended. This entails providing an overview of the task, detailing the dataset, and introducing the proposed evaluation framework.

3.1 The Task

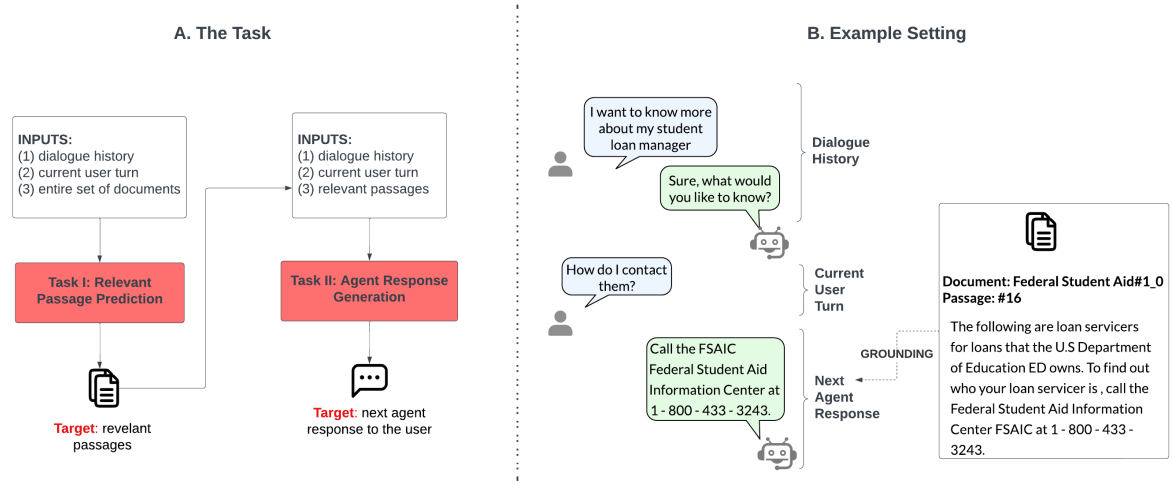


Figure 3.1 A general overview of the task with inputs and outputs (left) and an example of the task being performed (right).

The task is split into two subtasks: Relevant Passage Prediction and Agent Responses Generation. The input for Task I is the (1) dialogue history, (2) current user turn, and (3) entire

set of documents available to the LLM. The goal of this subtask is to predict the passages relevant to the next agent response which not only provides detailed grounding information for the agent response but also increases the interpretability and trustworthiness of the system. The inputs for Task II are the (1) dialogue history, (2) current user turn, and (3) the relevant passages from Task I. This task is more difficult since the agent utterance varies in style and cannot be directly extracted from the retrieved document (Feng et al., 2021). Figure 3.1 gives a clear outline of this framework (left) and provides an example of the task being performed (right).

3.2 The Dataset

Having outlined the real-world tasks our systems are designed to handle, we need to identify a dataset that encompasses all the elements shown in Figure 3.1.

3.2.1 Choice Rationale

As explained in the introduction, much of the work in augmenting LLMs with external knowledge has framed the task as a question-answering task using open-domain QA datasets (Baudiš and Šedivỳ, 2015; Berant et al., 2013; Joshi et al., 2017; Kwiatkowski et al., 2019; Yang et al., 2018). This work fails to include the dialogue history as input to Task I or Task II. The example setting in Figure 3.1 shows that the task-oriented dialogue for QA task is more challenging as the agent must understand what the pronoun 'them' is referring to in the last user response and also generate an agent response that is coherent in the context of the dialogue history.

Responding to this limitation, several works have created task-oriented dialogue datasets that are grounded in external information, such as QuAc (Choi et al., 2018), ShARC (Saeidi et al., 2018), CoQA (Reddy et al., 2019), OR-QuAC (Qu et al., 2020), and Doc2dial (Feng et al., 2020). Although this work is more closely aligned with our task, these datasets are formulated as machine reading comprehension tasks corresponding to a dialogue history but only *a single document or text snippet* (i.e. instead of a set of documents for input to Task I, only a single document would be given). Therefore, these datasets are not representative of the complex information retrieval tasks required in many real-world scenarios where several subgoals are addressed in different documents.

There are very few datasets that require a system to both take into account the **dialogue history** and read **multiple documents** to generate knowledge-grounded agent responses. To our knowledge, the existing datasets for this particular task are MultiDoc2Dial (Feng et al.,

2021), TopicalChat (Gopalakrishnan et al., 2019), Wizards of Wikipedia (WoW)[1] (Dinan et al., 2018), and CMU-DoG (Zhou et al., 2018). We selected MultiDoc2Dial because it uniquely possesses two key characteristics: it grounds dialogues in documents beyond Wikipedia and introduces a robust benchmark model based on the dataset. The benefits of these characteristics are explained below.

3.2.2 MultiDoc2Dial

MultiDoc2Dial includes approximately 4,800 dialogues with an average of 14 turns that are grounded in 488 documents from four different domains, namely the Department of Motor Vehicles, Social Security Administration, Veterans Affairs, and Federal Student Aid. The dialogues are annotated with information about the documents that were used to respond to each user utterance. By grounding dialogues in documents from public government service websites, MultiDoc2Dial is more generalizable and aligned with real-world settings than datasets grounded in strictly Wikipedia data. Moreover, the documents used in MultiDoc2Dial are on average much longer than the Wikipedia data used in comparable datasets, making the task of MultiDoc2Dial more realistic. In addition to the dataset, Feng et al. (2021) also releases a benchmark system based on RAG that involves retrieving relevant context information *without* access to the gold context or gold response (benchmark systems in TopicalChat and CMU-DoG utilize gold information making them less robust). This allows us to compare our prompt-tuned GPT-3 model to a robust fine-tuned BERT/BART architecture.

Within MultiDoc2Dial there are 3,474 dialogues in the train set and 661 in both the validation and test set. The creators of MultiDoc2Dial also release a related shared task competition hosted by the Second DialDoc Workshop on Documented-grounded Dialogue and Conversational Question Answering[2], where the systems are evaluated on a smaller version of the MultiDoc2Dial test set in which one turn is randomly selected from each of the 661 dialogues. We will be using the shared task dataset for experimentation unless otherwise specified due to resource constraints.

3.3 Evaluation Framework

Now that we have defined *what* we are evaluating we need to define *how* to evaluate it. We demonstrate that the baseline generation evaluation metrics used for MultiDoc2Dial are inadequate for measuring the overall quality of generated responses and then propose an improved,

[1] Dziri et al. (2022) releases FaithDial, an improved version of WoW

[2] an overview of the results from the shared task is presented in Feng et al. (2022)

multidimensional evaluation framework that leverages automated, LLM self-assessment, and human evaluation.

3.3.1 Baseline Evaluation Metrics: Shared Task Evaluation

As a starting point, we will describe the baseline evaluation metrics used by the authors of MultiDoc2Dial. Note that the original MultiDoc2Dial paper approaches Task I and Task II as completely separate tasks, evaluating both retrieval and generation for both tasks. This framework measures generation on a grounding span prediction from retrieved documents in Task I and on the next agent response output in Task II. The shared task competition, on the other hand, only scores generation metrics on the next agent response, while still using Task I to score intermediate retrieval. We adopt the shared task framework as we are focused on retrieval quality in Task I and generation quality in Task II.

Retrieval

The baseline evaluation metric used for the retrieval component of the MultiDoc2Dial task is recall (@k) which measures the fractions of times the correct document is found in the top-k predictions. Feng et al. (2021) reports recall @1, @5, and @10 for both passage level retrieval, where the target is the correct passage id, and document level retrieval, where the target is the correct document title. Since the process for splitting documents into passages differs across our implemented approaches, we will report document-level recall throughout our experimentation. This metric allows us to gain insight into the factual accuracy of systems, as higher retrieval should lead to more factual responses if systems are successfully grounding answering in retrieved context.

Generation

In the shared task, next agent response outputs are evaluated based on F1, SacreBLEU (Post, 2018), METEOR (Banerjee and Lavie, 2005), and RougeL (Lin, 2004) scores [3]. These metrics all measure the similarity between the generated answer and the ground truth answer, or 'gold' answer, using some formulation of word overlap.

- **F1** combines precision and recall. Precision is the ratio of common tokens between generated and gold answers to tokens in the generated answer. Recall is the ratio of

[3] the original MultiDoc2Dial paper uses F1 score, Exact Match, and SacreBLEU

common tokens to tokens in the gold answer.

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{3.1}$$

- **SacreBLEU** evaluates text generation by computing n-gram precision between a candidate translation and reference translations. It incorporates various n-gram lengths to capture phrasal and sentence similarities and introduces a brevity penalty to penalize overly short outputs that may boost precision artificially.
- **METEOR** evaluates text generation using unigram precision and recall, considering exact word matches, synonyms, and variations in word order. It aligns the candidate with the reference translations and computes the score as the harmonic mean of its precision and recall metrics.
- **RougeL** evaluates text generation by identifying the longest common sub-sequence of words (in order, but not necessarily consecutive) between the model output and reference, with longer shared sequences indicating higher similarity.

While baseline retrieval metrics provide insight into factual accuracy, the generation metrics, although widely used as the predominant means of assessment within QA (Cheng et al., 2023; Joshi et al., 2017; Kwiatkowski et al., 2019; Rajpurkar et al., 2016) and document-grounded task-oriented dialogue (Choi et al., 2018; Feng et al., 2020; Schick et al., 2023), are increasingly seen as insufficient for evaluating system responses (Dinan et al., 2018; Mehri and Eskenazi, 2020; Shuster et al., 2021). The reasons for this are twofold. First, they fail to identify hallucinations. A hallucinated response may match the overall length, structure, and wording of the gold response, leading to high similarity-based scores even with factually incorrect and hallucinated information (Figure 3.2). Second, as LLM scale in size, output tends to become more verbose, especially in comparison to the gold reference for MutliDoc2Dial, which leads to lower scores for longer, but still correct, responses (Figure 3.3).

3.3.2 Improved Automatic Metrics: UniEval

Various recent works have attempted to alleviate the issues associated with word overlap metrics by leveraging contextualized embeddings from pre-trained models such as BERT to measure embedding-based similarity, (Clark et al., 2019b; Zhang et al., 2019; Zhao et al., 2019), but this still fails to address the issue that similarity to the reference may not indicate the overall quality of the output. As a response to these shortcomings, *single-dimensional* evaluator models have been trained to specifically evaluate the degree of hallucination. Examples include FaithCritic

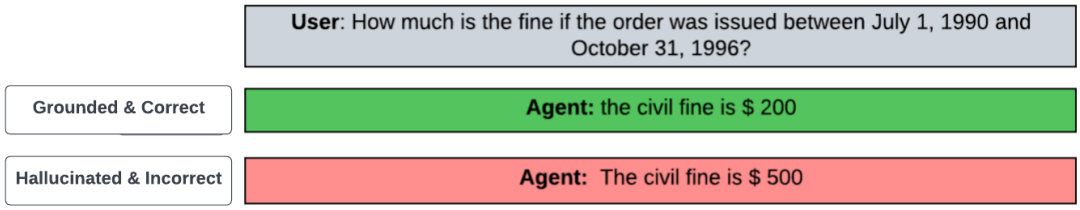


Figure 3.2 Example illustrating the limitations of similarity-based scores in identifying hallucination. Here, the system would receive high similarity-based scores for both responses despite the red response being grounded in incorrect information outside of the retrieved passage.

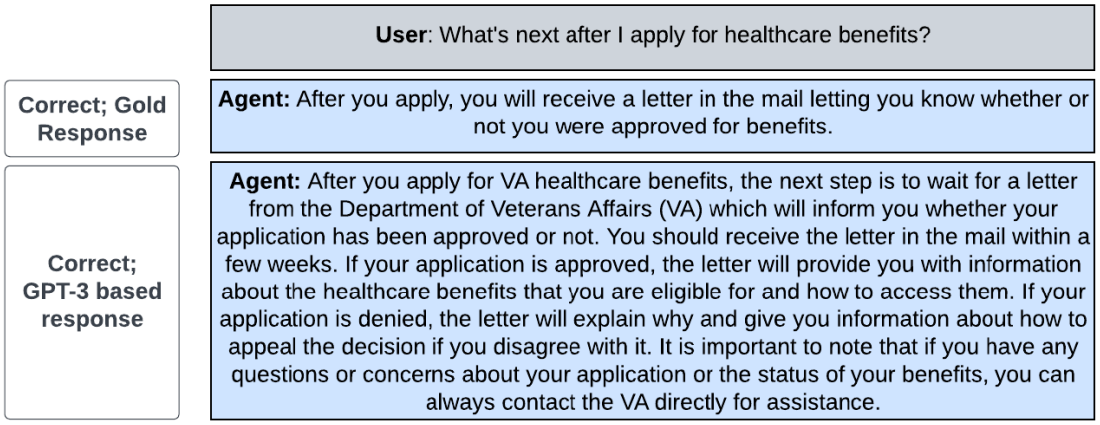


Figure 3.3 Example illustrating the limitations of similarity-based scores in penalizing verbosity. Here, the system would receive a low similarity-based score due to the verbosity of the GPT-3 response despite being grounded, correct, and well formatted.

(Dziri et al., 2022) and Q^2 (Honovich et al., 2021) which train models to return a percentage of utterances identified as unfaithful. Although we attempted to leverage both FaithCritic and Q^2 in our evaluation framework, these models use RoBERTa (Liu et al., 2019) as a pre-trained model and therefore only allow context with 512 or fewer tokens. As our retrieved passages often surpass this length, these were not useful metrics for our task. Fortunately, Zhong et al. (2022) builds on the concept of *single-dimensional* evaluators and proposes a unified *multi-dimensional* evaluator that outputs scores for **naturalness**, **coherence**, **engagingess**, and **groundedness**. They also release a single-dimensional evaluator for **factual consistency** that handles longer context token lengths (1024 tokens).

UniEval unifies all evaluation dimensions into a Boolean Question Answering problem (Clark et al., 2019a), meaning that each dimension (d_i) of the evaluation is associated with a question (q_i) that can be answered with yes or no. The question associated with each dimension of the evaluator is shown in Table 3.1. With the evaluation formulated as a Boolean QA problem, we have to evaluate our 5 dimensions $\mathbf{d} = (d_1, ..., d_5)$ of the model output. The model input includes the generated output, x, the context, c, and the dialogue history, h. The trained UniEval model outputs scores $\mathbf{s} = (s_1, ..., s_5)$ where s_i is calculated as:

$$s_i = \frac{P(\text{"}Yes\text{"}|x,c,h,q_i)}{P(\text{"}Yes\text{"}|x,c,h,q_i) + P(\text{"}No\text{"}|x,c,h,q_i)} \tag{3.2}$$

Engagingness is the only dimension that uses summation scores, as it indicates the total volume of interesting fact presented in the response, so it is in the range [0, +∞), while all others are [0, 1] for which we multiple by 100 to produce a percentage value when reporting results.

Zhong et al. (2022) shows that this evaluation framework outperforms USR (Mehri and Eskenazi, 2020), the state-of-the-art unified evaluator in the dialogue response generation task, in terms of correlation with human-annotated responses. We also show in Section 5.5.2 that UniEval outperforms similarity-based metrics in terms of correlation to human-annotated responses from our human evaluation. Therefore, we conclude that the dimensions of the multi-dimensional evaluator are a robust and meaningful suite of metrics for our MultiDoc2Dial task. We use naturalness, coherence, understandability, and engagingness to evaluate the linguistic quality of generated responses, and groundedness and factual consistency will be used as faithfulness metrics within our evaluation framework.

3.3.3 LLM Self-Evaluation: SelfEval

Alongside UniEval, we present a framework called SelfEval and incorporate it into our proposed evaluation approach. This framework uses an LLM to determine if a response contains hallucinated information, drawing inspiration from the emerging, yet not extensively explored,

Dimension	**Question**
Naturalness	Is this a natural response in the dialogue?
Coherence	Is this a coherent response given the dialogue history?
Understandability	Is this response understandable in the context of the dialogue history?
Engagingness	Is this an engaging and informative response according to the dialogue history and fact?
Groundedness	Does this response use knowledge from the fact?

Table 3.1 Questions used to train each dimension of the UniEval model.

realm of using LLM models for self-evaluation. Prominent examples of this concept are Constitutional AI (Bai et al., 2022), where an LLM critiques and recommends revisions for model outputs, and Perez et al. (2022)'s study, in which LLMs assess samples produced during dataset creation.

While evaluation by human annotators remains the gold standard, the process can be both time-consuming and expensive, particularly when dealing with large-scale assessments. As a result, LLMs—especially models trained with human preferences via RLHF—offer a viable alternative during system development. Indeed, Zheng et al. (2023) shows that robust LLMs like GPT-4 closely align with both controlled and crowd-sourced human preferences, achieving an impressive agreement rate of over 80%, comparable to human consensus. Yet, despite its success and scalability, it is important to note that LLM self-evaluation has shown biases, including verbosity—favoring longer responses—and self-enhancement—favoring its own outputs (Bubeck et al., 2023; Zheng et al., 2023).

During the initial exploration of self-evaluating our systems with an LLM, we leverage gpt-3.5-turbo in a zero-shot setting but found that this framework showed difficulty evaluating edge cases. For example, it wrongly classified many follow-up questions as hallucinations and did not differentiate models[4]. Based on these observations and related work finding that few-shot prompts and GPT-4 perform best for self-evaluation (Bubeck et al., 2023; Svikhnushina and Pu, 2023; Zheng et al., 2023), we shifted to an approach that utilized this setting. Our final UniEval framework uses a prompt made up of

1. The definition of hallucination: 'The definition of a hallucination is when an agent response contains additional factual information that is unsupported by the reference or dialogue history. If the agent asks a follow-up question grounded in the reference or dialogue history or admits it does not have enough information to answer, this is not a hallucination.'

[4]This setting predicted hallucination scores of 52-54% for all proposed models evaluated with the approach.

2. 5 domain-specific examples for few-shot learning
3. dialogue history
4. retrieved context
5. the system response

To construct the few-shot examples we first classify the 661 shared task samples into 4 categories: 1) **clear fact** where the response contains a clear entity or short phrase, 2) **general fact** where the response contains factual information that cannot be expressed with a simple short phrase, for example providing instructions or explaining a topic, 3) **follow-up question**, and 4) **non-informative** where the response includes information such as a greeting or thank you. The counts for these categories are 167, 360, 135, and 5, respectively. For each different domain, we construct an example of hallucination and non-hallucination for a case of clear fact and general fact as well as providing an example of how follow-up questions are not hallucinations. The few-shot example are made available on this Supplementary Material GitHub Page. As these prompts include many tokens and we use GPT-4, the SelfEval framework is only applied to our benchmark and best models due to resource constraints.

3.3.4 Human Evaluation Design

In addition to evaluating models with UniEval and our SelfEval framework, we conduct a thoughtfully designed and comprehensive human evaluation. The purpose of this evaluation is two-fold. First, the human evaluation is designed to compare the system performance in terms of linguistic quality, faithfulness, and accuracy to the MultiDoc2Dial gold-standard responses. Although the automatic metrics measure these dimensions, they just serve as a proxy for human evaluation, which remains the gold-standard means for evaluating and validating the quality of system-generated responses. Secondly, we will use our human evaluation to confirm that the UniEval metrics are a better proxy for evaluating system quality as opposed to the shared task evaluation metrics.

For our human evaluation, we set up two different surveys on the Amazon Mechanical Turk (AMT) Sandbox. We chose this platform because it's web-based, making it convenient for participants to complete tasks either at work or home, offers a user-friendly survey creation interface, and logs responses in neatly organized CSV files. The surveys are described here.

Survey I: Linguistic Quality and Faithfulness We assess responses generated from the top two proposed systems (as measured by retrieval and UniEval metrics), the gold responses from the dataset, and responses from the RAG baseline model (as a lower threshold for comparison). The annotator is first given the dialogue history along with the last agent response and asked to

rate the last agent response in terms of naturalness, coherence, and understandability. Then the user is provided with the context information retrieved by the particular system and asked whether the model hallucinates according to our definition of hallucination and whether the system adequately uses the retrieved context. The exact instructions, questions, and possible answers used can be found in Appendix A.1. These 5 questions make up 1 human intelligence task, or HIT, in our AMT survey.

We randomly select 25 different turns, each from a different dialogue, where the dialogue history is less than 10 turns long, the associated context information is less than 400 words, and the agent response is less than 200 words. Although these heuristics bias our survey, they are necessary in order to make the human evaluation feasible and retain annotator focus. Since we evaluate 4 different models, we have 100 dialogue samples. We set up the AMT task so that the questions for each dialogue sample must be answered by 3 different human annotators for a total of 300 HITS.

Survey II: Factual Accuracy We randomly select 50 examples from the shared task samples that were classified as 'Clear Fact' from Section 3.3.3. For each of these samples, the user is given the last user utterance with the following gold agent response from the MultiDoc2Dial dataset. The user is then presented with three alternate responses, response A is from the RAG baseline system, and responses B and C and from the two best-proposed systems. For each alternate response, the user is asked 1) whether the response contains all, some, or contradictory relevant information from the gold response and 2) whether the response has additional information that is not in the gold response. The instructions and questions for this task are shown in Appendix A.2. We again require three responses for each HIT for a total of 150 HITs.

23 and 21 subjects for Survey I and Survey II, respectively, were recruited from University students and Toshiba Cambridge Research Laboratory researchers to participate in Survey I. Participation was voluntary and unpaid.

Chapter 4

Response Generation Systems

Now that we have established a framework for our approach to the task, we can transition to an overview of the different response generation systems leveraged in experimentation. We first outline our baseline system which is a replication of the RAG system trained and evaluated by the authors of MultiDoc2Dial (Feng et al., 2021). Next, we describe our system using the Chat Completion API which allows for the use of GPT-3 as a generator with different retrieval approaches (i.e. DPR). Finally, we explore LlamaIndex, an innovative method that seamlessly integrates its unique retrieval mechanism with GPT-3 for generation.

4.1 Baseline

The first step of replicating the RAG system from the original MultiDoc2Dial paper is preprocessing the documents by segmenting them into passages. We implement two techniques for passage segmentation as done in Feng et al. (2021): (1) **token** where a document is split on a sliding window size of 100 tokens and (2) **structure** where the document is split based on original paragraphs indicated by HTML mark-up tags. Next, we select a retriever and generator. Feng et al. (2021) found that DPR-based biencoders outperformed sparse encoders such as BM25, so we explore two variations of DPR, DPR pre-trained on the Natural Question dataset[1] and DPR finetuned on the MultiDoc2Dial train and validation set[2] and then use BART as the generator.

Although the replicated RAG model will be used as a primary baseline, it is important to note that participants from the shared task improved upon this model, setting a higher standard for the baseline. In the shared task, all top teams adopted a retriever-generator architecture leveraging either BERT or BART as a generator. Importantly, no teams leveraged few-shot

[1] https://github.com/facebookresearch/DPR#new-march-2021-retrieval-model
[2] https://github.com/IBM/multidoc2dial/tree/main#run-baselines

learning approaches or GPT-based models. The winning team, CPII-NLP, specifically improved upon the baseline by adding a re-ranker after the retrieval step, jointly training BART-large with a grounding span predictor for the generator, and adding passage dropout to training (Li et al., 2022). Unfortunately, the retrieval and generation results of the winning team were not publicly released, but we contacted the team and were able to gain access to the generation results. Therefore, in cases where evaluation metrics are independent of retrieved context, we will also compare systems to the winning team, CPII-NLP, generation results, but in cases where retrieval results are necessary it is not possible to provide a comparison.

4.2 Chat Completion

The first set of approaches that we propose to improve upon the baseline uses Chat Completion[3], an API offered by OpenAI that allows users to interact with OpenAI's GPT models. Given a list of messages comprising a conversation, the API call will return a model response for the given chat conversation. When calling the API we specify gpt-3.5-turbo as the model.

Chat Completion is well suited for dialogue tasks as the input is a list of messages that contain the role (system, user, or assistant) and the content of the message. At the time of experimentation, the Chat Completion AI had no mechanism for document or passage retrieval, so to include context information it is necessary to perform the retrieval task beforehand and subsequently include retrieved information in the 'system' message via prompt engineering. Figure 4.1 shows an example use of the API where we include a brief introductory phrase and retrieved information in the systems message and then provide the user and assistant messages from the dialogue history. The model would then return a response to the final user question.

```
import openai

response = openai.ChatCompletion.create(
  model="gpt-3.5-turbo",

  messages=[
    {"role": "system", "content": "You are a helpful assistant at the Department of Motor Vehicles. You have access
to the following knowledge: Sign up or log into MyDMV online here."},
    {"role": "user", "content": "Hello, I forgot to update my address, can you help me with that?"},
    {"role": "assistant", "content": "Hi, you have to report any change of address to DMV within 10 days after
moving. You should do this both for the address associated with your license and all the addresses associated with
all your vehicles."},
    {"role": "user", "content": "Can I do my DMV transactions online?"}
  ]
)
```

Figure 4.1 An example of leveraging the Chat Completion API. A brief introductory phrase and retrieved information is included in the systems message and the dialogue history is incorporated in user and assistant messages.

[3] https://platform.openai.com/docs/guides/gpt/chat-completions-api

We first leverage Chat Completion to explore the effects of GPT-3 as a generator as opposed to BART. We also experiment with using either no retrieved information or gold retrieved information in the prompt to study the effect of external knowledge on GPT-3 responses. Next, we experiment with different styling prompt engineering approaches where we ask the model to mimic the language, structure, and length of responses from the MultiDoc2Dial dataset through different approaches. Finally, we augment GPT-3 with DPR retrieved information and study its effects.

4.3 LlamaIndex

After exploring different retrieval and styling approaches with Chat Completion, we attempt to boost system performance by leveraging LlamaIndex. LlamaIndex is a data framework for LLM applications to ingest, structure, and access private or domain-specific data (Liu, 2022). The LlamaIndex codebase[4] is being continually updated with new features and plug-ins. Whereas Chat Completion is more targeted for incorporating dialogue history, LlamaIndex is more targeted for retrieving and incorporating external knowledge. We will follow a 4 step basic usage pattern for our LlamaIndex approaches defined as follows:

Step 1: Load in the Documents

Documents are loaded into a 'Document' struct, a lightweight container around the data source, by either a dataloader or manually. The most commonly used dataloader is the SimpleDirectoryReader which takes a directory of files as input, selects the best file reader based on the extension and then reads each file in as a Document object. For manual implementation, each document text is assigned to a Document struct. We experimented with both approaches and found no significant difference between them, so chose to load in documents manually as this allows one to add metadata to each document such as 'title' and 'domain' which can be leveraged during retrieval.

Step 2: Parse the Document into Node

Document objects are then parsed into Node objects. A node represents "chunks" of source documents and contains metadata as well as relationship information between nodes. The NodeParser class takes in two optional arguments, chunk_size and chunk_overlap, which specify the maximum amount of tokens included in each node and how much content overlaps between consecutive nodes. The default parameters are chunk_size = 1,024 and chunk_overlap

[4] https://gpt-index.readthedocs.io/en/latest/index.html

= 20. For our experimentation, we use the default parameters, unless otherwise stated, because when experimenting with chunk_size = 512 there was an 8% decrease in recall @k=2.

Step 3: Index Construction

Next, an index is built over the nodes. We use the Vector Store Index which stores each node and its embedding (Figure 4.2). By default, LlamaIndex creates the embedding for each node by calling OpenAI's text-embedding-ada-002 embedding model and then stores embedding with a simple, in-memory dictionary.

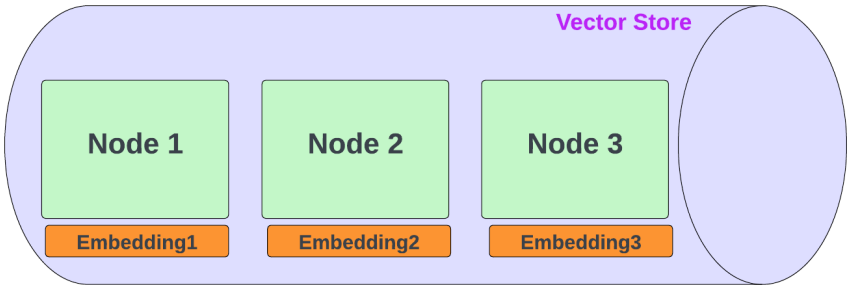


Figure 4.2 Illustration of Vector Store Index inspired from LlamaIndex documentation (Liu, 2022).

Step 4: Query the Index

The index can now be queried by creating an embedding for the query, again using OpenAI's text-embedding-ada-002 embedding model, and then fetching the top-k most similar nodes (Figure 4.3) where similarity (s) is measured by cosine similarity between the query embedding (q) and a given node (n_i):

$$s = \frac{q \cdot n_i}{||q||\ ||n_i||} \tag{4.1}$$

Throughout experimentation use the default number of documents retrieved (k=2) due to the expense of adding more documents to prompts sent to GPT-3. The retrieved nodes are then sent to the Response Synthesizer which generates a response from an LLM, in our case gpt-3.5-turbo. The response of the synthesizer is a Response object from which one can access the retrieved nodes and the LLM response.

Throughout our LlamaIndex experiments, we will adhere to this 4 step framework but will integrate and combine various LlamaIndex features to create a more robust, and faithful system.

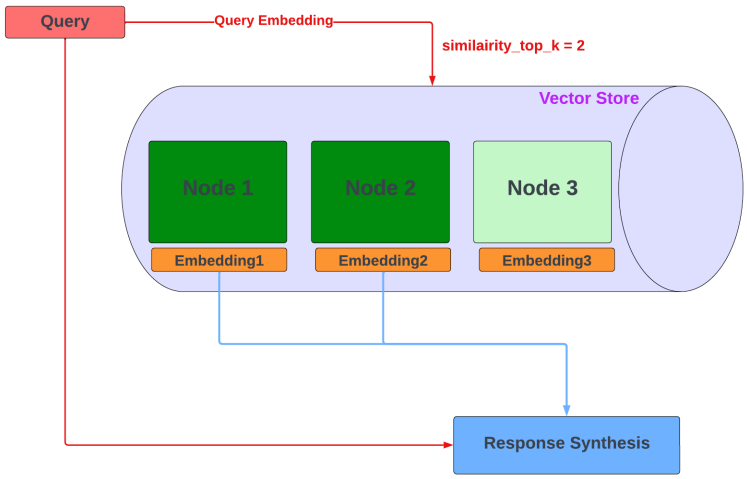


Figure 4.3 Illustration of Querying the Vector Store Index inspired by the LlamaIndex documentation (Liu, 2022).

4.4 Additional LlamaIndex Features

We will make use of two LlamaIndex features: Node Postprocessors, specifically for reranking retrieved nodes, and the ReAct Chat Engine. Both tools are designed to enhance retrieval accuracy and improve overall system generation. Then, drawing insights from ReAct results and relevant literature, we introduce a framework called ReAct & ReGround, which builds upon ReAct.

4.4.1 Node Postprocessors: Reranking

Node postprocessors are a set of modules within LlamaIndex that take a set of nodes and apply a transformation before returning them. These modules are most commonly applied within a query engine, after the node retrieval step, and before the response synthesis step. LlamaIndex offers several node postprocessors, but we will specifically experiment with LLM Rerank and Cohere Rerank.

LLM Rerank uses an LLM to re-order nodes by asking the LLM to return the relevant documents and a score of how relevant they are and subsequently returns the top N-ranked nodes. The LlamaIndex documentation uses their 'Default Choice Select Prompt'[5], which we edit slightly as their prompt is meant for nodes that contain document summaries where our nodes contain document passages (i.e. content snippets). The prompt that we use for our implementation of LLM Rerank is shown in Figure 4.4. This module requires specifying initial

[5]https://github.com/jerryjliu/llama_index/blob/main/llama_index/prompts/choice_select.py

retrieval and post-reranking node counts. For all LLM rerank experiments, we first retrieve 7 nodes, then utilize the top 2 nodes from the LLM rerank for response generation. Since the default chunk size is 1,024, and the maximum token length for gpt-3.5-turbo is 4,096 (including the model response), we were required to explore the use of smaller chunk sizes (i.e. 128 and 512) for these experiments.

```
LLM_Rerank_PROMPT_TMPL = (
    "A list of document is shown below. Each document has a number next to it along "
    "with the text content of the document. A question is also provided. \n"
    "Respond with the numbers of the documents "
    "you should consult to answer the question, in order of relevance, as well \n"
    "as the relevance score. The relevance score is a number from 1-10 based on "
    "how relevant you think the document is to the question.\n"
    "Do not include any documents that are not relevant to the question. \n"
    "Example format: \n"
    "Document 1:\n<text content of document 1>\n\n"
    "Document 2:\n<text content of document 2>\n\n"
    "...\n\n"
    "Document 10:\n<text content of document 10>\n\n"
    "Question: <question>\n"
    "Answer:\n"
    "Doc: 9, Relevance: 7\n"
    "Doc: 3, Relevance: 4\n"
    "Doc: 7, Relevance: 3\n\n"
    "Let's try this now: \n\n"
    "{context_str}\n"
    "Question: {query_str}\n"
    "Answer:\n"
)
```

Figure 4.4 The prompt used for our LLM Reranking.

In addition to LLM Rerank, we experiment with Cohere Rerank which implements the 'Cohere ReRank' functionality[6] from the startup company Cohere to re-order nodes, and returns the top N nodes. As this reranker is not opened source the exact implementation details are unknown. For Cohere experiments we instruct the Cohere functionality to rerank the top-10 retrieved nodes, as token length is no longer an issue, and use the top 2 for response synthesis.

4.4.2 ReAct Chat Engine

A recent feature added to the LlamaIndex documentation is the ReAct Chat Engine interface. The ReAct mode on LlamaIndex is inspired by the original ReAct paper (Yao et al., 2022) and is an agent-based chat mode built on top of the LlamaIndex query engine explained in Step 4. For each chat interaction in our implementation, the agent enters a ReAct loop where it has the option to reason (i.e. produce an observation such as 'I need to use the query engine tool'), act

[6] https://docs.cohere.com/docs/reranking

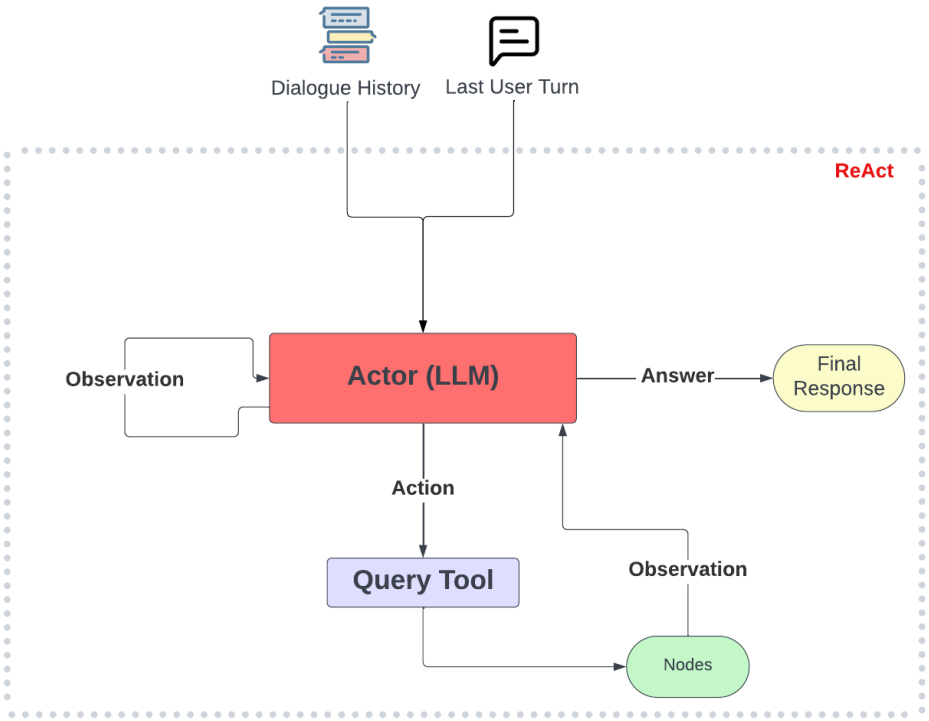


Figure 4.5 ReAct Framework implemented in experimentation. Upon entering the ReAct Chain the Actor has the option to either reason (observation), use a tool (action), or output a final response (answer).

which in our case entails using the query engine tool with a query specified by the LLM and then synthesizing the nodes into an observation, or produce an answer which is used as the final response. The chain repeats until a final response is reached or the max number of iterations, in our case 10, is reached. The ReAct chat engine takes in an optional chat_history parameter so that the chat engine can keep track of the conversation history and answer questions with the past context in mind, as well as a query. At each step, the observations are appended to the chat history to create a running memory available to the LLM at each step of the chain.

As explained in Section 2.3.2, the original ReAct implementation uses few-shot demonstrations to guide the LLM to reason an act, but LlamaIndex implements ReAct in a zero-shot setting. We edit the zero-shot prompt[7] provided in the LlamaIndex codebase to more adequately adapt to our task-oriented dialogue setting (our prompt is shown in Figure 4.6). Specifically, we edit the prompt by adding the instruction 'You have access to a Query Engine Tool. You must always use this tool to retrieve information and ground your answer in this information. This may require breaking the task into subtasks and using the tool multiple times to complete each subtask.' This encourages the ReAct system to remain faithful to the retrieved context.

As the ReAct chat engine feature in LlamaIndex is new and contained several bugs we edited the source code to create a module to extract retrieved nodes for evaluating retrieval, defined a more robust postprocessor for LLM reasoning responses, and incorporated reranking into the Query Engine tool.

4.4.3 ReAct & ReGround

One major advantage of the ReAct framework is that it gives researchers the flexibility to coerce the system to take specific actions at different points in the chain. We already use this flexibility to guide the chain to begin by using the Query Engine within our edited prompt, but in this section, we build on this flexibility by coercing the chain to reground its answer when the final answer is below a given heuristic. We call this model ReAct & ReGround (Figure 4.7).

For the ReAct & ReGround framework, when the ReAct chain decides on an answer, an evaluator module is used to evaluate the response based on a given heuristic. If the heuristic is above a chosen threshold (X), the answer is used as the final response, but if the heuristic is below the threshold, the evaluator returns a linguistic feedback response to the LLM which is stored in memory and used by the ReAct agent to decide on the next reasoning or action step. We will define the heuristic, threshold, and linguistic feedback in Section 5.3.4 as the choices are motivated by results from other proposed models in our study.

[7] https://github.com/jerryjliu/llama_index/blob/main/llama_index/agent/react/prompts.py

```
REACT_CHAT_SYSTEM_HEADER = """\

You are designed to help respond to a user and answer questions in a dialogue setting \
by grounding your answers in retrieved information.

## Tools
You have access to a Query Engine Tool. You must always use this tool to retrieve information
and ground your answer in this information. This may require breaking the task into
subtasks and using the tool multiple times to complete each subtask.


You have access to the following tool:
{tool_desc}

## Output Format
To respond to the user, please use the following format.

```
Thought: I need to use a tool to help me answer the question.
Action: tool name (one of {tool_names})
Action Input: the input to the tool, in a JSON format representing the kwargs (e.g. {{"input": "hello world"}})
```
Please use a valid JSON format for the action input. Do NOT do this {{'input': 'hello world'}}.

If this format is used, the user will respond in the following format:

```
Observation: tool response
```

You should keep repeating the above format until you have enough information
to respond to the user or answer the question without using any more tools. At that point, you MUST respond
in the following format:

```
Thought: I can answer without using any more tools.
Answer: [your answer here]
```

Current Conversation
Below is the current conversation consisting of interleaving human and assistant messages.

"""
```

Figure 4.6 ReAct prompt used in our zero-shot implementation where the LLM is coerced into querying the Query Engine Tool and ground its answers in retrieved context.

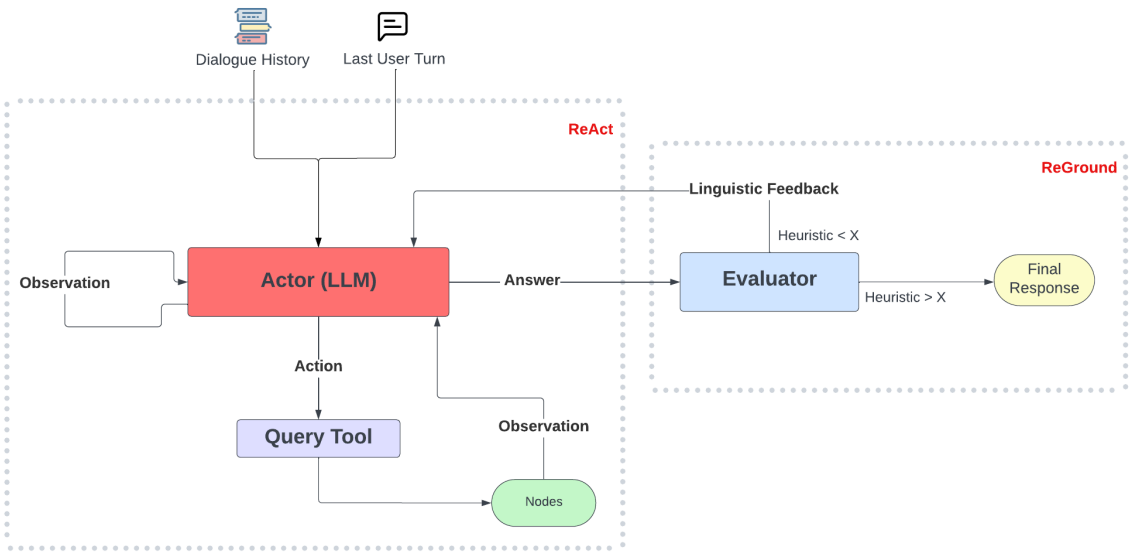


Figure 4.7 ReAct & ReGround Framework where answers are evaluated on a given heuristic and based on the results relation to the threshold, either linguistic feedback is returned back to the actor to continue the chain or the final response is returned.

This work most closely resembles the framework of Reflexion proposed in Shinn et al. (2023). This work proposes the idea of verbal reinforcement learning, where scalar feedback from the environment is converted into verbal feedback and then used to help an augmented LLM learn from prior mistakes. Of particular relevance to our work is their application of this framework in conjunction with ReAct for open-domain QA. However, not only does the paper fail to extend the framework to a dialogue setting, but their heuristic is inherently biased, as it relies on the correct answer to guide the system as they use Exact Match (EM) with the gold response as their heuristic. In real-world scenarios, the gold agent response would not be available, rendering this heuristic impractical and potentially leading to skewed results. Therefore, our heuristic is not dependent on gold responses.

Chapter 5

Results and Discussion

In this chapter, we present and discuss experimental results from response generation systems outlined in the previous chapter. We first present our findings using retrieval and UniEval metrics, and then report results from the SelfEval framework and human evaluation.

5.1 Baseline Model- RAG

We first present the results obtained through the replication of the baseline model from Feng et al. (2021). As stated in the set up we experiment with 2 approaches for document segmentation and 2 variations of DPR retrievers, resulting in 4 different baseline models for replication:

- D^{token}-nq - token segmentation, DPR pre-trained on the Natural Question dataset
- D^{struct}-nq - structure segmentation, DPR pre-trained on the Natural Question dataset
- D^{token}-ft - token segmentation, DPR finetuned on MultiDoc2Dial
- D^{struct}-ft - structure segmentation, DPR finetuned on MultiDoc2Dial

For each setting, we create a FAISS index using the segmented passages and then fine-tune RAG on MultiDoc2Dial data. We replicate Table 6 from Feng et al. (2021) for the test set by reporting F1, EM, and SacreBLEU (BL) scores and recall @k=1,5,10 for Task II where the target output is the next agent response 5.1.

Table 5.1 shows the alignment between our reproduced results and the outcomes reported in Feng et al. (2021). In both the original paper and our replication, models with structure segmentation and DPR fine-tuned on MultiDoc2Dial significantly outperform other models both in terms of retrieval and generation metrics. Therefore, D^{struct}-ft, will now be referred to as our baseline model and will be used for comparison in subsequent experiments, along with CPII-NLP results when possible.

Model	Version	F1	EM	BL	@1	@5	@10
D^{token}-nq	Original	32.5	3.2	16.9	25.9	51.0	61.6
	Replication	30.7	1.9	15.4	24.7	48.8	58.8
D^{struct}-nq	Original	33.0	3.6	17.6	27.3	52.6	62.7
	Replication	31.4	2.3	16.8	25.6	50.1	61.6
D^{token}-ft	Original	35.0	3.7	20.4	36.8	68.3	77.8
	Replication	34.1	3.0	19.8	35.7	67.5	77.2
D^{struct}-ft	Original	36.0	4.1	21.9	39.7	69.3	79.0
	Replication	35.5	3.4	21.7	40.9	69.8	79.2

Table 5.1 Comparison of evaluation results of Task II on agent response generation task for our replicated RAG model and the originally reported results on the test dataset.

5.2 Chat Completion Approaches

Now that a baseline model that leverages BART as the generator has been established, GPT-3 response generation systems are explored. Both shared task metrics and UniEval metrics for generation evaluation are reported in this section to demonstrate the need for a new evaluation standard and then UniEval metrics only are used in the subsequent sections.

5.2.1 Generator-Only Evaluation

We begin by freezing the retrieval process so that GPT-3 is prompted in a 'generator-only' setting shown in Figure 5.1. This system is evaluated in two settings, one where no information is used and another where the correct passage from the human-annotated MultiDoc2Dial corpus is incorporated into the prompt. The shared task evaluation results are shown in Table 5.2.

System	F1	SacreBLEU	METEOR	RougeL
CPII-NLP	52.06	37.41	51.64	50.19
Baseline	35.85	22.26	34.28	33.82
No Info	16.43	1.82	23.59	13.13
Correct Passage	23.57	5.78	33.86	20.67

Table 5.2 Shared Task Evaluation for generator-only Chat Completion versions.

The difference in metrics for No Information and Correct Passage demonstrates that incorporating domain-specific knowledge affects the output of GPT-3 responses for the MultiDoc2Dial dataset, indicating that not all information for these domains is stored in the LLM weights. Even

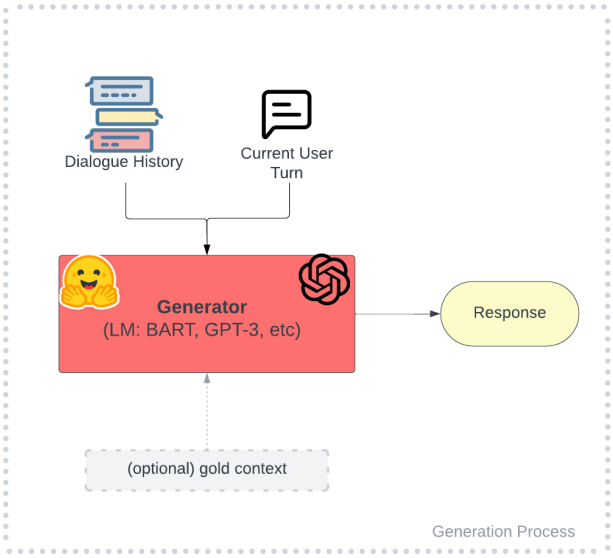


Figure 5.1 The framework for a 'generator-only' scenario where the retrieval process is frozen and replaced by either 1) no context information or 2) correct passage information.

though the Correct Passage system outperforms the No Information system, it significantly underperforms compared to the baseline model in terms of the shared task metrics. This shows that the metrics are most likely unfit for the task, as the Correct Passage System has access to the relevant information and therefore should perform better. As discussed in Section 3.3.2, this is partially a result of the verbosity of GPT-based responses.

5.2.2 Prompt Engineering for Styling

To confirm that lower shared task metrics for GPT-3 responses are partly attributed to differences in styling and length, we prompt the model to formulate responses that imitate the response style from the dataset. This process involves prompt engineering to formulate prompts that most successfully replicate response styling. After analyzing utterances from the MultiDoc2Dial corpus and the Correct Passage system, we find that two major styling differences are that GPT-3 responses are longer (Figure 5.2) and contain more numbered lists. Based on this information, we formulate and evaluate the following prompts:

- **none**: no styling (same as Correct Passage from Table 5.2)
- **mimic**: "You are a helpful assistant for the {domain}. Please mimic the style and approximate length of previous assistant content (if available) while still providing

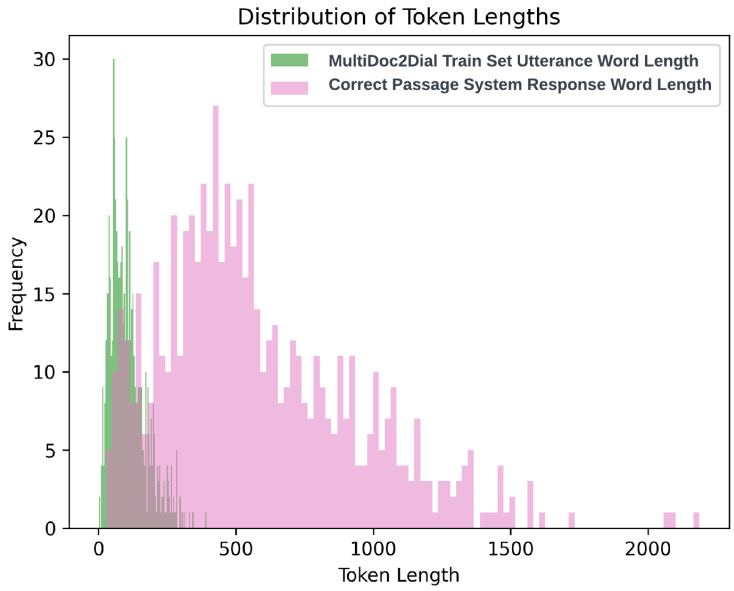


Figure 5.2 Token length distribution for the utterances from the MultiDoc2Dial training set compared to the GPT-3 generated responses using the Chat Completion version with Correct Passages.

complete responses to the user. You have access to the following relevant information: {relevant_info} "

- **stats**: "You are a helpful assistant for the {domain}. Please mimic the style and approximate response length of another assistant whose minimum answer length was 1 word, maximum answer length was 265 words, and average answer length was 20.2 words while still providing complete responses to the user. Also, please refrain from using numbered lists in your answer. You have access to the following relevant information: {relevant_info}"[1]

- **concise**: "You are a helpful assistant for the {domain}. Please make your answers as concise as possible and refrain from using numbered lists. You have access to the following relevant information: {relevant_info}"

- **length**: "You are a helpful assistant for the {domain}. Please provide a response to the user containing approximately 20 words. You have access to the following relevant information: {relevant_info}"

[1]The statistics are calculated using the MultiDoc2Dial training utterances.

- **max** prompt: "You are a helpful assistant for the {domain}. Provide a response to the user using only the following information: {relevant_info}. Limit your response to 25 or fewer words." [2]

- **max_tokens**: Same as length+, but max_tokens=50 was used as an argument in the Chat Completion call.

Table 5.3 shows that as the average length and the number of numbered lists decrease, the shared task evaluation of GPT-3 responses moves closer to the results of the benchmark models. This confirms our hypothesis that the shared task evaluations have a tendency to provide information into how well-generated responses replicate the styling of the dataset as opposed to providing meaningful insight into the quality of the response. Although we show that the automated metrics are inadequate, styling approaches may still provide more factual answers (i.e. by only retaining factual information) or may produce higher quality results (i.e. by producing more coherent responses due to the decreased length), so both non-styled and styled-based systems are implemented in the remaining experiments, using the **max** prompt for styling as it most successfully reduces the distance between GPT-3 and BART generated responses.

System	**Avg Len** ↓	**# lists** ↓	**F1** ↑	**BL** ↑	**METEOR** ↑	**RougeL** ↑
CPII-NLP	20.6	0	52.06	37.41	51.64	50.19
Baseline	51.52	0	35.85	22.26	34.28	33.82
none	95.89	153	23.57	5.78	33.86	20.67
mimic	105.13	79	24.08	6.53	35.81	21.43
stats	73.54	26	28.28	8.70	38.69	24.98
concise	66.71	37	29.23	9.17	39.01	26.01
length	55.39	22	31.95	11.16	40.42	28.17
max	**31.53**	**1**	**36.33**	**16.18**	40.76	**32.58**
max_tokens	36.89	17	36.16	16.08	**43.12**	32.31

Table 5.3 Shared Task Evaluation for different styling approaches.

5.2.3 DPR + GPT-3

Next, the retrieval process is unfrozen and the Chat Completion system is run with fine-tuned DPR as the retriever while keeping GPT-3 as the generator. As empirical evidence has been given to support the need for the new evaluation schema, we will transition to reporting UniEval

[2]This was used because 82.23% of the training data utterances are between 0 and 25 tokens.

metrics for this experiment. Since UniEval is a reference-free metric, an evaluation of the utterances from the 661 shared task samples, which we call the 'Gold' system, is also given in subsequent experiments.

System	Style?	natural	coherent	engage	understnd	grounded	fact
Gold		83.09	92.09	1.16	82.33	95.24	68.48
Baseline		83.89	95.73	1.23	83.19	87.18	67.72
CPII-NLP		86.67	93.78	NA	85.85	NA	NA
No Info		84.77	99.12	4.57*	86.90	60.94*	55.37*
	✓	**93.88**	**99.30**	1.64*	**93.59**	29.72*	52.83*
Correct Info		85.20	99.10	**4.45**	87.29	**96.64**	67.03
	✓	89.70	98.29	1.66	89.27	96.09	**74.35**
DPR		84.67	99.23	4.31	86.72	93.94	63.11
	✓	90.16	98.74	1.67	89.67	92.47	69.60

Table 5.4 Comparison of UniEval evaluation for benchmark models and all Chat Completion models. For context-dependent metrics, '*' denotes models where no retrieved context was incorporated so gold context is used for the calculation, and 'NA' denotes models where intermediate retrieval results are simply unavailable.

From Table 5.4 we make the following observations:

Overall Quality of Benchmarks: As this is the first report of UniEval results for our benchmark models (gold, baseline, CPII-NLP) we must assess these models given the new metrics. As expected, the Gold system shows high groundedness the samples are annotated with the correct relevant passages, whereas the baseline method lags behind. Interestingly, the linguistic quality of the Gold system is low, indicating that the crowd-sourced methodology to create agent utterances in MultiDoc2Dial may produce many undesirable utterances. The CPII-NLP results offer limited comparative value since their linguistic quality aligns closely with the baseline and measures of faithfulness cannot be calculated. Therefore, they will be omitted from subsequent tables.

Linguistic Quality of Proposed Models: The naturalness, coherence, engagingness, and understandability scores are higher for all GPT-3 models compared to the benchmark models. Additionally, naturalness increases when styling is applied, but engagingness decreases due to the decreased length.

Faithfulness of Proposed Models: A comparison of groundedness and factual consistency scores between the model with Correct Info and DPR info shows that models with higher retrieval accuracy remain more grounded in the retrieved context, a trend also seen in other studies (Lewis et al., 2020; Shuster et al., 2021). We observe that when styling approaches are

applied, factual consistency scores increase, indicating that styling may reduce hallucinated information in Chat Completion models.

5.3 LlamaIndex Approaches

Although Chat Completion with a fine-tuned DPR retriever achieves improved linguistic quality over the baseline, it lags in faithfulness metrics. We attempt to address this limitation by boosting retrieval results with LlamaIndex which utilizes its own retrieval mechanism and GPT-3 as a generator.

5.3.1 Query Techniques

Within the LlamaIndex framework, it is possible to perform the retrieval step without the generation step by using the 'no_text' mode for the response synthesizer, which saves costs. This is used to explore the effects on retrieval for four different methods of incorporating dialogue history into the query. All four method includes the introductory phrase, 'You are a helpful assistant for the domain', and the last user turn. DH_none includes no dialogue history. **DH_full** includes the entire dialogue history. **DH_full** includes a summary of the dialogue history produced by GPT-3 by using the following system message with Chat Completion 'Please create a one-sentence summary of the following dialogue history that includes all relevant details and can be passed on to another agent.' **DH_token** includes the last X user/assistant turns where X is the number that includes the most turns but remains under 100 tokens.

Version	**r@1**	**r@2**
Baseline	**50.68**	62.78
DH_none	31.62	38.58
DH_full	41.15	55.82
DH_summary	48.56	63.84
DH_tokens	50.53	**65.96**

Table 5.5 Recall @k=1,2 for methods of querying with LlamaIndex.

The retrieval results in Table 5.5 show that **DH_token performs best, even outperforming the baseline model for recall @2**. These results indicate that the dialogue history contains necessary information for retrieval as all versions including some form of the dialogue history have over a 15 percentage point increase in recall @2 over the version with no dialogue history. Additionally, if the dialogue history contains too many tokens, retrieval suffers, most likely due to the fact that dialogue history tokens dilute the tokens of the last user turn.

We then perform the end-to-end retrieval and generation pipeline with DH_tokens, with and without styling, as well as DH_none for comparison. The results are shown in Table 5.6 which again highlights higher linguistic quality for GPT-3 generated results, higher groundedness for models with better retrieval scores, and higher factual consistency for approaches where styling is applied. To compare the integration of different features into the LlamaIndex framework, we will use DH_token without styling (i.e. 'original' in subsequent experiments), as it shows the highest groundedness score while still maintaining a higher factual consistency score than the Gold System.

System	Style?	natural	coherent	engaging	understand	grounded	fact
Gold		83.09	92.09	1.16	82.33	95.24	68.48
Baseline		83.89	95.73	1.23	83.19	87.18	67.72
DH_none		88.46	86.04	2.96	89.65	77.38	66.64
DH_token		88.56	98.87	3.35	89.52	**94.63**	69.39
DH_token	✓	92.36	98.83	1.24	91.88	89.78	**72.42**

Table 5.6 Comparison of UniEval evaluation for benchmark models, LlamaIndex with no dialogue context, and best LlamaIndex model in terms of retrieval. 'NA' is used for scores that cannot be calculated due to the inability to access the context retrieved for the system.

5.3.2 Reranking

Although the retrieval of the original LlamaIndex system surpasses the baseline in terms of recall @2, we wanted to explore whether reranking passages after retrieval could further improve retrieval and therefore increase system faithfulness.

As mentioned in the experimental setup, large tokens lengths pose an issue when sending nodes to GPT-3 to be reranked, therefore, we experiment with limiting chunk size to 512 (medium) and 128 (small) so that we can send 7 nodes to the LLM reranker (LLM_RR) which will return 2 nodes. Again, for Cohere token length is not an issue and we use the default node length of 1,024 (large) and send 10 and return 2 nodes. The retrieval results in Table 5.7 show that Cohere does not significantly change results from the original LlamaIndex retrieval, but LLM Reranking with the medium storage size significantly improves the retrieval results by over 10% for r@1 and nearly 6% for r@2. This means that **GPT-3 is able to successfully differentiate the relevancy of retrieved documents, boosting retrieval**. Note that using smaller storage sizes decreases accuracy, indicating that longer passages are needed for retrieval.

We use LLM Reranking with the medium storage size for response generation and show results in Table 5.8. The increase in retrieval results has a strong positive impact on faithful-

Version	Storage Size	r@1	r@2
Baseline		50.68	62.78
Original		50.53	65.96
LLM_RR	small	56.73	67.47
LLM_RR	medium	**60.06**	**71.86**
Cohere	large	50.68	66.26

Table 5.7 Retrieval results for systems with reranking.

ness metrics. In fact, the **groundedness of LLM reranking without styling exceeds the groundedness scores of the gold system**. Upon manual observation, it is apparent that this is primarily because when the system encounters a dialogue that is not understandable it expresses uncertainty rather than replying with an additional poor, and not grounded, response as is done in cases of MultiDoc2Dial dataset samples. The linguistic quality reranking remains high, although naturalness drops slightly due to the fact that the reranking system occasionally includes '<assistant>' tags in the generated response, mimicking the styling used in our prompt. As LLM_RR without styling achieves the best groundedness scores while maintaining a high factual consistency score, we use this as the Reranking model for comparison in subsequent experiments.

System	Style?	natural	coherent	engaging	understand	grounded	fact
Gold		83.09	92.09	1.16	82.33	95.24	68.48
Baseline		83.89	95.73	1.23	83.19	87.18	67.72
Original		88.56	98.87	3.35	89.52	94.63	69.39
LLM_RR		85.88	99.30	4.80	88.50	**96.58**	72.19
LLM_RR	✓	92.00	98.25	1.23	91.49	90.70	**73.53**

Table 5.8 Comparison of UniEval metrics for the benchmark systems, original LlamaIndex system, and systems using reranking.

5.3.3 ReAct

Although the LlamaIndex system with reranking attains improved retrieval results and higher faithfulness metrics, we wanted to test whether introducing CoT through ReAct could improve system generation as we still observed several cases where the system grounded responses in the wrong information within a retrieved node.

As it is not possible to implement ReAct in a retrieval-only manner, we evaluated retrieval and generation results for all designed ReAct systems. We ran ReAct mode without additional features, ReAct with the incorporation of Reranking in the Query Engine tool, and React with Reranking and our prompt change discussed in the experimental setup. The retrieval results are shown in Table 5.9 and the generation results are shown in Table 5.10.

Version	**Additional Features**	**r@1**	**r@2**
Baseline		50.68	62.78
Original		50.53	65.96
Reranking		**60.06**	**71.86**
ReAct	None	44.93	54.77
ReAct	Rerank	49.62	57.94
ReAct	Rerank with Styling	42.81	49.47
ReAct	Rerank with Prompt Change	49.17	59.76
ReAct	Rerank with Prompt Change + Styling	52.34	60.67

Table 5.9 Retrieval scores for ReAct systems.

System	**Style?**	**natural**	**coherent**	**engage**	**understnd**	**grounded**	**fact**
Gold		83.09	92.09	1.16	82.33	95.24	68.48
Baseline		83.89	95.73	1.23	83.19	87.18	67.72
Original		88.56	98.87	3.35	89.52	94.63	69.39
Reranking		85.88	99.30	4.80	88.50	**96.58**	72.19
ReAct		91.90	98.00	1.44	91.47	82.91	61.98
ReAct+RR		89.30	98.69	2.45	90.02	88.35	68.79
ReAct+RR	✓	90.24	98.75	1.23	89.59	79.95	67.19
ReAct+RR+PC		89.55	99.37	2.40	90.17	**92.19**	69.57
ReAct+RR+PC	✓	91.74	98.60	1.45	91.31	90.41	**70.09**

Table 5.10 Comparison of UniEval metrics for ReAct systems with previous systems.

Although Reranking and the Prompt Change improve retrieval, and therefore generation results, the system still significantly falls behind the LlamaIndex with the Reranking model in terms of both measures. Despite these results, we find that ReAct performs very well in the samples defined as 'Clear Fact' when we grouped the shared task data into different categories in Section 3.3.3 as it can identify relevant information in the context better than other systems. This finding is supported by Shuster et al. (2022a) and Adolphs et al. (2021) which find that using a modular approach that first finds the relevant parts of the documents and then generates

the final response has shown to help alleviate the problem of generating a factually incorrect response from retrieved documents. This is the exact approach taken in ReAct as the system synthesizes retrieved information before generating a final response. We suspect that this ability of ReAct to synthesize information would have more pronounced effects for a task where multiple documents have to be incorporated into a single response (multi-hop response) whereas for our dataset each individual turn is annotated with one gold document (even though a given dialogue is grounded in multiple documents).

Finally, another benefit of the ReAct mode is that it allows for more flexibility in guiding the system as there are several observation and synthesis steps as opposed to the retrieve and generate paradigm of earlier systems. We attempt to build off the benefits of both the Rerank system and our best ReAct system.

5.3.4 ReAct & ReGround

Before designing our algorithmic approach, a thorough analysis of the best ReAct model was conducted. As the reranking system uses the last 100 tokens of dialogue we wanted to confirm that the dialogue length was not causing poor retrieval in ReAct. We found that the length of the dialogue did not affect retrieval, as the average turn was 6.4 for correct retrieval and 7.6 for incorrect retrieval. Next, we wanted to pinpoint a heuristic to differentiate failed vs. successful retrieval without incorporating gold context or responses. We found that the LlamaIndex retrieval scores calculated by cosine similarity did not differentiate retrieval performance with an average retrieval score of 84.96 for correct and 82.81 for incorrect retrievals. The same was the case for LLM reranking relevance scores with an average relevance score of 8.40 for correct and 7.77 for incorrect retrievals.

Interestingly, groundedness and factual consistency scores differentiated successful vs. failed retrieval extraordinarily well. Successful retrieval showed average scores of 98.17 for groundedness and 83.37 for factual consistency while failed retrieval showed an average of 75.80 for groundedness and 60.37 for factual consistency. As groundedness is hypothesized to be more robust since it is a part of the multi-dimensional UniEval approach (as opposed to uni-dimensional), we decided to use groundedness as our **heuristic** for our algorithmic approach. Furthermore, since 95% of the ReAct cases with correct retrieval have a groundedness score of 84.45 or above, we use 84.45 as our **Threshold**. We test two different versions of our algorithmic approach, ReAct & ReGround and ReAct & Reground +Force, which each use different processes for providing **linguistic feedback**:

- **ReAct & ReGround**: If the max iteration has not been reach and groundedness is less than 84.45, the verbal response returned to the LLM is ‘The information retrieved from

the Query Engine Tool is most likely incorrect. Rephrase the action input and make sure to include all relevant information from the chat history in your action input. Use the Query Engine Tool to retrieve new information and ground your answer in the newly retrieved information.' If the max iteration is reached and groundedness is still below 84.45, we fall back to the query which includes 100 words of the dialogue history.

- **ReAct & ReGround +Force**: If it is the first iteration and groundedness is less than 84.45, the verbal response FORCES the ReAct chain to query with from Rerank returning the feedback, 'Thought: I need to use a tool to help me answer the question. Action: Query Engine Tool Action Input: {"input": {Rerank_query}}'. This string will automatically force the output parser to interpret this as a call to the query engine. For every subsequent iteration, the system uses the verbal response used in ReAct & ReGround.

The retrieval results of these two proposed are compared to the previous systems in Table 5.11. Interestingly, ReAct & ReGround slightly boost the performance of ReAct, but to reach the retrieval results seen with the Reranking, forcing the chain to begin with the 100 tokens of dialogue history query, as opposed to a self-formulated query by the LLM, is necessary. This indicates that the most important information for retrieval is contained in the last few turns of a dialogue. As the ReAct & ReGround +Force method is nearly identical in terms of retrieval to LlamaIndex with Reranking, we analyze the generation response evaluation for both models.

Version	**r@1**	**r@2**
Baseline	50.68	62.78
Original	50.53	65.96
Reranking	**60.06**	**71.86**
ReAct	49.17	59.76
ReAct & ReGround	51.29	62.63
ReAct & ReGround +Force	**59.46**	**70.95**

Table 5.11 Retrieval scores ReAct & Reground systems.

Table 5.12 shows that **both ReAct & ReGround models succeed in outperforming all other models in terms of both groundedness and factual consistency**. As ReAct & ReGround +Force shows higher retrieval this is the model that most optimally balances accuracy and faithfulness. Additionally, ReAct & ReGround +Force shows slightly higher linguistic features than Reranking. Overall this indicates that the self-reflection framework can be used in order to guide systems to be more faithful and factually grounded in retrieved information.

System	natural	coherent	engaging	understand	grounded	fact
Gold	83.09	92.09	1.16	82.33	95.24	68.48
Baseline	83.89	95.73	1.23	83.19	87.18	67.72
Original	88.56	98.87	3.35	89.52	94.63	69.39
Reranking	85.88	99.30	4.80	88.50	96.58	72.19
ReAct	89.55	99.37	2.40	90.17	92.19	69.57
ReGround	88.76	99.34	2.75	89.64	98.08	**73.05**
ReGround+	89.45	99.60	3.08	90.61	**98.14**	72.96

Table 5.12 Comparison of UniEval metrics for ReAct & ReGround systems with previous systems.

5.4 LLM Self Evaluation

As our SelfEval framework leverages few-shot prompting and GPT-4 we had to be selective about which models to evaluate. We evaluated the Gold system as this should contain few hallucinations along with the Baseline model and Chat Completion using no retrieved information, as these two models serve as higher bounds containing many hallucinations. We then compare these benchmarks to our best models: Rerank and ReAct & ReGround +Force. The results in Table 5.13 show that the Gold system has the few hallucinations while the Baseline and Chat Completion with No Information have high hallucination scores, providing evidence that the self-evaluation framework succeeds in measuring hallucinations. The SelfEval finds that ReAct & ReGround +Force hallucinates slightly less than Rerank, supporting the UniEval results. Both the Rerank and ReAct & ReGround +Force systems show lower hallucination scores than the Gold systems, most likely due to the several poorly annotated MultiDoc2Dial samples when the last user turn is incoherent.

System	Hallucination Scores↓
Gold	10.89
Baseline	36.31
Chat Completion with No Information	51.74
LlamaIndex Rerank	9.23
ReAct & ReGround+	**6.21**

Table 5.13 Self Evaluation Scores with few-shot prompting and GPT-4 model.

5.5 Human Evaluation

Finally, we analyze the results of our human evaluation which acts as the gold standard and gives us additional insight from UniEval and SelfEval into system quality. We use the Human Evaluation to compare our two best models, Rerank and ReAct & ReGround +Force, to the Gold and Baseline systems and then give a short analysis of how well automated metrics correlate with the human annotations.

5.5.1 Model Comparison

The results of Survey I are shown in Table 5.14. All metrics are the average overall responses[3]. To measure inter-coder reliability we calculate Krippendorff's alpha (Krippendorff, 2018) over all 5 question categories and find Krippendorff alpha = 0.8569 indicating annotator agreement is high and the annotations are considered reliable.

System	natural↑	coherent↑	understnd↑	halluc.↓	Adequacy↑
Gold	80.00	68.67	73.33	**17.33**	61.33
Baseline	84.00	60.67	70.67	28.00	22.67
Rerank	90.67	**93.33**	**97.33**	26.67	**82.67**
ReAct & ReGround+	**92.67**	86.00	92.00	18.67	70.67

Table 5.14 Results for Survey I: Linguistic Quality and Faithfulness. All scores are the average overall HITS for the corresponding question.

The results show that the GPT-3 based models are significantly more natural, coherent, and understandable than the Gold and Baseline systems. The poor linguistic quality of the Gold system highlights the limitations of using a crowd-sourced dataset and the higher performance of proposed systems over the Baseline confirms that GPT-3 responses are preferred over BART-generated responses. Next, ReAct & ReGround +Force shows the lowest hallucination scores and demonstrates similar hallucination scores to the dataset itself. Interestingly, both proposed models use retrieved context more adequately than the gold and baseline systems, but Rerank shows a 12 percentage point increase over ReAct & ReGround +Force. This is mostly likely due to the fact that when ReAct & ReGround +Force retrieves irrelevant information, it is coerced into grounding its answer in the retrieved information, whereas Rerank is more likely to admit that the retrieved context is inadequate. Based on these results, ReAct & ReGround +Force may be a more appropriate system where groundedness is essential (i.e. the legal domain

[3]Naturalness and coherence were on a scale of 1-3 but were normalized to the range 0-1.

scenario), whereas Rerank may be more appropriate when some flexibility is acceptable (i.e. a conversational setting).

The results of Survey II are shown in Table 5.15 with Krippendorff's alpha = 0.6869 indicating relatively strong inter-coder agreement. These results show that our best-proposed models reduce contradictory information by 45% and increase all or some by nearly 300% compared to the baseline. Additionally, the proposed systems incorporate additional information more frequently, but notably, this is not a hallucination if the additional information is grounded in the retrieved information.

System	**All↑**	**Some↑**	**Contradictory↓**	**Additional**
Baseline	30.00	20.00	36.67	45.00
Rerank	**61.33**	17.33	**16.67**	74.00
ReAct & ReGround+	58.00	**21.33**	**16.67**	56.00

Table 5.15 Results for Survey II: Factual Accuracy. Reports the number of cases where human annotators found generated responses contained all, some, or contradictory information compared to the gold reference. The 'additional' column counts cases where responses have all or some relevant info plus additional information."

5.5.2 Metric Correlations with Human Annotation

Metrics	Natural		Coherence		Understand		Grounded	
	r	ρ	r	ρ	r	ρ	r	ρ
F1	-0.281	-0.360	-0.285	-0.343	-0.333	-0.388	*0.063*	*-0.033*
SacreBLEU	-0.233	-0.310	-0.234	-0.286	-0.255	-0.317	*0.084*	*0.032*
METEOR	-0.199	-0.246	*-0.156*	*-0.153*	-0.223	-0.244	0.224	*0.176*
RougeL	-0.267	-0.357	-0.274	-0.346	-0.288	-0.366	*0.079*	*-0.042*
UniEval(multi)	**0.299**	***0.118***	**0.314**	**0.548**	***0.172***	***0.174***	**0.504**	0.336
UniEval(consis)							0.344	0.414
SelfEval							0.459	**0.416**

Table 5.16 Turn-level Pearson (r) and Spearman (ρ) correlations of different metrics with human annotation on MultiDoc2Dial over all four systems evaluated in the Human Evaluation. All p-values ≥ 0.05 are italicized.

Finally, we calculate the Pearson and Spearman correlations between each automated metric and the human-annotated responses for each HIT from Survey I over all 4 human-evaluated

systems. These correlations give empirical insight into the meaningfulness of different metrics. Note that to transform the groundedness-related questions from Survey 1 into a single number we take 1 minus hallucination and average it with adequacy.

We calculated these correlations for the similarity-based metrics from the shared evaluation task, as well as the components of our evaluation framework, multi-dimensional UniEval, the factual consistency model released with UniEval, and SelfEval. Note that for multi-dimensional UniEval we calculate the correlation for metrics that align, namely naturalness, coherence, understandability, and groundedness. Next, we calculate the correlation for UniEval factual consistency and SelfEval results with this groundedness metric. To transform the hallucination score from the SelfEval into a score that is directly comparable to groundedness we again take 1 minus the hallucination score.

The results in Table 5.16 highlight the difficulty of our task as many correlations scores are low or even negative for word overlap scores/ Negative scores are a result of similarity metrics having value 1 for gold system responses (as they are compared to themselves) and corresponding low scores from human evaluators. Additionally, the misalignment of similarity-based metrics benchmarks underscores the urgency for a new robust and meaningful automated evaluation approach to assess LLM alignment with human preferences. Our study reveals that UniEval, along with the SelfEval framework, offers a promising path for advancing research in this domain with moderately positive correlations.

Chapter 6

Conclusion and Future Work

6.1 Summary

In the introduction we set out two main goals for this thesis: 1) improve the linguistic quality, accuracy, and faithfulness of state-of-the-art task-oriented dialogues for QA and 2) propose an improved evaluation schema that can guide the development of these improved models.

We initially approached this task from a high level, defining important definitions, framing the task, and deciding on an appropriate dataset, ultimately choosing MultiDoc2Dial. We then perform a thorough literature review of popular evaluation metrics used in the field and propose an improved framework made up of automated, LLM self-assessment, and human evaluation. We allow these high-level decisions to guide our system development.

For system development, we begin by replicating the RAG baseline released with the MultiDoc2Dial dataset, showing that we reach comparable results to those published in Feng et al. (2021). Using these results as a baseline, we then set out to explore the few-shot learning capabilities of GPT-3 to improve upon the baseline. We use Chat Completion to show that augmentation with external knowledge improves GPT-3 responses, shared task automatic metrics are insufficient for evaluation, and utilizing a DPR + GPT-3 pipeline is unable to improve upon the baseline in terms of groundedness and factual consistency.

We then shift to LlamaIndex-based approaches, where we propose two systems that satisfy the first goal of this thesis which is to design a system that exceeds in accuracy, linguistic quality, and faithfulness. The first of these systems, LlamaIndex with Rerank, shows that we can use LLMs themselves to rerank retrieved documents to improve the accuracy of retrieval methods such as DPR which are fine-tuned on the dataset itself. Next, ReGround & ReAct +Force shows that we can leverage the modularity and flexibility of the ReAct framework to leverage a self-reflection mechanism that produces answers that are more grounded in the retrieved knowledge of the system. We then compare these two best systems to the Gold and

Baseline systems in a LLM self-assessment and human evaluation and show that Rerank & Ground +Force both reduces hallucination over the baseline and increase linguistic quality, accuracy, and adequacy of incorporating retrieved information.

We conclude that the ReAct & ReGround +Force methodology may be more useful for use cases where it is crucial that the system grounds answers in retrieved information (i.e. in the legal domain setting) whereas rerank may be more useful for use cases where it is possible or encouraged for the system to have more freedom in its responses.

6.2 Future Work

Our results, along with past work (Lewis et al., 2020; Shuster et al., 2021) show that increasing retrieval accuracy increases the overall quality of responses. Although our best retrieval mechanism improves upon the baseline, we still only reach 71.86% for recall @k=2. Therefore, we believe future work in this field should be focused on improving retrieval accuracy. Furthermore, we have shown that by leveraging the modularity of ReAct, the system can be guided by self-reflection, so we specifically propose leveraging this framework for future work in improving retrieval.

Additionally, there were several limitations in our experimentation due to our resource constraints, and with more funding and time, there are multiple experiments we believe future research should explore. First, we run all proposed systems using gpt-3.5-turbo, but we believe that results would improve with gpt-4 as it has increased linguistic abilities due to the increase in model size. We suspect that this would specifically improve the results of ReAct models, as the effects of ReAct have been shown to increase with increasing model size (Yao et al., 2022). Additionally, we were forced to be selective in the models run in the LLM self-evaluation, but based on the ability to differentiate hallucination scores, we encourage researchers to leverage this approach in future work with more models.

Finally, our human evaluation showed that the Gold system, with utterances taken from the MultiDoc2Dial dataset, performs poorly in terms of human evaluation. This shows that the crowd-sourced nature of the dataset leads to poor linguistic quality and even contradicts the information that it is meant to be grounded in 36.67% of the time. Therefore, we would like to test our approaches to an improved dataset, such as FaithDial (Dziri et al., 2022).

Overall, our research underscores the potential of few-shot frameworks to enhance task-oriented dialogue systems and we hope that future researchers will leverage our findings to continue increasing the quality and faithfulness of these systems.

Bibliography

Adolphs, L., Shuster, K., Urbanek, J., Szlam, A., and Weston, J. (2021). Reason first, then respond: Modular generation for knowledge-infused dialogue. *arXiv preprint arXiv:2111.05204*.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. (2022). Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Banerjee, S. and Lavie, A. (2005). Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Baudiš, P. and Šedivỳ, J. (2015). Modeling of the question answering task in the yodaqa system. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 6th International Conference of the CLEF Association, CLEF'15, Toulouse, France, September 8-11, 2015, Proceedings 6*, pages 222–228. Springer.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., Van Den Driessche, G. B., Lespiau, J.-B., Damoc, B., Clark, A., et al. (2022). Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., et al. (2023). Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Cheng, D., Huang, S., Bi, J., Zhan, Y., Liu, J., Wang, Y., Sun, H., Wei, F., Deng, D., and Zhang, Q. (2023). Uprise: Universal prompt retrieval for improving zero-shot evaluation. *arXiv preprint arXiv:2303.08518*.

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P., and Zettlemoyer, L. (2018). Quac: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2022). Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. (2019a). Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Clark, E., Celikyilmaz, A., and Smith, N. A. (2019b). Sentence mover's similarity: Automatic evaluation for multi-sentence texts. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2748–2760.

Creswell, A., Shanahan, M., and Higgins, I. (2022). Selection-inference: Exploiting large language models for interpretable logical reasoning. *arXiv preprint arXiv:2205.09712*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dinan, E., Roller, S., Shuster, K., Fan, A., Auli, M., and Weston, J. (2018). Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Dziri, N., Kamalloo, E., Milton, S., Zaiane, O., Yu, M., Ponti, E. M., and Reddy, S. (2022). Faithdial: A faithful benchmark for information-seeking dialogue. *Transactions of the Association for Computational Linguistics*, 10:1473–1490.

Feng, S., Patel, S., and Wan, H. (2022). Dialdoc 2022 shared task: Open-book document-grounded dialogue modeling. In *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 155–160.

Feng, S., Patel, S. S., Wan, H., and Joshi, S. (2021). Multidoc2dial: Modeling dialogues grounded in multiple documents. In *EMNLP*.

Feng, S., Wan, H., Gunasekara, C., Patel, S. S., Joshi, S., and Lastras, L. A. (2020). doc2dial: A goal-oriented document-grounded dialogue dataset. *arXiv preprint arXiv:2011.06623*.

Gopalakrishnan, K., Hedayatnia, B., Chen, Q., Gottardi, A., Kwatra, S., Venkatesh, A., Gabriel, R., and Hakkani-Tür, D. (2019). Topical-chat: Towards knowledge-grounded open-domain conversations.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M.-W. (2020). Realm: Retrieval-augmented language model pre-training.

He, H., Zhang, H., and Roth, D. (2022). Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Honovich, O., Choshen, L., Aharoni, R., Neeman, E., Szpektor, I., and Abend, O. (2021). Q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. *arXiv preprint arXiv:2104.08202*.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Izacard, G. and Grave, E. (2020). Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Dwivedi-Yu, J., Joulin, A., Riedel, S., and Grave, E. (2022). Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*.

Johnson, J., Douze, M., and Jégou, H. (2019). Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. (2017). Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2022). Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Krippendorff, K. (2018). *Content analysis: An introduction to its methodology*. Sage publications.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. (2019). Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Lee, K., Chang, M.-W., and Toutanova, K. (2019). Latent retrieval for weakly supervised open domain question answering.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Li, K., Zhang, T., Tang, L., Li, J., Lu, H., Wu, X., and Meng, H. (2022). Grounded dialogue generation with cross-encoding re-ranker, grounding span prediction, and passage dropout. In *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, pages 123–129.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Liu, J. (2022). LlamaIndex.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Maynez, J., Narayan, S., Bohnet, B., and McDonald, R. (2020). On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661*.

Mehri, S. and Eskenazi, M. (2020). Usr: An unsupervised and reference free evaluation metric for dialog generation. *arXiv preprint arXiv:2005.00456*.

Mialon, G., Dessì, R., Lomeli, M., Nalmpantis, C., Pasunuru, R., Raileanu, R., Rozière, B., Schick, T., Dwivedi-Yu, J., Celikyilmaz, A., et al. (2023). Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. (2021). Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback.

Perez, E., Ringer, S., Lukošiūtė, K., Nguyen, K., Chen, E., Heiner, S., Pettit, C., Olsson, C., Kundu, S., Kadavath, S., et al. (2022). Discovering language model behaviors with model-written evaluations. *arXiv preprint arXiv:2212.09251*.

Petroni, F., Rocktäschel, T., Lewis, P., Bakhtin, A., Wu, Y., Miller, A. H., and Riedel, S. (2019). Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Post, M. (2018). A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W. B., and Iyyer, M. (2020). Open-retrieval conversational question answering. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 539–548.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Reddy, S., Chen, D., and Manning, C. D. (2019). Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Roberts, A., Raffel, C., and Shazeer, N. (2020). How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Robertson, S., Zaragoza, H., et al. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Roller, S., Dinan, E., Goyal, N., Ju, D., Williamson, M., Liu, Y., Xu, J., Ott, M., Shuster, K., Smith, E. M., et al. (2020). Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Saeidi, M., Bartolo, M., Lewis, P., Singh, S., Rocktäschel, T., Sheldon, M., Bouchard, G., and Riedel, S. (2018). Interpretation of natural language rules in conversational machine reading. *arXiv preprint arXiv:1809.01494*.

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., and Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Shinn, N., Labash, B., and Gopinath, A. (2023). Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*.

Shuster, K., Komeili, M., Adolphs, L., Roller, S., Szlam, A., and Weston, J. (2022a). Language models that seek for knowledge: Modular search generation for dialogue and prompt completion.

Shuster, K., Poff, S., Chen, M., Kiela, D., and Weston, J. (2021). Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*.

Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E. M., Roller, S., Ung, M., Chen, M., Arora, K., Lane, J., et al. (2022b). Blenderbot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*.

Svikhnushina, E. and Pu, P. (2023). Approximating human evaluation of social chatbots with prompting. *arXiv preprint arXiv:2304.05253*.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. (2022). Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Trivedi, H., Balasubramanian, N., Khot, T., and Sabharwal, A. (2022). Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. (2022). Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Welleck, S., Kulikov, I., Roller, S., Dinan, E., Cho, K., and Weston, J. (2019). Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Weng, L. (2023). Prompt engineering. *lilianweng.github.io*.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., and Manning, C. D. (2018). Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2022). React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Ye, X. and Durrett, G. (2022). The unreliability of explanations in few-shot prompting for textual reasoning. *Advances in neural information processing systems*.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Zhao, R., Chen, H., Wang, W., Jiao, F., Do, X. L., Qin, C., Ding, B., Guo, X., Li, M., Li, X., et al. (2023a). Retrieving multimodal information for augmented generation: A survey. *arXiv preprint arXiv:2303.10868*.

Zhao, R., Li, X., Joty, S., Qin, C., and Bing, L. (2023b). Verify-and-edit: A knowledge-enhanced chain-of-thought framework. *arXiv preprint arXiv:2305.03268*.

Zhao, W., Peyrard, M., Liu, F., Gao, Y., Meyer, C. M., and Eger, S. (2019). Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *arXiv preprint arXiv:1909.02622*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Zhong, M., Liu, Y., Yin, D., Mao, Y., Jiao, Y., Liu, P., Zhu, C., Ji, H., and Han, J. (2022). Towards a unified multi-dimensional evaluator for text generation. *arXiv preprint arXiv:2210.07197*.

Zhou, K., Prabhumoye, S., and Black, A. W. (2018). A dataset for document grounded conversations. *arXiv preprint arXiv:1809.07358*.

Appendix A

Human Evaluation

A.1 Survey I: Linguistic Quality and Faithfulness

The **instructions** for Survey I are: 'You will be given a conversation that took place between a user and an agent from a specified social services domain. Throughout the conversation, the agent has access to some documents and uses snippets of these documents to respond to the user. This information is called the reference information.'

The survey then begins with the introduction 'Please read this conversation between a user and an agent from the insert domain. You will then be asked to rate the FINAL AGENT RESPONSE on several metrics.' The **questions and possible answers** for Survey I are then as follows:

1. Is the response a valid continuation of the conversation history (i.e. does it remain on topic and address the user's question/statement)?
   (a) No, response drastically changes topic or ignores conversational history
   (b) Somewhat, response refers to the conversational history in a limited capacity and shifts the conversation topic
   (c) Yes, response is on topic and strongly acknowledges the conversational history
2. Is the response understandable in the context of the conversation history?
   (a) No, response is difficult to understand
   (b) Yes, you know what the agent is trying to say
3. Is the response naturally written (i.e. human-like and without grammatical errors)?
   (a) Unnatural

(b) Somewhat Unnatural, sounds strange or contains a few grammatical errors

(c) Natural with no grammatical errors

4. Does the Final Agent Response contain additional factual information that is unsupported by the reference information?

   (a) No

   (b) Yes

5. Do you think the response is adequate given the context?

   (a) No, could have used context better

   (b) Yes, it is adequate

As Q4 tended to be the most confusing and complex for human users, we provided the following example to assist the annotator:

Examples for Question #4

Example 1 (Correct Answer: 'NO'):

Dialogue History: USER: What can we get forgiven on a teacher loan?

Reference: The maximum forgiveness amount is either $17,500 or $5,000, depending on the subject area taught.

Agent Response: The maximum forgiveness amount can be either $17,500 or $5,000. The amount depends on the subject area taught.

Example 2 (Correct Answer: 'YES')

Dialogue History: USER: What can we get forgiven on a teacher loan?

Reference: The maximum forgiveness amount is either $17,500 or $5,000, depending on the subject area taught.

Agent Response: The maximum forgiveness amount is $15,000

The dialogues used for Survey I along with human responses are made available on this Supplementary Material GitHub Page.

A.2 Survey II: Factual Accuracy

The **Instructions** for survey II are: 'Below there is a snippet of a conversation between a user and a social services agent (i.e. from the department of motor vehicles, student aid, veterans affairs, or social security aid). The text snippet contains a user question and the correct agent response. You will be given three additional responses to the user question and be asked to compare them to the correct response.'

For each alternate response, the **questions and possible answers** are:

1. Does Response (A/B/C) contain all/some/contradictory RELEVANT information from the Correct Response?
   (a) All relevant info from the correct response
   (b) Some of the relevant info from the correct response
   (c) Contradicts the correct response
2. Does Response (A/B/C) contain additional information that is NOT in the Correct Response?
   (a) Yes
   (b) NO

The dialogues used for Survey II along with human responses are made available on this Supplementary Material GitHub Page.