Doctoral theses at NTNU, 2024:75

Yujie Xing

# Multi-Turn Generation-Based Conversational Agents in Open Domains

Doctoral thesis

NTNU
Norwegian University of Science and Technology
Thesis for the Degree of
Philosophiae Doctor
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Yujie Xing

# Multi-Turn Generation-Based Conversational Agents in Open Domains

Thesis for the Degree of Philosophiae Doctor

Trondheim, March 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science



NTNU
Norwegian University of
Science and Technology

**NTNU**
Norwegian University of Science and Technology

Thesis for the Degree of Philosophiae Doctor

Faculty of Information Technology and Electrical Engineering
Department of Computer Science



ISBN 978-82-326-7742-9 (printed ver.)
ISBN 978-82-326-7741-2 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)

Doctoral theses at NTNU, 2024:75

Printed by NTNU Grafisk senter

# Abstract

This PhD thesis focuses on open-domain generation-based conversational agents, which are chatbots that generate responses to any input or question using natural language processing and deep learning techniques. The thesis identifies three major challenges faced by these conversational agents.

(1) Generating appropriate responses for a wide range of topics and domains. Current studies have focused on single-corpus training, which limits the model's ability to generate relevant responses for certain topics.

(2) Improving a model's performance of context attention distribution in multi-turn settings. The ability to distribute attention and assign importance to relevant information is necessary to generate appropriate responses. However, most existing works have treated multi-turn conversations as one-turn contexts, limiting the performance of the agents.

(3) Integrating knowledge under the conversational question-answering task perspective. There is a gap in research on integrating extractive question-answering techniques with instruction-based tuning and prompt-based tuning. The thesis proposes several approaches to address these challenges.

For (1), the thesis proposes Document-specific Frequency (DF) as an evaluation metric and proposes several methods for balancing multiple corpora. The best method, which integrates DF with the training, achieves an improvement by 34.1% on F1 performance and at least 20.0% on DF. A thorough human evaluation shows a highly significant (p ¡ 0.001) improvement in all of our proposed methods.

For (2), the thesis proposes Distracting Attention Score ratio (DAS ratio) as an evaluation metric and employs self-contained negative samples and summarization techniques to improve a system's performance on context attention distribution. The proposed self-contained negative samples are applied as a training strategy, resulting in about 10% better DAS ratio. The best summarization technique setting with ORACLE gains a 23% improvement on the DAS ratio.

For (3), the thesis explores various settings of integrating extractive question answering with instruction-based tuning, prompt-based tuning, and multi-task learning. When combining prompt-based tuning with either instruction-based tuning or multi-task learning, the F1 performance is improved by about 18% over the baseline.

Together, these techniques have improved the overall performance of multi-turn conversational agents on open domains.

# Preface

This thesis is submitted to the Norwegian University of Science and Technology (NTNU) for partial fulfillment of the requirements for the degree of philosophiae doctor.

This doctoral work has been performed at the Department of Computer Science (IDI), NTNU, Trondheim, with Professor Jon Atle Gulla as the main supervisor and with Professor Kjetil Nørvåg, Doctor Aria Rahmati, and Doctor Peng Liu as co-supervisors.

The thesis has been part of the DNB-collaboration project with project number 90393101 and has received funds from the Norwegian Research Center for AI Innovation (NorwAI) with project number 990109102.

# Acknowledgments

I sincerely thank all my friends, who have supported me throughout my PhD journey, both inside and outside of IDI, for their supervision, kind words, delicious food, shared moments, and especially for all the help provided. I couldn't have gotten through it without each one of you. I wish you all the brightest and most prosperous future.

I extend my gratitude to my chinchillas and hamsters for their emotional support.

I am also deeply thankful to my family, who have been my backbone throughout this journey. I wish for you all to remain healthy and happy.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

This chapter gives an overview of the work conducted during my PhD study. In Section 1.1, we introduce the background and motivation of my research. In Section 1.2, we discuss the research questions and the approaches to solving them. In Section 1.3, we summarize the research publications and contributions. In Section 1.4, the structure of the thesis is introduced. Finally, Section 1.5 briefly explains the research context.

## 1.1 Challenges on Open-Domain Generation-Based Conversational Agents

Open-domain generation-based conversational agents are a type of chatbot that use natural language processing and deep learning techniques to generate responses to any question or input, regardless of topic or goal. Unlike retrieval-based chatbots that rely on pre-defined responses based on specific keywords or phrases, generation-based conversational agents construct responses from scratch based on the context of the conversation. These systems are designed to be more versatile and adaptable than task-specific conversational agents, which are intended to perform specific tasks, such as making a reservation or providing customer support. Despite notable advancements in recent years, open-domain generation-based conversational agents still face several challenges that must be addressed to enhance their effectiveness.

An important challenge for open-domain generation-based conversational agents is their ability to generate appropriate responses for a wide range of topics and domains. Most studies have been limited to single-corpus training and evaluation, which can result in models that are unable to generate relevant responses for certain topics. For instance, while the PersonaChat corpus provides examples of everyday conversations, it does not cover technical topics found in Ubuntu chat logs. Therefore, when asked a technical question about Ubuntu, the generated responses may not be relevant. To overcome this challenge, open-domain conversational systems need to learn from multiple corpora using effective learning techniques. This thesis explores various approaches for balancing multi-domain corpora and a metric for evaluating the relevance of generated responses for each specific corpus.

Another challenge faced by open-domain generation-based conversational agents is related to multi-turn conversations. To generate relevant responses in the context of such conversations, the system must possess a good ability to distribute attention, namely context attention distribution. This requires the system to distribute more attention to important information in a multi-turn context, while ignoring unimportant utterances. However, most existing works in this area have overlooked multi-turn modeling by treating a multi-turn context as a 1-turn context. Some approaches have attempted to address this issue by using modified attention mechanisms, hierarchical structures, and utterance tokens. Despite these efforts, the performance of multi-turn conversational agents on context attention distribution remains a challenge. In this thesis, we explore several architectures for a model's attention mechanism that addresses context attention distribution in multi-turn settings. We also propose an evaluation metric to measure a model's ability to distribute context attention and improve its performance through self-contained distractions addressing the proposed metric. Furthermore, we view the task of improving context attention distribution as a summarization task and employ extractive summarization techniques to enhance a model's context attention distribution.

The third challenge for open-domain generation-based conversational agents is the integration of knowledge. To address this challenge, we propose using a conversational question answering task perspective. Conversational question answering is a specialized dialogue system that can answer users' questions by leveraging a given document. It extends

traditional question answering systems to a conversational setting of multi-turn conversations to fulfill a user's information needs. The conversational question answering task comprises two parts: extractive question answering, which uses answer spans as responses, and generative question answering, which employs a generation-based conversational agent to generate answers from scratch. In this thesis, we aim to enhance generation-based conversational agents for conversational question answering using an extractive question answering system. We propose integrating instruction-based tuning, prompt-based tuning, and multi-task learning to improve performance.

In summary, this thesis aims to tackle three challenges: balancing multi-domain corpora to generate more relevant responses, improving a model's performance of context attention distribution in multi-turn settings, and integrating knowledge using extractive question answering with instruction-based and prompt-based tuning. We propose three research questions with corresponding sub-questions and answer them through four publications, each containing novel methodologies and extensive experimentation. While each of the three challenges can be addressed independently, our work suggests a synergistic approach to yield optimal performance. Specifically, balancing multi-domain corpora and improving context attention distribution are fundamentally interrelated: a well-balanced corpus allows the model to learn a more accurate context attention mechanism, which in turn enables the generation of more relevant responses. Additionally, the integration of knowledge through extractive question answering is influenced by how well the model understands the context, thereby showing a dependency on effective attention distribution.

Industrial needs align closely with these challenges, particularly as businesses seek to deploy conversational agents capable of complex, multi-turn dialogues across diverse subject matters. Therefore, we see these three challenges as relevant to each other and equally important. Solving these challenges collectively thus presents an opportunity for substantial advancements, both academically and industrially.

## 1.2 Research Questions and Approaches

The general goal of the thesis can be summarized as **integrating multi-domain corpora, multi-turn context, and knowledge into open-domain generation-based conversational**

**agents**. Specifically, we focus on three research questions as listed below:

- **RQ1** - *How can we balance multi-domain training corpora for generation-based conversational agents to improve the relevance of the generated responses?*

  *RQ1.1* What kind of approaches can be integrated into generation-based conversational agents to balance the training corpora? How do they perform?

  *RQ1.2* How do we evaluate the relevance of the generated responses corresponding to different corpora?

  *RQ1.3* How can we optimize the relevance of the generated responses for a generation-based conversational agent based on the proposed evaluation metric?

- **RQ2** - *How can we improve the awareness of multi-turn context on generation-based conversational agents?*

  *RQ2.1* How do we evaluate the context awareness for a generation-based conversational agent?

  *RQ2.2* How can we optimize the context awareness for a generation-based conversational agent based on the proposed evaluation metric?

  *RQ2.3* How can we integrate summarization techniques into generation-based conversational agents to improve the context awareness of the multi-turn context?

- **RQ3** - *How can we improve the quality of generated responses on knowledge for generation-based conversational agents under multi-turn conversational question answering context?*

  *RQ3.1* How can answer spans from the extractive question answering task be integrated into generation-based conversational agents and improve the quality of generated responses on knowledge?

  *RQ3.2* How can prompt-based tuning and instruction-based tuning improve the quality of generated responses on knowledge?

### 1.2.1 Approaches

We address the three main research questions through separate experiments regarding balancing multi-domain corpora for generation-based conversational agents, improving context awareness of multi-turn generation-based conversational agents, and integrating knowledge into multi-turn generation-based conversational agents, respectively. All the experiments adopt state-of-the-art deep learning models of generation-based conversational agents as the base models, namely LSTM (Hochreiter and Schmidhuber, 1997) and GPT-2 (Radford et al., 2019), whose structures are introduced in Chapter 2.2. Based on the state-of-the-art generation-based conversational agents, we examine a variety of techniques in the experiments to address the three main research questions respectively, which are introduced in Section 1.3. The techniques are then evaluated on common open-source English datasets such as Ubuntu (Lowe et al., 2015) and CoQA (Reddy et al., 2018), which addresses the problem each research question focuses on. The datasets are introduced in detail in Chapter 3 to Chapter 6. For evaluation, we propose two novel metrics: the Domain-specific Frequency (DF) for evaluating the relevance of a generated response corresponding multi-domain corpora, and Distracting Attention Score (DAS) ratio for evaluating the context awareness of a multi-turn conversational agent. These evaluation metrics are introduced in Chapter 3 and Chapter 4.

## 1.3 Publications and Contributions

In this section, we present the list of research papers published during the PhD studies, and summarize the contributions brought by them regarding the research questions. For each paper, we refer to the corresponding chapter where we include the content of the paper, and point out the relevance of the aforementioned research questions.

**Paper 1.** Yujie Xing, Jinglun Cai, Nils Barlaug, Peng Liu, and Jon Atle Gulla. 2022. *Balancing Multi-Domain Corpora Learning for Open-Domain Response Generation*. In Findings of the Association for Computational Linguistics: NAACL 2022, pages 2104-2120.

***Summary:*** The content of this paper is included in Chapter 3 and aims to answer the

research question **RQ1**.

***Contributions:*** We focus on the problem of open-domain response generation and the challenge of training and evaluating on multiple corpora from different domains. We explore several methods for making models generate relevant responses for each of the multiple corpora. The first method is interleaved learning, which intermingles the training data instead of simply concatenating them. This method ensures that the model learns from all corpora evenly. Interleaved learning serves as a baseline for the proposed methods. In addition to interleaved learning, we explore two multi-domain learning methods: labeled learning and multi-task labeled learning. Labeled learning is inspired by a control technique in response generation that focuses on controlling persona and style; however, it controls corpus information with a given corpus embedding. Multi-task labeled learning, although similar to labeled learning, minimizes both losses from both the corpus classifier and response generator, which enables a model to use a corpus classifier to choose a corpus embedding by itself.

Furthermore, we propose a novel method called weighted learning with Domain-specific Frequency (DF). DF is a word-level importance weight that assigns different weights to the same words from different corpora. In the training process, the model's loss is weighted with DF, so that the model focuses on the most important words for a specific corpus and thus improve the relevance of the generated responses. In the evaluation process, DF can be used for measuring the relevance of the generated responses. Results show that the best method (weighted learning) improves precision by 27.4%, recall by 45.5%, and F1 by 34.1%. Furthermore, it has 20.0% higher DF, indicating that it uses more important words from the most relevant corpus. To verify the automatic results, we also conduct an extensive human evaluation on 2400 generated responses. The human evaluation shows a highly significant improvement on all of the proposed methods, especially weighted learning.

**Paper 2.** Yujie Xing and Jon Atle Gulla. 2023. *Evaluating and Improving Context Attention Distribution on Multi-Turn Response Generation using Self-Contained Distractions*. In Computer Science & Information Technology (CS & IT), volume 13, number 02, pages 127-143.

***Summary:*** The content of this paper is included in Chapter 4 and aims to answer the

research question **RQ2.1** and **RQ2.2**.

***Contributions:*** We address evaluating and enhancing context attention distribution for multi-turn conversational agents. We propose a novel evaluation metric tailored for multi-turn conversational agents, Distracting Attention Score (DAS) ratio, to measure a model's performance on context attention distribution. Specifically, we propose the distracting test as an evaluation method that adds distracting utterances to the context of each dialogue and compares the attention scores of distracting utterances and original utterances. DAS ratio is then calculated as the ratio of the average attention score of distracting utterances to original utterances.

To further improve the performance of multi-turn conversational agents, we propose a self-contained optimization strategy that inserts randomly picked utterances into the current dialogue and teaches a model to minimize the attention paid to these unimportant extraneous utterances. Extensive experiments on 23 model variants and 9 distracting test sets show that the proposed optimization strategy has improved 10% on DAS ratio, meaning that the attention paid to unimportant utterances is reduced and the models' ability to distribute attention to important utterances is improved.

**Paper 3.** Yujie Xing and Jon Atle Gulla. 2023. *Improving Context-Awareness on Multi-Turn Dialogue Modeling with Extractive Summarization Techniques*. In 28th International Conference on Natural Language and Information Systems (NLDB 2023).

***Summary:*** The content of this paper is included in Chapter 5 and aims to answer the research question **RQ2.3**.

***Contributions:*** Following the aforementioned work, we investigate the application of extractive summarization techniques to enhance the context awareness of multi-turn conversational models. Specifically, we filter out unimportant utterances using extractive summarization techniques using the last utterance in the context, the query, as the reference, since the responses to be generated are primarily focused on answering the query. To achieve this, we use a PMI topic model to extract keywords from the context, which are then passed to the dialogue model. Additionally, we employ the ORACLE algorithm, a widely-used algorithm for generating gold labels for extractive summarization, to filter out

utterances unrelated to the query before passing them to the dialogue model. We examine the effectiveness of these techniques on both non-hierarchical and hierarchical models, where the best setting with ORACLE gains a 23% improvement on the DAS ratio, meaning that unimportant utterances are filtered out properly.

**Paper 4.** Yujie Xing and Peng Liu. 2023. *Prompt and Instruction-Based Tuning for Response Generation in Conversational Question Answering*. In 28th International Conference on Natural Language and Information Systems (NLDB 2023).

***Summary:*** The content of this paper is included in Chapter 6 and aims to answer the research question **RQ3**.

***Contributions:*** We propose a novel approach for enhancing response generation in conversational question answering by integrating prompt-based and instruction-based tuning. This approach represents the first application of instruction tuning to response generation in conversational question answering. In this paper, we first distinguish two angles from the conversational question answering task: the extractive one and the generative one, where the first employs answer spans as the expected response while the second asks for a generated response. We then adopt prompt-based tuning to improve the extractive angle in the conversational question answering task and instruction-based tuning to enhance the generative angle. Finally, we investigate the integration of these two angles through multi-task learning.

The experiments conducted in this study verify the influence of prompt-based tuning, instruction-based tuning, and multi-task learning on conversational question answering performance. Various settings, including prompt-based tuning with or without multi-task learning, prompt-based with or without instruction-based tuning, and prompt-based tuning with both multi-task learning and instruction-based tuning, are evaluated on GPT-2 using two modes of evaluation on the F1 score. The results show that the prompt-based tuning combined with either instruction-based tuning or multi-task learning improves the F1 score by about 18% over the baseline. The extractive question answering angle of the settings is assessed comparing with a GPT-2 fine-tuned on the extractive question answering task, and the results show a maximum 13% improvement when combining all the examined techniques.

The relationship between the publications and the research questions is illustrated in Table 1.1:

| | RQ1 | RQ2 | RQ3 |
|---|---|---|---|
| **Paper 1** | • | | |
| **Paper 2** | | • | |
| **Paper 3** | | • | |
| **Paper 4** | | | • |

Table 1.1: Relationship between ublications and research questions

## 1.4 Thesis Structure

The thesis is constituted of 7 chapters.

- Chapter 1 gives an overview and summarization of the thesis. Chapter 2 introduces the technical background and recent progresses in generation-based conversational agents.
- Chapter 3 includes **Paper 1**, which focuses on balancing multi-domain corpora for improving the relevance of generated responses.
- Chapter 4 includes **Paper 2**, which focuses on improving and evaluating context awareness in multi-turn context with self-contained distractions.
- Chapter 5 includes **Paper 3**, which focuses on improving context awareness with extractive summarization techniques.
- Chapter 6 includes **Paper 4**, which focuses on integrating knowledge using the conversational question answering task with prompt-based and instruction-based tuning.
- Finally, Chapter 7 gives a conclusion regarding the proposed research questions.

## 1.5 Research Context

The research work in this PhD thesis has been carried out as part of a four-year PhD program at the Department of Computer Science at Norwegian University of Science and Technology within the DNB collaboration project. The DNB collaboration project is funded by DNB (Den Norske Bank) with project number 90393101. The main objective of the DNB collaboration project is to research and develop the next-generation conversational agents for the bank system. The work is also carried out in collaboration with the Norwegian Research Center for AI Innovation (NorwAI) at NTNU, with project number 990109102.

# Chapter 2

# Background

In this section, we introduce the background of the thesis. We start with literature reviews in Section 2.1, where we give a thorough review of the topics covered in this thesis. We then illustrate the technical background in Section 2.2, where we introduce the architecture of all the basic models that we use as baselines in this thesis. Finally, we describe the challenges in Section 2.3, where we identify the challenges that the research questions come from.

## 2.1 Literature Reviews

### 2.1.1 A Brief Introduction to Conversational Agents

Conversational agents are specialized software programs designed to engage with users in natural language dialogue, facilitating various tasks or simply providing information. They are rooted in Natural Language Processing (NLP), a subfield of artificial intelligence (AI), and they focus on the interaction between computers and humans through natural language (Diederich et al., 2022). Conversational agents are often designed to understand context, manage conversation flow, and even adapt to users over time. They are typically more advanced than traditional chatbots, which are usually scripted and rule-based, lacking the ability to handle a wide variety of conversational contexts. Chatbots generally follow predetermined pathways or use simple keyword matching to interact with users.

On the other hand, a dialogue system is a broader term that encompasses both chatbots

and conversational agents (Arora et al., 2013). It refers to any system that is designed to converse with humans, whether through text, voice, or other modalities. Dialogue systems can range from rudimentary systems that rely on pre-defined scripts to highly sophisticated ones that use machine learning and NLP to interpret and generate human-like responses. While all conversational agents and chatbots are dialogue systems, not all dialogue systems can be accurately described as conversational agents or chatbots. For example, automated phone systems that navigate through a series of preset options are technically dialogue systems but lack the natural language understanding and generation capabilities of conversational agents.

Some conversational agents can interact through various modes such as speech, text, multi-modal interfaces, or even videos (Pelachaud, 2005). Speech-based conversational agents often find applications in voice-activated systems like smart home devices and are integral to technologies like Siri, Alexa, or Google Assistant. They focus on understanding spoken language and providing audio responses. Text-based conversational agents are more commonly found in customer service applications, messaging platforms, and other written forms of communication. Multi-modal conversational agents are more versatile, combining text, speech, and sometimes even gestures to interact with users, often found in advanced robotics or augmented reality interfaces. Video-based conversational agents add another layer of complexity by incorporating visual cues, such as facial expressions or body language, to make the interaction more human-like.

In this thesis, we will focus on pure text-based conversational agents. These agents deal solely with written language, interpreting text input from users and generating text responses. Despite not having the auditory or visual cues available in other modes, text-based conversational agents have seen widespread adoption across various industries, primarily due to their lightweight and efficient nature. Unlike video or speech-based agents, text-based systems don't require heavy computational resources for processing audio-visual data, making them more accessible and easier to deploy.

An example of a generation-based conversational agent (Brown et al., 2020) is as follows:

*User:* "What's your favorite color?"

*Agent:* "I don't have preferences like humans do, but I can tell you that blue is often

associated with calmness and red with passion. What's your favorite color?"

*User:* "I like green."

*Agent:* "Green is a color often associated with nature and renewal. It's interesting that you like green; it's a color that has a lot of positive connotations."

*User:* "Tell me a joke."

*Agent:* "Sure, why did the scarecrow win an award?"

*User:* "I don't know, why?"

*Agent:* "Because he was outstanding in his field!"

### 2.1.2 Generation-Based Conversational Agents

As mentioned above, conversational agents can be categorized into three types: rule-based conversational agents, information-retrieval-based (IR-based) conversational agents, and generation-based conversational agents. Conventional rule-based conversational agents, exemplified by ELIZA (Weizenbaum, 1966), rely on manually crafted templates to address diverse query types. Despite garnering affirmative evaluations (Colby et al., 1972), the reliance on such handcrafted templates imposes significant constraints on both the range of feasible answers and response patterns. The inherent limitation lies in the fact that these templates can only furnish a finite array of outputs. In contrast, IR-based conversational agents leverage extensive corpora to derive responses, drawing from the semantic likeness between inquiries and potential retorts within the corpus (Jurafsky and Martin, 2009). By capitalizing on the voluminous expanse of the corpus, these agents surpass rule-based counterparts in generating a wider spectrum of responses. Nonetheless, these agents are confined to extracting information from the corpus, which hinders their ability to generate entirely novel responses.

Generation-based conversational agents, in contrast to their corpus-dependent counterparts, utilize words from a vocabulary list to "synthesize" responses autonomously, enabling the production of novel responses. The conceptual framework underlying these agents is akin to that of machine translation, wherein the translation references are replaced by the responses expected to be generated. The initial strides in open-domain response generation were heavily influenced by the work of Ritter et al. (2011), treating the task akin to

machine translation. Subsequently, propelled by the advancements in neural network technologies, the application of sequence-to-sequence models (Sutskever et al., 2014) gained prominence in the realm of generation-based conversational agents, as demonstrated by the works of Vinyals and Le (2015), Shang et al. (2015) and Sordoni et al. (2015). However, the recent proliferation of robust large-scale language models like GPT-2 (Radford et al., 2018, 2019) catalyzed a surge in the development of generation-based conversational agents, outperforming their sequence-to-sequence predecessors by a considerable margin (Zhang et al., 2020).

Figure 2.1 illustrates the difference in the architecture of rule-based, IR-based, and generation-based conversational agents.

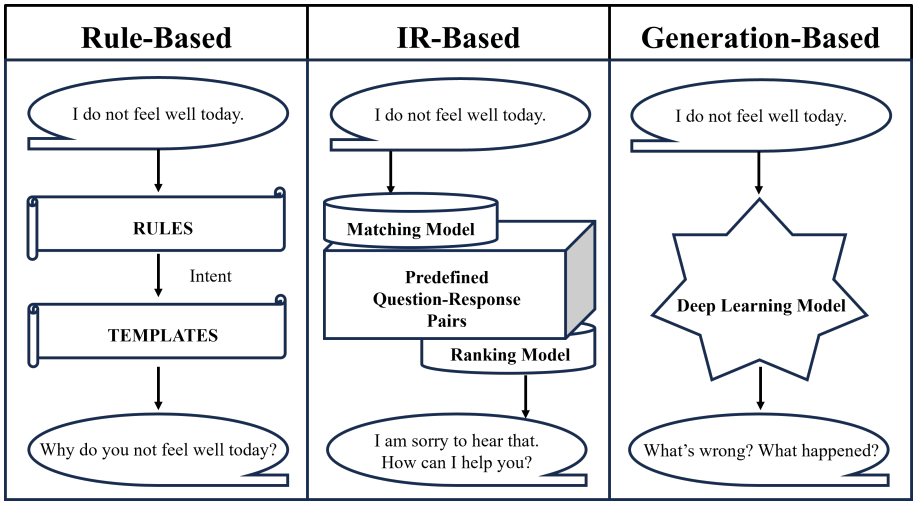



Figure 2.1: Different architecture of rule-based, IR-based and generation-based conversational agents

While generation-based methods offer an impressively fluid dialogue, there are scenarios where the two other methods remain favored. For instance, in financial institutions where the margin for error is infinitesimal, rule-based systems can ensure 100% accuracy, eliminating the risk of "hallucinations" or inaccurately generated responses. Similarly, IR-based agents are indispensable when detailed, lengthy, and specific answers are required, as they can pull this information directly from verified sources. Moreover, these older

methodologies may be preferable in environments where data collection is restricted or sparse, making it challenging to train a robust generation-based model (Ji et al., 2023).

In aware of the above, Generation-based conversational agents are particularly well-suited for scenarios that require a high degree of contextual understanding and flexibility in responses, since they can adapt to the nuances of human language, provide more personalized responses, and manage a wide range of queries, all while maintaining a natural-sounding conversation flow (Brown et al., 2020). Additionally, these agents can better handle ambiguous or unclear questions by either generating a clarifying question or making educated guesses based on context, something that rule-based or IR-based systems might struggle with. An important scenario where generation-based methods shine is in interactive applications that aim to provide a more human-like experience. In these contexts, the goal is often to emulate human interaction as closely as possible, making the dynamic and adaptive nature of generation-based agents invaluable. To achieve these goals, a generation-based conversational agent should have the ability to balance multiple knowledge domains, deal with multiple question-response turns, and integrate knowledge into the dialogue, which we will address in this thesis.

### 2.1.3 Multi Domains for Conversational Agents

Multi-domain is a concept in the field of natural language processing that refers to the ability of a system, such as a conversational agent, to operate effectively across multiple topics, contexts, or types of data (Ben-David et al., 2007). In the field of conversational agents, multi-domain competence is not just a luxury but a necessity. Conversational agents are increasingly integrated into various aspects of our daily lives—from customer service and healthcare to personal assistance and social interaction. As these applications span multiple domains, from daily chats to diagnosing system malfunctions, the ability of a conversational agent to understand and adapt to different domains becomes crucial.

In a single-domain setting, the conversational agent is specialized in handling tasks or discussions related to one specific area or subject. A customer support bot for a telecom company can help you with queries about your bill, new plans, or technical issues, but it won't be able to book a flight ticket for you or offer medical advice. Its domain or area

of expertise is 'telecom customer support'. On the contrary, **multi-domain** conversational agents extend this functionality by being skilled in more than one domain. It's not as specialized as a single-domain agent but has broader capabilities. Multi-domain conversational agents need to manage which domain to refer to when they receive questions from different domains.

An open-domain conversational agent aims to engage in conversation across any topic you can think of—be it politics, philosophy, pop culture, or even personal advice. These agents require a deeper understanding of human language, context, and intent as they aim to generate meaningful, coherent, and contextually relevant responses across an unlimited set of domains. Open-domain conversational agents are assumed to give stable responses across multiple domains, but this is not always the case. Techniques like multi-domain learning (Joshi et al., 2012) and multi-task learning (Luan et al., 2017; Niu and Bansal, 2018), and fine-tuning (Akama et al., 2017) have been employed for domain adaptation.

To sum it up, generation-based conversational agents often serve open domains, offering versatility without the complexity of handling any possible topic, which means that they need to change among multiple domains smoothly and generate responses accordingly.

### 2.1.4 Multi Turns for Conversational Agents

Not all conversational agents are created equal, and one significant differentiator is their ability to handle multi-turn conversations versus single-turn conversations.

Single-turn conversational agents (Zhang et al., 2020) respond to a single user input with a single output and do not maintain any context or history of previous interactions. Each exchange between the user and the agent is self-contained. For example, if you ask a single-turn agent what the weather is like, it might respond, "It's sunny and 75 degrees." If you then ask, "How about tomorrow?", the agent will not understand that "tomorrow" refers to the weather because it has no memory or context of the previous interaction.

On the other hand, multi-turn conversational agents (Serban et al., 2016) can maintain a dialogue context over multiple turns of exchanges. So, if you ask the same multi-turn agent what the weather is like and then ask, "How about tomorrow?", it would understand that you are still talking about the weather and provide a relevant answer, such as, "Tomorrow,

it's expected to be cloudy with a high of 68 degrees."

Multi-turn capability is crucial for more complex interactions and tasks. In real-life conversations, questions and answers are often not self-contained but rely on the context built up over multiple turns. For example, you might ask a travel agent, "Can you recommend a holiday destination?", followed by a series of related questions like, "How's the weather there?", "What activities can we do?", and so on. A multi-turn conversational agent can provide much richer, context-aware interactions, closely mimicking the dynamics of human conversation. While single-turn conversational agents can be useful for simple, isolated tasks, multi-turn conversational agents offer a more sophisticated, natural, and satisfying user experience, particularly for complex queries and tasks that require context and ongoing dialogue.

### 2.1.5 Conversational Question Answering

Conversational Question Answering (CQA) (Reddy et al., 2018; Choi et al., 2018) is a specialized area within Natural Language Processing (NLP) that focuses on enabling machines to understand, process, and respond to questions posed in natural language during a dialogue or conversation. Unlike traditional question-answering systems that provide answers to isolated queries based on a given document, conversational agents built for CQA also need to keep track of the context on the basis of a given document to generate contextually appropriate and coherent answers.

An example of traditional question answering can be as follows.

*User:* "What is the capital of France?"

*System:* "The capital of France is Paris."

An example of conversational question answering can be as follows.

*User:* "Tell me about France."

*System:* "France is a country in Western Europe known for its culture, history, and landmarks like the Eiffel Tower."

*User:* "What's its capital?"

*System:* "The capital of France is Paris."

In the second example, the system recognizes that "What's its capital?" refers to France,

which was mentioned earlier in the conversation. This is a simple instance, but CQA can get quite complex, especially in lengthy dialogues or those that involve multiple entities, changing topics, or nested questions.

### 2.1.6 Prompt-Based Tuning and Prompt-Based Tuning

On one hand, prompt-based tuning is essentially the practice of fine-tuning a conversational model based on the way you structure the prompt or the query you send to the model, which usually introduces special characters to the prompts (Zhong et al., 2022). On the other hand, instruction tuning, similar to prompt-based tuning, involves the refinement and optimization of the *instructions* provided to a language model (Gupta et al., 2022). The difference is that it seeks to improve the model's understanding of the task or context through clear and specific instructions, which are in natural language. Here, you introduce a set of example inputs and their instructions along with the expected outputs during the training phase. The model then adjusts its internal parameters to better match these example outputs when given similar inputs in the future.

Both methods aim to improve the performance of conversational agents, but they do so in different ways. Prompt-based tuning is more structured while instruction-based tuning can be formed as natural languages. A technical introduction to prompt-based tuning and instruction-based tuning can be found in Section 6.2.4.

## 2.2 Technical Background

The basic task of generation-based conversational agents is to predict the next token given all the past and current tokens from the context and response, and to make the predicted response as similar to the original response as possible. Formally, the probability of response $Y$ given context $X$ is predicted as:

$$P(Y|X) = \prod_{t=1}^{n} p(y_t|y_1, \ldots, y_{t-1}, X), \tag{2.1}$$

where $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ are a context-response pair.

### 2.2.1 RRN-Based Models

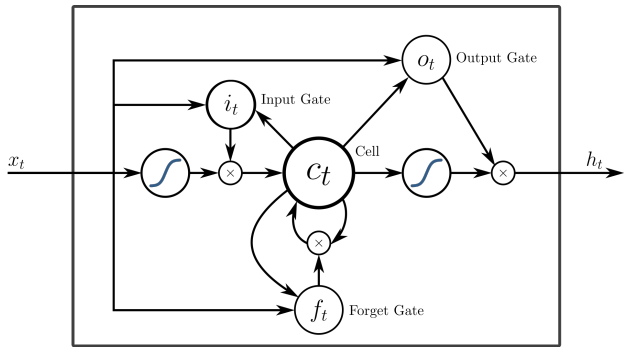



Figure 2.2: The architecture of an LSTM unit[1], adapted from Hochreiter and Schmidhuber (1997)

Recurrent Neural Networks (RNNs) are a type of neural network that is designed to handle sequential data, which suits the task of generation-based conversational agents. Unlike traditional feedforward neural networks, which process a fixed-size input and output, RNNs allow for the processing of variable-length input sequences by sharing the same set of weights across all time steps. This allows the network to maintain an internal state, which can capture important contextual information from the previous time steps and use it to inform the current prediction. Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) is a type of RNN that was designed to address the vanishing gradient problem, which can occur when training traditional RNNs. The vanishing gradient problem is caused by the fact that gradients can become exponentially small or large when backpropagating through many time steps. LSTMs use a more complex gating mechanism than traditional RNNs, which allows them to selectively retain or forget information from the previous time steps based on the current input. This gating mechanism, which is composed of input, forget, and output gates, helps LSTMs to maintain long-term dependencies and prevent the gradients from vanishing or exploding during training. The structure of an

LSTM unit is illustrated in Figure 2.2 and can be described as:

$$i_t = \sigma(W_{zi}E(z_t) + W_{hi}h_{t-1} + W_{ci}c_t + b_i) \tag{2.2}$$

$$f_t = \sigma(W_{zf}E(z_t) + W_{hf}h_{t-1} + W_{cf}c_t + b_f) \tag{2.3}$$

$$\tilde{C}_t = \tanh(W_{zc}E(z_t) + W_{hc}h_{t-1} + W_{cc}c_t + b_c) \tag{2.4}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.5}$$

$$o_t = \sigma(W_{zo}E(z_t) + W_{ho}h_{t-1} + b_o) \tag{2.6}$$

$$h_t = o_t \odot \tanh(C_t) \tag{2.7}$$

where $E(z_t)$ is the word embedding for word $z_t \in (x_1, \ldots, x_m, y_1, \ldots, y_{n-1})$, $h_{t-1}$ is the hidden state vector from the previous step, and $c_t$ is the context vector passed only to the decoder at step $t$. $i_t$ is the input gate vector at step $t$, $f_t$ is the forget gate vector at step $t$, $\tilde{C}_t$ is the candidate cell state vector at step $t$, $C_t$ is the cell state vector at step $t$, $o_t$ is the output gate vector at step $t$, and $h_t$ is the hidden state vector at step $t$. $W_{zi}, W_{zf}, W_{zc}, W_{xo}$ are weight matrices applied to input vector $z_t$. $W_{hi}, W_{hf}, W_{hc}, W_{ho}$ are the weight matrices applied to the hidden state vector $h_{t-1}$. $b_i, b_f, b_c, b_o$ are the bias terms. $\sigma$ is the sigmoid function, and $\odot$ denotes element-wise multiplication.

In this thesis, we apply dot multiple in the attention mechanism when calculating the context vector $c_t$:

$$c_t = H \cdot (softmax(H^\top \cdot h_{t-1}) \tag{2.8}$$

where $H \in \mathbb{R}^{d \times m}$ is the concatenation of hidden vectors from the encoder. $c_t$ is input to step $t$ in the decoder.

In the following chapters, we simplify the structure containing multiple layers of the above-described LSTM units with attention unit as $LSTM^*$. We calculate the hidden vector $h_t$ at step $t$ as:

$$h_t = LSTM^*(E(z_t), h_{t-1}, c_t) \tag{2.9}$$

where $E(z_t)$ is the word embedding for word $z_t \in (x_1, \ldots, x_m, y_1, \ldots, y_{n-1})$, $h_{t-1} \in \mathbb{R}^{dim}$ is the hidden vector at step $t-1$, $dim$ is the dimension of hidden vectors, and $c_t$ is the

[1] https://en.wikipedia.org/wiki/Long_short-term_memory

context vector at step $t$.

### 2.2.2 Transformer-Based Models

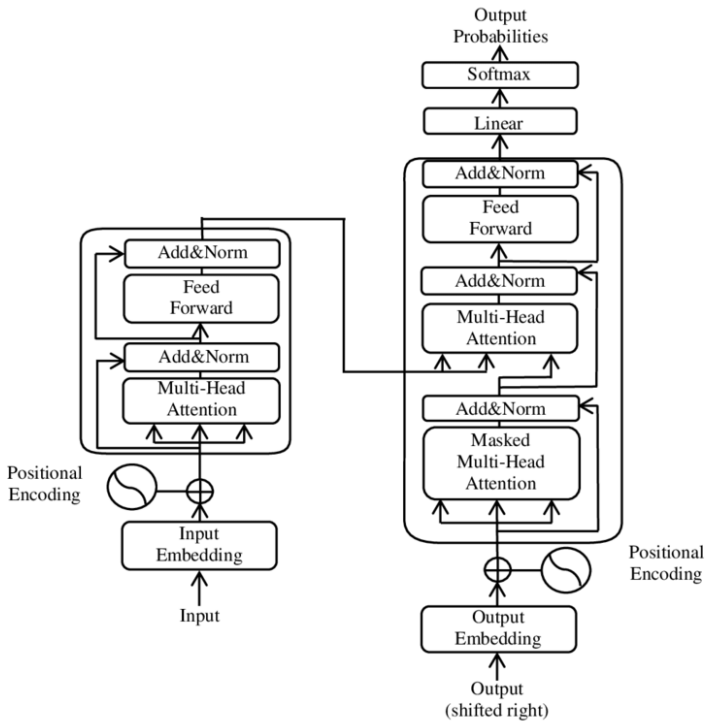



Figure 2.3: The architecture of Transformer[2], adapted from Vaswani et al. (2017)

The Transformer model (Vaswani et al., 2017) is a type of neural network architecture that is based solely on the attention mechanism. Unlike previous sequence models, the Transformer does not rely on recurrent neural networks (RNNs) or convolutions. Instead, it uses multi-head attention to capture dependencies between different parts of the input sequence, and position-wise fully connected feed-forward layers to transform the representations. The model consists of an encoder and a decoder, each of which contains multiple

layers of self-attention and feedforward neural networks. The architecture of the Transformer model is illustrated in Figure 2.3.

The Multi-Head Attention module based on the Scaled Dot-Product Attention is the core of the Transformer model. The structure of this module is illustrated in Figure 2.4. It is clear from the figure that the multi-head attention consists of several "heads," each of which applies Scaled Dot-Product Attention to a different projection of the input.

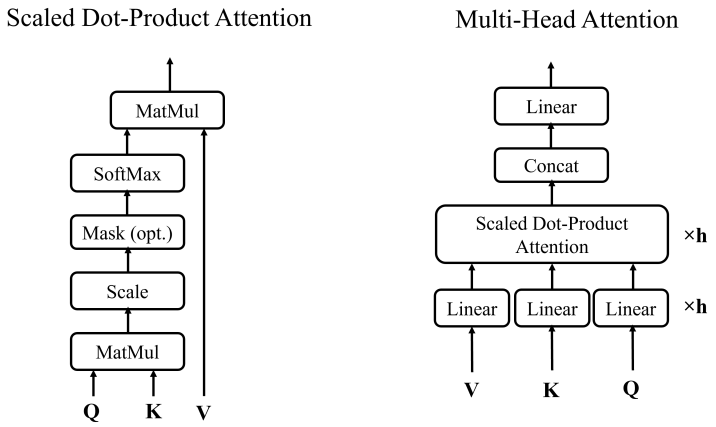



Figure 2.4: The architecture of the Scaled Dot-Product Attention and the Multi-Head Attention, adapted from (Vaswani et al., 2017)

The Scaled Dot-Product Attention conducts self-attention by transforming the input sequence into three parts: the queries $Q$, the keys $K$, and the values $V$, all of which have the same dimensionality $dim$. The Scaled Dot-Product Attention can be described as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{dim}}\right)V. \tag{2.10}$$

This equation can be interpreted as a weighted sum of $V$, where the weights are determined by the dot product between each $Q$ and each $K$, scaled by the square root of the dimensionality of each $K$. In practice, the transforming of the input vectors into $Q$,$K$,

[2] https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)

and $V$ is achieved through linear transformation, where each weight matrix has the same dimensionality as the input vectors, and is learned during the training process.

Instead of performing a single self-attention on the input vectors with the model's dimension, the multi-head attention splits the model dimension $dim$ evenly $h$ designated heads with dimension $hdim$, which allows the model to attend to multiple aspects of the input sequence simultaneously. The queries $Q$, the keys $K$, and the values $K$ are split into $h$ sets and are projected into these $hdim$-dimensional subspaces using learned linear projection matrices, after which the Scaled Dot-Product Attention is applied separately to each set of $Q$, $K$ and $V$. It can be described as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_h\right) W^O, \tag{2.11}$$

where $Q, K, V \in \mathbb{R}^{dim}$, $W^O \in \mathbb{R}^{dim^h \times dim}$, and $dim_h$ denotes for the sum for all $hdim$.

For each $\text{head}_i$ ($i \in \{1, \ldots, h\}$), the Scaled Dot-Product Attention is applied as follows:

$$\text{head}_i = \text{Attention}(Q_i, K_i, V_i) = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{hdim}}\right) V, \tag{2.12}$$

where $Q_i$, $K_i$ and $V_i$ are the projections from $Q$, $K$ and $V$ to the $i$-th head with the learned linear projection matrices. In practice, usually $hdim = dim/h$.

The Multi-Head Attention module allows the model to jointly attend to different parts of the input sequence in parallel, which can improve the model's ability to capture complex dependencies and long-range dependencies. Each of the multiple heads in the module focuses on a different aspect of the input sequence, which allows the model to capture both local and global dependencies between words, making it particularly suitable for the response generation task.

After the Multi-Head Attention module, there follows the Position-Wise Feed-Forward module, which consists of two linear transformations followed by a non-linear activation function. Specifically, given an input vector $x \in \mathbb{R}^{n \times dim}$, where $n$ is the length of the sequence and $dim$ is the hidden dimensionality of the model, the FFN can be expressed as follows:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2, \tag{2.13}$$

where $W_1 \in \mathbb{R}^{dim \times d_{ff}}$, $b_1 \in \mathbb{R}^{d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff} \times dim}$, and $b_2 \in \mathbb{R}^{dim}$ are learnable weight matrices and bias vectors, and $d_{ff}$ is the size of the hidden layer in the feed-forward network. The non-linear activation function used in the Transformer is the ReLU function, which is applied element-wise.

The purpose of the position-wise FFNs is to provide a simple yet powerful mechanism for modeling complex relationships between input tokens, independent of their positions. By applying a non-linear transformation to each token independently, the FFNs can capture higher-order interactions between tokens that might not be captured by the self-attention mechanism alone.

The Transformer has achieved state-of-the-art performance on a wide range of natural language processing tasks, including machine translation, language modeling, and question answering. For the response generation task, a sub-structure of the Transformer consisting of only the decoder part is widely used, since the decoder part is designed to predict the next word in a sequence based on the context of the previous words, which is exactly what the task asks for. In this thesis, we employ GPT (Generative Pre-trained Transformer) (Radford et al., 2018, 2019), which is a pre-trained transformer using only the decoder part of the Transformer architecture. Specifically, we utilize GPT for a response generation task following Wolf et al. (2019) and calculate the hidden vector to be input to the Transformer block as follows:

$$h_{0[t]} = E(X, Y_{[1:t]}) + (E_0, E_1) + W_p, \tag{2.14}$$

where $Y_{[1:t]}$ is $(y_1, \ldots, y_t)$, $E(X, Y_{[1:t]})$ is the sub-word embedding for context $X$ and response $Y_{[1:t]}$. $E_0$ and $E_1$ are dialogue-state embeddings, which tutor the model to distinguish between contexts and responses. $W_p$ is a pre-trained position embedding. The probability of the subword to generate is then calculated as:

$$h_{[t]} = \text{GPT\_Block}(h_{0[t]}) \tag{2.15}$$

$$P(y)_{t+1} = \text{softmax}(E^{\top}(h_{[t]})), \tag{2.16}$$

where $y \in V$, and $V$ stands for the sub-word vocabulary. We simplify the Transformer block of GPT as GPT_Block. We fill a mask to the attention matrix to ban past words from attending to future words, which ensures that the model follows the traditional language

modeling. The hidden vector of $t_{\text{th}}$ sub-word is used to generate the probability distribution for the vocabulary ($P(y),\ y \in V$) for $(t+1)_{\text{th}}$ sub-word. $E^{\top}$ means that the model uses the sub-word embeddings in calculating sub-word probabilities for generation.

## 2.3 Research Challenges

### 2.3.1 Balancing Multi Domains for Conversational Agents

In the last ten years, substantial improvements have been made (Serban et al., 2017; Li et al., 2016; Wolf et al., 2019) on generation-based conversational agents; however, most works are restricted to single-corpus training and evaluating, and there is a lack of work that balance generation-based conversational agents over multiple corpora. We thus propose research question **RQ1**: How can we balance multi-domain training corpora for generation-based conversational agents to improve the relevance of the generated responses? We use the first paper **Paper 1** to answer the question. **Paper 1** follows the common models of open-domain conversational systems while studying the problem of multiple corpora where each corpus comes from a different domain.

Previous works use embeddings to control response generation on extra information such as persona (Li et al., 2016), profiles (Yang et al., 2017), coherence (Xu et al., 2018), emotions (Huang et al., 2018), and dialogue attributes like response-relatedness (See et al., 2019). Nevertheless, there is a dearth of studies that leverage embeddings to control response generation across multiple corpora.

Another method to balance multi-domain over a generation-based conversational agent is through multi-domain learning, which aims at making a conversational model learn from multiple domains to prevent the performance from degrading due to domain differences (Ben-David et al., 2007). There are two categories of solutions for multi-domain learning (Joshi et al., 2012): (i) capturing domain-specific characteristics with additional parameters while preserving parameters that captured domain-general behaviors (Daumé III, 2007); (ii) capturing the relationship among different domains using tools like task-relationship matrix (Saha et al., 2011). Some work of natural language generation and machine translation is related to multi-domain learning. Luan et al. (2017) and Niu and Bansal (2018)

use multi-task learning for domain adaption respectively on speaker-role and politeness. Wen et al. (2016) and Akama et al. (2017) utilize fine-tuning as a common way of domain adaption for language generator and style transferer. For machine translation, in order to deal with the mixed-domain parallel corpus, Zeng et al. (2018) adjust the weights of target words in the training objective based on their relevance to different domains. **Paper 1** differs in that we propose DF and we deal with the response generation task. Chu et al. (2017) propose mixed fine-tuning, which adds the out-of-domain pre-training data to the fine-tuning dataset, and they observe an improvement of performance. In **Paper 1**, we also mix small-scale fine-tuning datasets with out-of-domain training data, while the data we add is not necessarily used during pre-training. Shi et al. (2015) state that fine-tuning can be done by placing the corpus to be fine-tuned at the end of the entire corpus, which is an extension of curriculum learning proposed by Bengio et al. (2009). We also explore how the order of multiple corpora influences the result in **Paper 1**, but our focus is on balancing performance. Recently, Smith et al. (2020) investigated blending conversational skills with knowledge and empathy skills, where they mix 3 corpora. They focus on selecting appropriate skills and they propose a blended corpus with labels, while **Paper 1** focuses on generating responses that are most relevant to a specific corpus.

### 2.3.2 Dealing with Multi-Turn Conversational Agents

The majority of generation-based conversational agents employ a straightforward concatenation technique for modeling multi-turn conversations (Zhao et al., 2020a; Zhang et al., 2020). Unfortunately, this method treats a multi-turn context as though it were a solitary utterance, thereby impeding the conversational agent's proficiency in handling multi-turn contexts. In this thesis, we delve into the prospect of this problem guided by the following research question **RQ2**: How can we improve the awareness of multi-turn context on generation-based conversational agents? We answer **RQ2** with papers **Paper 2** and **Paper 3**.

A conventional approach to modeling multi-turn conversations involves utilizing a hierarchical structure. Serban et al. (2016) and Serban et al. (2017) first introduce the hierarchical structure to dialogue models. Tian et al. (2017) evaluate different methods for

integrating context utterances in hierarchical structures. Zhang et al. (2018b) further evaluate the effectiveness of static and dynamic attention mechanism. Gu et al. (2021) apply a similar hierarchical structure on Transformer, and propose masked utterance regression and distributed utterance order ranking as the training objectives. Different from hierarchical models, Li et al. (2021) encode each utterance with a special token $[C]$ and apply a flow module to train the model to predict the next $[C]$; then they use semantic influence (the difference of the predicted and original tokens) to support generation. In **Paper 2**, instead of modelling the relations of inter-context utterances as Gu et al. (2021) or the dialogue flow as Li et al. (2021), we propose an optimization strategy that improves multi-turn modelling by distinguishing important/unimportant utterances directly on the attention mechanism, which is also used as a novel evaluation metric. Common evaluation metrics for conversational agents measure the similarity between the generated responses and the gold responses, while they do not gauge a generation-based conversational agent's proficiency in handling multi-turn contexts. Liu et al. (2016) summarizes commonly used metrics: word overlap-based metrics (e.g. BLEU) and embedding-based metrics. Bruni and Fernandez (2017) propose an adversarial evaluation method, which uses a classifier to distinguish human responses from generated responses. Lowe et al. (2017) propose a model that simulates human scoring for generated responses. Zemlyanskiy and Sha (2018) examine the quality of generated responses in a different direction: how much information the speakers exchange with each other. Recently, Li et al. (2021) propose a metric that evaluates the human-likeness of the generated response by measuring the gap between the corresponding semantic influences. Different from the above, the evaluation metric proposed by **Paper 2** is based on the attention mechanism and is intended to measure a model's performance on attributing attention to important utterances in a multi-turn context.

In **Paper 3**, we examine using context-summarization modules to improve the awareness of multi-turn context for conversational agents with and without hierarchical structure. A similar direction of combining summarization and multi-turn dialogue modeling is the integration of topic models, though current works in this direction are all on single-turn dialogues. Li and Sun (2018) uses a classifier to select the keyword for a given query from a pre-generated keyword list. Yao et al. (2017b) and Mou et al. (2016) use PMI to choose a keyword for a given query from a big corpus. Similarly, Xing et al. (2017) and

Baheti et al. (2018) uses a topic model to predict the keyword out from vocabulary words. In **Paper 3**, we also examine if a topic model can improve the context-awareness of dialogue models. As mentioned in Yao et al. (2017a), a typical way to construct labeled data for extractive summarization is to set ROUGE. Most works including Kedzie et al. (2018) construct gold label sequences by greedily optimizing ROUGE-1, which is the algorithm ORACLE. Further, although in **Paper 3** we stick to extractive summarization due to lack of suitable conversational datasets for abstractive summarization, we look forward to this kind of dataset from Gliwa et al. (2019).

### 2.3.3 Conversational Question Answering, prompt-based tuning and instruction-based tuning

For generation-based conversational agents, integrating knowledge is a hot topic to answer (Parthasarathi and Pineau, 2018; Ghazvininejad et al., 2018), and we deal with it using the task of generative conversational question answering (QA), which provides documents as a knowledge source and require a model to answer a question based on the knowledge source. We propose the research question **RQ3**: How can we improve the quality of generated responses on knowledge for generation-based conversational agents under multi-turn conversational question answering context? In addressing this query, **Paper 4** explores the impact of incorporating instruction tuning, prompt tuning, and multi-task learning on the performance of generation-based conversational agents under the task of multi-turn conversational question-answering.

Generative QA models (Izacard and Grave, 2021; Khashabi et al., 2020; Lewis et al., 2020; Raffel et al., 2020) have shown remarkable performance, where the goal is to generate answers by autoregressively predicting tokens. Generative methods are more often used in open domains (Izacard and Grave, 2021; Lewis et al., 2020; Raffel et al., 2020; Xiong et al., 2021) and unified settings (Khashabi et al., 2020; Tafjord and Clark, 2021). Raffel et al. (2020) proposed to use large pre-trained generative models, without using additional knowledge, for open-domain question answering. Lewis et al. (2020) introduced retrieval-augmented generative models for open-domain question answering. Khashabi et al. (2020) and Tafjord and Clark (2021) proposed to learn various QA formats in a unified way to

alleviate the manual effort of task-specific design.

In earlier times, recurrent neural networks (RNN) and attention variations were used to model multi-turn conversational QA tasks (Reddy et al., 2018; Zhu et al., 2018). Modern approaches leverage transformer-based pre-trained language models for QA by fine-tuning the models on annotated data from downstream QA tasks (Joshi et al., 2020; Lan et al., 2019; Chada and Natarajan, 2021; Ram et al., 2021). Capitalizing on pre-trained large language models, recent efforts have incorporated prompt-based tuning into the realm of multi-turn conversational question answering, which enhances the performance of generation through refining or optimizing the prompts given to a large language model. For instance, Chada and Natarajan (2021) proposed to cast QA as a text-generation problem by designing a prompt of a concatenation of the question and a special mask token representing the answer span. Similarly, Chen et al. (2023) proposed to use Masked Language Model on entities to enhance few-shot QA learning. A technical introduction of prompt-based tuning can be found in Section 6.2.3.

However, none of the aforementioned research works adopt instructions in prompt-based tuning for QA tasks. Recent literature has been motivated by building models that are generalizable across a variety of NLP tasks when prompted with a few examples (Bragg et al., 2021; Min et al., 2022a,b) or language definitions and constraints. Weller et al. (2020); Xu et al. (2022) introduced natural language instructions to improve the performance of LMs such as BART and GPT-3 for cross-task. Followed by this, FLAN Wei et al. (2022) has been proposed, which uses instructions to achieve generalization across unseen tasks. Recently, Khashabi et al. (2022) have shown that reframing instructional prompts can boost both few-shot and zero-shot model performance. The InstructGPT model is proposed, which is fine-tuned with human feedback (Ouyang et al., 2022). Puri et al. (2022) introduced instruction augmentation to improve model performance in task-specific, multi-task and cross-task learning paradigms. Prasad et al. (2022) introduced Gradient-free Instructional Prompt Search (GrIPS) to improve task instructions for large language models. Motivated by the effectiveness of instruction tuning. in **Paper 4**, we explore the potential application of employing the combination of instruction-based and prompt-based tuning for conversational question answering response generation.

# Chapter 3

# Balancing Multi-Domain Corpora

Open-domain conversational systems are assumed to generate equally good responses on multiple domains. Previous work achieved good performance on the single corpus, but training and evaluating on multiple corpora from different domains is less studied. This paper explores methods of generating relevant responses for corpora from different domains. We first examine interleaved learning which intermingles multiple corpora as the baseline. We then investigate two multi-domain learning methods, labeled learning and multi-task labeled learning, which encode each corpus through a unique corpus embedding. Furthermore, we propose Domain-specific Frequency (DF), a novel word-level importance weight that measures the relative importance of a word for a specific corpus compared to other corpora. Based on DF, we propose weighted learning, a method that integrates DF to the loss function. We also adopt DF as a new evaluation metric. Extensive experiments show that our methods gain significant improvements on both automatic and human evaluation. We share our code and data for reproducibility.[1]

## 3.1 Introduction and Related Works

Recent work has achieved improvements in general performance for open-domain response generation (Vinyals and Le, 2015; Serban et al., 2017; Li et al., 2016; Xu et al., 2018).

[1] https://github.com/yujie-xing/Balancing_Multi_Domain_Corpus_Learning_for_Open_Domain_Response_Generation

| Context | What are you going to do on the remote system exactly? |
|---|---|
| PersonaChat | I am going to be a pilot. I am going to fly planes. |
| 4 corpora (concatenated) | I am going to go to the beach. |

Table 3.1: Irrelevant responses generated from fine-tuned GPT-2. The GPT-2 model is fine-tuned respectively on PersonaChat / concatenated 4 corpora (OpenSubtitles, Twitter, Ubuntu, PersonaChat)

| | Test set | | | |
|---|---|---|---|---|
| Fine-tune corpus | OSDB | Twitter | Ubuntu | PersonaChat |
| PersonaChat | 478.8 | 159.6 | 264.7 | 19.6 |
| 4 corpora (concatenated) | 392.8 | 110.7 | 199.2 | 19.0 |

Table 3.2: Imbalanced perplexity performance of fine-tuned GPT-2. The GPT-2 model is fine-tuned on PersonaChat / concatenated 4 corpora (OpenSubtitles, Twitter, Ubuntu, PersonaChat)

However, most studies are restricted to single-corpus training and evaluating, while there is a lack of studies that train and evaluate corpora from different domains. Single-corpus training has intrinsic limitations. For example, a corpus of everyday chats, e.g., the PersonaChat corpus (Dinan et al., 2019), does not cover technical topics discussed in Ubuntu chatlogs (Lowe et al., 2015). A conversational system that learns only from PersonaChat or from multiple corpora without an appropriate technique is not likely to generate relevant responses for certain topics (see Table 3.1). Therefore, it is necessary for an open-domain conversational system to learn from multiple corpora, and to learn with good techniques.

Furthermore, the case of using a single small-scale open-domain corpus has apparent weaknesses. A common way of dealing with a small-scale corpus is through fine-tuning (Li et al., 2016; Akama et al., 2017; Chu et al., 2017). Fine-tuning on a single corpus tends to make the model overfit on that specific corpus while performing worse on other corpora. Table 3.2 shows the result of a GPT-2 model gaining good performance on PersonaChat while performing poorly on other corpora.

This paper explores how to train and evaluate on multiple corpora from different domains for the open-domain response generation task. We propose several methods to make

a model generate relevant responses for each of the multiple corpora.

Since simply training multiple corpora one by one does not solve the imbalanced performance (as shown in Table 3.1 and 3.2), we first investigate *interleaved learning*, a method that intermingles the training data instead of simply concatenating, which ensures a model learns from all corpora evenly. We use this method as a baseline. Additionally, we explore two multi-domain learning methods: *labeled learning* and *multi-task labeled learning*. Labeled learning comes from a control technique in response generation (Li et al., 2016; Johnson et al., 2017; Yang et al., 2017). Previous works focus on controlling persona and style, while our method controls the corpus' information with the corpus embedding. Multi-task labeled learning is inspired by work of domain adaption (Luan et al., 2017; Niu and Bansal, 2018; Chu and Wang, 2018), where multiple losses from both the corpus classifier and response generator are minimized. To the best of our knowledge, this paper is the first that uses corpus embeddings on the open-domain response generation task for multiple corpora.

Furthermore, we propose a novel *weighted learning* with Domain-specific Frequency (DF). DF is a word-level importance weight (Leopold and Kindermann, 2002) that assigns different weights (importance) to the same words from different corpora. In the training process, we weight the loss of a model with DF, so that the model focuses on the most important words for a specific corpus.

For automatic evaluation metrics, we eliminate the stop words and use ROUGE-1 (precision, recall, F1) (Lin, 2004) to measure the **relevance** of the generated responses. In addition, we adopt DF to see how relevant the generated response of a model is to a specific corpus. We will explain DF as an evaluation metric in Section 3.2.4. Results show that for overall performance, the best method (weighted learning) improves $27.4\%$ on precision, $45.5\%$ on recall, and $34.1\%$ on F1. Further, it has at least $20.0\%$ higher DF, stating that it uses more important words from the "correct" corpus. We also conduct an extensive human evaluation on 2400 generated responses. The human evaluation shows a highly significant ($p < 0.001$) improvement on all of our proposed methods, especially the weighted learning method.

We summarize our work as follows:

- We explore the problem of training and evaluating on multiple corpora from different

domains for open-domain response generation. The task is to make the conversational models generate relevant responses for **each** corpus.

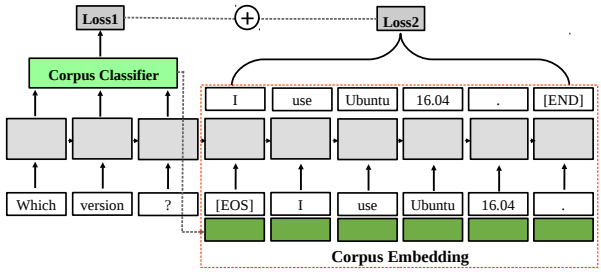
- We examine several multi-domain corpora learning methods for their ability to solve the proposed task.
- We propose Domain-specific Frequency (DF) as in weighted learning and as an evaluation metric. DF distinguishes important words for each corpus and helps a model to focus on these important words in the training process.

## 3.2 Methodology

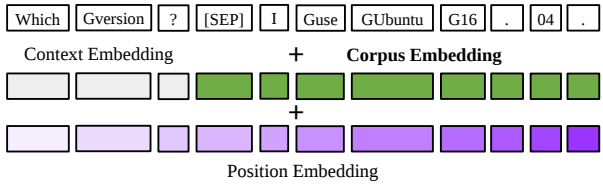
We use two base models: an LSTM Seq2Seq model with attention (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014; Bahdanau et al., 2015) and a pre-trained GPT-2 model (Radford et al., 2019). The LSTM Seq2Seq model with attention is a common model for conversational systems (Li et al., 2016; See et al., 2019), and the GPT2 model is a state-of-the-art model for the response generation task (Zhang et al., 2020; Zhao et al., 2020b). The structure of the two base models are described in Section 2.2. We simplify an LSTM with attention unit as $LSTM^*$, and the structure of the transformer block in GPT2 as GPT_block.

### 3.2.1 Interleaved Learning

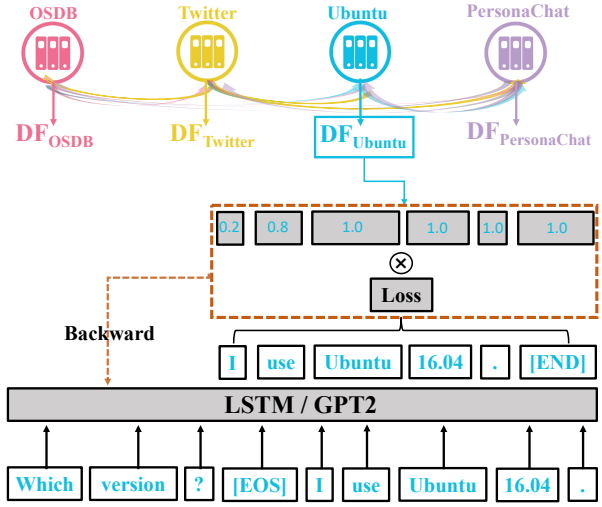
Interleaving is a concept in cognitive psychology proven to be efficient for learning (Kornell and Bjork, 2008): intermingling learning material of different topics helps students to gain better learning results than learning the material topic by topic. Previous work from machine learning also shows that training order greatly influences the performance (Bengio et al., 2009). When the training is conducted on a simple concatenation of multiple corpora, the model tends to concentrate on the last corpus (Shi et al., 2015). To address this issue, we propose interleaved learning as an alternative: each time we collect one context-response pair from each of the corpora, and we randomly shuffle them. For example, if there are 3 corpora $(a_1, a_2, ...), (b_1, b_2, ...), (c_1, c_2, ...)$ where $a_i, b_i$ and $c_i$ are context-response pairs, the resulting mixed corpus might be $(b_1, a_1, c_1, c_2, b_2, a_2, ...)$. Interleaved learning guarantees




(a) Structure of multi-task labeled learning on LSTM model




(b) Corpus embeddings with sub-word embeddings on GPT-2




(c) Structure of weighted learning

Figure 3.1: Adapted models with labeled learning, multi-task labeled learning and weighted learning

that the combined corpus is evenly distributed, which helps the model learn from multiple corpora evenly.

### 3.2.2 Labeled Learning

We propose our labeled learning as follows: each corpus is assigned a randomly initialized unique embedding, and the conversational model learns these embeddings together with conversations during the training period. We denote these embeddings as "corpus embedding", or $E_c$. A model captures each corpus's characteristics through the corpus embedding and uses it to control the generated responses. To know which corpus embedding to use, each context-utterance is labeled with which corpus it comes from, and these labels are provided to the model both in the training and generation period. We propose an approach for each of our base models for encoding corpus embeddings.

For the LSTM model, following Li et al. (2016), we input the corpus embedding $E_c$ into the first layer of the decoder LSTM at every step, together with the response words. Calculation of a hidden vector $h_t$ in the decoder LSTM is then adapted to:

$$h_t = LSTM^*(h_{t-1}, E(y_t), E_c). \tag{3.1}$$

The structure is illustrated in the dashed red rectangle of Figure 3.1a.

For the GPT-2 model, our method is based on Wolf et al. (2019). Instead of two kinds of dialogue-state embeddings (context embedding $E_0$ and response embedding $E_1$), we replace the response embedding with corpus embeddings $E_c$. As a result, the model is aware of which corpus the response belongs. Calculation of a hidden vector to be input to the transformer block is adapted to:

$$h_{0[t]} = E(X, Y_{[1:t]}) + (E_0, E_c) + W_p. \tag{3.2}$$

The structure is illustrated in Figure 3.1b.

### 3.2.3 Multi-Task Labeled Learning

Labeled learning needs corpus labels for both training and generation processes. To avoid providing labels in the generation process, we combine multi-task learning with labeled learning on multiple corpora. Here, the conversational model has to predict by itself which corpus a context-utterance belongs to, which is expected to result in worse performance, but less information is required. In the encoder, we have a classifier layer that uses the sum of hidden vectors from the encoder ($\sum H$) to predict the corpus of a context-utterance. The loss of the classifier is calculated as:

$$\mathcal{L}_c = -\log\left(softmax\left(\left(\sum H\right) \cdot W_{[c]}\right)\right), \tag{3.3}$$

where $W_{[c]} \in \mathbb{R}^{dim}$ is the part from the classifier layer for target corpus $c$. $\mathcal{L}_c$ is summed up with the loss from the response generator. The predicted corpus embedding is input into the decoder like labeled learning (see Section 3.2.2). The simplified structure is illustrated in Figure 3.1a.

### 3.2.4 Document-specific Frequency (DF)

We propose Domain-specific Frequency (DF) to measure how important a word is with respect to a different corpus under a collection of corpora. DF is used for weighted learning and evaluation. It is calculated as follows:

$$f(w)_d = \text{freq}(w)_d - \min_v\{\text{freq}(v)_d\} \tag{3.4}$$

$$\text{df}(w)_d = \begin{cases} 0 & f(w)_d = 0 \\ \frac{f(w)_d}{\sum_{d \in D} f(w)_d} & f(w)_d \neq 0 \end{cases} \tag{3.5}$$

$$\text{DF}(w)_d = \frac{\text{df}(w)_d}{\max_v\{\text{df}(v)_d\}}, \tag{3.6}$$

where $\text{freq}(w)_d$ is the relative frequency of a word $w$ in a corpus $d$, and $D$ represents the set of all corpora. It is easy to see from Equation 3.5 that $\text{DF}(w)_d$ represents the importance of word $w$ for corpus $d$ compared to other corpora. For a word $w$ that frequently appears in

corpus $d$ but seldom in other corpora (e.g., "upgrade" from Ubuntu corpus), $\sum_{d \in D} f(w)_d$ is close to $f(w)_d$, making DF$(w)_d$ approach 1. A word that frequently appears in all corpora (e.g., "I", "you") is punished, resulting in a lower DF$(w)_d$. A word that seldom appears in corpus $d$ but frequently appears in other corpora (e.g., "music" seldom appears in Ubuntu corpus, but is common in other corpora) has the lowest DF$(w)_d$. Words that appear minimal times (e.g., once) in a corpus are ignored with Equation 3.4. Words that appear few times (e.g., twice or three times) are not dealt with, yet they are not of great influence in our experiments. We apply a normalization in the final step (Equation 3.6) to make DF$(w)_d$ of each corpus $d$ range from 0 to 1.

We show DF$(w)_{\text{Ubuntu}}$ and DF$(w)_{\text{PersonaChat}}$ of some words in Table 3.3. We also show the results of TF-IDF (log normalization variant), a commonly used word importance weight, as a comparison. As expected, for the corpus Ubuntu and PersonaChat, most unique words $w$ have very different DF$(w)_{\text{Ubuntu}}$ and DF$(w)_{\text{PersonaChat}}$. Unique words of each corpus get the highest values for the corresponding corpus, like "upgrade" for the Ubuntu corpus and "music" for the PersonaChat corpus; these words receive the lowest values for *incorrect* corpora, like "upgrade" for PersonaChat and "music" for Ubuntu. The stress on unique words makes DF more suitable for our task.

| Word | TF-IDF(%) | | DF(%) | | $\alpha$DF$_{(\alpha=100)}$ | |
|---|---|---|---|---|---|---|
| | Ubuntu | PersonaChat | Ubuntu | PersonaChat | Ubuntu | PersonaChat |
| i | 100.0 | 62.6 | 20.8 | 42.1 | 2.6 | 7.3 |
| to | 64.6 | 32.8 | 26.9 | 24.9 | 3.8 | 3.1 |
| it | 83.2 | 21.7 | 38.5 | 14.5 | 5.1 | 2.1 |
| laptop | 5.4 | 0.2 | 89.8 | 4.5 | 76.0 | 1.0 |
| upgrade | 6.8 | 0.1 | 95.6 | 0.4 | 91.2 | 1.0 |
| file | 15.7 | 0.1 | 96.0 | 0.3 | 86.4 | 0 |
| windows | 12.2 | 0.1 | 97.1 | 0.1 | 86.3 | 1.0 |
| ubuntu | 27.5 | 0 | 99.9 | 0 | 99.5 | 0 |
| teacher | 0.1 | 2.2 | 0.7 | 77.8 | 1.0 | 53.5 |
| music | 1.5 | 7.6 | 4.8 | 82.9 | 1.2 | 49.1 |
| travel | 0.1 | 3.1 | 0.3 | 88.9 | 1.0 | 57.1 |
| hobby | 0.1 | 1.6 | 0.6 | 94.3 | 1.1 | 81.7 |
| hiking | 0 | 1.5 | 0 | 97.6 | 0 | 91.8 |

Table 3.3: Normalized TF-IDF (%), DF (%) and $\alpha$DF of some words for Ubuntu and PersonaChat (more examples on other corpora can be found in Section A.1)

**Weighted Learning with DF** Weighted learning weights the loss of the predication $y'$ for each target word $w$ using $\text{DF}(w)_d$. In the training period, each context is labeled with the corpus $d$ it belongs to, so that the model can use the $\text{DF}(w)_d$ of the corresponding corpus. Here DF is calculated only on the training sets. In the generation step, corpus labels are not provided, so DF is not used. The loss is weighted as follows :

$$\mathcal{L}_{\text{weighted}} = \text{DF}(w)_d \cdot \left(-\log\left(softmax(y'_w)\right)\right), \tag{3.7}$$

where $y'_w$ represents the model's predicted score for the target word $w$. With the weighted loss, the model concentrates on words that are important to the corpus of the current context, and focuses less on frequent words or words that are not important to the current corpus. The structure is illustrated in Figure 3.1c.

**Evaluation with DF** For the generated responses to be relevant to a specific corpus, they have to be similar to that corpus, which includes using important words of that corpus (e.g., responses generated for the Ubuntu corpus should have more technical words than other corpora). Thus, we propose DF as an evaluation metric that shows to what extent the generated responses use important words of the corresponding corpus. We want to decrease the influence of common words like "i", "to", etc., and thus address the important words. So we adopt exponential DF with $\alpha$ as the base ($\alpha$DF):

$$\alpha\text{DF}(w)_d = \begin{cases} 0 & \text{DF}(w)_d = 0 \\ \alpha^{\text{DF}(w)_d} & \text{DF}(w)_d \neq 0, \end{cases} \tag{3.8}$$

where $\alpha$ is a constant. $\alpha\text{DF}(w)_d$ rescales $\text{DF}(w)_d$ by exponent with $\alpha$ as a base. In our experiments, we set $\alpha$ to be $100$, which transforms the range of the metric from $(0, 1)$ to $(0, 100)$. This makes the difference between high and low $\alpha$DF more significant than DF and gives a $100$-scale score. For each corpus $d \in D$, we average $\alpha\text{DF}(w)_d$ on word $w$ from the generated responses of each test set, which gives us $\alpha\text{DF}_d$ scores ($d \in D$) for each test set. Ideally, the generated responses of a specific corpus $d$ should have a higher $\alpha\text{DF}_d$ score and lower $\alpha\text{DF}_{\overline{d}}$ score ($\overline{d} \in \{d' \in D \mid d' \neq d\}$). For example, generated responses of the Ubuntu test set should have a higher $\alpha\text{DF}_{\text{Ubuntu}}$ score, while a lower $\alpha\text{DF}_{\overline{\text{Ubuntu}}}$ score

($\overline{\text{Ubuntu}} \in \{d' \in D \mid d' \neq \text{Ubuntu}\}$). $\alpha\text{DF}_d$ scores for responses from the original test sets are the standard scores.

We show $\alpha\text{DF}(w)_{\text{Ubuntu}}$ and $\alpha\text{DF}(w)_{\text{PersonaChat}}$ (calculated purely on test set) in Table 3.3. As expected, $\alpha$DF has a more significant difference between important words and common words.

**Is DF a Legal Evaluation Metric?** Although DF is used for both weighted learning and evaluation, we see DF as a suitable evaluation metric for our task and not biased in favor of weighted learning due to: 1) A word receives multiple DF values in the training process given the corpus that a context belongs to; 2) in the generation process, DF is never used. 3) In the evaluation process, DF can be calculated purely on the test sets. Note that since a word receives multiple DF values in the training step, it is equivalently likely for the model trained with weighted learning to be influenced by DF weights of **incorrect** corpus. Above all, in the evaluation step, if the trained model is influenced more by DF weights from the correct corpus, it already means that the model is good at distinguishing which corpus a given context is from, thus is suitable for our task.

## 3.3 Experiment Setup

### 3.3.1 Datasets

**Data Collection** We collected 4 commonly used English corpora of different domains from the ParlAI platform (Miller et al., 2017): OpenSubtitles corpus (OSDB)[2] (Lison et al., 2018), Twitter corpus[3] (Miller et al., 2017), Ubuntu chatlogs corpus (Lowe et al., 2015)[4] , and PersonaChat corpus (Zhang et al., 2018a) from the NeurIPS 2018 ConvAI2 Challenge (Dinan et al., 2019). Each corpus contains $250$K context-response pairs, as much as the size of the original PersonaChat used in ConvAI2 competition. This gives us $1$M context-response pairs in total. The corpus for training is a combination of these 4 corpora. For

[2] http://www.opensubtitles.org/
[3] https://github.com/Marsan-Ma/chat_corpus/
[4] https://github.com/rkadlec/ubuntu-ranking-dataset-creator

comparison, we have a single corpus–PersonaChat–trained on both base models. For testing, each of the 4 corpora has a test set of 30K context-response pairs, which is the same size of the test set of PersonaChat.

The OpenSubtitles corpus (OSDB) is a noisy dataset of film subtitles. We removed films that belonged to genres that usually had few conversations, such as musical and documentary films. We regarded two neighboring sentences as a context-response pair following Vinyals and Le (2015). The Twitter corpus contains one-turn dialogues extracted from Twitter. The original author has already cleaned it, so we only removed special symbols such as hashtags, Emojis, and @. The Ubuntu corpus contains dialogues about solving technical problems of Ubuntu. The PersonaChat corpus contains dialogues between two workers acting as specific personas; we focused on the dialogue part and ignored the persona part. This corpus allows us to compare our base models with state-of-the-art performance. These 4 corpora have very different characteristics, confirmed by the imbalanced performance of GPT-2 fine-tuned on a single corpus (see Table 3.2).

### 3.3.2 Training and Decoding

We used Pytorch (Paszke et al., 2017) to implement the LSTM Seq2Seq model with attention and the pre-trained GPT-2 models. For GPT-2, we adapted our model from the implementation of the HuggingFace team[5]. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, learning rate of 1.0, dropout rate of 0.2, and gradient clip threshold of 5. The vocabulary size is 50000. GPT-2 has 12 layers, 12 heads, and the dimension is 768, the same as the pre-trained model. The training procedure was with Adam and we adopted a similar setup as Wolf et al. (2019): the batch size was 32, learning rate was $6 \times 10^{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, L2 weight decay set to $0.01$, learning rate linearly decreased to zero at the end. We followed these hyper-parameters to ensure state-of-the-art performance for the base models. We use the same hyper-parameters for both base models and models with our proposed methods, so the proposed methods work slightly (but not much) worse than it should be. This is to avoid the extra improvement caused by hyper-parameters. We pre-trained the LSTM model on

[5] `https://huggingface.co/`.

| Model | Corpus / Method | Test set: OSDB Precision | OSDB Recall | OSDB F1 | Twitter Precision | Twitter Recall | Twitter F1 | Ubuntu Precision | Ubuntu Recall | Ubuntu F1 | PersonaChat Precision | PersonaChat Recall | PersonaChat F1 | Overall Precision | Overall Recall | Overall F1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LSTM | PersonaChat (single) | 11.8 | 8.9 | 8.1 | 12.4 | 8.6 | 8.9 | 12.1 | 8.1 | 7.7 | 56.7 | 43.4 | 45.8 | 23.2 | 17.2 | 17.6 |
| | Concatenated | 11.0 | 7.7 | 7.2 | 15.7 | 10.9 | 11.4 | 36.5 | 17.8 | 20.1 | 57.7 | **44.0** | 46.4 | 30.2 | 20.1 | 21.3 |
| | Interleaved | 24.1 | 10.1 | 11.7 | 24.3 | 12.5 | 14.9 | 58.4 | 24.9 | 29.6 | 56.1 | 41.5 | 44.3 | 40.7 | 22.3 | 25.1 |
| | Labeled | 23.9 | 10.1 | 11.3 | 24.5 | **13.2** | 15.5 | 61.6 | 26.5 | 31.6 | 56.4 | 43.0 | 45.4 | 41.6 | 23.2 | 26.0 |
| | Multi-task Labeled | 23.2 | 9.6 | 11.1 | 23.2 | 12.3 | 14.5 | 56.4 | 23.8 | 28.3 | 53.2 | 40.6 | 42.7 | 39.0 | 21.6 | 24.2 |
| | Weighted | **26.6** | **11.9** | **13.4** | **29.7** | 12.2 | **15.6** | **78.4** | **35.2** | **41.2** | **62.4** | 42.5 | **47.1** | **49.3** | **25.5** | **29.3** |
| GPT-2 | PersonaChat (single) | 15.0 | 12.4 | 10.8 | 19.6 | 13.2 | 13.9 | 24.8 | 16.2 | 15.5 | 70.0 | 57.1 | 58.8 | 32.4 | 24.7 | 24.7 |
| | Concatenated | 17.4 | 14.1 | 12.6 | 24.5 | 16.4 | 17.2 | 35.0 | 22.5 | 22.4 | 66.8 | 55.4 | 56.3 | 35.9 | 27.1 | 27.1 |
| | Interleaved | 40.0 | 20.5 | 22.3 | 31.0 | 17.9 | 20.1 | 81.7 | 38.1 | 44.3 | 68.7 | 56.2 | 57.6 | 55.3 | 33.2 | 36.1 |
| | Labeled | 38.6 | 19.9 | 21.6 | 31.4 | **19.4** | 21.1 | 84.2 | 38.4 | 45.0 | **70.7** | **57.2** | **59.0** | 56.2 | 33.7 | 36.7 |
| | Multi-task Labeled | 38.4 | 19.8 | 21.4 | 31.2 | 18.6 | 20.6 | 80.9 | 37.8 | 43.8 | 68.0 | 56.0 | 57.3 | 54.6 | 33.0 | 35.8 |
| | Weighted | **41.9** | **21.2** | **23.4** | **39.9** | 18.4 | **22.3** | **86.8** | **43.3** | **48.6** | 69.0 | 53.2 | 55.8 | **59.4** | **34.0** | **37.5** |

Table 3.4: Precision, recall and F1 of ROUGE-1 (‰) for baselines and proposed methods fine-tuned on 4 corpora (stop words eliminated)

3 large-scale corpora (OSDB, Twitter and Ubuntu) with interleaved learning until converging. GPT-2 is already pre-trained, so we directly used it for fine-tuning (details about

| Model | Corpus / Method | Test Set: OSDB | | | | Test Set: Twitter | | | | Test Set: Ubuntu | | | | Test Set: PersonaChat | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | PersonaChat | | Twitter | | PersonaChat | | Ubuntu | | PersonaChat | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test |
| | Test Set (Standard Score) | **7.0** | **9.7** | 3.6 | 3.7 | **9.1** | **11.0** | 3.6 | 3.8 | **19.4** | **23.2** | 2.7 | 2.8 | **9.5** | **12.0** |
| LSTM | PersonaChat (single) | 2.9 | 3.4 | 9.2 | 9.9 | 2.8 | 3.4 | 8.6 | 9.2 | 2.7 | 3.1 | 8.6 | 9.1 | 11.9 | 12.6 |
| | Concatenated | 2.9 | 3.3 | 7.6 | 8.6 | 3.6 | 4.3 | 8.0 | 8.7 | 7.6 | 7.7 | 5.6 | 6.0 | 12.5 | 13.6 |
| | Interleaved | 3.9 | 4.1 | 5.0 | 5.3 | 4.7 | 4.9 | 4.1 | 4.5 | 11.8 | 11.3 | 3.7 | 4.0 | 11.5 | 12.5 |
| | Labeled | 3.9 | 4.2 | 5.0 | 5.3 | 5.0 | 5.3 | 3.9 | 4.3 | 12.5 | 11.8 | 3.4 | 3.8 | 12.1 | 13.1 |
| | Multi-task Labeled | 3.8 | 4.0 | 5.0 | 5.4 | 4.5 | 4.7 | 4.1 | 4.5 | 11.2 | 10.7 | 3.8 | 4.1 | 11.4 | 12.6 |
| | Weighted | **5.6** | **6.3** | 4.1 | 4.5 | **9.9** | **10.1** | 3.8 | 4.3 | **27.7** | **25.4** | 2.7 | 3.0 | **17.7** | **18.3** |
| GPT-2 | PersonaChat (single) | 2.8 | 3.2 | 10.5 | 11.1 | 2.9 | 3.3 | 9.5 | 9.8 | 4.1 | 4.6 | 8.3 | 8.4 | 12.9 | 13.7 |
| | Concatenated | 3.1 | 3.6 | 8.8 | 9.4 | 3.3 | 3.9 | 8.2 | 8.7 | 6.5 | 7.1 | 7.0 | 7.4 | 12.1 | 13.0 |
| | Interleaved | 4.9 | 5.8 | 4.8 | 5.0 | 4.6 | 5.1 | 4.4 | 4.7 | 15.7 | 16.0 | 3.1 | 3.4 | 12.1 | 12.9 |
| | Labeled | 4.9 | 5.8 | 4.8 | 5.0 | 4.7 | 5.2 | 4.1 | 4.3 | 16.7 | 17.0 | 2.9 | 3.2 | 12.4 | 13.1 |
| | Multi-task Labeled | 4.8 | 5.7 | 4.8 | 5.1 | 4.6 | 5.1 | 4.4 | 4.6 | 15.5 | 15.8 | 3.1 | 3.4 | 12.1 | 12.9 |
| | Weighted | **6.0** | **7.5** | 4.1 | 4.4 | **8.1** | **8.8** | 3.7 | 4.1 | **25.7** | **24.4** | 2.4 | 2.6 | **16.0** | **17.1** |

Table 3.5: $\alpha\text{DF}_d$ scores for generated responses from multiple corpora. The columns "train" indicate train-set-$\alpha\text{DF}_d$. The columns "test" indicate test-set-$\alpha\text{DF}_d$.

pre-training convergence can be found in Section A.2). For decoding, we adopted greedy decoding for all the models to ensure an equal condition.

### 3.3.3 Evaluation

For automatic metrics, to measure the **relevance** of the generated responses, we eliminated punctuation and stop words, and adopted Rouge-1[6] (precision, recall, F1) as multi-grams become meaningless without stop words. However, Rouge-1 compares the generated responses with the golden ones, while there is never a standard response for any context, so in addition to Rouge, we use $\alpha$DF score that shows to what extent the generated responses use important words of the corresponding corpus, as stated in Section 3.2.4. Due to the limitation of automatic evaluation methods (Liu et al., 2016), we also conduct an extensive human evaluation on the relevance of generated responses to contexts (see Section 3.4.1 for details).

| Model \ Corpus | OSDB | Twitter | Ubuntu | PersonaChat | Overall |
|---|---|---|---|---|---|
| PersonaChat (single) | 1.53 | 1.43 | 1.21 | 2.09 | 1.56 |
| Concatenated | 1.67 | 1.71 | 1.60 | 2.16 | 1.78 |
| Interleaved | 2.04 | 1.89 | 2.18 | 2.24 | 2.09 |
| Labeled | 2.10 | 2.10 | 2.32 | 2.24 | 2.19 |
| Multi-task Labeled | 2.05 | 1.98 | 2.11 | 2.24 | 2.10 |
| Weighted | **2.40** | **2.45** | **2.61** | **2.47** | **2.48** |

Table 3.6: Average scores of human evaluation for GPT-2 based models on each corpus

## 3.4 Results

Our base models achieve perplexity scores of $28.9$ (LSTM model) and $19.6$ (GPT-2) on the test set of the PersonaChat dataset from the ConvAI2 competition when fine-tuned with the single PersonaChat corpus (more details can be found in Section A.3). These results would likely advance the models to the second round in the competition.

Table 3.4 shows that models trained with our proposed methods gain better performance on Rouge than baselines. Baselines concentrate on the last trained corpus (PersonaChat), while with the proposed methods, performance is more balanced on multiple corpora. Weighted learning has the best overall performance on all metrics, and it performs

[6]We used implementation from `https://github.com/google-research/google-research/tree/master/rouge`.

| Model \ Model | PersonaChat | Concatenated | Interleaved | Labeled | Multi-Task Labeled | Weighted |
|---|---|---|---|---|---|---|
| PersonaChat | 1.00 | \ | \ | \ | \ | \ |
| Concatenated | $2.54 \times 10^{-7**}$ | 1.00 | \ | \ | \ | \ |
| Interleaved | $4.71 \times 10^{-34**}$ | $2.09 \times 10^{-12**}$ | 1.00 | \ | \ | \ |
| Labeled | $1.08 \times 10^{-46**}$ | $9.41 \times 10^{-21**}$ | $1.18 \times 10^{-2*}$ | 1.00 | \ | \ |
| Multi-task Labeled | $6.65 \times 10^{-35**}$ | $6.96 \times 10^{-13**}$ | $8.86 \times 10^{-1}$ | $1.17 \times 10$ | 1.00 | \ |
| Weighted | $1.65 \times 10^{-103**}$ | $2.86 \times 10^{-63**}$ | $6.54 \times 10^{-26**}$ | $1.59 \times 10^{-15**}$ | $2.01 \times 10^{-25**}$ | 1.00 |

Table 3.7: P-value for t-test on overall human evaluation scores of GPT-2 based models, $^{**}\ p < 0.001$

especially well on the Ubuntu corpus, indicating that it might be good at distinguishing the unique technical words from the Ubuntu corpus. Labeled learning is the second best

with stable improvement from interleaved learning, indicating that the corpus embeddings function as expected. Multi-task labeled learning has slightly worse performance than interleaved learning, indicating that predicting the corpus of a contexts is not easy, and wrong predictions result in worse performance.

Table 3.5 shows $\alpha\text{DF}_d$ scores for generated responses of each corpus. Full results can be found in Section A.5. We use both $\alpha\text{DF}_d$ calculated purely on the train set (train-set-$\alpha$DF) and $\alpha\text{DF}_d$ calculated purely on the test set (test-set-$\alpha$DF). The black scores are scores for the corresponding corpus (we expect high scores for these parts), while the grey scores are scores for non-related corpus–PersonaChat (we expect low scores for these parts). Note that scores for different corpora are in different scales. From the table, we can see that train-set-DF scores and test-set-DF scores are similar, and weighted learning always has the highest score, indicating that weighted learning distinguishes well which corpus a context comes from. Labeled learning is the second best, indicating that the learned corpus embeddings help the model to use more important words of the corresponding corpus. Compared to the concatenated corpus, the improvement is at least $20\%$, while the decrease in PersonaChat is just $9\%$ at most.

### 3.4.1 Human Evaluation

We conducted a human evaluation on all GPT-2 models: base models and models adapted with our proposed methods. We randomly picked $2400$ responses: $400$ different contexts evenly from 4 corpora with $6$ responses generated by each of our models. 3 judges[7] are asked to pick the most and the least relevant response(s) for the given context. The most relevant response(s) are given score $3$, the least relevant response(s) are given score $1$, and the other(s) are given score $2$. Table 3.6 shows the overall scores of all GPT-2 based models. Table 3.7 shows the p-value for the t-test conducted between every two models. The overall scores of our proposed methods are all highly significantly ($p < 0.001$) higher than the concatenated models, especially the weighted learning method.

[7]Similar to previous work like Zhang et al. (2020), we have 3 judges. We have one random worker from `https://www.mturk.com/worker`, one bachelor student, and one graduate student. An example of the mTurk interface can be found in Section A.6.

### 3.4.2 Response Examples

The generated responses from better methods are more relevant to the corresponding corpus, while worse methods cannot distinguish contexts from different corpora (e.g., they may answer any questions in a "PersonaChat" way). To show an intuition of the difference among our proposed methods, we present some response examples generated by GPT-2 in Section A.7.

## 3.5 Conclusions

We have experimented with 4 methods–interleaved learning (baseline), labeled learning, multi-task labeled learning, and weighted learning–to help common open-domain conversational systems generate relevant responses for multiple corpora of different domains. We adopted Rouge (precision, recall, F1) for auto evaluation. In addition, we used DF to evaluate how well a model uses relevant words for a corresponding corpus. We also did an extensive human evaluation. Our results show significant improvement in performance for our proposed methods, especially weighted learning.

## Acknowledgements

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assistance on computing resources from the IDUN cluster of NTNU (Själander et al., 2019). We would like to thank Aria Rahmati, Zhirong Yang (Norwegian Research Council, 287284) and Özlem Özgöbek for their helpful comments.

# Chapter 4

# Context Attention Distribution

Open-domain generation-based conversational agents have achieved great improvements in recent years. Despite the rapid progress, most deployed systems continue to treat dialogue contexts as single-turns, while systems dealing with multi-turn contexts are less studied. There is a lack of a reliable metric for evaluating multi-turn modelling, as well as an effective solution for improving it. In this paper, we focus on an essential component of multi-turn generation-based conversational agents: **context attention distribution**, i.e. how systems distribute their attention on the dialogue's context. For evaluation of this component, we introduce a novel attention-mechanism-based metric: **DAS ratio**. To improve performance on this component, we propose an optimization strategy that employs self-contained distractions. Our experiments on the Ubuntu chatlogs dataset show that models with comparable perplexity can be distinguished by their ability on context attention distribution. Our proposed optimization strategy improves both non-hierarchical and hierarchical models on the proposed metric by about $10\%$ from baselines.

## 4.1 Introduction and Related Works

In recent years, generation-based conversational agents have shown a lot of progress. However, multi-turn generation-based conversational agents are still facing challenges. Most recent works ignore multi-turn modelling by considering a multi-turn context as a 1-turn context Zhang et al. (2020); Zhao et al. (2020a). Some works try to deal with multi-turn

| User | utterances |
|---|---|
| Taru | Haha sucker. |
| Kuja | ? |
| Taru | Anyways, you made the changes right? |
| Kuja | Yes. |
| Taru | Then from the terminal type: sudo apt-get update |
| Kuja | I did. |

Table 4.1: An example of important utterances and unimportant utterances under the same context in the Ubuntu chatlog dataset Lowe et al. (2015). Unimportant utterances are marked in red.

modelling using modified attention mechanisms, hierarchical structures, utterance tokens, etc. Serban et al. (2016, 2017); Li et al. (2021). The main difference between multi-turn conversational agents and regular (1-turn) conversational agents is that instead of dealing with an utterance in a context on *word-level*, multi-turn models deal with a dialogue on *utterance-level*, so that models can understand an utterance as a whole and focus on important *utterances* rather than important *words*. This paper focuses on an essential utterance-level component for multi-turn modelling: **context attention distribution**, i.e. how much attention is distributed respectively to important and unimportant utterances in a context. An example of important/unimportant utterances existing in the same context is given by Table 4.1. In this example, the first two utterances ("Haha sucker." and "?") are irrelevant to the main topic of the context, thus are unimportant utterances. Human dialogues naturally have many such unimportant utterances, which can distract a multi-turn model from generating responses relevant to the main topic of a context. Therefore, it is crucial that a multi-turn model pays less attention to these unimportant utterances, and more attention to the important utterances in a context, which we define as a good ability on context attention distribution.

We first propose an evaluation metric to measure a conversational agent's performance on context attention distribution. Recent works lack a measurement for multi-turn modelling performance, especially for context attention distribution. They rely on general evaluation metrics such as BLEU Papineni et al. (2002), which measures the quality of generated responses. These metrics cannot directly describe a model's ability on dealing

with multi-turn contexts. The quality of the generated responses is influenced by many aspects, including the performance of multi-turn modelling; better performance in dealing with multi-turn context may result in better general performance, while not vice versa. Thus, a general metric like BLEU is insufficient for analyzing all relevant aspects of the conversational agents.

Since most multi-turn conversational agents have an attention mechanism and rely on it to distribute attention to different utterances in a context, we propose the **distracting test** as the evaluation method to examine if a model pays more attention to the important utterances. The test adds distracting utterances to the context of each dialogue and compares the attention scores of distracting utterances (i.e., unimportant utterances) and original utterances (i.e., important utterances). The ratio of average attention score of distracting utterances and original utterances is defined as the distracting attention score ratio (**DAS ratio**). It is the evaluation metric for a model's performance on context attention distribution. A model with good ability on context attention distribution should have higher scores on original utterances and lower scores on distracting utterances, thus a lower DAS ratio.

Further, we propose a self-contained optimization strategy for improving a conversational agent's performance on context attention distribution. For each dialogue, we randomly pick some utterances from the training corpus outside the current dialogue as self-contained distractions, and insert them into the current dialogue with different levels of possibilities. The attention paid to these distractions is minimized through multi-task learning. With this optimization strategy, a model learns to distribute less attention to unimportant utterances and more attention to important utterances.

In this paper, we examine the following research questions: 1) How do existing multi-turn modelling structures perform on context attention distribution? 2) Can the proposed optimization strategy improve performance on context attention distribution? 3) Which probability level of inserting distractions is the best for the proposed optimization strategy? Our contributions are as follows:

- We deal with a less studied problem: evaluating and improving context attention distribution for multi-turn conversational agents.
- We propose a novel evaluation metric for context attention distribution: DAS ratio.

It is tailored for multi-turn conversational agents by measuring a model's ability on context attention distribution.

- We propose an optimization strategy that minimizes the attention paid to self-contained distractions. The strategy can easily be added and adapted to existing models.

Extensive experiments on 23 model variants and 9 distracting test sets show an overall improvement in the performance of context attention distribution for the proposed strategy.

## 4.2 Methodology

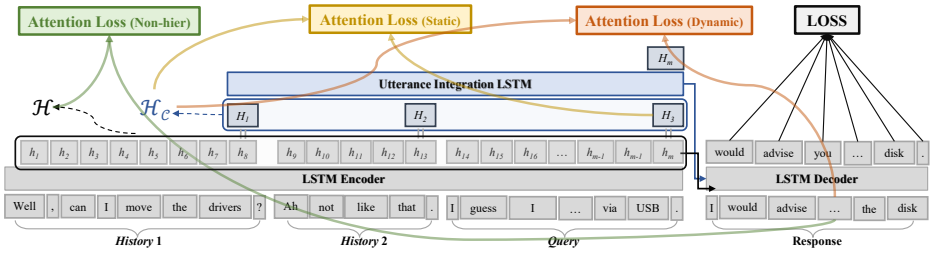



Figure 4.1: Structure of non-hierarchical, static and dynamic attention loss.

Our proposed evaluation metric and optimization strategy can work on attention mechanisms including Transformers. In this paper, we choose an LSTM Seq2Seq model with attention mechanism Hochreiter and Schmidhuber (1997); Sutskever et al. (2014); Bahdanau et al. (2015) as the base model, since most hierarchical structured multi-turn conversational agents are based on LSTM Serban et al. (2016, 2017); Tian et al. (2017); Zhang et al. (2018b) while few are based on Transformers. The architecture of an LSTM model with the attention mechanism and the description of the context vector $c_t$ in the attention mechanism can be found in Section 2.2.

### 4.2.1 Attention Mechanism & Utterance Integration (UI)

We examine both non-hierarchical and hierarchical structures. For hierarchical structures, following Zhang et al. (2018b), we develop two attention mechanisms: static and dynamic.

Following Tian et al. (2017), we develop models that are both with and without utterance integration LSTM units.

For the non-hierarchical structured model, there are no hidden vectors for utterances. All hidden vectors of tokens in the encoder are concatenated and used in the attention mechanism. Denoting the concatenated vector $\mathcal{H} = [h_1, h_2, ..., h_m]$, we calculate the context vector $c_t$ for each decoding step $t$ as:

$$c_t = \mathcal{H} \cdot (\text{softmax}(\mathcal{H}^\top \cdot h_t)) \,. \tag{4.1}$$

For the hierarchical models, we use the hidden vector of each utterance's last token as the hidden vector of the utterance, and we discard the hidden vectors for the other tokens. Thus, compared to the non-hierarchical structured model, we have much fewer hidden vectors from the encoder.

The context vector of static attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of the last utterance in the context. Denoting the hidden vector of $k$th utterance as $H_k$, and the hidden vector of the last utterance in the context as $H_q$, we have the context's concatenated vector $\mathcal{H}_\mathcal{C} = [H_1, H_2, ..., H_q]$. We calculate the context vector $c_t$ for static attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot H_q)), \tag{4.2}$$

where it is easy to see that the static context vector remains unchanged by the decoder.

The context vector of dynamic attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of each token in the decoding step. We calculate the context vector $c_t$ for dynamic attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot h_t)) \,. \tag{4.3}$$

Compared to the static attention mechanism, the context vector $c_t$ varies at each decoding step.

Finally, with the utterance integration LSTM unit, we calculate $H_m$ from $H_1$, $H_2$, ...

$H_q$:

$$H_m = LSTM(H_1, H_2, ..., H_q). \tag{4.4}$$

For models with utterance integration (UI), $H_m$ is input to the first step of the decoder, while for models without UI, regular $h_m$ is input instead.

### 4.2.2 Distracting Test & Attention Score (AS)

We examine if a multi-turn conversational agent distributes more attention to important utterances through the **distracting test** and attention scores.

In the distracting test, for each dialogue, we insert several distracting utterances before the end of the context. The distracting utterances can be randomly picked utterances from the training corpus (**random**), be formed by frequent words from the training corpus (**frequent**), or be formed by rare words from the training corpus (**rare**). We compare the attention scores of the distracting utterances with the attention scores of the original utterances. A well-performing model should distribute less attention to the distracting utterances but more attention to the original utterances. For an utterance $H_k$, the corresponding attention score $\text{AS}(H_k)$ is calculated as:

$$\text{AS}(H_k) = \begin{cases} \dfrac{m}{q} \cdot \texttt{mean}_t \left( \dfrac{\sum_{h_i \in H_k} \exp(h_i^\top \cdot h_t)}{\sum_{i=1}^{m} \exp(h_i^\top \cdot h_t)} \right) \\ \qquad \text{Non-hierarchical} \\ \dfrac{q \cdot \exp(H_k^\top \cdot H_q)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot H_q)} \\ \qquad \text{Static attention} \\ \texttt{mean}_t \left( \dfrac{q \cdot \exp(H_k^\top \cdot h_t)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot h_t)} \right) \\ \qquad \text{Dynamic attention} \end{cases} . \tag{4.5}$$

$h_i$ denotes hidden vectors from the encoding steps and $h_t$ denotes hidden vectors from the decoding steps. $m$ is the number of tokens in a context, and $q$ denotes the number of utterances in a context. Note that for non-hierarchical models we multiply by an $m$ in each $\text{AS}(H_k)$ to avoid bias caused by the total number of tokens in different contexts. Similarly

for hierarchical models, we multiply by a $q$ in each $\text{AS}(H_k)$ to avoid bias caused by the number of total utterances in different contexts. As a result, for an utterance $H_q$, $\text{AS}(H_q)$ will be $100\%$ (or approximately $100\%$ for non-hierarchical models) if the model assigns $H_q$ an about average attention score among all utterances.

We denote the last utterance in a context as *Query* and the rest of utterances in the context as *History*. Since different models have different scalars on attention scores, we calculate the average AS for all distracting utterances and all *History* in each dialogue, and use the ratio of them for evaluation. This ratio is denoted as distracting attention score ratio (**DAS ratio**), which measures a model's ability on context attention distribution:

$$\text{DAS ratio} = \texttt{mean}_{d \in D} \left( \frac{\texttt{mean}(\text{AS}(H_{\text{Distraction}}))}{\texttt{mean}(\text{AS}(H_{\textit{History}}))} \right), \tag{4.6}$$

where $d$ means a single dialogue, and $D$ denotes all dialogues in a test set. $H_{\text{Distraction}}$ are distracting utterances, and $H_{\textit{History}}$ are utterances in *History*.

### 4.2.3 Optimization with Self-Contained Distractions on Attention Mechanism

To train a conversational model to distribute more attention to important and less attention to unimportant utterances, we propose the following optimization strategy: 1) For each dialogue, we select some random utterances from other dialogues in the training corpus as **self-contained distractions**. We decide whether to insert these distractions into the current dialogue or not stochastically by a probability level. We denote the probability level as the training inserting probability. The locations of inserting distractions are randomly decided, while the locations are always before *Query* (the last utterance of the context). 2) We create a bitmask $M$ to track whether an utterance is original ($0$) or distracting ($1$). During the training period, the model uses the bitmask to calculate the attention loss $\mathcal{L}^t_{\text{attention}}$, which is summed up with the loss from the response generator. For each decoding step $t$,

the attention loss is calculated as:

$$\mathcal{L}^{t}_{attention} = \begin{cases} \texttt{MSE}(\text{softmax}(\mathcal{H}^{\top} \cdot h_t) \otimes M, 0) \\ \qquad\qquad \text{Non-hierarchical} \\ \texttt{MSE}(\text{softmax}(\mathcal{H}^{\top}_{\mathcal{C}} \cdot H_q) \otimes M, 0) \\ \qquad\qquad \text{Static attention} \\ \texttt{MSE}(\text{softmax}(\mathcal{H}^{\top}_{\mathcal{C}} \cdot h_t) \otimes M, 0) \\ \qquad\qquad \text{Dynamic attention} \end{cases} \tag{4.7}$$

where $\otimes$ means dimensional multiplication. As shown in Equation (4.7), our goal is to minimize the attention assigned to all the self-contained distractions. During the distracting test, no bitmask is offered to the model. The illustration of attention loss on both non-hierarchical and hierarchical models is shown in Figure 4.1.

## 4.3 Experiments

### 4.3.1 Dataset

We use the Ubuntu chatlogs dataset Lowe et al. (2015) as the training and testing corpus, which contains dialogues about solving technical problems of Ubuntu. We choose this dataset because the dialogues have both technical topics and casual chats, meaning that it is easier to distinguish important/unimportant utterances than datasets whose topics are consistent. We use about $0.48M$ dialogues for training, $20K$ dialogues for validation, and $10K$ dialogues for testing. These are the original settings of the Ubuntu chatlogs dataset. We removed all single-turn dialogues.

### 4.3.2 Training

Our methods are built on an LSTM Seq2Seq model with attention mechanism. We used Pytorch Paszke et al. (2017) for implementation. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, a learning rate of

| | **Random: 0.5** | **Random: 0.7** | **Random: 1.0** |
|---|---|---|---|
| *History* | \ | Well, can I move the drives? | Yes. |
| | Or kill all speedlink. | Anyways, you made the changes right? | Well, can I move the drives? |
| | Well, can I move the drives? | Ah not like that. | Then from the terminal type: sudo apt-get update. |
| | Ah not like that. | I did. | Ah not like that. |
| | **Frequent: Begin** | **Frequent: Middle** | **Frequent: End** |
| *History* | Why should I help you? | Well, can I move the drives? | Well, can I move the drives? |
| | I have my right. | Why should I help you? | Ah not like that. |
| | Well, can I move the drives? | I have my right. | Why should I help you? |
| | Ah not like that. | Ah not like that. | I have my right. |
| | **Rare: Begin** | **Rare: Middle** | **Rare: End** |
| *History* | Would you have lunch? | Well, can I move the drives? | Well, can I move the drives? |
| | I should have lunch. | Would you have lunch? | Ah not like that. |
| | Well, can I move the drives? | I should have lunch. | Would you have lunch? |
| | Ah not like that. | Ah not like that. | I should have lunch. |
| *Query* | **I guess I could just get an enclosure and copy via USB.** | | |
| Response | **I would advise you to get the disk.** | | |

Table 4.2: Examples of distracting test sets. Distracting utterances are marked red.

1.0, and a gradient clip threshold of 5. The vocabulary size is 25000 and the dropout rate is 0.2. The learning rate is halved when the perplexity stops dropping, and the training is stopped when the model converges.

### 4.3.3 Examined Models

We examine our proposed evaluation metric on 5 models: non-hierarchical LSTM (Non-hier), static attention without utterance integration LSTM unit (Static), static attention with utterance integration LSTM unit (StaticUI), dynamic attention without utterance integration LSTM unit (Dynamic), and dynamic attention with utterance integration LSTM unit (DynamicUI). In addition, we examine our proposed optimization strategy on these 5 models with 3 training inserting probabilities–0.5, 0.7, and 1.0. Models with a training inserting probability of 0 are regarded as baselines. For comparison, we pick the best overall model and train the model with self-contained distractions but without training on the attention

loss (Non-atten-loss), i.e. the model does not know which utterances are distractions. In total, we train and evaluate 23 model variants.

### 4.3.4 Evaluation

For the distracting test, we set the number of distracting utterances for each dialogue to 2. We have 3 distracting test sets: 1) random distracting test set. Distracting utterances in this test set are randomly picked from the training corpus (outside the current dialogue), and they are randomly picked in every evaluation step, which means that there is not a pre-prepared random distracting test set; 2) frequent distracting test set. Distracting utterances in this test set are formed by frequent words in the training corpus, but these utterances do not appear in the training corpus. In our experiments, we use "why should I help you" and "I have my right" as examples of distracting utterances with frequent words; 3) rare distracting test set. Distracting utterances in this test set have words that are rare in the training corpus, and these utterances do not appear in the training corpus. In our experiments, we use "would you have lunch?" and "I should have lunch" as examples of distracting utterances with rare words.

In the distracting test, we insert distracting utterances into different locations. For 1) random, we insert utterances to a random location before *Query* in each context. Similar to the optimization strategy, we use different probability levels to decide whether a distracting utterance is to be inserted or not. We denote these as testing inserting probability. In our experiments, we set the probability levels to be 0.5, 0.7, and 1.0. We expect the model to perform stably on all different probability levels. For 2) frequent and 3) rare, we have three kinds of inserting locations: at the beginning of a context (marked as Begin), in the middle of the context (marked as Middle), and at the end of the context (before *Query* and after *History*, marked as End). In total, we have 9 test sets for evaluation. See Table 4.2 for the example of each test set.

## 4.4 Results and Discussions

Table 4.3 shows DAS ratios of 23 trained model variants on 9 distracting test sets. Figure 4.2 shows the DAS ratios of 3 example model variants (StaticUI with training inserting probability of 0.0 as the baseline, Non-atten-loss StaticUI with training inserting probability of 0.7, and StaticUI with training inserting probability of 0.7) on 9 distracting test sets. For the full results containing average AS (Attention Score) of distracting utterances and average AS of *History*, please refer to Appendix B.1.

In Table 4.3, we show the perplexity and *History*'s average AS of each model on the non-distracted test set under the "Original" column. Since perplexity scores on the distracting test sets are similar, we show the perplexity scores on the non-distracted test set only. We show the DAS ratios of each model on each of the distracting test sets under the "DAS ratio for distracting test set" column. A lower DAS ratio means that a model distributes less attention to distracting utterances (unimportant utterances) and more attention to the original utterances in *History* (important utterances), from which it can be inferred that the model has a better performance on context attention distribution. Both perplexity and DAS ratio are the lower, the better.

### 4.4.1 Perplexity and Average AS on Non-Distracted Test Set

Perplexity scores are shown in the "Perp." column, under the "Original" column in Table 4.3. Perplexity scores of the examined 23 models are similar; the Static models trained with our proposed optimization strategy and a higher training inserting probability level achieves slightly better performance than other models.

Average AS are shown in the "Avg." column, under the "Original" column in Table 4.3. The average AS of *History* tells about a model's attention distribution for *History* and *Query*. A higher score indicates that less attention is distributed to *Query*. Recall that AS of an utterance is $100\%$ (or approximately $100\%$ for non-hierarchical models) if the utterance is paid about average attention among the dialogue. Overall, the models distribute attention of lower than average to *History*, especially for models with static attention (i.e. the Static model and StaticUI model), which distribute more attention to *Query* than non-hierarchical models and models with dynamic attention. This is apparent from the structure of static

| Prob | Model Structure | Original Perp. | Original Avg. | DAS ratio on distracting test sets: Random 0.5 | Random 0.7 | Random 1.0 | Frequent Begin | Frequent Middle | Frequent End | Rare Begin | Rare Middle | Rare End |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.0 | Non-hier | 43.2 | 91.3 | 0.93 | 0.93 | 0.93 | 0.75 | 0.80 | 0.84 | 0.80 | 0.92 | 1.01 |
| | Static | 44.1 | 61.4 | 0.82 | 0.82 | 0.79 | 0.37 | 0.80 | 1.31 | 0.37 | 0.77 | 1.21 |
| | StaticUI | 44.6 | 57.5 | 0.79 | 0.76 | 0.76 | **0.32** | 0.75 | 1.32 | 0.30 | 0.75 | 1.22 |
| | Dynamic | 45.4 | 81.4 | 0.89 | 0.89 | 0.88 | 0.65 | 0.86 | 1.02 | 0.66 | 0.89 | 1.06 |
| | DynamicUI | 44.7 | 91.6 | 0.94 | 0.94 | 0.93 | 0.72 | 0.84 | 0.86 | 0.73 | 0.93 | 0.97 |
| 0.5 | Non-hier | 43.4 | 87.2 | 0.84 | 0.83 | 0.81 | 0.63 | 0.74 | **0.76** | 0.69 | 0.81 | 0.86 |
| | Static | 44.5 | 66.5 | 0.70 | 0.69 | 0.67 | 0.42 | 0.78 | 1.12 | 0.34 | 0.71 | 0.99 |
| | StaticUI | 44.3 | 47.7 | 0.74 | 0.74 | 0.70 | 0.39 | 0.71 | 1.08 | 0.40 | **0.69** | 0.96 |
| | Dynamic | 44.6 | 81.9 | 0.79 | 0.78 | 0.77 | 0.64 | 0.74 | 0.84 | 0.61 | 0.77 | 0.85 |
| | DynamicUI | 43.9 | 86.7 | 0.82 | 0.81 | 0.80 | 0.60 | 0.84 | 0.87 | 0.61 | 0.80 | 0.83 |
| | Non-atten-loss StaticUI | 44.7 | 71.1 | 0.73 | 0.73 | 0.72 | 0.39 | 0.68 | 0.93 | 0.40 | 0.80 | 1.11 |
| 0.7 | Non-hier | 43.2 | 86.9 | 0.84 | 0.82 | 0.80 | 0.72 | 0.82 | 0.82 | 0.71 | 0.85 | 0.87 |
| | Static | **44.0** | 57.6 | 0.73 | 0.72 | 0.69 | 0.40 | 0.70 | 1.08 | 0.41 | 0.70 | 0.98 |
| | StaticUI | 44.9 | 43.7 | **0.67** | **0.67** | **0.65** | 0.36 | **0.66** | 1.02 | 0.36 | 0.70 | 0.99 |
| | Dynamic | 44.3 | 82.0 | 0.76 | 0.75 | 0.73 | 0.58 | 0.71 | 0.86 | 0.58 | 0.73 | 0.83 |
| | DynamicUI | 44.8 | 85.3 | 0.93 | 0.93 | 0.93 | 0.45 | 0.78 | 0.80 | 0.60 | 0.80 | **0.81** |
| | Non-atten-loss StaticUI | 44.1 | 55.4 | 0.72 | 0.70 | 0.69 | 0.45 | 0.70 | 0.98 | 0.43 | 0.73 | 0.97 |
| 1.0 | Non-hier | 47.3 | 95.9 | 0.91 | 0.90 | 0.90 | 0.84 | 0.86 | 0.85 | 0.85 | 0.87 | 0.88 |
| | Static | **44.0** | 65.4 | 0.70 | 0.70 | 0.68 | 0.49 | 0.74 | 1.08 | 0.46 | 0.71 | 0.88 |
| | StaticUI | 49.6 | 73.5 | 0.96 | 0.95 | 0.94 | 0.66 | 0.86 | 1.53 | **0.21** | 0.86 | 1.50 |
| | Dynamic | 44.7 | 88.8 | 0.79 | 0.78 | 0.77 | 0.63 | 0.75 | 0.82 | 0.65 | 0.77 | 0.82 |
| | DynamicUI | 45.2 | 90.2 | 0.87 | 0.86 | 0.85 | 0.73 | 0.81 | 0.83 | 0.75 | 0.88 | 0.88 |
| | Non-atten-loss StaticUI | 44.1 | 76.5 | 0.72 | 0.71 | 0.69 | 0.49 | 0.74 | 0.98 | 0.49 | 0.77 | 0.98 |

Table 4.3: Results of perplexity (Perp.) and average AS of *History* (Avg.) on the original test set (%) are shown in the "Original" column. We also show results of DAS ratios on 9 distracting test sets and 23 model variants.

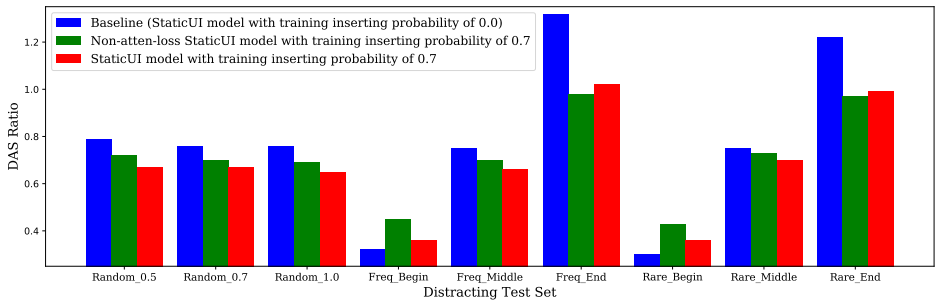



Figure 4.2: DAS ratios of 3 example model variants on 9 distracting test sets. The lower the DAS ratio, the better the performance.

attention. We also show the results of a StaticUI model without training on the attention loss (Non-atten-loss StaticUI model) as a comparison. The StaticUI model trained with our optimization strategy distributes more attention to *query* than the Non-atten-loss StaticUI model. This is because the optimization strategy decreases the model's attention distributed to distracting utterances in *History*, thus decreasing the overall attention distributed to *History*.

### 4.4.2 Distracting Test: Random

Results of the random distracting test with different testing inserting probabilities (0.5, 0.7, and 1.0) are shown in the "Random" column in Table 4.3. Models with training inserting probabilities of 0.0 (shown in the row where "Prob" is 0.0) are baseline models to which our proposed optimization strategy is not applied. In general, our proposed optimization strategy with training inserting probabilities of 0.5 or 0.7 achieves better performance on DAS ratios (i.e. the models achieve lower DAS ratios) on random distracting test sets of all 3 testing inserting probabilities. The Static model and the DynamicUI model achieves the best performance with a training inserting probability of 0.5, while the Non-hier model, the StaticUI model and the Dynamic model achieve the best performance with a training inserting probability of 0.7. A training inserting probability of 1.0 leads to worse performance. One reasons is that it assumes there must be some distracting utterances in a context, while that is not always the case.

The StaticUI model with a training inserting probability of $0.7$ achieves the best overall performance on DAS ratio. As shown in Figure 4.2, on all the random distracting test sets (probabilities of $0.5$, $0.7$, and $1.0$), the StaticUI model is better than the baseline StaticUI model and the Non-atten-loss StaticUI model. The baseline model is not trained with any self-contained distractions (training inserting probability is $0.0$), and it gets the worst performance. The Non-atten-loss model is trained with self-contained distractions (with a training inserting probability of $0.7$) while not knowing which utterances are distractions, and it achieves a better performance than the baseline. The StaticUI model with a training inserting probability of $0.7$ is trained to minimize the attention loss of self-contained distractions and it achieves the best performance. Naturally since the optimization strategy minimizes the attention loss of distractions, the StaticUI model distributes less attention to *History* and more attention to *Query* (refer to the "Avg" column in Appendix B.1 for more details); nevertheless, a lower DAS ratio shows that the model distributes even less attention to the distracting utterances compared to the original utterances in *History*.

Note that even if both our proposed strategy and the random distracting test use the same trick: insert random distracting utterances among original utterances in *History*, the random utterances inserted in the distracting test are different from those inserted in the training process, thus it is difficult for the test to be biased in favor of models with our proposed strategy. It is apparent that less attention is distributed to *History*, while DAS ratio calculates the ratio between the distracting utterances and the original utterances in *History*, so it shows the attention distributed to the distracting utterances regardless of the total attention distributed to *History*. Moreover, we adopt three testing inserting probability levels to ensure stable evaluation results for each model.

### 4.4.3 Distracting Test: Frequent and Rare

Results of the frequent and the rare distracting test are shown in the "Frequent" and "Rare" columns in Table 4.3. Different from the random distracting test, the inserting locations of these two tests are decided manually. As a nature of LSTM model, all models distribute more attention to utterances near *Query* and less attention to utterances far away from *Query*, as can be seen in Table 4.3 and Figure 4.2 that DAS ratios are higher for End test

set (near *Query*) and lower for Begin test set (far away from *Query*). Since the results on Begin and End test sets are biased by the structure of LSTM, we mainly analyze the results on Middle test sets.

For the Middle test sets of both the frequent and rare distracting test, the best models are still those trained with our proposed optimization strategy. StaticUI model with training inserting probabilities of $0.5$ and $0.7$ achieve the best performance (lowest DAS ratios) on the Frequent Middle and Rare Middle test sets. The Non-atten-loss models can be better than the models trained with a wrong training inserting probability. Telling from similar DAS ratios, the frequent distracting test set is as difficult for the trained models to distinguish as the rare distracting test set, although for humans, the rare distracting utterances are much easier to distinguish than the frequent ones.

### 4.4.4 Summary of Results

DAS ratio can distinguish conversational agents with similar perplexity on their ability of context attention distribution. In general, models trained with our proposed optimization strategy focus less on distracting utterances and more on original utterances in *History*. For most models, DAS ratios decrease by about $10\%$ when trained with our proposed strategy with a $0.5$ or $0.7$ probability level. $0.7$ performs best as a training inserting probability.

## 4.5 Conclusions

We have studied context attention distribution, an essential component of multi-turn modelling for open-domain conversational agents. We have proposed an evaluation metric for context attention distribution based on the distracting test: DAS ratio. We have also improved the performance of context attention distribution for common multi-turn conversational agents through an optimization strategy via reducing the attention loss of self-contained distracting utterances. Extensive experiments show that our proposed strategy achieves improvements on most models, especially with a training inserting probability level of $0.7$.

## Acknowledgements

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assistance on computing resources from the IDUN cluster of NTNU (Själander et al., 2019). We would like to thank Pinar Øzturk, Benjamin Kille, and Peng Liu for their helpful comments.

# Chapter 5

# Context-Awareness and Summarization

The study of context-awareness in multi-turn generation-based dialogue modeling is an important but relatively underexplored topic. Prior research has employed hierarchical structures to enhance the context-awareness of dialogue models. This paper aims to address this issue by utilizing two extractive summarization techniques, namely the PMI topic model and the ORACLE algorithm, to filter out unimportant utterances within a given context. Our proposed approach is assessed on both non-hierarchical and hierarchical models using the *distracting test*, which evaluates the level of attention given to each utterance. Our proposed methods gain significant improvement over the baselines in the distracting test.

## 5.1 Introduction and Related Works

Although generation-based dialogue models have achieved much progress in recent years, multi-turn dialogue models are still facing challenges. Recent works deal with multi-turn using modified attention mechanisms and hierarchical structures. One focus of dealing with multi-turn is the ability of context-awareness on a dialogue model, which requires a model to pay more attention to important utterances while less attention to unimportant ones. An example of important/unimportant utterances is given by Table 5.1.

In Table 5.1, the first two utterances ("Haha sucker." and "?") are unimportant utterances that are irrelevant to the main topic of the context. A multi-turn dialogue model with good ability on context awareness should identify and ignore these unimportant utterances

| User | utterances |
|---|---|
| Taru | Haha sucker. |
| Kuja | ? |
| Taru | Anyways, you made the changes right? |
| Kuja | Yes. |
| Taru | Then from the terminal type: sudo apt-get update |
| Kuja | I did. |

Table 5.1: An example of important utterances and unimportant utterances under the same context in the Ubuntu chatlog dataset Lowe et al. (2015). Unimportant utterances are marked in red.

and focus only on the important ones. Thus, we propose that one way to improve the context awareness of a model is to **filter out** the unimportant utterances, which is a task similar to summarization: given a reference and a source, an extractive summarization algorithm extracts all utterances related to the reference and eliminate all others in the source. In the case of dialogue models, we do not have a reference for the context; nevertheless, the last utterance in the context, i.e., *Query*, plays a crucial role in generating the response. In most cases, responses aim to provide answers to *Query* while utilizing other utterances in the context (we denote them as *Source*) as the source for answering. This paper investigates improving context awareness for multi-turn dialogue models by filtering out unimportant utterances using extractive summarization techniques with *Query* as the reference.

There are a few works that combine summarization with dialogue models. One of the techniques used in these works is the topic model, where a keyword is predicted from *Query* and the entire corpus to help a model generate detailed responses. In our paper, we also use a PMI topic model to extract keywords from *Source*, while instead of using the keywords to support the generation task, we pass the keywords directly to the dialogue model. Additionally, we explore the ORACLE algorithm, a widely-used algorithm for generating gold labels for extractive summarization, to filter out utterances unrelated to *Query* before passing them to the dialogue model.

For evaluation, we use an evaluation method tailored for multi-turn dialogue models. Since most multi-turn dialogue models have attention mechanisms and they rely on the mechanism to assign different extents of focus to each utterance in the context, we use the

**distracting test** to measure if a model pays more attention to the important utterances and less to the unimportant ones. The test simply adds distracting utterances to each dialogue and compares the attention scores on these distracting (unimportant) utterances with the original (important) utterances in *Source*, thus measures the ability of context awareness for a dialogue model.

In Section 5.2, we introduce the summarization techniques to be integrated. In Section 5.3, we describe our experiment settings, and we report the results in Section 5.4. Finally, in Section 5.5, we give a conclusion to this chapter.

## 5.2 Proposed Methods

In this paper, we choose an LSTM Seq2Seq model with attention mechanism Hochreiter and Schmidhuber (1997); Sutskever et al. (2014); Bahdanau et al. (2015) as the base model, since most hierarchical structured multi-turn conversational agents are based on LSTM Serban et al. (2016, 2017); Tian et al. (2017); Zhang et al. (2018b) while few are based on Transformers. The architecture of an LSTM model with the attention mechanism and the description of the context vector $c_t$ in the attention mechanism can be found in Section 2.2. The description of various context vectors (e.g. static and dynamic context vectors) and the utterance integration unit can be found in Section 4.2.1.

### 5.2.1 PMI-context

The method PMI-context uses a Pointwise Mutual Information (PMI) to select the $k$ most relevant words in a *History* given a *Query*. Given a word $x_s$ in *Source*, the total PMI of $x_s$ given a *query* $= x_{q1}, ..., x_{ql}$ is calculated following Yao et al. (2017b):

$$\text{PMI}(x_{q1}, ..., x_{ql}, x_s) \approx \sum_{i}^{l} \text{PMI}(x_{qi}, x_s) \ . \tag{5.1}$$

The selected $k$ keywords $x_{s1}, ..., x_{sk}$ and the $query$ are combined through the static attention mechanism described in Equation (4.2) to calculate the context vector $c_t$. Note that here a $query$ does not attend to itself, but only to the selected keywords. The context

vector $c_t$, the selected $k$ keywords, and the $query$ are then inputted into the LSTM unit as described in the following adapted version of Equation (2.9):

$$h_t = LSTM^*(E(z'_t), h_{t-1}, c_t) \,, \tag{5.2}$$

where $z'_t \in \{x_{s1}, ..., x_{sk}, x_{q1}, ..., x_{ql}, y_1, ..., y_{n-1}\}$.

### 5.2.2 ORACLE-context

The method ORACLE-context is based on an extractive summarization algorithm named the ORACLE algorithm. It uses the ORACLE algorithm to extract relevant utterances from *Source* by greedily optimizing ROUGE-1 using *Query* as the summarization reference. The extracted $k$ most relevant utterances are then inputted into the LSTM unit as described in the following adapted version of Equation (2.9):

$$h_t = LSTM^*(E(z''_t), h_{t-1}, c_t) \,, \tag{5.3}$$

where $z''_t \in \{x^1_{s1}, x^1_{s2}, ..., x^k_{s1}, x^k_{s2}, ..., x_{q1}, ..., x_{ql}, y_1, ..., y_{n-1}\}$, and $X_i = x^i_{s1}, x^i_{s2}, ...$ ($i \in \{1, ..., k\}$) denotes for each of the extracted $k$ most relevant utterances.

This method intends to filter out irrelevant utterances from *Source* given *Query* and delete them from the inputs to the dialogue model, which helps the model to pay attention correctly to the important utterances.

### 5.2.3 Evaluation

For the evaluation, since perplexity is considered not a good measure of how good a conversation is (Liu et al., 2016), besides perplexity, we examine whether the model pays attention to the correct utterance through the distracting test, attention scores, and DAS score. The description of these can be found in Section 4.2.2.

## 5.3 Experiment Setup

### 5.3.1 Dataset

We use the Ubuntu chatlogs dataset Lowe et al. (2015), which contains dialogues about solving technical problems of Ubuntu, as the training and testing corpus. We have about $0.48M$ dialogues for training, $20K$ dialogues for validation, and $10K$ dialogues for testing. These are the original settings of the Ubuntu chatlogs dataset. We removed all single-turn dialogues, since single-turns do not have contexts that we need to study on. The last utterance in the context is treated as *Query*, and the other utterances are treated as *Source*.

For the distracting test, we set the amount of distracting utterances for each dialogue as 2. We have 3 distracting test datasets: 1) dataset distracted with utterances containing frequent words, which are "why should I help you" and "I have my right"; 2) dataset distracted with utterances containing rare words, which are "would you have lunch?" and "I should have lunch"; 3) dataset distracted with utterances randomly picked from the training set.

### 5.3.2 Training

Our methods are built on a basic LSTM Seq2Seq model. We used Pytorch Paszke et al. (2017) for implementation. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, a learning rate of 1.0, and a gradient clip threshold of 5. The vocabulary size is 25000 and the dropout rate is 0.2.

### 5.3.3 Models to be examined

For the method PMI-context, we examine the maximum keyword amounts of both 10-word level and 30-word level. For the method ORACLE-context, we examine the maximum extracted utterance amounts of both 5-utterance level and 10-utterance level. Also, we examine ORACLE-context on 5 model variants, namely static attention with utterance integration LSTM unit, static attention without utterance integration LSTM unit, dynamic attention with utterance integration LSTM unit, and dynamic attention without utterance

integration LSTM unit. Among these variants, one is non-hierarchical structured, and the other four are hierarchical structured.

## 5.4 Results

We show the perplexity and attention scores of the models to be examined. For comparison, we also show scores of non-hierarchical model trained on either the whole context (*Source* and *Query*) or only *Query*. The results are shown in Table5.2.

For the distracting test, besides the attention scores of the distracting utterances, we also show the average attention scores of *Source*. A lower score indicates that more attention is paid to *Query* instead of *Source*. In addition, we calculate the ratio between the attention scores of the distracting utterances and those of the original utterances in *Source*, to show how much attention is paid to the distracting utterances compared to *Source*. A lower ratio indicates that the model is less distracted by the distracting utterances.

Table 5.2 shows that the non-hierarchical model with the ORACLE-context method of 10-utterance level has the best perplexity and the lowest attention scores' ratio for the frequent and rare distracting datasets, which indicates that this model is the least distracted from frequent and rare distracting utterances. Among the four kinds of hierarchical models, the variant of static attention mechanism with utterance integration LSTM unit (Static+UttLSTM) gets the best performance on the random distracting dataset, and most of the other variants manage to exceed the non-hierarchical model on the random distracting dataset, from which we can infer that the hierarchical models are less distracted from random distracting utterances. PMI-context method of the 30-word level also gains a good perplexity, but since perplexity is not a good method for evaluating responses' quality, more evaluation is needed.

It is easy to notice that while the perplexity scores of the ORACLE-context models show marginal improvement over the baselines, they outperform the baselines in the distracting test, which is a better evaluation metric for the ability of context-awareness. To assess the efficacy of the ORACLE algorithm, we further investigated the filtered-out and extracted utterances. Results show that approximately 79%, 84%, and 82% of the distracting utterances were filtered out in each of the three distracting datasets, respectively. In

| Method | Model | Original | | Distract: random | | |
|---|---|---|---|---|---|---|
| | | Perp | Avg. | Perp | Distract (ratio) | Avg. |
| \ | Non-hier (*Query* only) | 49.5 | 100 | \ | \ | |
| | Non-hier | 49.8 | 94.7 | 49.8 | 94.4 (0.99) | 95.4 |
| PMI | PMI-10 | 49.5 | \ | 49.5 | \ | |
| | PMI-30 | 47.8 | \ | 47.8 | \ | |
| ORACLE-5 | Non-hier | 48.1 | 86.2 | 48.7 | 82.4 (0.94) | 87.2 |
| | static | 49.0 | 68.0 | 49.3 | 56.8 (0.81) | 70.0 |
| | static+UttLSTM | 51.3 | 52.8 | 51.6 | **41.2 (0.76)** | 54.1 |
| | dynamic | 49.7 | 86.8 | 50.2 | 81.4 (0.93) | 88.0 |
| | dynamic+UttLSTM | 50.7 | 93.8 | 51.2 | 91.3 (0.97) | 94.4 |
| ORACLE-10 | Non-hier | **47.1** | 86.5 | **47.7** | 82.5 (0.94) | 87.4 |
| | static | 49.5 | 60.7 | 49.9 | **47.1 (0.75)** | 62.4 |
| | static+UttLSTM | 47.7 | 54.1 | 48.0 | 43.5 (0.79) | 55.3 |
| | dynamic | 49.9 | 85.5 | 50.3 | 80.0 (0.92) | 86.7 |
| | dynamic+UttLSTM | 49.6 | 95.0 | 49.9 | 93.4 (0.98) | 95.3 |

(a) Results on the random distract testset

| Method | Model | Distract: frequent | | | Distract: rare | | |
|---|---|---|---|---|---|---|---|
| | | Perp | Distract (ratio) | Avg. | Perp | Distract (ratio) | Avg. |
| \ | Non-hier (*query only*) | \ | \ | | \ | \ | |
| | Non-hier | 49.7 | 94.3 (0.98) | 95.8 | 49.8 | 94.4 (0.99) | 95.5 |
| PMI | PMI-10 | 49.5 | \ | | 49.5 | \ | |
| | PMI-30 | 47.8 | \ | | 47.8 | \ | |
| ORACLE-5 | Non-hier | 48.3 | 74.8 (0.86) | 86.9 | 48.4 | 78.1 (0.90) | 86.3 |
| | static | 49.1 | 65.1 (0.95) | 68.7 | 49.2 | 63.0 (0.91) | 69.3 |
| | static+UttLSTM | 51.4 | 46.9 (0.88) | 53.4 | 51.4 | **48.3 (0.90)** | 53.5 |
| | dynamic | 49.9 | 79.3 (0.90) | 88.3 | 50.0 | 83.0 (0.95) | 87.5 |
| | dynamic+UttLSTM | 50.8 | 89.3 (0.95) | 94.6 | 50.9 | 94.3 (1.01) | 93.0 |
| ORACLE-10 | Non-hier | **47.3** | **69.9 (0.80)** | 87.3 | **47.3** | **74.3 (0.86)** | 86.8 |
| | static | 49.7 | 51.0 (0.83) | 61.7 | 49.7 | 55.3 (0.90) | 61.5 |
| | static+UttLSTM | 47.7 | **46.8 (0.86)** | 54.7 | 47.9 | 51.1 (0.95) | 54.1 |
| | dynamic | 50.1 | 79.3 (0.92) | 86.4 | 50.1 | 87.9 (1.03) | 85.0 |
| | dynamic+UttLSTM | 49.7 | 91.1 (0.95) | 95.9 | 49.8 | 94.6 (1.00) | 94.3 |

(b) Results on the frequent and rare distracting dataset

Table 5.2: Perplexity (Perp), attention score of distracting utterances (Distract, %), attention score of average original utterances in *Source* (Avg., %), and their ratio (ratio). The best attention scores of distracting utterances and the best ratios are bolded.

contrast, the algorithm extracted a considerable portion of the first and second utterances closest to *Query*, which are typically regarded as important utterances in a *Source*, and these make up 30% and 43% of the total extracted utterances, respectively. This means that

the ORACLE algorithm does filter out unimportant utterances to some extent.

It is surprising to see that the models have the worst performance for the distracting dataset with rare utterances. It is obvious for humans to identify "Would you have lunch?" and "I should have lunch" as distracting utterances, while although the ORACLE algorithm only keeps 16% of these distracting utterances, the model still cannot learn to pay less attention to them.

## 5.5 Conclusions

We have integrated extractive summarization techniques with multi-turn dialogue models to improve their ability of context-awareness. The techniques that we have examined are PMI topic model and ORACLE algorithm; we have integrated them with both non-hierarchical and hierarchical dialogue models. For evaluation, we have employed the distracting test to evaluate the context-awareness of each model. With extractive summarization techniques integrated, we find significant improvements in distracting tests for the multi-turn conversational agents.

## Acknowledgements

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assistance on computing resources from the IDUN cluster of NTNU (Själander et al., 2019). We would like to thank Pinar Øzturk for her helpful comments.

# Chapter 6

# Integrating Knowledge

In recent years, prompt-based tuning and instruction-based tuning have emerged as popular approaches for natural language processing. In this paper, we investigate the application of prompt and instruction-based tuning approaches for response generation in conversational question answering. We approach this task from both extractive and generative angles, where we adopt prompt-based tuning for the extractive angle and instruction-based tuning for the generative angle. Additionally, we utilize multi-task learning to integrate these two angles. To evaluate the performance of our proposed approaches, we conduct experiments on the GPT-2 model. The results show that the approaches improve performance by $18\%$ on F1 score over the baseline.

## 6.1 Introduction and Related Works

Conversational Question Answering (CQA) is a QA dialogue system that can answer user questions based on a given document. CQA is an extension of traditional QA systems to a conversational setting and engages in multi-turn conversation to satisfy a user's information needs. According to the types of QA, CQA is studied in two settings: extractive and generative. In the extractive setting, the answer is marked as a span in the text paragraph, whereas in the generative setting, i.e. response generation in CQA, the answer is free-form text generated by autoregressively predicting tokens.

With the rapid development of language modeling techniques, a lot of pre-trained language models have been successfully applied to extractive CQA (Chen et al., 2023; Ram et al., 2021), generative CQA (Izacard and Grave, 2021; Xiong et al., 2021) and unified systems that solve various CQA tasks through a single model (Khashabi et al., 2020; Tafjord and Clark, 2021). Recently, Gekhman et al. (2022) have conducted a comprehensive robustness study of history modeling approaches for CQA and propose a prompt-based history highlighting method to improve robustness while maintaining overall high performance. However, prompts are generally short and do not generalize well to reformulations and new tasks.

Instruction tuning is an emergent paradigm where models are trained on a variety of tasks with natural language instructions. Instructions in natural language make it easy for questioners to ask questions, and are proven to achieve a good performance due to the nature of the language model (Gupta et al., 2022). To the best of our knowledge, we are the first to apply instruction tuning for response generation on conversational question answering. Our paper proposes approaches for enhancing the response generation of conversational question answering by integrating prompt-based and instruction-based tuning. We adopt the prompt-based tuning method introduced by Gekhman et al. (2022) to improve from the extractive angle on the multi-turn scenario. Additionally, we propose an instruction-based tuning method to enhance from the generative angle, based on the work of Zhong et al. (2022) and Gupta et al. (2022). Furthermore, we investigate the integration of these two angles through multi-task learning.

In our experiments, we verify the influence of prompt-based tuning, instruction-based tuning, and multi-task learning for the task. We evaluate the performance of various settings, including prompt-based tuning with or without multi-task learning, prompt-based with or without instruction-based tuning, and prompt-based tuning with both multi-task learning and instruction-based tuning. We conduct the experiments on GPT-2 and evaluate the results on F1 score with 2 modes: the decoding mode and the evaluation mode. Additionally, we assess the extractive question answering part of the settings with a GPT-2 fine-tuned on the extractive question answering task.

The results show that our prompt-based tuning together with other approaches has improved the performance by about $18\%$ on F1 score over the baseline, and the instruction-based tuning and multi-task learning settings have improved further at about $1\%$ compared to pure prompt-based tuning approach.

The main contributions of this work are:

- To the best of our knowledge, we are the first to incorporate instruction tuning in conversational question answering.
- We investigate tuning approaches based on prompt and instruction for the response generation task on conversational question answering. The approaches are simple and easy to be adapted to other models.
- We conduct comprehensive experiments on the influence of instruction-based tuning, prompt-based tuning and multi-task learning for this task. The results show that the best approach improves about $18\%$ on F1 score than the baseline.

We define our task and introduce the approaches used in our research in Section 6.2. In Section 6.3 we describe the setups of our experiments, and in Section 6.4 we present our results. Finally, in Section 6.5, we give a conclusion to this chapter.

## 6.2 Methodology

In this section, we first define the task of conversational question answering, and we introduce how this task and response generation is realized under GPT-2. After that, we explain the proposed multi-task learning, prompt tuning, and instruction tuning in detail.

### 6.2.1 Conversational Question Answering

The task of conversational question answering is to predict the answer span (start position, end position) in a passage for the given question and the previous questions and answer spans. The question answering task can be transferred to two classification tasks: one for the start position, and the other for the end position. Given a question $Q$ and a passage

$X$, the tasks are to calculate the probability of the $t$-th token in the passage $X$ is the start position $P_{x_t=\text{start}}$ and is the end position $P_{x_t=\text{end}}$:

$$P(x_t = \text{start} \mid Q, X) \tag{6.1}$$

$$P(x_t = \text{end} \mid Q, X), \tag{6.2}$$

where $Q = q_1, \ldots, q_k$, $X = x_1, \ldots, x_m$ are sequences of tokens.

The difference between the task of conversational question answering with regular question answering is that there are conversation histories, i.e. multiple turns of questions and answer spans.

The question answering task is dealt with the GPT-2 model as follows. First, a hidden vector that is to be input to the transformer block is calculated as:

$$h_0 = E(Q, X) + (E_0, E_1) + W_p, \tag{6.3}$$

where $E(Q, X)$ is the sub-word embedding for question $Q$ and passage $X$. $E_0$ and $E_1$ are state embeddings, where $E_0$ is assigned to the question, and $E_1$ is assigned to the passage. $W_p$ is a pre-trained position embedding. Then, the probability of the subword $t$ to be the start or end position is calculated as:

$$h_X = \text{GPT\_block}(h_0)[X] \tag{6.4}$$

$$P(x_t = start) = \text{softmax}(A \cdot h_X)[t] \tag{6.5}$$

$$P(x_t = end) = \text{softmax}(B \cdot h_X)[t], \tag{6.6}$$

where $A \in \mathbb{R}^{1\times\dim(h)}$ and $B \in \mathbb{R}^{1\times\dim(h)}$, $h_X$ denotes for slice of the passage $X$ part in the hidden vector, and $[t]$ denotes for the $t$-th subword token in the passage $X$. We simplify the structure of the transformer block as GPT_block. In the block, a mask bans past words from attending to future words. Equation 6.5 and Equation 6.6 transfer $h_X \in \mathbb{R}^{\dim(h)\times|X|}$ into sequences of probabilities for each subword token in $X$, where the probability of a subword $t$ being the start position or the end position can be obtained.

### 6.2.2 Response Generation

The description of the task of response generation and the use of GPT-2 on this task can be found in Section 2.2.

### 6.2.3 Prompt-Based Tuning

Following Gekhman et al. (2022), we add prompts to the passage for the conversational question answering task, where the prompts indicate the answers to the previous questions. For any turn $i$, all the answer spans of the previous turns $(S_j, A_j)$ ($j \in [1, \ldots, i-1]$) are marked in the passage $X$ with the prompts $<j>$. Examples of prompt-based tuning can be found in the following table:

Table 6.1: An example of prompt-based tuning

| Turn | Question | Text of Answer Span | Prompted Passage |
|---|---|---|---|
| 1 | What color was Cotton? | a little white kitten named Cotton | Once upon a time, in a barn near a farm house, there lived a little white kitten named Cotton. Cotton lived high up... |
| 2 | Where did she live? | in a barn near a farm house, there lived a little white kitten | Once upon a time, in a barn near a farm house, there lived <1>a little white kitten named Cotton <1>. Cotton lived high up... |
| 3 | Did she live alone? | Cotton wasn't alone | Once upon a time, <2>in a barn near a farm house, there lived <1>a little white kitten <2>named Cotton <1>. Cotton lived high up... |

Note that for any turn $j$ that does not have an answer span, there is not a prompt $<j>$for it.

### 6.2.4 Instruction-Based Tuning

Furthermore, following Zhong et al. (2022) and Gupta et al. (2022), we add instructions to the inputs. We use two kinds of instructions: an *instruction* at the beginning of the input, and several *guidances* among the sections that constitute the input. The instruction at the beginning of the input is word-based, and it introduces what the task is about. The guidances are word-based with symbols, such as "[Instruction]:", "[Question]:", "[Passage]:" and "[Answer]:", which separate each section and clarify what each section is. We denote an instruction as a sequence of tokens: $I = I_1, \ldots, I_j$, and guidances for each section as $G_{\text{Section 1}}, G_{\text{Section 2}}, \ldots$. The instruction and the guidances are inserted into the original input as follows:

$$[G_{\text{instruction}}, I, G_{\text{question}}, Q, G_{\text{passage}}, X, G_{\text{answer}}, Y], \tag{6.7}$$

where $Q$ is the question, $X$ is the passage, and $Y$ is the answer. $Q$, $X$ and $Y$ are all sequences of tokens, and in Equation 6.7 they are concatenated. We denote $X_I = [G_{\text{instruction}}, I, G_{\text{question}}, Q, G_{\text{passage}}, X, G_{\text{answer}}]$, then the hidden vector to be input to the transformer block is calculated as:

$$h_{0[t]} = E(X_I, Y_{[1:t]}) + (E_0, E_1) + W_p, \tag{6.8}$$

### 6.2.5 Multi-Task Learning

To fully leverage the extractive question answering task, we employ a multi-task learning approach to integrate it with the response generation task. Specifically, we use the same hidden vector as described in Equation 2.16 as input to the transformer block, which is then used for calculating the probability distribution of the vocabulary for the next token, as well as the probability of the start and end position for each token in the passage. The multi-task learning approach optimizes both answer span extraction and response generation simultaneously. The loss is then integrated as:

$$\mathcal{L}_{\text{QA}} = \frac{\mathcal{L}_{\text{start position}} + \mathcal{L}_{\text{end position}}}{2} \tag{6.9}$$

$$\mathcal{L} = \mathcal{L}_{\text{QA}} + \mathcal{L}_{\text{response generation}}. \tag{6.10}$$

## 6.3 Experimental Setup

### 6.3.1 Dataset

We employ the CoQA (Conversational Question Answering) dataset (Reddy et al., 2018) for our research. The CoQA dataset is a collection of conversational question answering instances spanning a broad range of domains, such as literature, news, and Wikipedia articles. The dataset is conversational because it includes conversational histories, i.e., the previous turns in a conversation leading up to the current question-answer pair. The answers in the dataset include both answer spans for extractive question answering and human-written free-form answers for generative question answering.

### 6.3.2 Model and Tuning

In the experiments, we will evaluate 5 models:

(1) Response generation (baseline)

(2) Response generation with prompt-based tuning (`prompt`)

(3) Response generation with prompt-based tuning & instruction-based tuning (`w instruct`)

(4) Response generation with prompt-based tuning & multi-task learning (`w multi-task`)

(5) Response generation with prompt-based tuning & instruction-based tuning & multi-task learning (`w multi-task & w instruct`)

We have excluded three other settings, namely response generation with instruction-based tuning, response generation with multi-task learning, and response generation with instruction-based tuning & multi-task learning, since prompts are necessary indicators for multi-turns. Our task–the conversational question answering–is based on multi-turns, so any model without prompt-based tuning, other than the baseline, is considered not relevant to the task.

The instructions and prompts that we used in the prompt-based tuning and instruction-based tuning are described in the following table:

Table 6.2: An example for prompt and instruction based tuning

| | Prompt-Based Tuning | Instruction-Based Tuning |
|---|---|---|
| Instruction | \ | [Instruction]:<br>Answer the question based on the given passage. |
| Question | Where did she live? | [Question]:<br>Where did she live? |
| Passage | Once upon a time, in a barn near a farm house, there lived <1>a little white kitten named Cotton <1>. Cotton lived high up... | [Passage]:<br>Once upon a time, in a barn near a farm house, there lived a little white kitten named Cotton. Cotton lived high up... |
| Answer | in a barn | [Answer]:<br>in a barn |

### 6.3.3 Training

Our implementation makes use of Pytorch (Paszke et al., 2017) and the HuggingFace Transformers [1]. We adopted GPT-2 basic[2] which has 12 layers and 12 heads with a dimension of 768. The training procedure was with a batch size of 16, 10 epochs, a learning rate of $3 \cdot 10^{-5}$, a weight decay of $0.01$, cross-entropy loss and AdamW. The input sequences are $1024$ tokens.

### 6.3.4 Evaluation

We evaluate the similarity between the human input answers and the generated answers using the F1 score. We compare the performance of five models, namely the baseline, `prompt`, `w instruct`, `w multi-task`, and `w multi-task & w instruct`,

[1] `https://huggingface.co/.`
[2] `https://huggingface.co/gpt2`

using the official dev dataset for evaluation. We compare the latter 4 models with the baseline and the latter 3 models with the prompt model. To ensure consistency, we limit the maximum output length to 64 tokens. We use two different evaluation modes, decoding mode and evaluation mode, to assess the performance of the models.

In decoding mode, models are not provided with any information about the previous turns and are required to use the predicted answer spans from the previous turn as prompts for generating responses. Only models with multi-task learning can generate answers under this mode. In contrast, the evaluation mode provides the correct information on previous turns to the models. This mode enables pure generation models to handle multi-turns with prompts, thus making them more accurate in generating responses. We employ prompt-tuning in the evaluation mode, whereby the correct information on the previous answers is prompted in the same way as introduced in Section 6.2.3.

By default, the evaluation mode generates better results than the decoding mode, given the correct information on previous turns. We provide results for both the evaluation mode and decoding mode to ensure a comprehensive evaluation. In many real-life scenarios, we cannot assume that we have access to the correct answer spans for previous questions, which makes evaluation using the evaluation mode impractical. Therefore, by including decoding mode results, we can provide a more realistic evaluation of our approach that reflects the real-life scenarios.

We also evaluate the performance of the extractive QA part of the two models with multi-task learning (`w multi-task` and `w multi-task & w instruct`) and compare them with an GPT-2 model fine-tuned on extractive question answering task. We measure the similarity between the predicted answer span text and the original answer span text using the F1 score.

We show which mode is applied for each model in the following table:

Table 6.3: Models and modes

| | Decoding Mode | Evaluation Mode |
|---|---|---|
| baseline | ✗ | ✓ |
| `prompt` | ✗ | ✓ |
| `w instruct` | ✗ | ✓ |
| `w multi-task` | ✓ | ✓ |
| `w multi-task & w instruct` | ✓ | ✓ |

Table 6.4: F1 results (%) for different models. Numbers in the brackets state F1 improvements compared to the baseline under evaluation mode.

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| baseline | \ | 53.8 |
| `prompt` | \ | 63.0 (+17.1) |
| `w instruct` | \ | 63.7 (+18.4) |
| `w multi-task` | 61.6 (+14.4) | 63.9 (+18.7) |
| `w multi-task & w instruct` | 56.5 (+5.0) | 57.8 (+7.4) |

## 6.4 Results

### 6.4.1 Automatic Results

Table 6.4 and Table 6.5 summarize the response generation performance of five models w.r.t. F1 score and its improvements. Since only models with multi-task learning can generate answers in the decoding mode, we use backslash '\' to denote this setting is not applicable to the first three models.

From the results, we have the following observations:

1) As shown in all the tables, the performance of the evaluation mode is better than decoding mode. This is because the evaluation mode can provide the correct answer

Table 6.5: F1 improvement (%) compared to `prompt` (evaluation mode)

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| `w instruct` | \ | +1.1 |
| `w multi-task` | −2.2 | +1.4 |
| `w multi-task & w instruct` | −10.3 | −8.2 |

spans from previous turns to the models for prompt-tuning.

2) In Table 6.4, prompt-based tuning outperforms baseline by a large margin, demonstrating that prompt can encode valuable information about the answers from previous conversation turns for model tuning. Besides, instruction-based tuning can further improve the response generation performance, which proves the usefulness of injecting task-specific guidance during fine-tuning. Apart from that, compared with the "`prompt`" model and the "`w instruct`" model, the "`w multi-task`" model achieves the best performance, from which we can find the conversational question answering task can significantly facilitate the response generation task.

3) The brackets of Table 6.4 show the F1 score improvements compared to the baseline under evaluation mode. As expected, all the models have certain performance improvements compared to the baseline. In particular, the "`w multi-task`" model has the highest performance improvement, which is 18.7% and 14.4% in the evaluation and decoding modes, respectively.

4) Table 6.5 shows the F1 score improvement compared to the "`prompt`" model (evaluation mode). We find that the performance of the "`w multi-task`" model drops by 2.2% in the decoding mode, suggesting that answer prediction errors from previous conversation turns can accumulate to have a large impact on the response generation task. Another interesting observation is that the performance of the "`w multi-task & w instruct`" model drops 10.3% and 8.2% in the decoding and evaluation modes, respectively. This is probably because the optimization of the multi-task learning and instruction-based tuning are conflicting with each other.

Table 6.6: F1 results and improvement (%) for the extractive question answering part. Answer span texts instead of human answers are used for evaluation.

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| GPT-2 fine-tuned on extractive QA | 63.9 (\) | 64.7 (\) |
| `w multi-task` (QA part) | 60.2 (−5.7) | 65 (+0.4) |
| `w multi-task & w instruct` (QA part) | 64.9 (+1.6) | 70.1 (+8.3) |

Table 6.6 reports the evaluation results of the extractive question answering part of a GPT-2 model fine-tuned on extractive question answering task and the two models with multi-task learning. Compared with the baseline (GPT-2 fine-tuned on extractive question answering), both multi-task learning models can improve the performance of question answering task, which demonstrates the effectiveness of prompt-based and instruction-based tuning and the boosting effect of the response generation task on the question answering task. We can also observe that the performance of the "`w multi-task`" model drops by 5.7% in the decoding mode, which is due to the accumulated answer prediction errors from previous turns.

### 6.4.2 Qualitative Results

Table 6.7: An example of the difference between extractive question answering and generated answers

| Question | Gold Answer Span Text | Human | Extractive QA Answer | Generated |
|---|---|---|---|---|
| Is it a small city? | the most populated city in the state of Nevada | No | is the 28th-most populated city in the United States | No |
| Which state is it in? | Vegas, is the 28th-most populated city in the United States, the most populated city in the state of Nevada | Nevada | is the 28th-most populated city in the United States, the most populated city in the state of Nevada | Nevada |
| What is it famous for? | The city bills itself as The Entertainment Capital of the World, and is famous for its mega casino hotel | mega casino hotel | famous for its mega casino hotels and associated activities | gambling, shopping, fine dining, entertainment, and nightlife |

Table 6.7 presents a comparative analysis between answer spans predicted by the question answering module and generated answers. The first question demonstrates that for

yes/no questions, the generated answer provides a more direct response, whereas the extractive QA answer only provides the information required to answer the question without a simple yes or no. The second question highlights that in cases where there is no direct answer in the passage, the generated answer provides a better response as it directly addresses the question. However, the third question illustrates that in some cases, extractive QA answers are superior, as the given answer is fully grounded in the passage. The generated answer may be based on the passage and relevant to the question, but not necessarily grounded in the passage.

Table 6.8: An example of answers generated by different models

| Question | Baseline | `prompt` | `w instruct` | `w multi-task` | `w multi-task & w instruct` |
|---|---|---|---|---|---|
| What is it famous for? | its the largest city within the greater Mojave Desert. | its real things | its gambling, shopping, fine dining, entertainment, and nightlife | gambling, shopping, fine dining, entertainment, and nightlife | a guitar hotels and associated activities |

Table 6.8 provides a comparative analysis of answers generated by different models. The baseline model generates answers that are not related to the question, while the "`prompt`" model generates answers that are related to the question but not grounded in the passage. In contrast, the "`w instruct`" and "`w multi-task`" models generate good quality answers that are grounded in the passage. The "`w multi-task & w instruct`" model generates an answer that is almost identical to the gold standard, however with a deviation in the form of "guitar hotels" instead of "mega casino hotels". Qualitatively, the "`w instruct`" and "`w multi-task`" models can generate better and more robust answers compared to the baseline and the "`prompt`" model.

## 6.5 Conclusions

This study aimed to explore different tuning approaches for response generation in conversational question answering. Specifically, we experimented with the effectiveness of prompt tuning, instruction tuning, and multi-task learning on GPT-2, under both decoding mode and evaluation mode. The F1 results demonstrated that prompt-based tuning outperformed the baseline, while models with instruction-based tuning and multi-task learning yielded slightly better results than those with prompt-based tuning alone.

## Acknowledgements

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). This paper also received travel funds from the Norwegian Research Center for AI Innovation (NorwAI). We also received assistance on computing resources from the IDUN cluster of NTNU (Själander et al., 2019). We would like to thank Jon Atle Gulla for his helpful comments.

# Chapter 7

# Discussion and Conclusion

This thesis explored solutions to the problems of balancing multi-domain corpora, awareness of multi-turn context, and knowledge integration under conversational question answering. The solutions are explained in detail below, corresponding to the proposed research questions.

## 7.1 Answers to Research Questions

**RQ1** *How can we balance multi-domain training corpora for generation-based conversational agents to improve the relevance of the generated responses?*

We explored three approaches to balance multi-domain training corpora: interleaved learning, labeled learning, and multi-task labeled learning. Additionally, we defined an evaluation metric, Document-specific Frequency (DF), to measure the relevance of the generated responses regarding each specific corpus. We then propose DF-based weighted learning to optimize the performance. A comprehensive explanation of these proposed methods follows below.

***RQ1.1*** *What kind of approaches can be integrated into generation-based conversational agents to balance the training corpora? How do they perform?*

We have examined three approaches for addressing the issue of multi-domain training corpora balance, namely interleaved learning, labeled learning, and multi-task labeled learning. Interleaved learning evenly distributes multiple training corpora to ensure that

conversational agents can learn from them evenly. Labeled learning assigns each corpus a corpus embedding, which is learned by the model alongside the generation task throughout the training process. The corpus embedding of each question's source corpus is provided to the model to support response generation. Multi-task labeled learning improves upon labeled learning by introducing an additional corpus classifier that predicts the source corpus of each question. This allows a question with an unknown source corpus, which labeled learning cannot handle, to be assigned a corpus embedding of a known source corpus. These approaches have all achieved better performance on the F1 score compared to the base models. Notably, labeled learning achieves better performance than interleaved learning on the technical corpus (the Ubuntu corpus) in terms of F1 score. Conversely, multi-task labeled learning is expected to exhibit worse performance relative to labeled learning.

***RQ1.2*** *How do we evaluate the relevance of the generated responses corresponding to different corpora?*

Given that the F1 score is not a suitable evaluation metric for assessing the relevance of generated responses, we have proposed Document-specific Frequency (DF) as an alternative evaluation metric. This metric measures the importance of a word within a given corpus by comparing the occurrence of the word in other corpora as well as its occurrence relative to other words within the same corpus, and assigns an importance score to each word regarding each corpus. The comparison between DF and TFIDF has demonstrated the efficiency of DF. Specifically, DF evaluates the relevance of a generated response to a corpus by computing the average importance score of the response with respect to the specific corpus and the average importance score of the response with respect to some irrelevant corpora. The former should be as high as possible, indicating that the generated response employs more words from the relevant corpus, while the latter should be as low as possible, indicating that the response uses fewer words from irrelevant corpora.

***RQ1.3*** *How can we optimize the relevance of the generated responses for a generation-based conversational agent based on the proposed evaluation metric?*

To optimize the relevance of generated responses using DF, we have proposed weighted learning, where we utilize the DF score of each word to weight the loss function for the generation task during the training process. This weighting mechanism is not applied during the decoding process. Our experiments have shown that weighted learning yields a

significant improvement in performance, as measured by both the F1 score and DF, when compared to both the baseline and all the other explored approaches.

**RQ2** *How can we improve the awareness of multi-turn context on generation-based conversational agents?*

We have defined the notion of context attention distribution, which serves as a means to assess the awareness of a multi-turn context for generation-based conversational agents. Specifically, the notion of context attention distribution is based on the attention a model pays to important turns in comparison to unimportant turns within a multi-turn context. To evaluate a model's ability in context attention distribution, we have proposed the distracting test and the Distracting Attention Score (DAS) ratio. We have examined the effectiveness of static and dynamic attention mechanisms, with and without utterance integration, by employing the distracting test and DAS ratio. We have also investigated training strategies for optimizing a model's ability on context attention distribution, which involves the use of self-contained distraction and extractive summarization techniques. A comprehensive explanation of these proposed methods follows below.

***RQ2.1*** *How do we evaluate the context awareness for a generation-based conversational agent?*

We proposed the distracting test and the Distracting Attention Score (DAS) ratio as the evaluation metric for assessing the context awareness of a generation-based conversational agent. The distracting test inserts extraneous distracting utterances into each dialogue, so that to examine the model's ability to maintain attention on the relevant information despite the presence of these distracting utterances. We have specified three distinct settings for the distracting test: random, frequent, and rare. In the random setting, the distracting utterances are drawn from the training corpus, while in the frequent and rare settings, they are formed from frequent and rare words in the training corpus, respectively. After inserting these distracting utterances, we compute the attention scores, which are assigned by each model's attention mechanism, to both the distracting utterances and the original utterances in the current dialogue. The DAS ratio is the quotient of the attention scores on the distracting utterances and original utterances, and serves as a measure of the model's ability to ignore distracting utterances while attending to the original utterances in the dialogue. A low DAS

ratio indicates a higher level of context awareness on the model.

***RQ2.2*** *How can we optimize the context awareness for a generation-based conversational agent based on the proposed evaluation metric?*

We introduced a novel training strategy aimed at enhancing the context awareness of generation-based conversational agents, which leverages the proposed distracting test. In the training process, we insert distracting utterances, chosen at random from the training corpus, into each dialogue. We then utilize a bitmask that differentiates between distracting and original utterances, assigning 1 to the former and 0 to the latter, to calculate the attention loss, which is to be minimized along with the loss for the generation task using multi-task learning. This enables the model to focus on the relevant information in the dialogue while ignoring the extraneous distracting utterances. Extensive experiments show the efficiency of the proposed training strategy, where models trained using this method achieve better performance on the DAS ratio in the majority of settings when compared to models trained without this technique.

***RQ2.3*** *How can we integrate summarization techniques into generation-based conversational agents to improve the context awareness of the multi-turn context?*

We explored two extractive summarization techniques, PMI and ORACLE, as means of enhancing the context awareness of generation-based conversational agents in the multi-turn context. To implement these techniques, we utilize the final turn in the context as the query and filter out any turns in the dialogue that are irrelevant to the query. We use the distracting test and DAS ratio to examine the effectiveness of the techniques in improving the context awareness of the models. Our experimental results show that models employing ORACLE to filter out the irrelevant turns perform better on the DAS ratio than those that do not utilize this technique.

**RQ3** *How can we improve the quality of generated responses on knowledge for generation-based conversational agents under multi-turn conversational question answering context?*

We investigated approaches to enhance the generation for the conversational question answering task, where the provided document is viewed as the knowledge to be acquired by

a generation-based conversational agent. Specifically, we examined the integration of extractive question answering through multi-task learning and prompt-based tuning, in addition to employing instruction-based tuning. To evaluate the effectiveness of these methods, we measured their performance with the F1 score. A detailed explanation of our proposed approaches is presented below.

***RQ3.1*** *How can answer spans from the extractive question answering task be integrated into generation-based conversational agents and improve the quality of generated responses on knowledge?*

We explored leveraging answer spans from extractive question answering by utilizing prompt-based tuning and multi-task learning. Prompt-based tuning adds prompts to the document at locations where answer spans occur, which is then learned by the model during the training process. Multi-task learning involves combining the losses from both the extractive question answering and the generation tasks to enable the model to learn from both simultaneously. We experimented with two settings: using only prompt-based tuning and using both prompt-based tuning and multi-task learning. In both cases, we observed significant improvements over in the F1 score compared to the baseline. Notably, the setting incorporating both approaches resulted in a subtle improvement over the setting using prompt-based tuning alone.

***RQ3.2*** *How can prompt-based tuning and instruction-based tuning improve the quality of generated responses on knowledge?*

We employed an instruction-based tuning approach that adds an instruction and several guidances to the document, and experimented with applying this approach both in conjunction with prompt-based tuning and with both prompt-based tuning and multi-task learning. In both cases, we observed significant improvements in the F1 score compared to the baseline. Notably, the setting incorporating instruction-based tuning upon prompt-based tuning resulted in a subtle improvement over the setting using prompt-based tuning alone.

## 7.2 Conclusions

### 7.2.1 Conclusion for Paper 1, Chapter 3

**Paper 1** has answered ***RQ1.2***. In **Paper 1**, to improve the relevance of generated responses in multi-domain conversational agents, we explored four distinct approaches: interleaved learning (serving as our baseline), labeled learning, multi-task labeled learning, and weighted learning. These methods aim to harmonize the influence of multiple training corpora from different domains, ensuring that the generated dialogue is contextually appropriate. To assess the performance of each approach, we employed Rouge metrics, covering precision, recall, and F1 score, as an automated evaluation method. As one of our main contributions, we proposed Domain Frequency (DF) as a metric to gauge the model's effectiveness in using domain-specific vocabulary. Beyond these automated metrics, we conducted an extensive human evaluation. Our findings indicate a notable enhancement in the relevance of the generated responses, with weighted learning emerging as the most effective technique among those tested.

### 7.2.2 Conclusions for Paper 2, Chapter 4

**Paper 2** has answered ***RQ2.1*** and ***RQ2.2***. In **Paper 2**, we have focused on enhancing multi-turn modeling in open-domain conversational agents by studying the role of context attention distribution. To objectively evaluate this component, we introduced an evaluation metric called the DAS ratio, which relies on a distracting test to measure the context attention distribution. Building on this metric, we've developed an optimization strategy aimed at minimizing the attention loss associated with self-contained distracting utterances. Our comprehensive experiments demonstrate that this approach significantly improves the performance of various conversational agent models, most notably when a training inserting probability level of $0.7$ is employed.

### 7.2.3 Conclusions for Paper 3, Chapter 5

**Paper 3** has answered ***RQ2.3***. In **Paper 3**, to enhance the context-awareness of multi-turn conversational agents, we have amalgamated extractive summarization methods, specifically the PMI topic model and the ORACLE algorithm, into both non-hierarchical and hierarchical dialogue models. To assess the effectiveness of this integration, we conducted distracting tests as our evaluation metric. The incorporation of these extractive summarization techniques has led to noteworthy advancements in the context-aware performance of multi-turn conversational agents, as evidenced by the distracting test results.

### 7.2.4 Conclusions for Paper 4, Chapter 6

**Paper 4** has answered ***RQ3***. In **Paper 4**, we investigated various techniques for enhancing the quality of generated responses in conversation-based question answering systems. Our experimentation focused on integrating answer spans from extractive question answering tasks and generation-based conversational agents to improve knowledge-based response quality. Furthermore, we evaluated the impact of prompt-based tuning, instruction-based tuning, and multi-task learning strategies on GPT-2 in both decoding and evaluation modes. Our findings indicated that prompt-based tuning surpassed the baseline in F1 scores, and models employing instruction-based tuning or multi-task learning showed an improvement over those using prompt-based tuning alone.

### 7.2.5 Reflections

This thesis made significant strides in addressing some key challenges in generation-based conversational agents. It addresses three pivotal challenges in the domain of generation-based conversational systems: 1) Balancing multi-domain training corpora to improve the relevance of generated responses, 2) Enhancing the model's context awareness in multi-turn conversations, and 3) Integrating knowledge through combining extractive question-answering and prompt-based and instruction-based tuning techniques. These challenges

are deeply rooted in real-world conversational systems. The proposed methods for balancing multi-domain corpora can be employed to improve the relevance of generated responses across a wide array of topics. Context awareness is indispensable for maintaining a coherent and meaningful dialogue, especially in multi-turn settings. The integration of question-answering methods can significantly improve the depth and quality of conversational agents' responses. Furthermore, these challenges are interconnected and can be combined to create a holistic solution. For instance, a balanced multi-domain corpus can be applied in conjunction with context awareness techniques and knowledge integration methods to produce a conversational agent that is not only relevant but also contextually aware and knowledge-rich. In all, we have managed to focus on specific, realistic challenges faced by conversational systems and offer substantial contributions to research, where these challenges are not only realistic but also interrelated.

**Balancing Multi-Domain Training Corpora** Open-domain conversational agents often struggle with relevancy across different domains. This thesis has provided methodologies like weighted learning to balance the training corpora and "Document-specific Frequency" to evaluate it. This is critical for an open-domain system where the range of possible conversations spans multiple subjects.

**Improving Multi-Turn Context Awareness** In real-world conversational scenarios, the conversational agent must be adapted to understanding the context of a conversation, which could be ever-changing and complex. This thesis introduced the notion of "context attention distribution" and devised a distracting test and the DAS ratio to measure this.

**Integrating Knowledge** This thesis also focused on improving the quality of answers using a document as the basis for knowledge. By combining extractive question answering, multi-task learning, and prompt-based and instruction-based tuning, we attempt to make the generation-based conversational agent more informative in multi-turn conversations.

Our research offers a strong indication of the potential superiority of generation-based systems over rule-based and information-retrieval-based systems. The use of metrics like

F1 score and DF, as well as the focus on context understanding and knowledge integration, are all factors that can't be easily handled by rule-based systems. Rule-based systems are generally not capable of understanding context or balancing multi-domain corpora, as they are generally designed to follow a set script. Information-retrieval-based systems, while better at pulling information, may lack the finesse required to generate conversational, relevant, and context-aware responses. The generation-based systems, particularly those trained with the aforementioned methodologies, show promise in being able to understand, learn, and adapt to a broader range of conversational cues and contexts.

This thesis deals with a critically important topic. As conversational agents become an integral part of various industries, the challenges addressed in the paper become increasingly relevant. The thesis has not only stated the problems but also proposed measurable solutions, which have been rigorously evaluated using novel metrics. This shows a methodical approach to research and contributes to the credibility of the findings. Overall, this thesis makes valuable contributions to the field of open-domain conversational systems by tackling three pertinent challenges through innovative methods.

### 7.2.6 Future Works

This thesis has proposed novel approaches and evaluation metrics to improve the performance of generation-based conversational agents, particularly by addressing the challenges of balancing the training corpora, handling multi-turn context, and integrating external knowledge within a question answering context. Despite these contributions, there are still some limitations that need to be highlighted for future works.

- This study has laid the groundwork for improving the agent's ability to generate contextually relevant responses through multi-domain training corpora, multi-turn context, and knowledge incorporation. As a natural progression, future studies have the exciting opportunity to delve into unexplored but crucial areas like emotion, personality, and hallucination management to create even more versatile conversational agents.

- The current study is rooted in existing base models, offering valuable insights into their performance enhancements. As the field rapidly evolves with the introduction

of cutting-edge base models like Alpaca (Taori et al., 2023), future work has the exciting prospect of evaluating and potentially extending the thesis' approaches on these newer, high-performance architectures.

- The current work has shown promising results using approaches such as multi-task learning, loss weighting, and prompt-based or instruction-based tuning. These findings open the door for future studies to investigate other compelling techniques, such as reinforcement learning, for further performance optimization, particularly with the advent of new and more capable base models.

# Appendix A

# Appendix of Chapter 3

## A.1 Comparison among TF-IDF, DF and $\alpha$F for 4 corpora on more example words

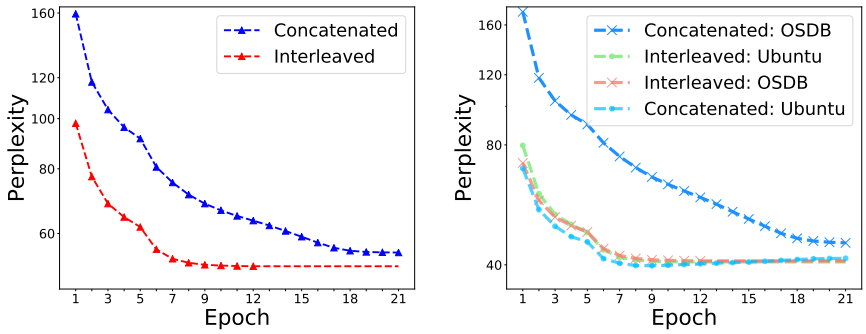
Example words are divided into five blocks. The first block has frequent words in all corpora, the second block has unique words from OSDB, the third block has unique words from Twitter, the fourth block has unique words from Ubuntu, and the fifth block has unique words from PersonaChat. The values of the corresponding corpus are marked with different colors.

From Table A.1, it is clear that the commonly used word importance weight, TF-IDF, is not suitable for our task. This is due to the vast range of frequency, which leads to a relatively small penalty for IDF (Inversed Document Frequency) over words with too large TF (Term Frequency).

| Word | TF-IDF(%) | | | | DF(%) | | | | $\alpha$DF$_{(\alpha=100)}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OSDB | Twitter | Ubuntu | PersonaChat | OSDB | Twitter | Ubuntu | PersonaChat | OSDB | Twitter | Ubuntu | PersonaChat |
| i | 91.39 | 100.00 | 100.00 | 62.63 | 21.40 | 15.68 | 20.80 | 42.12 | 2.62 | 2.01 | 2.59 | 7.32 |
| to | 54.46 | 77.55 | 64.59 | 32.80 | 24.85 | 23.40 | 26.87 | 24.89 | 3.00 | 2.88 | 3.76 | 3.08 |
| it | 61.77 | 74.10 | 83.20 | 21.74 | 25.02 | 22.02 | 38.49 | 14.46 | 3.44 | 2.67 | 5.11 | 2.13 |
| sword | 0.64 | 0.17 | 0.01 | 0.08 | 68.37 | 13.74 | 0.26 | 17.63 | 63.29 | 1.37 | 1.00 | 1.15 |
| forgive | 2.41 | 0.48 | 0.16 | 0.06 | 75.35 | 14.37 | 5.44 | 4.84 | 50.96 | 1.58 | 1.19 | 1.05 |
| hurry | 5.21 | 0.52 | 0.09 | 0.08 | 88.39 | 6.67 | 1.48 | 3.45 | 63.53 | 1.32 | 1.15 | 1.04 |
| darling | 2.54 | 0.39 | 0.00 | 0.01 | 90.88 | 8.42 | 0.11 | 0.58 | 57.10 | 1.45 | 0 | 1.21 |
| explain | 1.27 | 0.00 | 0.00 | 0.11 | 91.33 | 0 | 0 | 8.67 | 94.14 | 0 | 0 | 1.06 |
| tax | 0.21 | 2.52 | 0.05 | 0.09 | 6.77 | 87.06 | 1.09 | 5.07 | 1.28 | 71.26 | 1.05 | 1.04 |
| liberal | 0.03 | 1.71 | 0.01 | 0.10 | 2.06 | 88.19 | 0.25 | 9.50 | 1.21 | 59.65 | 0 | 1.38 |
| vote | 0.41 | 6.08 | 0.10 | 0.11 | 6.07 | 90.68 | 0.78 | 2.47 | 1.12 | 80.22 | 1.02 | 1.09 |
| trump | 0.04 | 18.66 | 0.00 | 0.13 | 0.11 | 99.16 | 0.00 | 0.73 | 1.00 | 96.63 | 0 | 1.03 |
| hillary | 0.05 | 8.61 | 0.00 | 0.01 | 0.42 | 99.53 | 0 | 0.05 | 0 | 99.38 | 0 | 1.01 |
| laptop | 0.10 | 0.40 | 5.39 | 0.15 | 1.33 | 4.37 | 89.88 | 4.42 | 1.07 | 1.22 | 76.02 | 1.01 |
| upgrade | 0.03 | 0.47 | 6.85 | 0.03 | 0.24 | 3.75 | 95.63 | 0.37 | 1.01 | 1.06 | 91.24 | 1.03 |
| file | 0.64 | 0.55 | 15.65 | 0.05 | 2.29 | 1.44 | 96.02 | 0.26 | 1.11 | 1.04 | 86.36 | 0 |
| windows | 0.33 | 0.44 | 12.18 | 0.06 | 1.09 | 1.37 | 97.13 | 0.41 | 1.04 | 1.10 | 86.33 | 1.01 |
| ubuntu | 0.00 | 0.01 | 27.47 | 0.00 | 0 | 0.01 | 99.99 | 0 | 0 | 1.01 | 99.48 | 0 |
| music | 1.90 | 3.29 | 1.53 | 7.66 | 4.01 | 8.20 | 4.84 | 82.94 | 1.18 | 1.40 | 1.23 | 49.14 |
| teacher | 1.48 | 0.74 | 0.07 | 2.20 | 14.53 | 7.01 | 0.68 | 77.78 | 1.39 | 1.32 | 1.01 | 53.49 |
| travel | 0.42 | 0.91 | 0.05 | 3.07 | 3.91 | 6.89 | 0.28 | 88.92 | 1.27 | 1.36 | 1.01 | 57.15 |
| hobby | 0.10 | 0.27 | 0.04 | 1.56 | 1.94 | 3.03 | 0.57 | 94.46 | 1.13 | 1.00 | 1.09 | 81.71 |
| hiking | 0.03 | 0.09 | 0.00 | 1.52 | 0.85 | 1.45 | 0 | 97.70 | 0 | 1.09 | 0 | 91.76 |

Table A.1: Normalized TF-IDF (%), DF (%) and $\alpha$DF of more example words for 4 corpora

## A.2 Convergence time of pre-training LSTM model on large-scale corpora




(a) Overall perplexity of 3 corpora (OpenSubtitles, Twitter, Ubuntu) per epoch

(b) Perplexity of OSDB corpus and Ubuntu corpus per epoch

Figure A.1: Convergence time of pre-training LSTM on large-scale corpora

Table A.2 shows the convergence time of pertaining LSTM on large-scale corpora. In the pre-training period, it takes 21 epochs for the concatenated corpus to converge on the base LSTM model, while only 12 epochs with interleaved learning, which is $43\%$ shorter. When trained on the concatenated corpus in the order of OSDB $\rightarrow$ Twitter $\rightarrow$ Ubuntu, it takes 20 epochs for the perplexity on OSDB and Ubuntu to be balanced, while with interleaved learning, it takes less than one epoch. For concatenated corpus, the performance of the Ubuntu corpus is sacrificed in order to balance the performance of the two corpora, which results in worse overall performance.

## A.3 Results of automatic evaluation with stop words

| Model | Corpus / Method | Test Set | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | | Twitter | | | Ubuntu | | | PersonaChat | | | Overall | | |
| | | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 |
| LSTM | PersonaChat (single) | 109.8 | 4.8 | 6.5 | 191.9 | 5.4 | 6.3 | 116.9 | 4.8 | 6.8 | 28.9 | 13.1 | 15.0 | 47.0 | 7.0 | 8.7 |
| | Concatenated | 57.0 | **4.8** | 6.3 | 111.4 | 5.9 | 6.1 | 50.0 | 5.1 | 6.8 | 27.8 | 13.2 | 15.1 | 36.8 | **7.2** | 8.6 |
| | Interleaved | 41.3 | 3.7 | **6.7** | 89.3 | 6.0 | 7.6 | 43.1 | 5.1 | 8.7 | 27.9 | 12.8 | 15.0 | 34.3 | 6.9 | 9.5 |
| | Labeled | **40.5** | 3.2 | 6.6 | **87.0** | **6.2** | 7.6 | **42.6** | **5.3** | **8.8** | **27.1** | **13.2** | **15.2** | **33.4** | 7.0 | **9.6** |
| | Multi-task Labeled | 41.7 | 3.5 | 6.6 | 89.7 | 6.1 | **7.7** | 43.5 | 5.0 | 8.6 | 27.8 | 12.6 | 14.8 | 34.3 | 6.8 | 9.4 |
| | Weighted | 46.1 | 3.6 | 6.6 | 102.5 | 4.6 | 6.7 | 49.4 | 3.8 | 6.6 | 32.8 | 11.4 | 15.0 | 39.9 | 5.8 | 8.7 |
| GPT-2 | PersonaChat (single) | 478.8 | 4.9 | 6.7 | 159.6 | 5.5 | 6.7 | 264.7 | 5.1 | 7.7 | 19.6 | 14.1 | 16.2 | 44.7 | 7.3 | 9.3 |
| | Concatenated | 392.8 | **5.0** | 6.9 | 110.7 | 5.8 | 7.0 | 199.2 | 5.8 | 8.5 | 19.0 | 13.9 | 16.0 | 40.1 | **7.6** | 9.6 |
| | Interleaved | 26.6 | 4.3 | 7.4 | 54.8 | 5.8 | 7.4 | 28.1 | 5.7 | 9.2 | 19.2 | 14.0 | 16.1 | 23.7 | 7.4 | 10.0 |
| | Labeled | **26.5** | 4.2 | 7.3 | **54.1** | **5.9** | **7.6** | **27.7** | 5.7 | 9.2 | **18.9** | **14.1** | **16.3** | **23.5** | 7.5 | **10.1** |
| | Multi-task Labeled | 26.9 | 4.1 | 7.2 | 55.4 | 5.8 | 7.5 | 38.5 | **5.8** | **9.4** | 20.7 | 14.0 | 16.1 | 25.1 | 7.4 | 10.1 |
| | Weighted | 29.6 | 4.3 | **7.5** | 64.1 | 5.1 | 7.4 | 44.1 | 4.1 | 7.0 | 23.4 | 13.0 | 15.7 | 28.4 | 6.6 | 9.4 |

Table A.2: Perplexity, BLEU (%) and F1 (%) scores for baselines and proposed methods fine-tuned on 4 corpora (**with** stop words). BLEU is from NLTK sentence BLEU

Models of labeled, multi-task labeled and weighted learning do not have the best hyper-parameters, but the same hyper-parameters as the base models. Their perplexity is slightly worse than it should be.

The results of the single corpus PersonaChat trained with the LSTM model confirm our concern on a small fine-tuning corpus. The LSTM model is pre-trained on OSDB, Twitter and Ubuntu; however, the performance for the 3 corpora greatly decreases after fine-tuning.

The automatic evaluation with stop words is not good for measuring relevance, since stop words are taken too much into account. See BLEU and F1 scores of PersonChat (single) and weighted learning as an example in Table A.2. Models trained on PersonaChat (single) cannot answer Ubuntu technical questions **at all**, yet they receive better scores than weighted learning. But once the stop words are removed, the scores of weighted learning surplus PersonaChat (single) a lot.

## A.4 Additional Results of automatic evaluation without stop words

| Model | Corpus / Method | Test Set | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | | Twitter | | | Ubuntu | | | PersonaChat | | | Overall | | |
| | | BLEU | ROUGE | DF-F1 | BLEU | ROUGE | DF-F1 | BLEU | ROUGE | F1 | BEU | ROUGE | DF-F1 | BLEU | ROUGE | DF-F1 |
| LSTM | PersonaChat (single) | 5.2 | 8.1 | 6.2 | 5.7 | 8.9 | 5.0 | 4.5 | 7.7 | 4.8 | 34.2 | 45.8 | 44.6 | 12.4 | 17.6 | 15.2 |
| | Concatenated | 4.5 | 7.2 | 5.6 | 7.4 | 11.4 | 8.8 | 11.6 | 20.1 | 17.4 | **34.6** | 46.4 | 44.2 | 14.5 | 21.3 | 19.0 |
| | Interleaved | 6.5 | 11.7 | 9.9 | 8.6 | 14.9 | 12.6 | 17.1 | 29.6 | 28.4 | 32.4 | 44.3 | 43.2 | 16.1 | 25.1 | 23.5 |
| | Labeled | 6.2 | 11.3 | 9.7 | **9.1** | 15.5 | 12.6 | 18.1 | 31.6 | 30.7 | 33.5 | 45.4 | 43.8 | 16.7 | 26.0 | 24.2 |
| | Multi-task Labeled | 6.2 | 11.1 | 9.5 | 8.4 | 14.5 | 11.7 | 16.0 | 28.3 | 27.2 | 31.5 | 42.7 | 41.9 | 15.5 | 24.2 | 22.6 |
| | Weighted | **7.6** | **13.4** | **12.2** | 7.6 | **15.6** | **18.7** | **24.2** | **41.2** | **44.1** | 33.2 | **47.1** | **46.9** | **18.2** | **29.3** | **30.5** |
| GPT-2 | PersonaChat (single) | 7.1 | 10.8 | 9.2 | 8.7 | 13.9 | 10.5 | 8.8 | 15.5 | 12.2 | 45.0 | 58.8 | 56.8 | 17.4 | 24.7 | 22.2 |
| | Concatenated | 8.4 | 12.6 | 11.0 | 10.8 | 17.2 | 13.7 | 13.4 | 22.4 | 23.3 | 43.0 | 56.3 | 55.7 | 18.9 | 27.1 | 25.9 |
| | Interleaved | 14.0 | 22.3 | 21.3 | 12.2 | 20.1 | 19.3 | 25.8 | 44.3 | 48.3 | 44.2 | 57.6 | 58.0 | 24.0 | 36.1 | 36.7 |
| | Labeled | 13.6 | 21.6 | 20.5 | **13.1** | 21.1 | 20.3 | 25.8 | 45.0 | 49.6 | **45.1** | **59.0** | **59.6** | 24.4 | 36.7 | 37.5 |
| | Multi-task Labeled | 13.4 | 21.4 | 20.4 | 12.7 | 20.6 | 20.1 | 25.4 | 43.8 | 47.6 | 44.0 | 57.3 | 57.4 | 23.9 | 35.8 | 36.4 |
| | Weighted | **14.5** | **23.4** | **23.4** | 11.9 | **22.3** | **25.2** | **29.2** | **48.6** | **52.5** | 42.4 | 55.8 | 57.6 | **24.5** | **37.5** | **39.7** |

Table A.3: BLEU (‰), ROUGE (‰) and DF-F1 (‰) scores for baselines and proposed methods fine-tuned on 4 corpora (**without** stop words). DF-F1 is ROUGE F1 weighted by test-set $\alpha$DF

## A.5 Full results of $\alpha$F for generated responses from multiple corpora

| Model | Corpus / Method | Test Set: OSDB | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | **7.01** | **9.66** | 3.75 | 3.75 | 2.82 | 2.86 | 3.59 | 3.75 |
| LSTM | PersonaChat (single) | 2.92 | 3.40 | 2.40 | 2.82 | 2.27 | 2.51 | 9.18 | 9.91 |
| | Concatenated | 2.92 | 3.35 | 2.49 | 2.94 | 2.41 | 2.71 | 7.65 | 8.55 |
| | Interleaved | 3.88 | 4.13 | 2.45 | 2.54 | 2.89 | 2.87 | 4.98 | 5.31 |
| | Labeled | 3.94 | 4.16 | 2.37 | 2.44 | 2.71 | 2.70 | 5.01 | 5.34 |
| | Multi-task Labeled | 3.78 | 4.02 | 2.41 | 2.49 | 2.91 | 2.88 | 5.02 | 5.36 |
| | Weighted | **5.60** | **6.29** | 2.65 | 2.84 | 2.89 | 2.84 | 4.14 | 4.47 |
| GPT-2 | PersonaChat (single) | 2.76 | 3.15 | 2.30 | 2.66 | 2.24 | 2.51 | 10.53 | 11.09 |
| | Concatenated | 3.07 | 3.59 | 2.52 | 2.96 | 2.30 | 2.55 | 8.75 | 9.35 |
| | Interleaved | 4.86 | 5.78 | 2.63 | 2.67 | 2.69 | 2.66 | 4.77 | 5.04 |
| | Labeled | 4.86 | 5.77 | 2.61 | 2.66 | 2.67 | 2.64 | 4.76 | 5.04 |
| | Multi-task Labeled | 4.81 | 5.70 | 2.60 | 2.64 | 2.69 | 2.65 | 4.83 | 5.1 |
| | Weighted | **6.02** | **7.46** | 2.71 | 2.83 | 2.47 | 2.48 | 4.12 | 4.38 |

(a) $\alpha\text{DF}_d$ scores for generated responses from OSDB

| Model | Corpus / Method | Test Set: Twitter | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 3.97 | 4.07 | **9.07** | **11.01** | 3.24 | 3.40 | 3.64 | 3.80 |
| LSTM | PersonaChat (single) | 2.79 | 3.21 | 2.78 | 3.36 | 2.35 | 2.59 | 8.60 | 9.18 |
| | Concatenated | 2.62 | 3.12 | 3.55 | 4.31 | 2.30 | 2.71 | 7.97 | 8.69 |
| | Interleaved | 3.28 | 3.68 | 4.66 | 4.95 | 3.11 | 3.34 | 4.11 | 4.51 |
| | Labeled | 3.30 | 3.68 | 4.97 | 5.27 | 3.00 | 3.24 | 3.89 | 4.26 |
| | Multi-task Labeled | 3.31 | 3.68 | 4.47 | 4.73 | 3.14 | 3.36 | 4.08 | 4.49 |
| | Weighted | 3.10 | 3.62 | **9.92** | **10.10** | 2.79 | 3.01 | 3.79 | 4.30 |
| GPT-2 | PersonaChat (single) | 2.74 | 3.04 | 2.87 | 3.33 | 2.45 | 2.66 | 9.47 | 9.77 |
| | Concatenated | 2.87 | 3.28 | 3.32 | 3.94 | 2.41 | 2.65 | 8.21 | 8.68 |
| | Interleaved | 3.42 | 3.67 | 4.59 | 5.08 | 3.05 | 3.13 | 4.39 | 4.68 |
| | Labeled | 3.48 | 3.74 | 4.66 | 5.16 | 3.08 | 3.19 | 4.06 | 4.35 |
| | Multi-task Labeled | 3.41 | 3.66 | 4.63 | 5.11 | 3.08 | 3.15 | 4.37 | 4.65 |
| | Weighted | 3.58 | 4.01 | **8.13** | **8.84** | 2.59 | 2.79 | 3.68 | 4.07 |

(b) $\alpha\text{DF}_d$ scores for generated responses from Twitter

| Model | Corpus / Method | Test Set: Ubuntu | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 2.69 | 2.74 | 2.96 | 2.85 | **19.36** | **23.20** | 2.67 | 2.78 |
| LSTM | PersonaChat (single) | 2.71 | 3.28 | 2.41 | 2.89 | 2.74 | 3.06 | 8.55 | 9.09 |
| | Concatenated | 2.61 | 2.89 | 2.27 | 2.53 | 7.60 | 7.74 | 5.59 | 5.99 |
| | Interleaved | 2.91 | 3.19 | 2.30 | 2.36 | 11.78 | 11.27 | 3.70 | 4.01 |
| | Labeled | 3.03 | 3.38 | 2.28 | 2.36 | 12.46 | 11.75 | 3.45 | 3.75 |
| | Multi-task Labeled | 2.91 | 3.17 | 2.30 | 2.35 | 11.19 | 10.72 | 3.77 | 4.09 |
| | Weighted | 2.16 | 2.84 | 2.05 | 2.16 | **27.73** | **25.42** | 2.68 | 3.01 |
| GPT-2 | PersonaChat (single) | 2.60 | 2.85 | 2.31 | 2.64 | 4.12 | 4.64 | 8.27 | 8.42 |
| | Concatenated | 2.67 | 3.03 | 2.45 | 2.82 | 6.54 | 7.10 | 7.04 | 7.37 |
| | Interleaved | 2.73 | 3.05 | 2.22 | 2.37 | 15.67 | 16.02 | 3.08 | 3.41 |
| | Labeled | 2.68 | 3.03 | 2.17 | 2.35 | 16.73 | 17.02 | 2.90 | 3.24 |
| | Multi-task Labeled | 2.73 | 3.06 | 2.22 | 2.37 | 15.45 | 15.78 | 3.12 | 3.44 |
| | Weighted | 2.26 | 2.56 | 2.16 | 2.28 | **25.73** | **24.42** | 2.37 | 2.60 |

(c) $\alpha\text{DF}_d$ scores for generated responses from Ubuntu

| Model | Corpus / Method | Test Set: PersonaChat | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 3.32 | 3.23 | 3.18 | 3.04 | 2.67 | 2.69 | **9.45** | **12.00** |
| LSTM | PersonaChat (single) | 2.59 | 3.02 | 2.31 | 2.73 | 2.15 | 2.35 | 11.86 | 12.62 |
| | Concatenated | 2.47 | 2.84 | 2.29 | 2.76 | 2.06 | 2.33 | 12.52 | 13.61 |
| | Interleaved | 2.57 | 2.92 | 2.30 | 2.71 | 2.17 | 2.45 | 11.48 | 12.52 |
| | Labeled | 2.51 | 2.88 | 2.27 | 2.68 | 2.08 | 2.36 | 12.06 | 13.11 |
| | Multi-task Labeled | 2.55 | 2.91 | 2.29 | 2.74 | 2.15 | 2.43 | 11.45 | 12.59 |
| | Weighted | 2.21 | 2.44 | 2.13 | 2.41 | 2.04 | 2.18 | **17.65** | **18.31** |
| GPT-2 | PersonaChat (single) | 2.54 | 2.79 | 2.28 | 2.59 | 2.12 | 2.34 | 12.85 | 13.74 |
| | Concatenated | 2.58 | 2.99 | 2.41 | 2.79 | 2.16 | 2.39 | 12.08 | 12.99 |
| | Interleaved | 2.64 | 2.89 | 2.37 | 2.65 | 2.23 | 2.42 | 12.13 | 12.87 |
| | Labeled | 2.57 | 2.84 | 2.32 | 2.62 | 2.16 | 2.37 | 12.37 | 13.10 |
| | Multi-task Labeled | 2.65 | 2.90 | 2.37 | 2.65 | 2.22 | 2.42 | 12.14 | 12.86 |
| | Weighted | 2.39 | 2.63 | 2.27 | 2.52 | 2.02 | 2.17 | **15.96** | **17.07** |

(d) $\alpha\text{DF}_d$ scores for generated responses from PersonaChat

Table A.4: Full results of $\alpha\text{DF}_d$ scores for generated responses from multiple corpora

## A.6 Example of human evaluation system

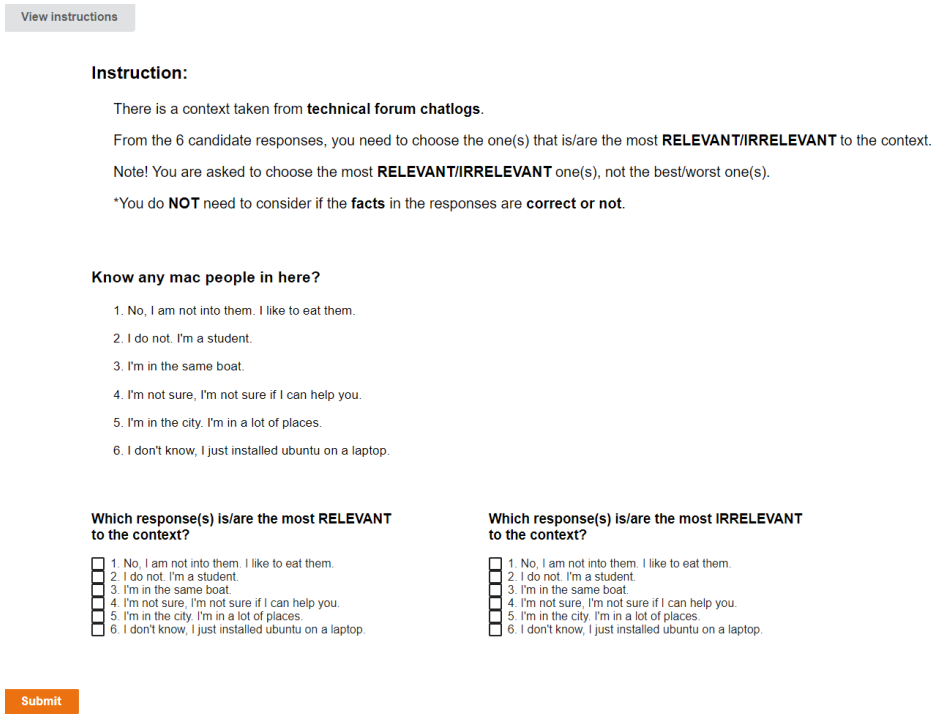



Figure A.2: Human evaluation system for Ubuntu contexts

## A.7 Examples of generated responses

| | |
|---|---|
| **OSDB Context:** You know the main reason parents kill kids? | **Ubuntu Context:** What are you going to do on the remote system exactly? |
| **PersonaChat (single):** I am not sure. I am a vegan.<br>**Concatenated:** I know. I've a daughter that just turned 18. | **PersonaChat (single):** I am going to be a pilot. I am going to fly planes.<br>**Concatenated:** I am going to go to the beach. |
| **Interleaved:** Because they're afraid of what they're gonna do.<br>**Labeled:** Because they're not going to be able to handle it.<br>**Multi-Task Labeled:** Because they're afraid of what they're gonna do.<br>**Weighted:** Because they're afraid of the consequences. | **Interleaved:** I'm going to install the latest version of Ubuntu.<br>**Labeled:** I'm going to install a new kernel.<br>**Multi-Task Labeled:** I'm going to install the latest version of Ubuntu.<br>**Weighted:** I'm trying to connect to the server. |
| **OSDB Context:** The big banks will fail, triggering a domino effect. | **Ubuntu Context:** Know any mac people in here? |
| **PersonaChat (single):** I am not sure yet. I am sure they will fail.<br>**Concatenated:** I'm not sure. I'm not sure what the big banks are doing.<br>**Interleaved:** What? | **PersonaChat (single):** No, I am not into them. I like to eat them.<br>**Concatenated:** I do not. I'm a student.<br>**Interleaved:** I'm in the same boat. |
| **Labeled:** I'm not sure what you're talking about.<br>**Multi-Task Labeled:**I'm not sure what you mean.<br>**Weighted:** What's going on? | **Labeled:** I'm not sure, I'm not sure if I can help you.<br>**Multi-Task Labeled:** I'm in the city. I'm in a lot of places.<br>**Weighted:** I don't know, I just installed ubuntu on a laptop. |
| **OSDB Context:** Rigor mortis has yet to set in. His core body temperature remains at 98 degrees. His skin color, lividity is healthy. | **Ubuntu Context:** Type sudo name of application (in short) |
| **PersonaChat (single):**I'm not sure if I like him or not. I like to eat meat.<br>**Concatenated:**That is good. I like to cook. I am a chef.<br>**Interleaved:** He is a very good doctor. | **PersonaChat (single):** I like to play games on my computer.<br>**Concatenated:** I did not see that. I am a programmer.<br>**Interleaved:** I have a root shell, but I can't login. |
| **Labeled:** He's a good guy.<br>**Multi-Task Labeled:** He is a man of great character.<br>**Weighted:** He's been in the hospital for two weeks. | **Labeled:** I have a root shell.<br>**Multi-Task Labeled:** I don't know what that is.<br>**Weighted:** It is Sudo apt-get install xubuntu-desktop. |

Table A.5: Responses generated from GPT-2 fine-tuned on OSDB and Ubuntu dataset with multiple methods

| | |
|---|---|
| **Twitter Context:** Hillary ran away from rain today. Trump is back on stage minutes after assassination attempt. | **PersonaChat Context:** That must be hard. I like a beer after driving my nissan pathfinder. |
| **PersonaChat (single):** I'm not sure what that means.<br>**Concatenated:** That is so sad. I hope she gets better soon.<br>**Interleaved:** I'm so sorry. I'm so sorry. | **PersonaChat (single):** I like to go to the movies. I have been to a few.<br>**Concatenated:** I am a huge gamer, I play dungeons and dragons on my computer.<br>**Interleaved:** I like to drive my nissan camry. |
| **Labeled:** I'm so sorry.<br>**Multi-Task Labeled:** I'm not sure what happened to her.<br>**Weighted:** She's a liar. | **Labeled:** I like to go to the movies with my friends.<br>**Multi-Task Labeled:** I like to drive my prius.<br>**Weighted:** I love beer too. I love to drink it while I am driving. |
| **Twitter Context:** It was gruesome & extremely rare to have two witnesses. It was linked to epstein. Media didn't report it. | **PersonaChat Context:** A food bank near my house. Do you ever volunteer? |
| **PersonaChat (single):** I'm sorry to hear that. I'm a nurse.<br>**Concatenated:** I'm not sure what that means. I'm a big fan of the show.<br>**Interleaved:** I'm not sure what happened to the epstein case. | **PersonaChat (single):** I do, I love to volunteer at the soup kitchen.<br>**Concatenated:** Yes I do. I love to help people.<br>**Interleaved:** I do. I have a lot of friends. |
| **Labeled:** I'm not sure what the link was, but it was a link to a video.<br>**Multi-Task Labeled:** I'm not sure what happened to the epstein case.<br>**Weighted:** The media is biased and biased. | **Labeled:** I do. I am a volunteer at the local soup kitchen.<br>**Multi-Task Labeled:** I do. I love it. I volunteer at the animal shelter.<br>**Weighted:** I do. I love volunteering at the food pantry. |
| **Twitter Context:** This is 100% true, but can we talk about it tomorrow and make argentina the focus today? Something needs to stick. | **PersonaChat Context:** Yes! My son is in junior high and I just started letting him watch them too. |
| **PersonaChat (single):** Yes, I think so. I'm going to go to the gym tomorrow.<br>**Concatenated:** I'm going to go see a movie.<br>**Interleaved:** I'm not sure what that's. I'm not sure what that's. | **PersonaChat (single):** I am a mom of two boys. I have a son.<br>**Concatenated:** I am a mom, I have three kids.<br>**Interleaved:** I love kids, I have a son who is in high school. |
| **Labeled:** I'm not sure what that means. I'm not sure what that means.<br>**Multi-Task Labeled:** I'm not sure what you mean by that. I'm not sure what you mean by that.<br>**Weighted:** I'm not sure if I can talk about it tomorrow. | **Labeled:** I bet he is a good kid. I am a teacher.<br>**Multi-Task Labeled:** I am sorry to hear that. I am a teacher.<br>**Weighted:** I bet you are a good mom. |

Table A.5: Responses generated from GPT-2 fine-tuned on Twitter and PersonaChat dataset with multiple methods

# Appendix B

# Appendix of Chapter 4

## B.1 Full results of distracting tests

| Model | | Original | | Distracting Test Set | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Random 0.5 | | | Random 0.7 | | | Random 1.0 | | |
| Probability | Structure | Perp. | Avg. | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. |
| 0.0 | Non-hier | 43.2 | 91.3 | 0.93 | 86.8 | 93.1 | 0.93 | 87.1 | 93.8 | 0.93 | 87.6 | 94.5 |
| | Static | 44.1 | 61.4 | 0.82 | 52.6 | 64.5 | 0.82 | 53.5 | 65.6 | 0.79 | 53.4 | 67.6 |
| | StaticUI | 44.6 | 57.5 | 0.79 | 47.4 | 60.2 | 0.76 | 46.3 | 61.3 | 0.76 | 47.7 | 62.7 |
| | Dynamic | 45.4 | 81.4 | 0.89 | 74.9 | 84.4 | 0.89 | 75.6 | 85.2 | 0.88 | 76.2 | 86.6 |
| | DynamicUI | 44.7 | 91.6 | 0.94 | 87.5 | 93.4 | 0.94 | 88.0 | 93.7 | 0.93 | 88.2 | 94.5 |
| 0.5 | Non-hier | 43.4 | 87.2 | 0.84 | 77.2 | 91.6 | 0.83 | 77.1 | 93.3 | 0.81 | 77.0 | 95.5 |
| | Static | 44.5 | 66.5 | 0.70 | 50.3 | 71.5 | 0.69 | 50.5 | 73.5 | 0.67 | 51.1 | 76.5 |
| | StaticUI | 44.3 | 47.7 | 0.74 | 38.1 | 51.2 | 0.74 | 39.2 | 53.1 | 0.70 | 39.1 | 55.5 |
| | Dynamic | 44.6 | 81.9 | 0.79 | 68.3 | 86.6 | 0.78 | 69.1 | 88.2 | 0.77 | 69.4 | 90.8 |
| | DynamicUI | 43.9 | 86.7 | 0.82 | 74.5 | 91.1 | 0.81 | 75.2 | 92.5 | 0.80 | 75.8 | 94.8 |
| | Non-labled StaticUI | 44.7 | 71.1 | 0.73 | 55.5 | 75.6 | 0.73 | 56.6 | 77.1 | 0.72 | 57.4 | 79.6 |
| 0.7 | Non-hier | 43.2 | 86.9 | 0.84 | 76.5 | 91.3 | 0.82 | 75.9 | 93.1 | 0.80 | 75.9 | 95.4 |
| | Static | **44.0** | 57.6 | 0.73 | 45.5 | 62.2 | 0.72 | 45.8 | 64.0 | 0.69 | 46.4 | 66.9 |
| | StaticUI | 44.9 | **43.7** | **0.67** | **32.4** | **48.1** | **0.67** | **33.2** | **49.9** | **0.65** | **34.0** | **52.1** |
| | Dynamic | 44.3 | 82.0 | 0.76 | 66.3 | 87.2 | 0.75 | 66.6 | 89.1 | 0.73 | 67.2 | 91.8 |
| | DynamicUI | 44.8 | 85.3 | 0.93 | 86.8 | 93.1 | 0.93 | 87.1 | 93.8 | 0.93 | 87.6 | 94.5 |
| | Non-labled StaticUI | 44.1 | 55.4 | 0.72 | 43.3 | 59.9 | 0.70 | 43.4 | 62.0 | 0.69 | 44.3 | 64.4 |
| 1.0 | Non-hier | 47.3 | 95.9 | 0.91 | 88.7 | 98.0 | 0.90 | 89.3 | 98.7 | 0.90 | 89.5 | 99.9 |
| | Static | **44.0** | 65.4 | 0.70 | 49.7 | 71.1 | 0.70 | 51.3 | 73.1 | 0.68 | 51.8 | 76.4 |
| | StaticUI | 49.6 | 73.5 | 0.96 | 74.8 | 77.8 | 0.95 | 75.2 | 79.4 | 0.94 | 76.7 | 81.2 |
| | Dynamic | 44.7 | 88.8 | 0.79 | 74.2 | 93.4 | 0.78 | 74.4 | 95.2 | 0.77 | 75.4 | 97.4 |
| | DynamicUI | 45.2 | 90.2 | 0.87 | 81.3 | 93.6 | 0.86 | 81.5 | 94.9 | 0.85 | 81.9 | 96.5 |
| | Non-labled StaticUI | 44.1 | 76.5 | 0.72 | 59.5 | 82.1 | 0.71 | 59.7 | 84.4 | 0.69 | 60.3 | 87.6 |

Table B.1: Results of perplexity (Perp.) and average AS of *History* (Avg.) on the original test set (%) are shown in the "Original" column. Besides, we show the results on the random distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), and average AS of original utterances in *history* (Avg.) (%).

| Model | | Distracting Test Set | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Frequent: Beginning | | | | Frequent: Middle | | | Frequent: End | | | | |
| Probability | Structure | DAS ratio | DAS | Avg. | 1st | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | Last |
| 0.0 | Non-hier | 0.75 | 70.8 | 94.1 | 82.3 | 0.80 | 74.7 | 93.2 | 0.84 | 78.0 | 92.7 | 98.4 |
| | Static | 0.37 | 29.3 | 79.5 | 47.9 | 0.80 | 56.7 | 71.2 | 1.31 | 82.0 | 62.4 | 72.6 |
| | StaticUI | **0.32** | **24.3** | **76.1** | **42.2** | 0.75 | 50.6 | 67.7 | 1.32 | 77.2 | 58.7 | 69.5 |
| | Dynamic | 0.65 | 60.9 | 93.3 | 72.7 | 0.86 | 75.5 | 88.0 | 1.02 | 87.5 | 85.1 | 88.8 |
| | DynamicUI | 0.72 | 72.2 | 100.7 | 81.0 | 0.84 | 81.6 | 97.0 | 0.86 | 84.1 | 98.2 | 98.8 |
| 0.5 | Non-hier | 0.63 | 57.3 | 91.0 | 84.0 | 0.74 | 66.4 | 89.7 | **0.76** | **68.7** | **90.8** | **99.5** |
| | Static | 0.42 | 35.0 | 84.5 | 47.4 | 0.78 | 59.8 | 77.0 | 1.12 | 80.3 | 71.5 | 81.7 |
| | StaticUI | 0.39 | 24.6 | 62.9 | 44.3 | 0.71 | 41.8 | 58.9 | 1.08 | 54.7 | 50.6 | 55.2 |
| | Dynamic | 0.64 | 60.4 | 94.6 | 76.7 | 0.74 | 68.2 | 92.1 | 0.84 | 76.2 | 91.1 | 94.4 |
| | DynamicUI | 0.60 | 60.1 | 100.4 | 77.7 | 0.84 | 78.4 | 92.9 | 0.87 | 82.3 | 94.2 | 93.0 |
| | Non-labled StaticUI | 0.39 | 35.1 | 90.3 | 53.7 | 0.68 | 56.1 | 82.7 | 0.93 | 72.6 | 78.4 | 91.2 |
| 0.7 | Non-hier | 0.72 | 64.7 | 90.1 | 83.1 | 0.82 | 72.6 | 88.8 | 0.82 | 73.4 | 89.4 | 97.1 |
| | Static | 0.40 | 30.3 | 75.2 | 48.3 | 0.70 | 48.7 | 69.3 | 1.08 | 68.1 | 62.9 | 69.2 |
| | StaticUI | 0.36 | 21.0 | 57.6 | 37.4 | **0.66** | **36.0** | **54.7** | 1.02 | 51.7 | 50.6 | 56.4 |
| | Dynamic | 0.58 | 56.3 | 96.8 | 73.9 | 0.71 | 66.4 | 93.4 | 0.86 | 76.0 | 88.8 | 91.8 |
| | DynamicUI | 0.45 | 44.4 | 98.8 | 76.1 | 0.78 | 73.2 | 93.8 | 0.80 | 75.8 | 95.2 | 95.0 |
| | Non-labled StaticUI | 0.45 | 31.7 | 70.6 | 51.0 | 0.70 | 46.6 | 66.6 | 0.98 | 60.2 | 61.2 | 65.8 |
| 1.0 | Non-hier | 0.84 | 82.0 | 97.8 | 92.9 | 0.86 | 83.7 | 97.7 | 0.85 | 83.2 | 97.4 | 100.0 |
| | Static | 0.49 | 40.2 | 82.5 | 60.3 | 0.74 | 57.1 | 76.8 | 1.08 | 73.3 | 67.8 | 72.3 |
| | StaticUI | 0.66 | 73.3 | 110.4 | 24.9 | 0.86 | 71.5 | 82.9 | 1.53 | 104.8 | 68.3 | 88.6 |
| | Dynamic | 0.63 | 64.0 | 102.1 | 81.7 | 0.75 | 73.9 | 98.5 | 0.82 | 79.5 | 97.5 | 99.7 |
| | DynamicUI | 0.73 | 72.5 | 100.0 | 83.5 | 0.81 | 79.2 | 97.4 | 0.83 | 81.6 | 98.7 | 97.6 |
| | Non-labled StaticUI | 0.49 | 46.1 | 95.0 | 67.1 | 0.74 | 65.0 | 87.7 | 0.98 | 79.9 | 81.6 | 86.6 |

Table B.2: Results on the frequent distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), average AS of original utterances in *history* (Avg.) (%), and AS of 1st/last utterance in *history* (%).

| Model | | Distracting Test Set | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Rare: Begin | | | | Rare: Middle | | | Rare: End | | | | |
| Probability | Structure | DAS ratio | DAS | Avg. | 1st | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | Last |
| 0.0 | Non-hier | 0.80 | 74.7 | 93.8 | 82.3 | 0.92 | 84.3 | 92.1 | 1.01 | 92.0 | 91.2 | 96.7 |
| | Static | 0.37 | 29.3 | 79.6 | 47.9 | 0.77 | 55.5 | 72.2 | 1.21 | 77.8 | 64.3 | 75.2 |
| | StaticUI | 0.30 | 22.9 | 76.4 | 42.2 | 0.75 | 51.1 | 67.9 | 1.22 | 74.2 | 61.0 | 73.2 |
| | Dynamic | 0.66 | 61.4 | 93.6 | 72.7 | 0.89 | 77.4 | 87.2 | 1.06 | 87.0 | 82.3 | 85.4 |
| | DynamicUI | 0.73 | 73.8 | 100.5 | 81.0 | 0.93 | 87.9 | 94.2 | 0.97 | 90.1 | 93.0 | 93.0 |
| 0.5 | Non-hier | 0.69 | 62.9 | 90.8 | 84.0 | 0.81 | 72.6 | 89.2 | 0.86 | 77.7 | 90.3 | 98.6 |
| | Static | 0.34 | 29.4 | 86.1 | 47.4 | 0.71 | 55.4 | 78.4 | 0.99 | 72.2 | 72.5 | 82.6 |
| | StaticUI | 0.40 | 24.7 | 62.2 | 44.3 | **0.69** | **40.4** | **58.6** | 0.96 | 53.0 | 55.4 | 60.5 |
| | Dynamic | 0.61 | 58.1 | 95.4 | 76.7 | 0.77 | 70.3 | 91.0 | 0.85 | 76.0 | 89.4 | 92.2 |
| | DynamicUI | 0.61 | 60.9 | 100.5 | 77.7 | 0.80 | 75.5 | 94.9 | 0.83 | 78.6 | 94.4 | 93.6 |
| | Non-labled StaticUI | 0.40 | 36.4 | 90.4 | 53.7 | 0.80 | 64.4 | 80.7 | 1.11 | 82.7 | 74.6 | 87.4 |
| 0.7 | Non-hier | 0.71 | 64.1 | 90.5 | 83.1 | 0.85 | 75.0 | 88.8 | 0.87 | 78.1 | 89.6 | 97.3 |
| | Static | 0.41 | 30.0 | 73.5 | 48.3 | 0.70 | 48.7 | 69.4 | 0.98 | 63.6 | 64.6 | 71.7 |
| | StaticUI | 0.36 | 20.9 | 57.7 | 37.4 | 0.70 | 37.6 | 54.1 | 0.99 | 50.6 | 51.1 | 57.3 |
| | Dynamic | 0.58 | 55.8 | 96.6 | 73.9 | 0.73 | 67.8 | 92.4 | 0.83 | 74.6 | 89.8 | 92.5 |
| | DynamicUI | 0.60 | 59.7 | 98.8 | 76.1 | 0.80 | 74.4 | 93.2 | **0.81** | **75.9** | **93.9** | **93.8** |
| | Non-labled StaticUI | 0.43 | 30.8 | 70.9 | 51.0 | 0.73 | 48.1 | 66.3 | 0.97 | 60.5 | 62.5 | 67.3 |
| 1.0 | Non-hier | 0.85 | 82.8 | 97.8 | 92.9 | 0.87 | 84.8 | 97.5 | 0.88 | 85.6 | 97.3 | 100.2 |
| | Static | 0.46 | 37.5 | 81.6 | 60.3 | 0.71 | 55.0 | 77.4 | 0.88 | 65.2 | 74.4 | 79.8 |
| | StaticUI | **0.21** | **22.4** | **105.6** | **24.9** | 0.86 | 71.0 | 83.0 | 1.50 | 103.1 | 68.7 | 89.0 |
| | Dynamic | 0.65 | 65.6 | 101.4 | 81.7 | 0.77 | 75.1 | 98.1 | 0.82 | 79.6 | 96.9 | 98.5 |
| | DynamicUI | 0.75 | 74.3 | 99.3 | 83.5 | 0.88 | 83.9 | 95.4 | 0.88 | 84.2 | 96.0 | 94.3 |
| | Non-labled StaticUI | 0.49 | 45.5 | 93.7 | 67.1 | 0.77 | 67.4 | 87.3 | 0.98 | 80.6 | 82.5 | 87.4 |

Table B.3: Results on the rare distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), average AS of original utterances in *history* (Avg.) (%), and AS of 1st/last utterance in *history* (%).

# Bibliography

Reina Akama, Kazuaki Inada, Naoya Inoue, Sosuke Kobayashi, and Kentaro Inui. 2017. Generating Stylistically Consistent Dialog Responses with Transfer Learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 408–412, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Suket Arora, Kamaljeet Batra, and Sarabjit Singh. 2013. Dialogue system: A brief review. *arXiv preprint arXiv:1306.4134*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating More Interesting Responses in Neural Conversation Models with Distributional Constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3970–3980. Association for Computational Linguistics.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of Representations for Domain Adaptation. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 137–144. MIT Press.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48. ACM.

Jonathan Bragg, Arman Cohan, Kyle Lo, and Iz Beltagy. 2021. Flex: Unifying evaluation for few-shot nlp. *Advances in Neural Information Processing Systems*, 34:15787–15800.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Elia Bruni and Raquel Fernandez. 2017. Adversarial evaluation for open-domain dialogue generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 284–288. Association for Computational Linguistics.

Rakesh Chada and Pradeep Natarajan. 2021. Fewshotqa: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6081–6090.

Xiusi Chen, Yu Zhang, Jinliang Deng, Jyun-Yu Jiang, and Wei Wang. 2023. Gotta: Generative few-shot question answering by prompt-based cloze data augmentation. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question Answering in Context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184. Association for Computational Linguistics.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An Empirical Comparison of Domain Adaptation Methods for Neural Machine Translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391. Association for Computational Linguistics.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Kenneth Mark Colby, Franklin Dennis Hilf, Sylvia Weber, and Helena C Kraemer. 1972. Turing-like indistinguishability tests for the validation of a computer simulation of paranoid processes. *Artificial Intelligence*, 3:199–221.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Stephan Diederich, Alfred Benedikt Brendel, Stefan Morana, and Lutz Kolbe. 2022. On the design of and interaction with conversational agents: An organizing and assessing review of human-computer interaction research. *Journal of the Association for Information Systems*, 23(1):96–138.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2019. The Second Conversational Intelligence Challenge (ConvAI2). *arXiv:1902.00098 [cs]*.

Zorik Gekhman, Nadav Oved, Orgad Keller, Idan Szpektor, and Roi Reichart. 2022. On the robustness of dialogue history representation in conversational question answering: A comprehensive study and a new prompt-based method. *arXiv preprint arXiv:2206.14796*.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wentau Yih, and Michel Galley. 2018. A Knowledge-Grounded Neural Conversation Model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum

Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*.

Xiaodong Gu, Kang Min Yoo, and Jung-Woo Ha. 2021. DialogBERT: Discourse-aware response generation via learning to recover and rank utterances. In *In Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021)*.

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey Bigham. 2022. InstructDial: Improving zero and few-shot generalization in dialogue through instruction tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 505–525, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Chenyang Huang, Osmar Zaiane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic Dialogue Generation with Expressed Emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 49–54. Association for Computational Linguistics.

Gautier Izacard and Édouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes,

and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Mahesh Joshi, Mark Dredze, William W. Cohen, and Carolyn Rose. 2012. Multi-Domain Learning: When Do Domains Matter? In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1302–1312. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Chris Kedzie, Kathleen R. McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Daniel Khashabi, Chitta Baral, Yejin Choi, and Hannaneh Hajishirzi. 2022. Reframing instructional prompts to gptk's language. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 589–612.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907.

Nate Kornell and Robert Bjork. 2008. Learning concepts and categories is spacing the "Enemy of induction"? *Psychological science*, 19:585–92.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Edda Leopold and Jörg Kindermann. 2002. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46(1):423–444.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Jingyuan Li and Xiao Sun. 2018. A Syntactically Constrained Bidirectional-Asynchronous Approach for Emotional Conversation Generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 678–683, Brussels, Belgium. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003. Association for Computational Linguistics.

Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie Zhou. 2021. Conversations are not flat: Modeling the dynamic information flow across dialogue utterances. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 128–138, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. 2018. OpenSubtitles2018: Statistical Rescoring of Sentence Alignments in Large, Noisy Parallel Corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*. European Language Resource Association.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An Empirical Study of

Unsupervised Evaluation Metrics for Dialogue Response Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132. Association for Computational Linguistics.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1116–1126. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294. Association for Computational Linguistics.

Yi Luan, Chris Brockett, Bill Dolan, Jianfeng Gao, and Michel Galley. 2017. Multi-Task Learning for Speaker-Role Adaptation in Neural Conversation Models. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 605–614. Asian Federation of Natural Language Processing.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A Dialog Research Software Platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84. Association for Computational Linguistics.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. Metaicl: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to Backward and Forward Sequences: A Content-Introducing Approach to Generative Short-Text Conversation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3349–3358. The COLING 2016 Organizing Committee.

Tong Niu and Mohit Bansal. 2018. Polite Dialogue Generation Without Parallel Data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Prasanna Parthasarathi and Joelle Pineau. 2018. Extending Neural Generative Conversational Model using External Knowledge Sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 690–695, Brussels, Belgium. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

Catherine Pelachaud. 2005. Multimodal expressive embodied conversational agents. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 683–689.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. Grips: Gradient-free, edit-based instruction search for prompting large language models. *arXiv preprint arXiv:2203.07281*.

Ravsehaj Singh Puri, Swaroop Mishra, Mihir Parmar, and Chitta Baral. 2022. How many data samples is an additional instruction worth? *arXiv preprint arXiv:2203.09161*.

Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Better Language Models and Their Implications. Technical report, Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3066–3079.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2018. CoQA: A Conversational Question Answering Challenge. *arXiv:1808.07042 [cs]*.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-Driven Response Generation in Social Media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593. Association for Computational Linguistics.

Avishek Saha, Piyush Rai, Hal DaumÃ© III, and Suresh Venkatasubramanian. 2011. Online Learning of Multiple Tasks and Their Relationships. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 643–651. PMLR.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? How controllable attributes affect human judgments. *arXiv:1902.08654 [cs]*.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586. Association for Computational Linguistics.

Yangyang Shi, Martha Larson, and Catholijn M. Jonker. 2015. Recurrent Neural Network Language Model Adaptation with Curriculum Learning. *Comput. Speech Lang.*, 33(1):136–154.

Magnus Själander, Magnus Jahre, Gunnar Tufte, and Nico Reissmann. 2019. EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure.

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can You Put it All Together: Evaluating Conversational Agents' Ability to Blend Skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030, Online. Association for Computational Linguistics.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Proceedings*

*of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Oyvind Tafjord and Peter Clark. 2021. General-purpose question-answering with macaw. *arXiv preprint arXiv:2109.02593*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Zhiliang Tian, Rui Yan, Lili Mou, Yiping Song, Yansong Feng, and Dongyan Zhao. 2017. How to Make Context More Useful? An Empirical Study on Context-Aware Neural Conversational Models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–236. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Oriol Vinyals and Quoc Le. 2015. A Neural Conversational Model. *arXiv:1506.05869 [cs]*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Joseph Weizenbaum. 1966. Eliza–a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Orion Weller, Nicholas Lourie, Matt Gardner, and Matthew E Peters. 2020. Learning from task descriptions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1361–1375.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain Neural Network Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 120–129. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A Transfer Learning Approach for Neural Network Based Conversational Agents. *arXiv:1901.08149 [cs]*.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic Aware Neural Response Generation. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, et al. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Yanggang Wang, Haiyu Li, and Zhilin Yang. 2022. Zeroprompt: Scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization. *arXiv preprint arXiv:2201.06910*.

Xinnuo Xu, Ondřej Dušek, Ioannis Konstas, and Verena Rieser. 2018. Better Conversations by Modeling, Filtering, and Optimizing for Coherence and Diversity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3981–3991, Brussels, Belgium. Association for Computational Linguistics.

Min Yang, Zhou Zhao, Wei Zhao, Xiaojun Chen, Jia Zhu, Lianqiang Zhou, and Zigang Cao. 2017. Personalized Response Generation via Domain Adaptation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 1021–1024. ACM.

Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. 2017a. Recent advances in document summarization. *Knowledge and Information Systems*.

Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017b. Towards Implicit Content-Introducing for Generative Short-Text Conversation Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2190–2199. Association for Computational Linguistics.

Yury Zemlyanskiy and Fei Sha. 2018. Aiming to Know You Better Perhaps Makes Me a More Engaging Dialogue Partner. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 551–561. Association for Computational Linguistics.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. 2018. Multi-domain neural machine translation with word-level domain context discrimination. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, Brussels, Belgium. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018a. Personalizing Dialogue Agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213. Association for Computational Linguistics.

Weinan Zhang, Yiming Cui, Yifa Wang, Qingfu Zhu, Lingzhi Li, Lianqiang Zhou, and Ting Liu. 2018b. Context-Sensitive Generation of Open-Domain Conversational Responses. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2437–2447. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020a. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020b. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

Wanjun Zhong, Yifan Gao, Ning Ding, Yujia Qin, Zhiyuan Liu, Ming Zhou, Jiahai Wang, Jian Yin, and Nan Duan. 2022. ProQA: Structural prompt-based pre-training for unified question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4230–4243, Seattle, United States. Association for Computational Linguistics.

Chenguang Zhu, Michael Zeng, and Xuedong Huang. 2018. Sdnet: Contextualized attention-based deep network for conversational question answering. *arXiv preprint arXiv:1812.03593*.

# Original Papers

# Paper I

# Balancing Multi-Domain Corpora Learning for Open-Domain Response Generation

**Yujie Xing**[1], **Jinglun Cai**[2*], **Nils Barlaug**[1], **Peng Liu**[1], **Jon Atle Gulla**[1]
[1]Department of Computer Science, Norwegian University of Science and Technology
[2]Amazon AWS AI
{yujie.xing,nils.barlaug,peng.liu,jon.atle.gulla}@ntnu.no
cjinglun@amazon.com

## Abstract

Open-domain conversational systems are assumed to generate equally good responses on multiple domains. Previous work achieved good performance on the single corpus, but training and evaluating on multiple corpora from different domains are less studied. This paper explores methods of generating relevant responses for each of multiple multi-domain corpora. We first examine interleaved learning which intermingles multiple corpora as the baseline. We then investigate two multi-domain learning methods, labeled learning and multi-task labeled learning, which encode each corpus through a unique corpus embedding. Furthermore, we propose Domain-specific Frequency (DF), a novel word-level importance weight that measures the relative importance of a word for a specific corpus compared to other corpora. Based on DF, we propose weighted learning, a method that integrates DF to the loss function. We also adopt DF as a new evaluation metric. Extensive experiments show that our methods gain significant improvements on both automatic and human evaluation. We share our code and data for reproducibility.[1]

## 1 Introduction

Recent work has achieved improvements in general performance for open-domain response generation (Vinyals and Le, 2015; Serban et al., 2017; Li et al., 2016; Xu et al., 2018). However, most studies are restricted to single-corpus training and evaluating, while there lacks studies for training and evaluating with multiple corpora from different domains. Single-corpus training has intrinsic limitations. For example, a corpus of everyday chats, e.g., the PersonaChat corpus (Dinan et al., 2019), does not cover technical topics discussed in Ubuntu chatlogs (Lowe et al., 2015). A conversational system that learns only from PersonaChat or from multiple corpora without an appropriate technique is not likely to generate relevant responses for certain topics (see Table 1). Therefore, it is necessary for an open-domain conversational system to learn from multiple corpora, and to learn with good techniques.

| Context | What are you going to do on the remote system exactly? |
|---|---|
| PersonaChat | I am going to be a pilot. I am going to fly planes. |
| 4 corpora (concatenated) | I am going to go to the beach. |

Table 1: Irrelevant responses generated from fine-tuned GPT-2. The GPT-2 model is fine-tuned respectively on PersonaChat / concatenated 4 corpora (OpenSubtitles, Twitter, Ubuntu, PersonaChat)

| | Test set | | | |
|---|---|---|---|---|
| Fine-tune corpus | OSDB | Twitter | Ubuntu | PersonaChat |
| PersonaChat | 478.8 | 159.6 | 264.7 | 19.6 |
| 4 corpora (concatenated) | 392.8 | 110.7 | 199.2 | 19.0 |

Table 2: Imbalanced perplexity performance of fine-tuned GPT-2. The GPT-2 model is fine-tuned on PersonaChat / concatenated 4 corpora (OpenSubtitles, Twitter, Ubuntu, PersonaChat)

Furthermore, the case of using a single small-scale open-domain corpus has apparent weaknesses. A common way of dealing with a small-scale corpus is through fine-tuning (Li et al., 2016; Akama et al., 2017; Chu et al., 2017). Fine-tuning on a single corpus tends to make the model overfit on that specific corpus while performing worse on other corpora. Table 2 shows the result of a GPT-2 model gaining good performance on PersonaChat while performing poorly on other corpora.

This paper explores how to train and evaluate on multiple corpora from different domains for the open-domain response generation task. We propose

* This work was done prior to the author joining Amazon.
[1]https://github.com/yujie-xing/Balancing_Multi_Domain_Corpus_Learning_for_Open_Domain_Response_Generation



*Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2104 - 2120
July 10-15, 2022 

several methods to make a model generate relevant responses for each of the multiple corpora.

Since simply training multiple corpora one by one does not solve the imbalanced performance (as shown in Table 1 and 2), we first investigate *interleaved learning*, a method that intermingles the training data instead of simply concatenating, which ensures a model learns from all corpora evenly. We use this method as a baseline. Additionally, we explore two multi-domain learning methods: *labeled learning* and *multi-task labeled learning*. Labeled learning comes from a control technique in response generation (Li et al., 2016; Johnson et al., 2017; Yang et al., 2017). Previous works focus on controlling persona and style, while our method controls corpus's information with the corpus embedding. Multi-task labeled learning is inspired by works of domain adaption (Luan et al., 2017; Niu and Bansal, 2018; Chu and Wang, 2018), where multiple losses from both the corpus classifier and response generator are minimized. To the best of our knowledge, this paper is the first that uses corpus embeddings on the open-domain response generation task for multiple corpora.

Furthermore, we propose a novel *weighted learning* with Domain-specific Frequency (DF). DF is a word-level importance weight (Leopold and Kindermann, 2002) that assigns different weights (importance) to the same words from different corpora. In the training process, we weight the loss of a model with DF, so that the model focuses on the most important words for a specific corpus.

For automatic evaluation metrics, we eliminate the stop words and use ROUGE-1 (precision, recall, F1) (Lin, 2004) to measure the **relevance** of the generated responses. In addition, we adopt DF to see how relevant the generated response of a model is to a specific corpus. We will explain DF as an evaluation metric in Section 4.4. Results show that for overall performance, the best method (weighted learning) improves $27.4\%$ on precision, $45.5\%$ on recall, and $34.1\%$ on F1. Further, it has at least $20.0\%$ higher DF, stating that it uses more important words from the "correct" corpus. We also conduct an extensive human evaluation on 2400 generated responses. The human evaluation shows a highly significant ($p < 0.001$) improvement on all of our proposed methods, especially the weighted learning method.

We summarize our work as follows:

- We explore the problem of training and evaluating on multiple corpora from different domains for open-domain response generation. The task is to make the conversational models generate relevant responses for **each** corpus.
- We examine several multi-domain corpora learning methods for their ability to solve the proposed task.
- We propose Domain-specific Frequency (DF) as in weighted learning and as an evaluation metric. DF distinguishes important words for each corpus and helps a model to focus on these important words in the training process.

## 2 Related Work

**Open-Domain Response Generation** Recent work of open-domain response generation generally follows the work of Ritter et al. (2011) where the task is treated as a machine translation task, and many of them use a Seq2Seq structure (Sutskever et al., 2014) following previous work (Vinyals and Le, 2015; Shang et al., 2015; Sordoni et al., 2015). In recent years, substantial improvements have been made (Serban et al., 2017; Li et al., 2016; Wolf et al., 2019), and embeddings are used to control response generation on extra information such as persona (Li et al., 2016), profiles (Yang et al., 2017), coherence (Xu et al., 2018), emotions (Huang et al., 2018), and dialogue attributes like response-relatedness (See et al., 2019). However, there is a lack of work that uses embeddings to control response generation over multiple corpora. Our work follows the common models of open-domain conversational systems, while we study the problem of multiple corpora of different domains.

**Multi-Domain Learning and Domain Adaption** Multi-domain learning aims at making a conversational model learn from multiple domains to prevent the performance from degrading due to domain differences (Ben-David et al., 2007). There are two categories of solutions for multi-domain learning (Joshi et al., 2012): (i) capturing domain-specific characteristics in the parameters (Daumé III, 2007); (ii) capturing the relationship among different domains (Saha et al., 2011).

Some work of natural language generation and machine translation is related to multi-domain learning. Luan et al. (2017) and Niu and Bansal (2018) use multi-task learning for domain adaption respectively on speaker-role and politeness. Wen

et al. (2016) and Akama et al. (2017) utilizes fine-tuning as a common way of domain adaption for language generator and style transferer. For machine translation, in order to deal with the mixed-domain parallel corpus, Zeng et al. (2018) adjust the weights of target words in the training objective based on their relevance to different domains. We differ in that we propose DF and we deal with the response generation task. Chu et al. (2017) propose mixed fine-tuning, which adds the out-of-domain pre-training data to the fine-tuning dataset, and they observe an improvement of performance. In this paper, we also mix small-scale fine-tuning datasets with out-of-domain training data, while the data we add is not necessarily used during pre-training. Shi et al. (2015) state that fine-tuning can be done by placing the corpus to be fine-tuned at the end of the entire corpus, which is an extension of curriculum learning proposed by Bengio et al. (2009). We also explore how the order of multiple corpora influences the result, but our focus is on balancing performance. Recently, Smith et al. (2020) investigated blending conversational skills with knowledge and empathy skills, where they mix 3 corpora. They focus on selecting appropriate skills and they propose a blended corpus with labels, while we focus on generating responses that are most relevant to a specific corpus.

## 3 Base Models

We use two base models: an LSTM Seq2Seq model with attention (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014; Bahdanau et al., 2015) and a pre-trained GPT-2 model (Radford et al., 2019). The LSTM Seq2Seq model with attention is a common model for conversational systems (Li et al., 2016; See et al., 2019), and the GPT2 model is a state-of-the-art model for the response generation task (Zhang et al., 2020; Zhao et al., 2020).

The basic task of response generation is to predict the next word given the past and current words of the context and response, and to make the generated response as similar to the original response as possible. The task can be described as follows. Probability of response $Y$ given context $X$ is predicted as:

$$P(Y|X) = \prod_{t=1}^{n} P(y_t|y_1, \ldots, y_{t-1}, X), \quad (1)$$

where $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ is a context-response pair.

### 3.1 LSTM Seq2Seq Model with Attention

We simplify an LSTM with attention unit as $LSTM^*$ since it is well introduced in previous work (Li et al., 2016). We calculate the hidden vector $h_t$ at step $t$ as:

$$h_t = LSTM^*(h_{t-1}, E(z_t)), \quad (2)$$

where $h_{t-1} \in \mathbb{R}^{dim}$ is the hidden vector at step $t-1$, $dim$ is the dimension of hidden vectors, and $E(z_t)$ is the word embedding for word $z_t \in (x_1, \ldots, x_m, y_1, \ldots, y_{n-1})$. We apply dot multiple in the attention mechanism when calculating the context vector $c_t$:

$$c_t = H \cdot (softmax(H^\top \cdot h_t))$$

where $H \in \mathbb{R}^{d \times m}$ is the concatenation of hidden vectors from the encoder. $c_t$ is input to the next step $t+1$ in the decoder. Each token's hidden vector $h_t$ in the decoder is combined with $c_t$ through a linear layer and an activation to predict the next token.

### 3.2 GPT-2

As for GPT-2, we follow the adaption of Wolf et al. (2019). The transformer block of GPT-2 captures the relation of multiple words in one sentence, which largely follows Vaswani et al. (2017). The hidden vector to be input to the transformer block is calculated as:

$$h_{0[t]} = E(X, Y_{[1:t]}) + (E_0, E_1) + W_p, \quad (3)$$

where $Y_{[1:t]}$ is $(y_1, \ldots, y_t)$, $E(X, Y_{[1:t]})$ is the sub-word embedding for context $X$ and response $Y_{[1:t]}$. $E_0$ and $E_1$ are dialogue-state embeddings, which tutor the model to distinguish between contexts and responses. $W_p$ is a pre-trained position embedding. The probability of the subword to generate is then calculated as:

$$h_{[t]} = transformer\_block(h_{0[t]}) \quad (4)$$

$$P(y)_{t+1} = softmax(E^\top(h_{[t]})), \quad (5)$$

where $y \in V$, and $V$ stands for the sub-word vocabulary. We simplify the structure of transformer block as $transformer\_block$. In the block, a mask is filled in the attention matrix, which bans past words from attending to future words. This ensures that the model follows the traditional language modeling. The hidden vector of $t_{\text{th}}$ sub-word is used to generate the probability distribution for the vocabulary ($P(y),\ y \in V$) for $(t+1)_{\text{th}}$ sub-word. $E^\top$ means that the model uses the sub-word embeddings in calculating sub-word probabilities for generation (Press and Wolf, 2017).

## 4 Proposed Methods

### 4.1 Interleaved Learning

Interleaving is a concept in cognitive psychology proven to be efficient for learning (Kornell and Bjork, 2008): intermingling learning material of different topics helps students to gain better learning results than learning the material topic by topic. Previous work from machine learning also shows that training order greatly influences the performance (Bengio et al., 2009). When the training is conducted on a simple concatenation of multiple corpora, the model tends to concentrate on the last corpus (Shi et al., 2015). To address this issue, we propose interleaved learning as an alternative: each time we collect one context-response pair from each of the corpora, and we randomly shuffle them. For example, if there are 3 corpora $(a_1, a_2, ...), (b_1, b_2, ...), (c_1, c_2, ...)$ where $a_i, b_i$ and $c_i$ are context-response pairs, the resulting mixed corpus might be $(b_1, a_1, c_1, c_2, b_2, a_2, ...)$. Interleaved learning guarantees that the combined corpus is evenly distributed, which helps the model learn from multiple corpora evenly.

### 4.2 Labeled Learning

We propose our labeled learning as follows: each corpus is assigned a randomly initialized unique embedding, and the conversational model learns these embeddings together with conversations during the training period. We denote these embeddings as "corpus embedding", or $E_c$. A model captures each corpus's characteristics through the corpus embedding and uses it to control the generated responses. To know which corpus embedding to use, each context is labeled with which corpus it comes from, and these labels are provided to the model both in the training and generation period. We propose an approach for each of our base models for encoding corpus embeddings.

For the LSTM model, following Li et al. (2016), we input the corpus embedding $E_c$ into the first layer of the decoder LSTM at every step, together with the response words. Calculation of a hidden vector $h_t$ in the decoder LSTM is then adapted to:

$$h_t = LSTM^*(h_{t-1}, E(y_t), E_c). \quad (6)$$

The structure is illustrated in the dashed red rectangle of Figure 1a.

For the GPT-2 model, our method is based on Wolf et al. (2019). Instead of two kinds of dialogue-state embeddings (context embedding $E_0$ and response embedding $E_1$), we replace the response embedding with corpus embeddings $E_c$. As a result, the model is aware of which corpus the response belongs. Calculation of a hidden vector to be input to the transformer block is adapted to:

$$h_{0[t]} = E(X, Y_{[1:t]}) + (E_0, E_c) + W_p. \quad (7)$$

The structure is illustrated in Figure 1b.

### 4.3 Multi-Task Labeled Learning

Labeled learning needs corpus labels for both training and generation processes. To avoid providing labels in the generation process, we combine multi-task learning with labeled learning on multiple corpora. Here, the conversational model has to predict by itself which corpus a context belongs to, which is expected to result in worse performance, but less information is required. In the encoder, we have a classifier layer that uses the sum of hidden vectors from the encoder ($\sum H$) to predict the corpus of a context. The loss of the classifier is calculated as:

$$\mathcal{L}_c = -\log\left(softmax\left(\left(\sum H\right) \cdot W_{[c]}\right)\right), \quad (8)$$

where $W_{[c]} \in \mathbb{R}^{dim}$ is the part from the classifier layer for target corpus $c$. $\mathcal{L}_c$ is summed up with the loss from the response generator. The predicted corpus embedding is input into the decoder like labeled learning (see Section 4.2). The simplified structure is illustrated in Figure 1a.

### 4.4 Document-specific Frequency (DF)

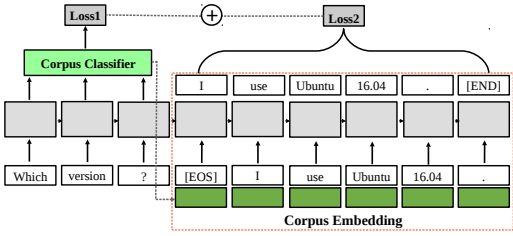
We propose Domain-specific Frequency (DF) to measure how important a word is with respect to a different corpus under a collection of corpora. DF is used for weighted learning and evaluation. It is calculated as follows:

$$f(w)_d = \text{freq}(w)_d - \min_v\{\text{freq}(v)_d\} \quad (9)$$

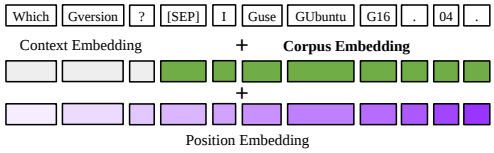
$$\text{df}(w)_d = \begin{cases} 0 & f(w)_d = 0 \\ \frac{f(w)_d}{\sum_{d \in D} f(w)_d} & f(w)_d \neq 0 \end{cases} \quad (10)$$

$$\text{DF}(w)_d = \frac{\text{df}(w)_d}{\max_v\{\text{df}(v)_d\}}, \quad (11)$$

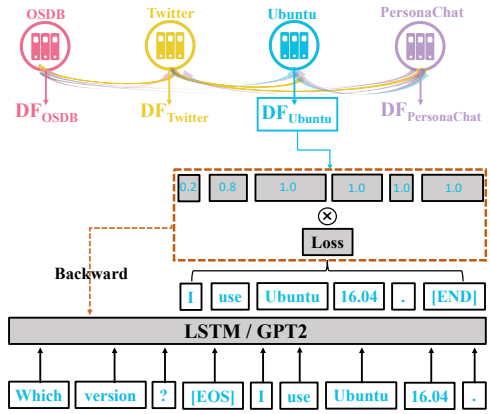
where $\text{freq}(w)_d$ is the relative frequency of a word $w$ in a corpus $d$, and $D$ represents the set of all corpora. It is easy to see from Equation 10 that $\text{DF}(w)_d$ represents the importance of word $w$ for corpus $d$ compared to other corpora. For a word $w$ that frequently appears in corpus $d$ but seldom




(a) Structure of multi-task labeled learning on LSTM model




(b) Corpus embeddings with sub-word embeddings on GPT-2




(c) Structure of weighted learning

Figure 1: Adapted models with labeled learning, multi-task labeled learning and weighted learning

in other corpora (e.g., "upgrade" from Ubuntu corpus), $\sum_{d \in D} f(w)_d$ is close to $f(w)_d$, making $\text{DF}(w)_d$ approach 1. A word that frequently appears in all corpora (e.g., "I", "you") is punished, resulting in a lower $\text{DF}(w)_d$. A word that seldom appears in corpus $d$ but frequently appears in other corpora (e.g., "music" seldom appears in Ubuntu corpus, but is common in other corpora) has the lowest $\text{DF}(w)_d$. Words that appear minimal times (e.g., once) in a corpus are ignored with Equation 9. Words that appear few times (e.g., twice or three times) are not dealt with, yet they are not of great influence in our experiments. We apply a normalization in the final step (Equation 11) to make $\text{DF}(w)_d$ of each corpus $d$ range from 0 to 1.

We show $\text{DF}(w)_{\text{Ubuntu}}$ and $\text{DF}(w)_{\text{PersonaChat}}$ of some words in Table 3. We also show the results of TF-IDF (log normalization variant), a commonly used word importance weight, as a comparison. As expected, for the corpus Ubuntu and PersonaChat, most unique words $w$ have very different $\text{DF}(w)_{\text{Ubuntu}}$ and $\text{DF}(w)_{\text{PersonaChat}}$. Unique words of each corpus get the highest values for the corresponding corpus, like "upgrade" for the Ubuntu corpus and "music" for the PersonaChat corpus; these words receive the lowest values for *incorrect* corpora, like "upgrade" for PersonaChat and "music" for Ubuntu. The stress on unique words makes DF more suitable for our task.

**Weighted Learning with DF** Weighted learning weights the loss of the predication $y'$ for each target word $w$ using $\text{DF}(w)_d$. In the training period, each context is labeled with the corpus $d$ it belongs to, so that the model can use the $\text{DF}(w)_d$ of the corresponding corpus. Here DF is calculated only on the training sets. In the generation step, corpus labels are not provided, so DF is not used. The loss is weighted as follows :

$$\mathcal{L}_{\text{weighted}} = \text{DF}(w)_d \cdot \left(-\log\left(softmax(y'_w)\right)\right), \quad (12)$$

where $y'_w$ represents the model's predicted score for the target word $w$. With the weighted loss, the model concentrates on words that are important to the corpus of the current context, and focuses less on frequent words or words that are not important to the current corpus. The structure is illustrated in Figure 1c.

**Evaluation with DF** For the generated responses to be relevant to a specific corpus, they have to

| Word | TF-IDF(%) | | DF(%) | | $\alpha$DF$_{(\alpha=100)}$ | |
|---|---|---|---|---|---|---|
| | Ubuntu | PersonaChat | Ubuntu | PersonaChat | Ubuntu | PersonaChat |
| i | 100.0 | 62.6 | 20.8 | 42.1 | 2.6 | 7.3 |
| to | 64.6 | 32.8 | 26.9 | 24.9 | 3.8 | 3.1 |
| it | 83.2 | 21.7 | 38.5 | 14.5 | 5.1 | 2.1 |
| laptop | 5.4 | 0.2 | 89.8 | 4.5 | 76.0 | 1.0 |
| upgrade | 6.8 | 0.1 | 95.6 | 0.4 | 91.2 | 1.0 |
| file | 15.7 | 0.1 | 96.0 | 0.3 | 86.4 | 0 |
| windows | 12.2 | 0.1 | 97.1 | 0.1 | 86.3 | 1.0 |
| ubuntu | 27.5 | 0 | 99.9 | 0 | 99.5 | 0 |
| teacher | 0.1 | 2.2 | 0.7 | 77.8 | 1.0 | 53.5 |
| music | 1.5 | 7.6 | 4.8 | 82.9 | 1.2 | 49.1 |
| travel | 0.1 | 3.1 | 0.3 | 88.9 | 1.0 | 57.1 |
| hobby | 0.1 | 1.6 | 0.6 | 94.3 | 1.1 | 81.7 |
| hiking | 0 | 1.5 | 0 | 97.6 | 0 | 91.8 |

Table 3: Normalized TF-IDF (%), DF (%) and $\alpha$DF of some words for Ubuntu and PersonaChat (more examples on other corpora can be found in Section A)

be similar to that corpus, which includes using important words of that corpus (e.g., responses generated for the Ubuntu corpus should have more technical words than other corpora). Thus, we propose DF as an evaluation metric that shows to what extent the generated responses use important words of the corresponding corpus. We want to decrease the influence of common words like "i", "to", etc., and thus address the important words. So we adopt exponential DF with $\alpha$ as the base ($\alpha$DF):

$$\alpha\text{DF}(w)_d = \begin{cases} 0 & \text{DF}(w)_d = 0 \\ \alpha^{\text{DF}(w)_d} & \text{DF}(w)_d \neq 0, \end{cases} \quad (13)$$

where $\alpha$ is a constant. $\alpha\text{DF}(w)_d$ rescales $\text{DF}(w)_d$ by exponent with $\alpha$ as a base. In our experiments, we set $\alpha$ to be 100, which transforms the range of the metric from $(0, 1)$ to $(0, 100)$. This makes the difference between high and low $\alpha$DF more significant than DF and gives a 100-scale score. For each corpus $d \in D$, we average $\alpha\text{DF}(w)_d$ on word $w$ from the generated responses of each test set, which gives us $\alpha\text{DF}_d$ scores ($d \in D$) for each test set. Ideally, the generated responses of a specific corpus $d$ should have a higher $\alpha\text{DF}_d$ score and lower $\alpha\text{DF}_{\overline{d}}$ score ($\overline{d} \in \{d' \in D \mid d' \neq d\}$). For example, generated responses of the Ubuntu test set should have a higher $\alpha\text{DF}_{\text{Ubuntu}}$ score, while a lower $\alpha\text{DF}_{\overline{\text{Ubuntu}}}$ score ($\overline{\text{Ubuntu}} \in \{d' \in D \mid d' \neq \text{Ubuntu}\}$). $\alpha\text{DF}_d$ scores for responses from the original test sets are the standard scores.

We show $\alpha\text{DF}(w)_{\text{Ubuntu}}$ and $\alpha\text{DF}(w)_{\text{PersonaChat}}$ (calculated purely on test set) in Table 3. As expected, $\alpha$DF has a more significant difference between important words and common words.

**Is DF a Legal Evaluation Metric?** Although DF is used for both weighted learning and evaluation, we see DF as a suitable evaluation metric for our task and not biased in favor of weighted learning due to: 1) A word receives multiple DF values in the training process given the corpus that a context belongs to; 2) in the generation process, DF is never used. 3) In the evaluation process, DF can be calculated purely on the test sets. Note that since a word receives multiple DF values in the training step, it is equivalently likely for the model trained with weighted learning to be influenced by DF weights of **incorrect** corpus. Above all, in the evaluation step, if the trained model is influenced more by DF weights from the correct corpus, it already means that the model is good at distinguishing which corpus a given context is from, thus is suitable for our task.

## 5 Experiment Setup

### 5.1 Datasets

**Data Collection** We collected 4 commonly used English corpora of different domains from the ParlAI platform (Miller et al., 2017): OpenSubtitles corpus (OSDB)[2] (Lison et al., 2018), Twitter corpus[3] (Miller et al., 2017), Ubuntu chatlogs corpus (Lowe et al., 2015)[4] , and PersonaChat corpus (Zhang et al., 2018) from the NeurIPS 2018 ConvAI2 Challenge (Dinan et al., 2019). Each corpus contains 250K context-response pairs, as much as the size of the original PersonaChat used in ConvAI2 competition. This gives us 1M context-response pairs in total. The corpus for training is a combination of these 4 corpora. For comparison, we have a single corpus–PersonaChat–trained on both base models. For testing, each of the 4 corpora has a test set of 30K context-response pairs, which is the same size of the test set of PersonaChat.

The OpenSubtitles corpus (OSDB) is a noisy dataset of film subtitles. We removed films that belonged to genres that usually had few conversations, such as musical and documentary films. We regarded two neighboring sentences as a context-response pair following Vinyals and Le (2015). The Twitter corpus contains one-turn dialogues extracted from Twitter. The original author has already cleaned it, so we only removed special symbols such as hashtags, Emojis, and @. The Ubuntu corpus contains dialogues about solving technical problems of Ubuntu. The PersonaChat corpus contains dialogues between two workers acting as specific personas; we focused on the dialogue part and ignored the persona part. This corpus allows us to compare our base models with state-of-the-art performance. These 4 corpora have very different characteristics, confirmed by the imbalanced performance of GPT-2 fine-tuned on a single corpus (see Table 2).

### 5.2 Training and Decoding

We used Pytorch (Paszke et al., 2017) to implement the LSTM Seq2Seq model with attention and the pre-trained GPT-2 models. For GPT-2, we adapted

[2] http://www.opensubtitles.org/

[3] https://github.com/Marsan-Ma/chat_corpus/

[4] https://github.com/rkadlec/ubuntu-ranking-dataset-creator

| Model | Corpus / Method | Test set | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | | Twitter | | | Ubuntu | | | PersonaChat | | | Overall | | |
| | | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 | Precision | Recall | F1 |
| LSTM | PersonaChat (single) | 11.8 | 8.9 | 8.1 | 12.4 | 8.6 | 8.9 | 12.1 | 8.1 | 7.7 | 56.7 | 43.4 | 45.8 | 23.2 | 17.2 | 17.6 |
| | Concatenated | 11.0 | 7.7 | 7.2 | 15.7 | 10.9 | 11.4 | 36.5 | 17.8 | 20.1 | 57.7 | **44.0** | 46.4 | 30.2 | 20.1 | 21.3 |
| | Interleaved | 24.1 | 10.1 | 11.7 | 24.3 | 12.5 | 14.9 | 58.4 | 24.9 | 29.6 | 56.1 | 41.5 | 44.3 | 40.7 | 22.3 | 25.1 |
| | Labeled | 23.9 | 10.1 | 11.3 | 24.5 | **13.2** | 15.5 | 61.6 | 26.5 | 31.6 | 56.4 | 43.0 | 45.4 | 41.6 | 23.2 | 26.0 |
| | Multi-task Labeled | 23.2 | 9.6 | 11.1 | 23.2 | 12.3 | 14.5 | 56.4 | 23.8 | 28.3 | 53.2 | 40.6 | 42.7 | 39.0 | 21.6 | 24.2 |
| | Weighted | **26.6** | **11.9** | **13.4** | **29.7** | 12.2 | **15.6** | **78.4** | **35.2** | **41.2** | **62.4** | 42.5 | **47.1** | **49.3** | **25.5** | **29.3** |
| GPT-2 | PersonaChat (single) | 15.0 | 12.4 | 10.8 | 19.6 | 13.2 | 13.9 | 24.8 | 16.2 | 15.5 | 70.0 | 57.1 | 58.8 | 32.4 | 24.7 | 24.7 |
| | Concatenated | 17.4 | 14.1 | 12.6 | 24.5 | 16.4 | 17.2 | 35.0 | 22.5 | 22.4 | 66.8 | 55.4 | 56.3 | 35.9 | 27.1 | 27.1 |
| | Interleaved | 40.0 | 20.5 | 22.3 | 31.0 | 17.9 | 20.1 | 81.7 | 38.1 | 44.3 | 68.7 | 56.2 | 57.6 | 55.3 | 33.2 | 36.1 |
| | Labeled | 38.6 | 19.9 | 21.6 | 31.4 | **19.4** | 21.1 | 84.2 | 38.4 | 45.0 | **70.7** | **57.2** | **59.0** | 56.2 | 33.7 | 36.7 |
| | Multi-task Labeled | 38.4 | 19.8 | 21.4 | 31.2 | 18.6 | 20.6 | 80.9 | 37.8 | 43.8 | 68.0 | 56.0 | 57.3 | 54.6 | 33.0 | 35.8 |
| | Weighted | **41.9** | **21.2** | **23.4** | **39.9** | 18.4 | **22.3** | **86.8** | **43.3** | **48.6** | 69.0 | 53.2 | 55.8 | **59.4** | **34.0** | **37.5** |

Table 4: Precision, recall and F1 of ROUGE-1 (‰) for baselines and proposed methods fine-tuned on 4 corpora (stop words eliminated)

| Model | Corpus / Method | Test set | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | | | Twitter | | | | Ubuntu | | | | PersonaChat | |
| | | OSDB | | PersonaChat | | Twitter | | PersonaChat | | Ubuntu | | PersonaChat | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | **7.0** | **9.7** | 3.6 | 3.7 | **9.1** | **11.0** | 3.6 | 3.8 | **19.4** | **23.2** | 2.7 | 2.8 | **9.5** | **12.0** |
| LSTM | PersonaChat (single) | 2.9 | 3.4 | 9.2 | 9.9 | 2.8 | 3.4 | 8.6 | 9.2 | 2.7 | 3.1 | 8.6 | 9.1 | 11.9 | 12.6 |
| | Concatenated | 2.9 | 3.3 | 7.6 | 8.6 | 3.6 | 4.3 | 8.0 | 8.7 | 7.6 | 7.7 | 5.6 | 6.0 | 12.5 | 13.6 |
| | Interleaved | 3.9 | 4.1 | 5.0 | 5.3 | 4.7 | 4.9 | 4.1 | 4.5 | 11.8 | 11.3 | 3.7 | 4.0 | 11.5 | 12.5 |
| | Labeled | 3.9 | 4.2 | 5.0 | 5.3 | 5.0 | 5.3 | 3.9 | 4.3 | 12.5 | 11.8 | 3.4 | 3.8 | 12.1 | 13.1 |
| | Multi-task Labeled | 3.8 | 4.0 | 5.0 | 5.4 | 4.5 | 4.7 | 4.1 | 4.5 | 11.2 | 10.7 | 3.8 | 4.1 | 11.4 | 12.6 |
| | Weighted | **5.6** | **6.3** | 4.1 | 4.5 | **9.9** | **10.1** | 3.8 | 4.3 | **27.7** | **25.4** | 2.7 | 3.0 | **17.7** | **18.3** |
| GPT-2 | PersonaChat (single) | 2.8 | 3.2 | 10.5 | 11.1 | 2.9 | 3.3 | 9.5 | 9.8 | 4.1 | 4.6 | 8.3 | 8.4 | 12.9 | 13.7 |
| | Concatenated | 3.1 | 3.6 | 8.8 | 9.4 | 3.3 | 3.9 | 8.2 | 8.7 | 6.5 | 7.1 | 7.0 | 7.4 | 12.1 | 13.0 |
| | Interleaved | 4.9 | 5.8 | 4.8 | 5.0 | 4.6 | 5.1 | 4.4 | 4.7 | 15.7 | 16.0 | 3.1 | 3.4 | 12.1 | 12.9 |
| | Labeled | 4.9 | 5.8 | 4.8 | 5.0 | 4.7 | 5.2 | 4.1 | 4.3 | 16.7 | 17.0 | 2.9 | 3.2 | 12.4 | 13.1 |
| | Multi-task Labeled | 4.8 | 5.7 | 4.8 | 5.1 | 4.6 | 5.1 | 4.4 | 4.6 | 15.5 | 15.8 | 3.1 | 3.4 | 12.1 | 12.9 |
| | Weighted | **6.0** | **7.5** | 4.1 | 4.4 | **8.1** | **8.8** | 3.7 | 4.1 | **25.7** | **24.4** | 2.4 | 2.6 | **16.0** | **17.1** |

Table 5: $\alpha\text{DF}_d$ scores for generated responses from multiple corpora. The columns "train" indicate train-set-$\alpha\text{DF}_d$. The columns "test" indicate test-set-$\alpha\text{DF}_d$.

our model from the implementation of the HuggingFace team[5]. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, learning rate of 1.0, dropout rate of 0.2, and gradient clip threshold of 5. The vocabulary size is 50000. GPT-2 has 12 layers, 12 heads, and the dimension is 768, the same as the pre-trained model. The training procedure was with Adam and we adopted a similar setup as Wolf et al. (2019): the batch size was 32, learning rate was $6 \times 10^{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, L2 weight decay set to 0.01, learning rate linearly decreased to zero at the end. We followed these hyper-parameters to ensure state-of-the-art performance for the base models. We use the same hyper-parameters for both base models and models with our proposed methods, so the proposed methods work slightly (but not much) worse than it should be. This is to avoid the extra improvement caused by hyper-parameters. We pre-trained the LSTM model on 3 large-scale corpora (OSDB, Twitter and Ubuntu) with interleaved learning until converging. GPT-2 is already pre-trained, so we directly used it for fine-tuning (details about pre-training convergence can be found in Section B). For decoding, we adopted greedy decoding for all the models to ensure an equal condition.

### 5.3 Evaluation

For automatic metrics, to measure the **relevance** of the generated responses, we eliminated punctuation and stop words, and adopted Rouge-1[6] (precision, recall, F1) as multi-grams become meaningless

[5] https://huggingface.co/.

[6] We used implementation from https://github.com/google-research/google-research/tree/master/rouge.

without stop words. However, Rouge-1 compares the generated responses with the golden ones, while there is never a standard response for any context, so in addition to Rouge, we use $\alpha$DF score that shows to what extent the generated responses use important words of the corresponding corpus, as stated in Section 4.4. Due to the limitation of automatic evaluation methods (Liu et al., 2016), we also conduct an extensive human evaluation on the relevance of generated responses to contexts (see Section 6.1 for details).

## 6 Results

Our base models achieve perplexity scores of 28.9 (LSTM model) and 19.6 (GPT-2) on the test set of the PersonaChat dataset from the ConvAI2 competition when fine-tuned with the single PersonaChat corpus (more details can be found in Section C). These results would likely advance the models to the second round in the competition.

Table 4 shows that models trained with our proposed methods gain better performance on Rouge than baselines. Baselines concentrate on the last trained corpus (PersonaChat), while with the proposed methods, performance is more balanced on multiple corpora. Weighted learning has the best overall performance on all metrics, and it performs especially well on the Ubuntu corpus, indicating that it might be good at distinguishing the unique technical words from the Ubuntu corpus. Labeled learning is the second best with stable improvement from interleaved learning, indicating that the corpus embeddings function as expected. Multi-task labeled learning has slightly worse performance than interleaved learning, indicating that predicting the corpus of a contexts is not easy, and wrong predictions result in worse performance.

Table 5 shows $\alpha\text{DF}_d$ scores for generated responses of each corpus. Full results can be found in Section E. We use both $\alpha\text{DF}_d$ calculated purely on the train set (train-set-$\alpha$DF) and $\alpha\text{DF}_d$ calculated purely on the test set (test-set-$\alpha$DF). The black scores are scores for the corresponding corpus (we expect high scores for these parts), while the grey scores are scores for non-related corpus–PersonaChat (we expect low scores for these parts). Note that scores for different corpora are in different scales. From the table, we can see that train-set-DF scores and test-set-DF scores are similar, and weighted learning always has the highest score, indicating that weighted learning distinguishes well which corpus a context comes from. Labeled learning is the second best, indicating that the learned corpus embeddings help the model to use more important words of the corresponding corpus. Compared to the concatenated corpus, the improvement is at least 20%, while the decrease in PersonaChat is just 9% at most.

### 6.1 Human Evaluation

We conducted a human evaluation on all GPT-2 models: base models and models adapted with our proposed methods. We randomly picked 2400 responses: 400 different contexts evenly from 4 corpora with 6 responses generated by each of our models. 3 judges[7] are asked to pick the most and the least relevant response(s) for the given context. The most relevant response(s) are given score 3, the least relevant response(s) are given score 1, and the other(s) are given score 2. Table 6 shows the overall scores of all GPT-2 based models. Table 7 shows the p-value for the t-test conducted between every two models. The overall scores of our proposed methods are all highly significantly ($p < 0.001$) higher than the concatenated models, especially the weighted learning method.

### 6.2 Response Examples

The generated responses from better methods are more relevant to the corresponding corpus, while worse methods cannot distinguish contexts from different corpora (e.g., they may answer any questions in a "PersonaChat" way). To show an intuition of the difference among our proposed methods, we present some response examples generated by GPT-2 in Section G.

### 6.3 Possible Limitations

Our proposed methods are meant to be able to work in most models, which is why we choose the most common conversational models as our base models. However, there are many variants of conversational models focusing on different aspects, such as integrating knowledge, avoiding dull responses, keeping the speech style, etc. We cannot ensure that our methods work for all of these variant models. Also, dialogues are always multi-turn, while we focus on a simpler task: single-turn response generation.

[7]Similar to previous work like Zhang et al. (2020), we have 3 judges. We have one random worker from https://www.mturk.com/worker, one bachelor student, and one graduate student. An example of the mTurk interface can be found in Section F.

| Model \ Corpus | OSDB | Twitter | Ubuntu | PersonaChat | Overall |
|---|---|---|---|---|---|
| PersonaChat (single) | 1.53 | 1.43 | 1.21 | 2.09 | 1.56 |
| Concatenated | 1.67 | 1.71 | 1.60 | 2.16 | 1.78 |
| Interleaved | 2.04 | 1.89 | 2.18 | 2.24 | 2.09 |
| Labeled | 2.10 | 2.10 | 2.32 | 2.24 | 2.19 |
| Multi-task Labeled | 2.05 | 1.98 | 2.11 | 2.24 | 2.10 |
| Weighted | **2.40** | **2.45** | **2.61** | **2.47** | **2.48** |

Table 6: Average scores of human evaluation for GPT-2 based models on each corpus

| Model \ Model | PersonaChat | Concatenated | Interleaved | Labeled | Multi-Task Labeled | Weighted |
|---|---|---|---|---|---|---|
| PersonaChat | 1.00 | \ | \ | \ | \ | \ |
| Concatenated | $2.54 \times 10^{-7**}$ | 1.00 | \ | \ | \ | \ |
| Interleaved | $4.71 \times 10^{-34**}$ | $2.09 \times 10^{-12**}$ | 1.00 | \ | \ | \ |
| Labeled | $1.08 \times 10^{-46**}$ | $9.41 \times 10^{-21**}$ | $1.18 \times 10^{-2*}$ | 1.00 | \ | \ |
| Multi-task Labeled | $6.65 \times 10^{-35**}$ | $6.96 \times 10^{-13**}$ | $8.86 \times 10^{-1}$ | $1.17 \times 10$ | 1.00 | \ |
| Weighted | $1.65 \times 10^{-103**}$ | $2.86 \times 10^{-63**}$ | $6.54 \times 10^{-26**}$ | $1.59 \times 10^{-15**}$ | $2.01 \times 10^{-25**}$ | 1.00 |

Table 7: P-value for t-test on overall human evaluation scores of GPT-2 based models, ** $p < 0.001$

Furthermore, the methods are trained and evaluated on English corpora. There can be a limitation on applying the methods to other languages.

## 7 Conclusions

We have experimented with 4 methods–interleaved learning (baseline), labeled learning, multi-task labeled learning, and weighted learning–to help common open-domain conversational systems generate relevant responses for multiple corpora of different domains. We adopted Rouge (precision, recall, F1) for auto evaluation. In addition, we used DF to evaluate how well a model uses relevant words for a corresponding corpus. We also did an extensive human evaluation. Our results show significant improvement in performance for our proposed methods, especially weighted learning. Future work of multi-turn response generation is potential. We have focused on one-turn response generation, while dialogue is naturally multi-turn so further research is needed.

## Acknowledgements

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assist on computing resources from the IDUN cluster of NTNU (Själander et al., 2019). We would like to thank Aria Rahmati, Zhirong Yang (Norwegian Research Council, 287284) and Özlem Özgöbek for their helpful comments.

## References

Reina Akama, Kazuaki Inada, Naoya Inoue, Sosuke Kobayashi, and Kentaro Inui. 2017. Generating Stylistically Consistent Dialog Responses with Transfer Learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 408–412, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of Representations for Domain Adaptation. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 137–144. MIT Press.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum Learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48. ACM.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An Empirical Comparison of Domain Adaptation Methods for Neural Machine Translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–391. Association for Computational Linguistics.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa

Fe, New Mexico, USA. Association for Computational Linguistics.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2019. The Second Conversational Intelligence Challenge (ConvAI2). *arXiv:1902.00098 [cs]*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Chenyang Huang, Osmar Zaiane, Amine Trabelsi, and Nouha Dziri. 2018. Automatic Dialogue Generation with Expressed Emotions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 49–54. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Mahesh Joshi, Mark Dredze, William W. Cohen, and Carolyn Rose. 2012. Multi-Domain Learning: When Do Domains Matter? In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1302–1312. Association for Computational Linguistics.

Nate Kornell and Robert Bjork. 2008. Learning concepts and categories is spacing the "Enemy of induction"? *Psychological science*, 19:585–92.

Edda Leopold and Jörg Kindermann. 2002. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46(1):423–444.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. 2018. OpenSubtitles2018: Statistical Rescoring of Sentence Alignments in Large, Noisy Parallel Corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*. European Language Resource Association.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132. Association for Computational Linguistics.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294. Association for Computational Linguistics.

Yi Luan, Chris Brockett, Bill Dolan, Jianfeng Gao, and Michel Galley. 2017. Multi-Task Learning for Speaker-Role Adaptation in Neural Conversation Models. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 605–614. Asian Federation of Natural Language Processing.

Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A Dialog Research Software Platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 79–84. Association for Computational Linguistics.

Tong Niu and Mohit Bansal. 2018. Polite Dialogue Generation Without Parallel Data. *Transactions of the Association for Computational Linguistics*, 6:373–389.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *EACL*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Better Language Models and Their Implications. Technical report, Technical report, OpenAI.

Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-Driven Response Generation in Social Media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593. Association for Computational Linguistics.

Avishek Saha, Piyush Rai, Hal DaumÃ© III, and Suresh Venkatasubramanian. 2011. Online Learning of Multiple Tasks and Their Relationships. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 643–651. PMLR.

Abigail See, Stephen Roller, Douwe Kiela, and Jason Weston. 2019. What makes a good conversation? How controllable attributes affect human judgments. *arXiv:1902.08654 [cs]*.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586. Association for Computational Linguistics.

Yangyang Shi, Martha Larson, and Catholijn M. Jonker. 2015. Recurrent Neural Network Language Model Adaptation with Curriculum Learning. *Comput. Speech Lang.*, 33(1):136–154.

Magnus Själander, Magnus Jahre, Gunnar Tufte, and Nico Reissmann. 2019. EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure.

Eric Michael Smith, Mary Williamson, Kurt Shuster, Jason Weston, and Y-Lan Boureau. 2020. Can You Put it All Together: Evaluating Conversational Agents' Ability to Blend Skills. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2021–2030, Online. Association for Computational Linguistics.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Oriol Vinyals and Quoc Le. 2015. A Neural Conversational Model. *arXiv:1506.05869 [cs]*.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016. Multi-domain Neural Network Language Generation for Spoken Dialogue Systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 120–129. Association for Computational Linguistics.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. TransferTransfo: A Transfer Learning Approach for Neural Network Based Conversational Agents. *arXiv:1901.08149 [cs]*.

Xinnuo Xu, Ondřej Dušek, Ioannis Konstas, and Verena Rieser. 2018. Better Conversations by Modeling, Filtering, and Optimizing for Coherence and Diversity. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3981–3991, Brussels, Belgium. Association for Computational Linguistics.

Min Yang, Zhou Zhao, Wei Zhao, Xiaojun Chen, Jia Zhu, Lianqiang Zhou, and Zigang Cao. 2017. Personalized Response Generation via Domain Adaptation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 1021–1024. ACM.

Jiali Zeng, Jinsong Su, Huating Wen, Yang Liu, Jun Xie, Yongjing Yin, and Jianqiang Zhao. 2018. Multi-domain neural machine translation with word-level domain context discrimination. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 447–457, Brussels, Belgium. Association for Computational Linguistics.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

## A Comparison among TF-IDF, DF and $\alpha$DF for 4 corpora on more example words

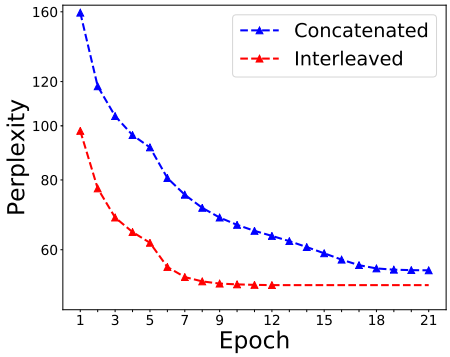
| Word | TF-IDF(%) | | | | DF(%) | | | | $\alpha$DF$_{(\alpha=100)}$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OSDB | Twitter | Ubuntu | PersonaChat | OSDB | Twitter | Ubuntu | PersonaChat | OSDB | Twitter | Ubuntu | PersonaChat |
| i | 91.39 | 100.00 | 100.00 | 62.63 | 21.40 | 15.68 | 20.80 | 42.12 | 2.62 | 2.01 | 2.59 | 7.32 |
| to | 54.46 | 77.55 | 64.59 | 32.80 | 24.85 | 23.40 | 26.87 | 24.89 | 3.00 | 2.88 | 3.76 | 3.08 |
| it | 61.77 | 74.10 | 83.20 | 21.74 | 25.02 | 22.02 | 38.49 | 14.46 | 3.44 | 2.67 | 5.11 | 2.13 |
| sword | 0.64 | 0.17 | 0.01 | 0.08 | 68.37 | 13.74 | 0.26 | 17.63 | 63.29 | 1.37 | 1.00 | 1.15 |
| forgive | 2.41 | 0.48 | 0.16 | 0.06 | 75.35 | 14.37 | 5.44 | 4.84 | 50.96 | 1.58 | 1.19 | 1.05 |
| hurry | 5.21 | 0.52 | 0.09 | 0.08 | 88.39 | 6.67 | 1.48 | 3.45 | 63.53 | 1.32 | 1.15 | 1.04 |
| darling | 2.54 | 0.39 | 0.00 | 0.01 | 90.88 | 8.42 | 0.11 | 0.58 | 57.10 | 1.45 | 0 | 1.21 |
| explain | 1.27 | 0.00 | 0.00 | 0.11 | 91.33 | 0 | 0 | 8.67 | 94.14 | 0 | 0 | 1.06 |
| tax | 0.21 | 2.52 | 0.05 | 0.09 | 6.77 | 87.06 | 1.09 | 5.07 | 1.28 | 71.26 | 1.05 | 1.04 |
| liberal | 0.03 | 1.71 | 0.01 | 0.10 | 2.06 | 88.19 | 0.25 | 9.50 | 1.21 | 59.65 | 0 | 1.38 |
| vote | 0.41 | 6.08 | 0.10 | 0.11 | 6.07 | 90.68 | 0.78 | 2.47 | 1.12 | 80.22 | 1.02 | 1.09 |
| trump | 0.04 | 18.66 | 0.00 | 0.13 | 0.11 | 99.16 | 0.00 | 0.73 | 1.00 | 96.63 | 0 | 1.03 |
| hillary | 0.05 | 8.61 | 0.00 | 0.01 | 0.42 | 99.53 | 0 | 0.05 | 0 | 99.38 | 0 | 1.01 |
| laptop | 0.10 | 0.40 | 5.39 | 0.15 | 1.33 | 4.37 | 89.88 | 4.42 | 1.07 | 1.22 | 76.02 | 1.01 |
| upgrade | 0.03 | 0.47 | 6.85 | 0.03 | 0.24 | 3.75 | 95.63 | 0.37 | 1.01 | 1.06 | 91.24 | 1.03 |
| file | 0.64 | 0.55 | 15.65 | 0.05 | 2.29 | 1.44 | 96.02 | 0.26 | 1.11 | 1.04 | 86.36 | 0 |
| windows | 0.33 | 0.44 | 12.18 | 0.06 | 1.09 | 1.37 | 97.13 | 0.41 | 1.04 | 1.10 | 86.33 | 1.01 |
| ubuntu | 0.00 | 0.01 | 27.47 | 0.00 | 0 | 0.01 | 99.99 | 0 | 0 | 1.01 | 99.48 | 0 |
| music | 1.90 | 3.29 | 1.53 | 7.66 | 4.01 | 8.20 | 4.84 | 82.94 | 1.18 | 1.40 | 1.23 | 49.14 |
| teacher | 1.48 | 0.74 | 0.07 | 2.20 | 14.53 | 7.01 | 0.68 | 77.78 | 1.39 | 1.32 | 1.01 | 53.49 |
| travel | 0.42 | 0.91 | 0.05 | 3.07 | 3.91 | 6.89 | 0.28 | 88.92 | 1.27 | 1.36 | 1.01 | 57.15 |
| hobby | 0.10 | 0.27 | 0.04 | 1.56 | 1.94 | 3.03 | 0.57 | 94.46 | 1.13 | 1.00 | 1.09 | 81.71 |
| hiking | 0.03 | 0.09 | 0.00 | 1.52 | 0.85 | 1.45 | 0 | 97.70 | 0 | 1.09 | 0 | 91.76 |

Table 8: Normalized TF-IDF (%), DF (%) and $\alpha$DF of more example words for 4 corpora

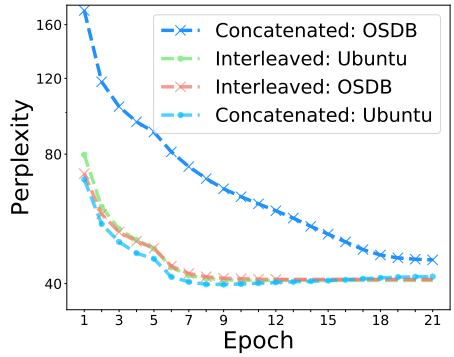
Example words are divided into five blocks. The first block has frequent words in all corpora, the second block has unique words from OSDB, the third block has unique words from Twitter, the fourth block has unique words from Ubuntu, and the fifth block has unique words from PersonaChat. The values of the corresponding corpus are marked with different colors.

From this table, it is clear that the commonly used word importance weight, TF-IDF, is not suitable for our task. This is due to the vast range of frequency, which leads to a relatively small penalty for IDF (Inversed Document Frequency) over words with too large TF (Term Frequency).

## B Convergence time of pre-training LSTM model on large-scale corpora




(a) Overall perplexity of 3 corpora (OpenSubtitles, Twitter, Ubuntu) per epoch




(b) Perplexity of OSDB corpus and Ubuntu corpus per epoch

Figure 2: Convergence time of pre-training LSTM on large-scale corpora

In the pre-training period, it takes 21 epochs for the concatenated corpus to converge on the base LSTM model, while only 12 epochs with interleaved learning, which is $43\%$ shorter. When trained on the concatenated corpus in the order of OSDB $\rightarrow$ Twitter $\rightarrow$ Ubuntu, it takes 20 epochs for the perplexity on OSDB and Ubuntu to be balanced, while with interleaved learning, it takes less than one epoch. For concatenated corpus, the performance of the Ubuntu corpus is sacrificed in order to balance the performance of the two corpora, which results in worse overall performance.

## C Results of automatic evaluation with stop words

| Model | Corpus / Method | Test set: OSDB | | | Twitter | | | Ubuntu | | | PersonaChat | | | Overall | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 | Perp | BLEU | F1 |
| LSTM | PersonaChat (single) | 109.8 | 4.8 | 6.5 | 191.9 | 5.4 | 6.3 | 116.9 | 4.8 | 6.8 | 28.9 | 13.1 | 15.0 | 47.0 | 7.0 | 8.7 |
| | Concatenated | 57.0 | **4.8** | 6.3 | 111.4 | 5.9 | 6.1 | 50.0 | 5.1 | 6.8 | 27.8 | 13.2 | 15.1 | 36.8 | **7.2** | 8.6 |
| | Interleaved | 41.3 | 3.7 | **6.7** | 89.3 | 6.0 | 7.6 | 43.1 | 5.1 | 8.7 | 27.9 | 12.8 | 15.0 | 34.3 | 6.9 | 9.5 |
| | Labeled | **40.5** | 3.2 | 6.6 | **87.0** | **6.2** | 7.6 | **42.6** | **5.3** | **8.8** | **27.1** | **13.2** | **15.2** | **33.4** | 7.0 | **9.6** |
| | Multi-task Labeled | 41.7 | 3.5 | 6.6 | 89.7 | 6.1 | **7.7** | 43.5 | 5.0 | 8.6 | 27.8 | 12.6 | 14.8 | 34.3 | 6.8 | 9.4 |
| | Weighted | 46.1 | 3.6 | 6.6 | 102.5 | 4.6 | 6.7 | 49.4 | 3.8 | 6.6 | 32.8 | 11.4 | 15.0 | 39.9 | 5.8 | 8.7 |
| GPT-2 | PersonaChat (single) | 478.8 | 4.9 | 6.7 | 159.6 | 5.5 | 6.7 | 264.7 | 5.1 | 7.7 | 19.6 | 14.1 | 16.2 | 44.7 | 7.3 | 9.3 |
| | Concatenated | 392.8 | **5.0** | 6.9 | 110.7 | 5.8 | 7.0 | 199.2 | 5.8 | 8.5 | 19.0 | 13.9 | 16.0 | 40.1 | **7.6** | 9.6 |
| | Interleaved | 26.6 | 4.3 | 7.4 | 54.8 | 5.8 | 7.4 | 28.1 | 5.7 | 9.2 | 19.2 | 14.0 | 16.1 | 23.7 | 7.4 | 10.0 |
| | Labeled | **26.5** | 4.2 | 7.3 | **54.1** | **5.9** | **7.6** | **27.7** | 5.7 | 9.2 | **18.9** | **14.1** | **16.3** | **23.5** | 7.5 | **10.1** |
| | Multi-task Labeled | 26.9 | 4.1 | 7.2 | 55.4 | 5.8 | 7.5 | 38.5 | **5.8** | **9.4** | 20.7 | 14.0 | 16.1 | 25.1 | 7.4 | 10.1 |
| | Weighted | 29.6 | 4.3 | **7.5** | 64.1 | 5.1 | 7.4 | 44.1 | 4.1 | 7.0 | 23.4 | 13.0 | 15.7 | 28.4 | 6.6 | 9.4 |

Table 9: Perplexity, BLEU (%) and F1 (%) scores for baselines and proposed methods fine-tuned on 4 corpora (**with** stop words). BLEU is from NLTK sentence BLEU

Models of labeled, multi-task labeled and weighted learning do not have the best hyper-parameters, but the same hyper-parameters as the base models. Their perplexity is slightly worse than it should be.

The results of the single corpus PersonaChat trained with the LSTM model confirm our concern on a small fine-tuning corpus. The LSTM model is pre-trained on OSDB, Twitter and Ubuntu; however, the performance for the 3 corpora greatly decreases after fine-tuning.

The automatic evaluation with stop words is not good for measuring relevance, since stop words are taken too much into account. See BLEU and F1 scores of PersonChat (single) and weighted learning as an example. Models trained on PersonaChat (single) cannot answer Ubuntu technical questions **at all**, yet they receive better scores than weighted learning. But once the stop words are removed, the scores of weighted learning surplus PersonaChat (single) a lot.

## D Additional Results of automatic evaluation without stop words

| Model | Corpus / Method | Test set: OSDB | | | Twitter | | | Ubuntu | | | PersonaChat | | | Overall | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | BLEU | ROUGE | DF-F1 | BLEU | ROUGE | DF-F1 | BLEU | ROUGE | F1 | BEU | ROUGE | DF-F1 | BLEU | ROUGE | DF-F1 |
| LSTM | PersonaChat (single) | 5.2 | 8.1 | 6.2 | 5.7 | 8.9 | 5.0 | 4.5 | 7.7 | 4.8 | 34.2 | 45.8 | 44.6 | 12.4 | 17.6 | 15.2 |
| | Concatenated | 4.5 | 7.2 | 5.6 | 7.4 | 11.4 | 8.8 | 11.6 | 20.1 | 17.4 | **34.6** | 46.4 | 44.2 | 14.5 | 21.3 | 19.0 |
| | Interleaved | 6.5 | 11.7 | 9.9 | 8.6 | 14.9 | 12.6 | 17.1 | 29.6 | 28.4 | 32.4 | 44.3 | 43.2 | 16.1 | 25.1 | 23.5 |
| | Labeled | 6.2 | 11.3 | 9.7 | **9.1** | 15.5 | 12.6 | 18.1 | 31.6 | 30.7 | 33.5 | 45.4 | 43.8 | 16.7 | 26.0 | 24.2 |
| | Multi-task Labeled | 6.2 | 11.1 | 9.5 | 8.4 | 14.5 | 11.7 | 16.0 | 28.3 | 27.2 | 31.5 | 42.7 | 41.9 | 15.5 | 24.2 | 22.6 |
| | Weighted | **7.6** | **13.4** | **12.2** | 7.6 | **15.6** | **18.7** | **24.2** | **41.2** | **44.1** | 33.2 | **47.1** | **46.9** | **18.2** | **29.3** | **30.5** |
| GPT-2 | PersonaChat (single) | 7.1 | 10.8 | 9.2 | 8.7 | 13.9 | 10.5 | 8.8 | 15.5 | 12.2 | 45.0 | 58.8 | 56.8 | 17.4 | 24.7 | 22.2 |
| | Concatenated | 8.4 | 12.6 | 11.0 | 10.8 | 17.2 | 13.7 | 13.4 | 22.4 | 23.3 | 43.0 | 56.3 | 55.7 | 18.9 | 27.1 | 25.9 |
| | Interleaved | 14.0 | 22.3 | 21.3 | 12.2 | 20.1 | 19.3 | 25.8 | 44.3 | 48.3 | 44.2 | 57.6 | 58.0 | 24.0 | 36.1 | 36.7 |
| | Labeled | 13.6 | 21.6 | 20.5 | **13.1** | 21.1 | 20.3 | 25.8 | 45.0 | 49.6 | **45.1** | **59.0** | **59.6** | 24.4 | 36.7 | 37.5 |
| | Multi-task Labeled | 13.4 | 21.4 | 20.4 | 12.7 | 20.6 | 20.1 | 25.4 | 43.8 | 47.6 | 44.0 | 57.3 | 57.4 | 23.9 | 35.8 | 36.4 |
| | Weighted | **14.5** | **23.4** | **23.4** | 11.9 | **22.3** | **25.2** | **29.2** | **48.6** | **52.5** | 42.4 | 55.8 | 57.6 | **24.5** | **37.5** | **39.7** |

Table 10: BLEU (‰), ROUGE (‰) and DF-F1 (‰) scores for baselines and proposed methods fine-tuned on 4 corpora (**without** stop words). DF-F1 is ROUGE F1 weighted by test-set $\alpha$DF

## E Full results of $\alpha$DF for generated responses from multiple corpora

| Model | Corpus / Method | Test set: OSDB | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | **7.01** | **9.66** | 3.75 | 3.75 | 2.82 | 2.86 | 3.59 | 3.75 |
| LSTM | PersonaChat (single) | 2.92 | 3.40 | 2.40 | 2.82 | 2.27 | 2.51 | 9.18 | 9.91 |
| | Concatenated | 2.92 | 3.35 | 2.49 | 2.94 | 2.41 | 2.71 | 7.65 | 8.55 |
| | Interleaved | 3.88 | 4.13 | 2.45 | 2.54 | 2.89 | 2.87 | 4.98 | 5.31 |
| | Labeled | 3.94 | 4.16 | 2.37 | 2.44 | 2.71 | 2.70 | 5.01 | 5.34 |
| | Multi-task Labeled | 3.78 | 4.02 | 2.41 | 2.49 | 2.91 | 2.88 | 5.02 | 5.36 |
| | Weighted | **5.60** | **6.29** | 2.65 | 2.84 | 2.89 | 2.84 | 4.14 | 4.47 |
| GPT-2 | PersonaChat (single) | 2.76 | 3.15 | 2.30 | 2.66 | 2.24 | 2.51 | 10.53 | 11.09 |
| | Concatenated | 3.07 | 3.59 | 2.52 | 2.96 | 2.30 | 2.55 | 8.75 | 9.35 |
| | Interleaved | 4.86 | 5.78 | 2.63 | 2.67 | 2.69 | 2.66 | 4.77 | 5.04 |
| | Labeled | 4.86 | 5.77 | 2.61 | 2.66 | 2.67 | 2.64 | 4.76 | 5.04 |
| | Multi-task Labeled | 4.81 | 5.70 | 2.60 | 2.64 | 2.69 | 2.65 | 4.83 | 5.1 |
| | Weighted | **6.02** | **7.46** | 2.71 | 2.83 | 2.47 | 2.48 | 4.12 | 4.38 |

(a) $\alpha\text{DF}_d$ scores for generated responses from OSDB

| Model | Corpus / Method | Test set: Twitter | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 3.97 | 4.07 | **9.07** | **11.01** | 3.24 | 3.40 | 3.64 | 3.80 |
| LSTM | PersonaChat (single) | 2.79 | 3.21 | 2.78 | 3.36 | 2.35 | 2.59 | 8.60 | 9.18 |
| | Concatenated | 2.62 | 3.12 | 3.55 | 4.31 | 2.30 | 2.71 | 7.97 | 8.69 |
| | Interleaved | 3.28 | 3.68 | 4.66 | 4.95 | 3.11 | 3.34 | 4.11 | 4.51 |
| | Labeled | 3.30 | 3.68 | 4.97 | 5.27 | 3.00 | 3.24 | 3.89 | 4.26 |
| | Multi-task Labeled | 3.31 | 3.68 | 4.47 | 4.73 | 3.14 | 3.36 | 4.08 | 4.49 |
| | Weighted | 3.10 | 3.62 | **9.92** | **10.10** | 2.79 | 3.01 | 3.79 | 4.30 |
| GPT-2 | PersonaChat (single) | 2.74 | 3.04 | 2.87 | 3.33 | 2.45 | 2.66 | 9.47 | 9.77 |
| | Concatenated | 2.87 | 3.28 | 3.32 | 3.94 | 2.41 | 2.65 | 8.21 | 8.68 |
| | Interleaved | 3.42 | 3.67 | 4.59 | 5.08 | 3.05 | 3.13 | 4.39 | 4.68 |
| | Labeled | 3.48 | 3.74 | 4.66 | 5.16 | 3.08 | 3.19 | 4.06 | 4.35 |
| | Multi-task Labeled | 3.41 | 3.66 | 4.63 | 5.11 | 3.08 | 3.15 | 4.37 | 4.65 |
| | Weighted | 3.58 | 4.01 | **8.13** | **8.84** | 2.59 | 2.79 | 3.68 | 4.07 |

(b) $\alpha\text{DF}_d$ scores for generated responses from Twitter

| Model | Corpus / Method | Test set: Ubuntu | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 2.69 | 2.74 | 2.96 | 2.85 | **19.36** | **23.20** | 2.67 | 2.78 |
| LSTM | PersonaChat (single) | 2.71 | 3.28 | 2.41 | 2.89 | 2.74 | 3.06 | 8.55 | 9.09 |
| | Concatenated | 2.61 | 2.89 | 2.27 | 2.53 | 7.60 | 7.74 | 5.59 | 5.99 |
| | Interleaved | 2.91 | 3.19 | 2.30 | 2.36 | 11.78 | 11.27 | 3.70 | 4.01 |
| | Labeled | 3.03 | 3.38 | 2.28 | 2.36 | 12.46 | 11.75 | 3.45 | 3.75 |
| | Multi-task Labeled | 2.91 | 3.17 | 2.30 | 2.35 | 11.19 | 10.72 | 3.77 | 4.09 |
| | Weighted | 2.16 | 2.84 | 2.05 | 2.16 | **27.73** | **25.42** | 2.68 | 3.01 |
| GPT-2 | PersonaChat (single) | 2.60 | 2.85 | 2.31 | 2.64 | 4.12 | 4.64 | 8.27 | 8.42 |
| | Concatenated | 2.67 | 3.03 | 2.45 | 2.82 | 6.54 | 7.10 | 7.04 | 7.37 |
| | Interleaved | 2.73 | 3.05 | 2.22 | 2.37 | 15.67 | 16.02 | 3.08 | 3.41 |
| | Labeled | 2.68 | 3.03 | 2.17 | 2.35 | 16.73 | 17.02 | 2.90 | 3.24 |
| | Multi-task Labeled | 2.73 | 3.06 | 2.22 | 2.37 | 15.45 | 15.78 | 3.12 | 3.44 |
| | Weighted | 2.26 | 2.56 | 2.16 | 2.28 | **25.73** | **24.42** | 2.37 | 2.60 |

(c) $\alpha\text{DF}_d$ scores for generated responses from Ubuntu

| Model | Corpus / Method | Test set: PersonaChat | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | OSDB | | Twitter | | Ubuntu | | PersonaChat | |
| | | $\alpha$DF Calculated From: | | | | | | | |
| | | Train | Test | Train | Test | Train | Test | Train | Test |
| Test Set (Standard Score) | | 3.32 | 3.23 | 3.18 | 3.04 | 2.67 | 2.69 | **9.45** | **12.00** |
| LSTM | PersonaChat (single) | 2.59 | 3.02 | 2.31 | 2.73 | 2.15 | 2.35 | 11.86 | 12.62 |
| | Concatenated | 2.47 | 2.84 | 2.29 | 2.76 | 2.06 | 2.33 | 12.52 | 13.61 |
| | Interleaved | 2.57 | 2.92 | 2.30 | 2.71 | 2.17 | 2.45 | 11.48 | 12.52 |
| | Labeled | 2.51 | 2.88 | 2.27 | 2.68 | 2.08 | 2.36 | 12.06 | 13.11 |
| | Multi-task Labeled | 2.55 | 2.91 | 2.29 | 2.74 | 2.15 | 2.43 | 11.45 | 12.59 |
| | Weighted | 2.21 | 2.44 | 2.13 | 2.41 | 2.04 | 2.18 | **17.65** | **18.31** |
| GPT-2 | PersonaChat (single) | 2.54 | 2.79 | 2.28 | 2.59 | 2.12 | 2.34 | 12.85 | 13.74 |
| | Concatenated | 2.58 | 2.99 | 2.41 | 2.79 | 2.16 | 2.39 | 12.08 | 12.99 |
| | Interleaved | 2.64 | 2.89 | 2.37 | 2.65 | 2.23 | 2.42 | 12.13 | 12.87 |
| | Labeled | 2.57 | 2.84 | 2.32 | 2.62 | 2.16 | 2.37 | 12.37 | 13.10 |
| | Multi-task Labeled | 2.65 | 2.90 | 2.37 | 2.65 | 2.22 | 2.42 | 12.14 | 12.86 |
| | Weighted | 2.39 | 2.63 | 2.27 | 2.52 | 2.02 | 2.17 | **15.96** | **17.07** |

(d) $\alpha\text{DF}_d$ scores for generated responses from PersonaChat

Table 11: Full results of $\alpha\text{DF}_d$ scores for generated responses from multiple corpora

## F Example of human evaluation system

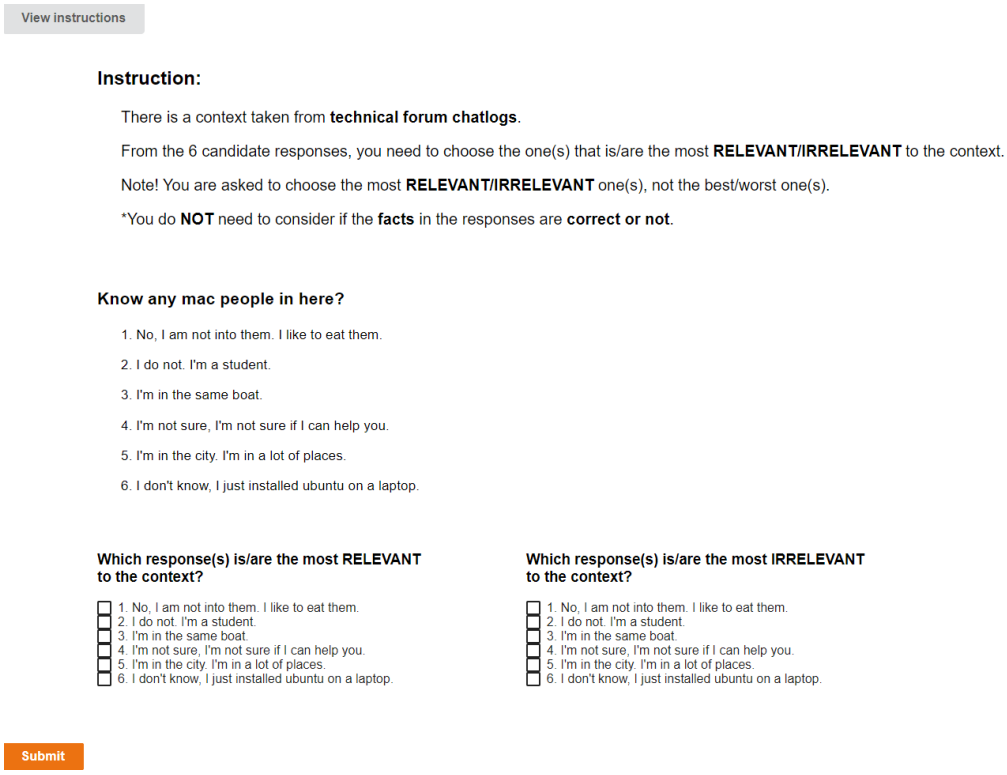



Figure 3: Human evaluation system for Ubuntu contexts

## G Examples of generated responses

**OSDB Context:** You know the main reason parents kill kids?

**PersonaChat (single):** I am not sure. I am a vegan.
**Concatenated:** I know. I've a daughter that just turned 18.
**Interleaved:** Because they're afraid of what they're gonna do.
**Labeled:** Because they're not going to be able to handle it.
**Multi-Task Labeled:** Because they're afraid of what they're gonna do.
**Weighted:** Because they're afraid of the consequences.

**OSDB Context:** The big banks will fail, triggering a domino effect.

**PersonaChat (single):** I am not sure yet. I am sure they will fail.
**Concatenated:** I'm not sure. I'm not sure what the big banks are doing.
**Interleaved:** What?
**Labeled:** I'm not sure what you're talking about.
**Multi-Task Labeled:**I'm not sure what you mean.
**Weighted:** What's going on?

**OSDB Context:** Rigor mortis has yet to set in. His core body temperature remains at 98 degrees. His skin color, lividity is healthy.

**PersonaChat (single):**I'm not sure if I like him or not. I like to eat meat.
**Concatenated:**That is good. I like to cook. I am a chef.
**Interleaved:** He is a very good doctor.
**Labeled:** He's a good guy.
**Multi-Task Labeled:** He is a man of great character.
**Weighted:** He's been in the hospital for two weeks.

**Twitter Context:** Hillary ran away from rain today. Trump is back on stage minutes after assassination attempt.

**PersonaChat (single):** I'm not sure what that means.
**Concatenated:** That is so sad. I hope she gets better soon.
**Interleaved:** I'm so sorry. I'm so sorry.
**Labeled:** I'm so sorry.
**Multi-Task Labeled:** I'm not sure what happened to her.
**Weighted:** She's a liar.

**Twitter Context:** It was gruesome & extremely rare to have two witnesses. It was linked to epstein. Media didn't report it.

**PersonaChat (single):** I'm sorry to hear that. I'm a nurse.
**Concatenated:** I'm not sure what that means. I'm a big fan of the show.
**Interleaved:** I'm not sure what happened to the epstein case.
**Labeled:** I'm not sure what the link was, but it was a link to a video.
**Multi-Task Labeled:** I'm not sure what happened to the epstein case.
**Weighted:** The media is biased and biased.

**Twitter Context:** This is 100% true, but can we talk about it tomorrow and make argentina the focus today? Something needs to stick.

**PersonaChat (single):** Yes, I think so. I'm going to go to the gym tomorrow.
**Concatenated:** I'm going to go see a movie.
**Interleaved:** I'm not sure what that's. I'm not sure what that's.
**Labeled:** I'm not sure what that means. I'm not sure what that means.
**Multi-Task Labeled:** I'm not sure what you mean by that. I'm not sure what you mean by that.
**Weighted:** I'm not sure if I can talk about it tomorrow.

**Ubuntu Context:** What are you going to do on the remote system exactly?

**PersonaChat (single):** I am going to be a pilot. I am going to fly planes.
**Concatenated:** I am going to go to the beach.
**Interleaved:** I'm going to install the latest version of Ubuntu.
**Labeled:** I'm going to install a new kernel.
**Multi-Task Labeled:** I'm going to install the latest version of Ubuntu.
**Weighted:** I'm trying to connect to the server.

**Ubuntu Context:** Know any mac people in here?

**PersonaChat (single):** No, I am not into them. I like to eat them.
**Concatenated:** I do not. I'm a student.
**Interleaved:** I'm in the same boat.
**Labeled:** I'm not sure, I'm not sure if I can help you.
**Multi-Task Labeled:** I'm in the city. I'm in a lot of places.
**Weighted:** I don't know, I just installed ubuntu on a laptop.

**Ubuntu Context:** Type sudo name of application (in short)

**PersonaChat (single):** I like to play games on my computer.
**Concatenated:** I did not see that. I am a programmer.
**Interleaved:** I have a root shell, but I can't login.
**Labeled:** I have a root shell.
**Multi-Task Labeled:** I don't know what that is.
**Weighted:** It is Sudo apt-get install xubuntu-desktop.

**PersonaChat Context:** That must be hard. I like a beer after driving my nissan pathfinder.

**PersonaChat (single):** I like to go to the movies. I have been to a few.
**Concatenated:** I am a huge gamer, I play dungeons and dragons on my computer.
**Interleaved:** I like to drive my nissan camry.
**Labeled:** I like to go to the movies with my friends.
**Multi-Task Labeled:** I like to drive my prius.
**Weighted:** I love beer too. I love to drink it while I am driving.

**PersonaChat Context:** A food bank near my house. Do you ever volunteer?

**PersonaChat (single):** I do, I love to volunteer at the soup kitchen.
**Concatenated:** Yes I do. I love to help people.
**Interleaved:** I do. I have a lot of friends.
**Labeled:** I do. I am a volunteer at the local soup kitchen.
**Multi-Task Labeled:** I do. I love it. I volunteer at the animal shelter.
**Weighted:** I do. I love volunteering at the food pantry.

**PersonaChat Context:** Yes! My son is in junior high and I just started letting him watch them too.

**PersonaChat (single):** I am a mom of two boys. I have a son.
**Concatenated:** I am a mom, I have three kids.
**Interleaved:** I love kids, I have a son who is in high school.
**Labeled:** I bet he is a good kid. I am a teacher.
**Multi-Task Labeled:** I am sorry to hear that. I am a teacher.
**Weighted:** I bet you are a good mom.

Table 12: Responses generated from GPT-2 fine-tuned on 4 corpora with multiple methods

# Paper II

# EVALUATING AND IMPROVING CONTEXT ATTENTION DISTRIBUTION ON MULTI-TURN RESPONSE GENERATION USING SELF-CONTAINED DISTRACTIONS

Yujie Xing and Jon Atle Gulla

Norwegian University of Science and Technology
{yujie.xing, jon.atle.gulla}@ntnu.no

***ABSTRACT***

*Despite the rapid progress of open-domain generation-based conversational agents, most deployed systems treat dialogue contexts as single-turns, while systems dealing with multi-turn contexts are less studied. There is a lack of a reliable metric for evaluating multi-turn modelling, as well as an effective solution for improving it. In this paper, we focus on an essential component of multi-turn generation-based conversational agents:* ***context attention distribution****, i.e. how systems distribute their attention on dialogue's context. For evaluation of this component, We introduce a novel attention-mechanism-based metric:* ***DAS ratio****. To improve performance on this component, we propose an optimization strategy that employs self-contained distractions. Our experiments on the Ubuntu chatlogs dataset show that models with comparable perplexity can be distinguished by their ability on context attention distribution. Our proposed optimization strategy improves both non-hierarchical and hierarchical models on the proposed metric by about* $10\%$ *from baselines.*

***KEYWORDS***

*Natural Language Processing, Response Generation, Dialogue System, Conversational Agent, Multi-Turn Dialogue System*

## 1. INTRODUCTION

In recent years, generation-based conversational agents have shown a lot of progress, while multi-turn generation-based conversational agents are still facing challenges. Most recent work ignores multiturn modelling by considering a multi-turn context as a 1-turn context [1, 2]. Some works try to deal with multi-turn modelling using modified attention mechanisms, hierarchical structures, utterance tokens, etc. [3, 4, 5]. The main difference between multi-turn conversational agents and regular (1-turn) conversational agents is that instead of dealing with an utterance in a context on the *word-level*, multi-turn models deal with a dialogue on the *utterance-level*, so that models can understand an utterance as a whole and focus on important *utterances* rather than important *words*. An example of important/unimportant utterances existing in the same context is given by Table 1.

Table 1: An example of important utterances and unimportant utterances under the same context in the Ubuntu chatlog dataset [6]. Unimportant utterances are marked in red.

| User | Utterances |
|---|---|
| Taru | Haha sucker. |
| Kuja | ? |
| Taru | Anyways, you made the changes right? |
| Kuja | Yes. |
| Taru | Then from the terminal type: sudo apt-get update |

| Kuja | I did. |
|---|---|

In this example, the first two utterances ("Haha sucker." and "?") are unimportant utterances that are irrelevant to the main topic of the context. Human dialogues naturally contain many of these unimportant utterances. These utterances do not distract humans from understanding the main idea of the context, since humans can easily ignore them and focus instead on important utterances; however, a model usually lacks this capability and can be distracted by these utterances, resulting in a lower performance in generating *relevant* responses to the main topic of a context. Therefore, it is crucial that a multi-turn model can decide which utterances in the context are important and which are unimportant, and distribute its attention accordingly. In this paper, we define the research topic as **context attention distribution**, which denotes how much attention is distributed respectively to important and unimportant utterances in a context. A model with a good performance on context attention distribution should pay more attention to important utterances and less attention to unimportant utterances.

Recent work lacks a measurement for the performance of multi-turn modelling. Common metrics rely on general evaluation metrics such as BLEU [7], which measures the quality of generated responses. These metrics cannot directly describe a model's ability on dealing with multi-turn contexts, since the quality of generated responses is influenced by many aspects. Better performance in dealing with multi-turn context may result in better general performance; however, a better general performance does not necessariy mean that the model has a better ability on dealing with multi-turn contexts. Thus, as a supplementary to general evaluation metrics like BLEU, we propose a metric that measures a conversational agent's performance on context attention distribution, which is specifically designed for evaluating a model's performance on multi-turn modelling.

Since most multi-turn conversational agents have the attention mechanism and rely on it to distribute attention to different utterances in a context, we propose **distracting test** as the evaluation method to examine if a model pays more attention to the important utterances. The test adds unrelated utterances as distractions to the context of each dialogue and compares the attention scores of distracting utterances (i.e., unimportant utterances) and original utterances (i.e., important utterances). The ratio of the average attention score of distracting utterances and original utterances is defined as the distracting attention score ratio (**DAS ratio**). We use DAS ratio as the evaluation metric for a model's performance on context attention distribution. A model with good capability on context attention distribution should have higher scores on original utterances and lower scores on distracting utterances, thus a lower DAS ratio.

Furthermore, we propose a self-contained optimization strategy to improve a conversational agent's performance on context attention distribution. For each dialogue, we randomly pick some utterances from the training corpus outside the current dialogue as self-contained distractions, and insert them into the current dialogue with different levels of possibilities. The attention paid to these distractions is minimized during the training process through multi-task learning. With this optimization strategy, a model learns to distribute less attention to unimportant utterances and thus more attention to important utterances.

In this paper, we examine the following research questions: 1) How do existing multi-turn modelling structures perform on context attention distribution? 2) Can the proposed optimization strategy improve a model's performance on context attention distribution? 3) Which probability level is the best for inserting distractions in the proposed optimization strategy?

Our contributions are as follows:

(1) We deal with a less studied problem: evaluating and improving context attention distribution for multi-turn conversational agents.

(2) We propose a novel evaluation metric for multi-turn conversational agents: DAS ratio. It measures a model's performance on context attention distribution, i.e. the capability of

distributing more attention to important utterances and less to unimportant ones.

(3) We propose an optimization strategy that minimizes the attention paid to self-contained distractions during the training process, and thus makes the model try to pay less attention to unimportant utterances. The strategy can easily be added and adapted to existing models.

Extensive experiments on 23 model variants and 9 distracting test sets show an overall improvement in the performance on context attention distribution for the proposed strategy. We will share our code for reproducibility.

Related work is introduced in Section 2. In Section 3, we introduce our base models and proposed methods. We show our experiments settings in Section 4 and results in Section 5. Finally, we give a conclusion in Section 6.

## 2. RELATED WORKS

Common evaluation metrics for conversational agents measure the similarity between the generated responses and the gold responses. Liu et al. [8] summarizes commonly used metrics: word overlap-based metrics (e.g. BLEU) and embedding-based metrics. Bruni et al. [9] propose an adversarial evaluation method, which uses a classifier to distinguish human responses from generated responses. Lowe et al. [10] propose a model that simulates human scoring for generated responses. Zemlyanskiy et al. [11] examine the quality of generated responses in a different direction: how much information the speakers exchange with each other. Recently, Li et al. [5] propose a metric that evaluates the human-likeness of the generated response by measuring the gap between the corresponding semantic influences. Different from the above, our proposed evaluation metric is based on the attention mechanism and is intended to measure a model's performance on context attention distribution.

Most generation-based conversational agents apply simple concatenation for multi-turn conversation modelling [2, 1], which regards a multi-turn context as a 1-turn utterance. Some works try to model multi-turn conversations through the hierarchical structure: Serban et al. [3, 4] first introduce the hierarchical structure to dialogue models. Tian et al. [12] evaluate different methods for integrating context utterances in hierarchical structures. Zhang et al. [13] further evaluate the effectiveness of static and dynamic attention mechanism. Gu et al. [14] apply a similar hierarchical structure on Transformer, and propose masked utterance regression and distributed utterance order ranking for the training objectives. Different from hierarchical models, Li et al. [5] encode each utterance with a special token $[C]$ and apply a flow module to train the model to predict the next $[C]$; then they use semantic influence (the difference of the predicted and original tokens) to support generation. In our paper, instead of modelling the relations of inter-context utterances as [14] or the dialogue flow as [5], our optimization strategy improves multi-turn modelling by distinguishing important/unimportant utterances directly on the attention mechanism.

## 3. METHODS

Our proposed evaluation metric and optimization strategy can work on attention mechanisms including Transformers. In this paper, we choose an LSTM Seq2Seq model with attention mechanism [15, 16, 17] as the base model, since most hierarchical structured multi-turn conversational agents are based on LSTM [3, 4, 12, 13] while few are based on Transformers.

The basic task of generation-based conversational agents is to predict the next token given all the past and current tokens from the context and response, and to make the predicted response as similar to the original response as possible. Formally, the probability of response $Y$ given context $X$ is predicted as:

$$P(Y|X) = \prod_{t=1}^{n} p(y_t|y_1, \ldots, y_{t-1}, X), \tag{1}$$

where $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ are a context-response pair.

### 3.1. LSTM Seq2Seq Model with Attention

We simplify an LSTM unit as $LSTM$, and we denote the attention version of an LSTM with an asterisk ($LSTM^*$). They are well introduced in previous work [18]. We calculate the hidden vector $h_t$ at step $t$ as:

$$h_t = LSTM^*(h_{t-1}, E(z_t), c_{t-1}), \tag{2}$$

where $h_{t-1} \in \mathbb{R}^{dim}$ is the hidden vector at step $t$-1, $dim$ is the dimensionality of hidden vectors, and $E(z_t)$ is the word embedding for token $z_t \in \{x_1, ..., x_m, y_1, ..., y_{n-1}\}$. $c_{t-1}$ is the context vector at step $t$-1, and it is input to the next step $t$ *only in the decoder*. Each $h_t$ and $c_t$ of the current step $t$ are combined through a linear layer and an activation to predict the next token.

### 3.2. Attention Mechanism & Utterance Integration (UI)

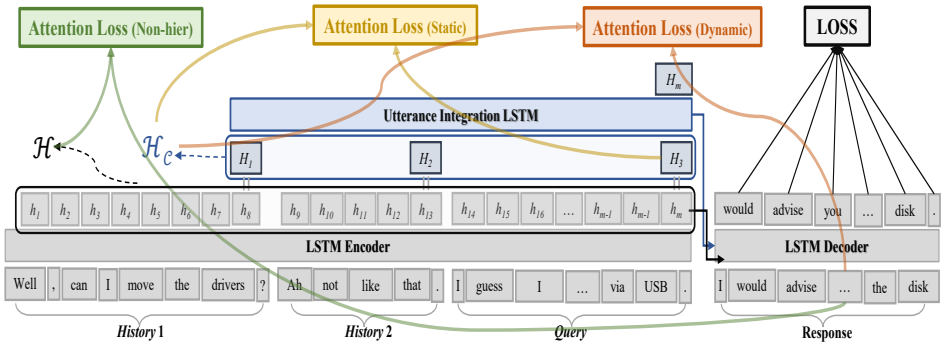



Figure 1: Structure of non-hierarchical, static and dynamic attention loss.

We examine both non-hierarchical and hierarchical structures. For hierarchical structures, following [13], we develop two attention mechanisms: static and dynamic. Following [12], we develop models that are both with and without utterance integration LSTM units.

For the non-hierarchical structured model, there are no hidden vectors for utterances. All hidden vectors of tokens in the encoder are concatenated and used in the attention mechanism. Denoting the concatenated vector $\mathcal{H} = [h_1, h_2, ..., h_m]$, we calculate the context vector $c_t$ for each decoding step $t$ as:

$$c_t = \mathcal{H} \cdot (\text{softmax}(\mathcal{H}^\top \cdot h_t)) \,. \tag{3}$$

For the hierarchical models, we use the hidden vector of each utterance's last token as the hidden vector of the utterance, and we discard the hidden vectors for the other tokens. Thus, compared to the non-hierarchical structured model, we have much fewer hidden vectors from the encoder.

The context vector of static attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of the last utterance in the context. Denoting the hidden vector of $k$th utterance as $H_k$, and the hidden vector of the last utterance in the context as $H_q$, we have the context's concatenated vector $\mathcal{H}_\mathcal{C} = [H_1, H_2, ..., H_q]$. We calculate the context vector $c_t$ for

static attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot H_q)), \tag{4}$$

where it is easy to see that the static context vector remains unchanged by the decoder.

The context vector of dynamic attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of each token in the decoding step. We calculate the context vector $c_t$ for dynamic attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot h_t)) \, . \tag{5}$$

Compared to the static attention mechanism, the context vector $c_t$ varies at each decoding step.

Finally, with the utterance integration LSTM unit, we calculate $H_m$ from $H_1$, $H_2$, ... $H_q$:

$$H_m = LSTM(H_1, H_2, ..., H_q). \tag{6}$$

For models with utterance integration (UI), $H_m$ is input to the first step of the decoder, while for models without UI, regular $h_m$ is input instead.

### 3.3. Distracting Test & Attention Score (AS)

We examine if a multi-turn conversational agent distributes more attention to important utterances through the distracting test and attention scores.

In the distracting test, for each dialogue before the end of the context, we insert several utterances that are irrelevant to the main idea of the dialogue as distractions. These utterances are named *distracting utterances*, and they can be randomly picked utterances from the training corpus (**random**), be formed by frequent words from the training corpus (**frequent**), or be formed by rare words from the training corpus (**rare**). We compare the attention scores of the distracting utterances with the attention scores of the original utterances. A well-performing model should distribute less attention to the distracting utterances while more attention to the original utterances. For an utterance $H_k$, the corresponding attention score $\text{AS}(H_k)$ is calculated as:

$$\text{AS}(H_k) = \begin{cases} \dfrac{m}{q} \cdot \texttt{mean}_t \left( \dfrac{\sum_{h_i \in H_k} \exp(h_i^\top \cdot h_t)}{\sum_{i=1}^{m} \exp(h_i^\top \cdot h_t)} \right) & \text{Non-hierarchical} \\ \dfrac{q \cdot \exp(H_k^\top \cdot H_q)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot H_q)} & \text{Static attention} \\ \texttt{mean}_t \left( \dfrac{q \cdot \exp(H_k^\top \cdot h_t)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot h_t)} \right) & \text{Dynamic attention} \end{cases} . \tag{7}$$

$h_i$ denotes hidden vectors from the encoding steps and $h_t$ denotes hidden vectors from the decoding steps. $m$ is the number of tokens in a context, and $q$ denotes the number of utterances in a context. Note that for non-hierarchical models we multiply by an $m$ in each $\text{AS}(H_k)$ to avoid bias caused by the total number of tokens in different contexts. Similarly for hierarchical models, we multiply by a $q$ in each $\text{AS}(H_k)$ to avoid bias caused by the number of total utterances in different contexts. As a result, for an utterance $H_q$, $\text{AS}(H_q)$ will be $100\%$ (or approximately $100\%$ for non-hierarchical models) if the model assigns $H_q$ an about average attention score among all utterances.

We denote the last utterance in a context as *Query* and the rest of utterances in the context as *History*. Since different models have different scalars on attention scores, we calculate the average AS for all distracting utterances and all *History* in each dialogue, and use the ratio of them for

evaluation. This ratio is denoted as distracting attention score ratio (**DAS ratio**), which measures a model's ability on context attention distribution:

$$\text{DAS ratio} = \texttt{mean}_{d \in D} \left( \frac{\texttt{mean}(\text{AS}(H_{\text{Distraction}}))}{\texttt{mean}(\text{AS}(H_{\textit{History}}))} \right), \tag{8}$$

where $d$ means a single dialogue, and $D$ denotes all dialogues in a test set. $H_{\text{Distraction}}$ denotes distracting utterances, and $H_{\textit{History}}$ denotes utterances in *History*.

### 3.4. Optimization with Self-Contained Distractions on Attention Mechanism

To train a conversational model to distribute more attention to important and less attention to unimportant utterances, we propose the following optimization strategy: 1) For each dialogue, we select some random utterances from other dialogues in the training corpus as **self-contained distractions**. We decide whether to insert these distractions into the current dialogue or not stochastically by a probability level. We denote the probability level as the training inserting probability. The locations of inserting distractions are randomly decided, while the locations are always before *Query* (the last utterance of the context). 2) We create a bitmask $M$ to track whether an utterance is original (0) or distracting (1). During the training period, the model uses the bitmask to calculate the attention loss $\mathcal{L}^t_{\text{attention}}$, which is summed up with the loss from the response generator. For each decoding step $t$, the attention loss is calculated as:

$$\mathcal{L}^t_{attention} = \begin{cases} \texttt{MSE}(\text{softmax}(\mathcal{H}^\top \cdot h_t) \circ M, 0) & \text{Non-hierarchical} \\ \texttt{MSE}(\text{softmax}(\mathcal{H}^\top_{\mathcal{C}} \cdot H_q) \circ M, 0) & \text{Static attention} \\ \texttt{MSE}(\text{softmax}(\mathcal{H}^\top_{\mathcal{C}} \cdot h_t) \circ M, 0) & \text{Dynamic attention} \end{cases} \tag{9}$$

where $\circ$ means Hadamard product, or elementwise multiplication. As shown in Equation (9), our goal is to minimize the attention assigned to all the self-contained distractions. During the distracting test, no bitmask is offered to the model. The illustration of attention loss on both non-hierarchical and hierarchical models is shown in Figure 1.

## 4. EXPERIMENTS

In this section, we introduce the setups of the experiment.

### 4.1. Dataset

We use the Ubuntu chatlogs dataset [6] as the training and testing corpus, which contains dialogues about solving technical problems of Ubuntu. We choose this dataset because the dialogues have both technical topics and casual chats, meaning that it is easier to distinguish important/unimportant utterances than datasets whose topics are consistent. We use about $0.48M$ dialogues for training, $20K$ dialogues for validation, and $10K$ dialogues for testing. These are the original settings of the Ubuntu chatlogs dataset. We removed all single-turn dialogues.

### 4.2. Training

Our methods are built on an LSTM Seq2Seq model with attention mechanism. We used Pytorch [19] for implementation. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, a learning rate of 1.0, and a gradient clip threshold of 5. The vocabulary size is 25000 and the dropout rate is 0.2. The learning rate is halved when the perplexity stops dropping, and the training is stopped when the model converges.

### 4.3. Examined Models

We examine our proposed evaluation metric on 5 models: non-hierarchical LSTM (Non-hier), static attention without utterance integration LSTM unit (Static), static attention with utterance integration LSTM unit (StaticUI), dynamic attention without utterance integration LSTM unit (Dynamic), and dynamic attention with utterance integration LSTM unit (DynamicUI). In addition, we examine our proposed optimization strategy on these 5 models with 3 training inserting probabilities–0.5, 0.7, and 1.0. Models with a training inserting probability of 0 are regarded as baselines. For comparison, we pick the best overall model and train the model with self-contained distractions but without training on the attention loss (Non-atten-loss), i.e. the model does not know which utterances are distractions. In total, we train and evaluate 23 model variants.

### 4.4. Evaluation

Table 2: Examples of distracting test sets. Distracting utterances are marked red.

| | **Random: 0.5** | **Random: 0.7** | **Random: 1.0** |
|---|---|---|---|
| *History* | \ | Well, can I move the drives? | Yes. |
| | Or kill all speedlink. | Anyways, you made the changes right? | Well, can I move the drives? |
| | Well, can I move the drives? | Ah not like that. | Then from the terminal type: sudo apt-get update. |
| | Ah not like that. | I did. | Ah not like that. |
| | **Frequent: Begin** | **Frequent: Middle** | **Frequent: End** |
| *History* | Why should I help you? | Well, can I move the drives? | Well, can I move the drives? |
| | I have my right. | Why should I help you? | Ah not like that. |
| | Well, can I move the drives? | I have my right. | Why should I help you? |
| | Ah not like that. | Ah not like that. | I have my right. |
| | **Rare: Begin** | **Rare: Middle** | **Rare: End** |
| *History* | Would you have lunch? | Well, can I move the drives? | Well, can I move the drives? |
| | I should have lunch. | Would you have lunch? | Ah not like that. |
| | Well, can I move the drives? | I should have lunch. | Would you have lunch? |
| | Ah not like that. | Ah not like that. | I should have lunch. |
| *Query* | **I guess I could just get an enclosure and copy via USB.** | | |
| Response | **I would advise you to get the disk.** | | |

For the distracting test, we set the number of distracting utterances for each dialogue to 2. We chose 2 to make the distracting utterances a complete turn and to make the number of distracting utterances the minimum, since dialogues from the corpus normally have only 4 to 8 utterances in the contexts. We have 3 distracting test sets. 1) Random distracting test set: distracting utterances in this test set are randomly picked from the training corpus (outside the current dialogue), and they are randomly picked in every evaluation step, which means that there is no pre-prepared random distracting test set. 2) Frequent distracting test set: distracting utterances in this test set are formed by frequent words in the training corpus, but these utterances do not appear in the training corpus. In our experiments, we use "why should I help you" and "I have my right" as examples of distracting utterances with frequent words. 3) Rare distracting test set: distracting utterances in this test set have words that are rare in the training corpus, and these utterances do not appear in the training corpus. In our experiments, we use "would you have lunch?" and "I should have lunch"

as examples of distracting utterances with rare words.

In the distracting test, we insert distracting utterances into different locations. For 1) random, we insert utterances to a random location before *Query* in each context. Similar to the optimization strategy, we use different probability levels to decide whether a distracting utterance is to be inserted or not. We denote these as testing inserting probability. In our experiments, we set the probability levels to be 0.5, 0.7, and 1.0. We expect the model to perform stably on all different probability levels. For 2) frequent and 3) rare, we have three kinds of inserting locations: at the beginning of a context (marked as Begin), in the middle of the context (marked as Middle), and at the end of the context (before *Query* and after *History*, marked as End). In total, we have 9 test sets for evaluation. See Table 2 for the example of each test set.

## 5. RESULTS AND DISCUSSIONS

Table 3 illustrates the main results on DAS ratios. It shows the DAS ratios of 23 trained model variants on 9 distracting test sets. Figure 2 shows the DAS ratios of 3 example model variants (StaticUI with training inserting probability of 0.0 as the baseline, Non-atten-loss StaticUI with training inserting probability of 0.7, and StaticUI with training inserting probability of 0.7) on 9 distracting test sets. Table 4, Table 5 and Table 6 show the detailed results on average Attention Score (average AS) of distracting utterances and average AS of *History*.

In Table 3, we show the perplexity and *History*'s average AS of each model on the non-distracted test set under the "Original" column. Since perplexity scores on the distracting test sets are similar, we show the perplexity scores on the non-distracted test set only. We show the DAS ratios of each model on each of the distracting test sets under the "DAS ratio for distracting test set" column. A lower DAS ratio means that a model distributes less attention to distracting utterances (unimportant utterances) and more attention to the original utterances in *History* (important utterances), from which it can be inferred that the model has better performance on context attention distribution. Both perplexity and DAS ratio are the lower, the better.

### 5.1. Perplexity and Average AS on Non-Distracted Test Set

Perplexity scores are shown in the "Perp." column, under the "Original" column in Table 3. Perplexity scores of the examined 23 models are similar; the Static models trained with our proposed optimization strategy and a higher training inserting probability level achieves slightly better performance than other models.

Average AS are shown in the "Avg." column, under the "Original" column in Table 3. The average AS of *History* tells about a model's attention distribution for *History* and *Query*. A higher score indicates that less attention is distributed to *Query*. Recall that AS of an utterance is $100\%$ (or approximately $100\%$ for non-hierarchical models) if the utterance is paid about average attention among the dialogue. Overall, the models distribute attention of lower than average to *History*, especially for models with static attention (i.e. the Static model and StaticUI model), which distribute more attention to *Query* than non-hierarchical models and models with dynamic attention. This is apparent from the structure of static attention. We also show the results of a StaticUI model without training on the attention loss (Non-atten-loss StaticUI model) as a comparison. The StaticUI model trained with our optimization strategy distributes more attention to *query* than the Non-atten-loss StaticUI model. This is because the optimization strategy decreases the model's attention distributed to distracting utterances in *History*, thus decreasing the overall attention distributed to *History*.

### 5.2. Distracting Test: Random

Results of the random distracting test with different testing inserting probabilities ($0.5$, $0.7$, and $1.0$) are shown in the "Random" column in Table 3. Models with training inserting probabilities of

0.0 (shown in the row where "Prob" is 0.0) are baseline models to which our proposed optimization strategy is not applied. In general, our proposed optimization strategy with training inserting probabilities of 0.5 or 0.7 achieves better performance on DAS ratios (i.e. the models achieve lower DAS ratios) on random distracting test sets of all 3 testing inserting probabilities. The Static model and the DynamicUI model achieves the best performance with a training inserting probability of 0.5, while the Non-hier model, the StaticUI model and the Dynamic model achieve the best performance with a training inserting probability of 0.7. A training inserting probability of 1.0 leads to worse performance. One reason is that it assumes there must be some distracting utterances in a context, while that is not always the case.

Table 3: Results of perplexity (Perp.) and average AS of *History* (Avg.) on the original test set (%) are shown in the "Original" column. We also show results of DAS ratios on 9 distracting test sets and 23 model variants.

| Model | | Original | | DAS ratio on distracting test sets | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Random | | | Frequent | | | Rare | | | | |
| Prob | Structure | Perp. | Avg. | 0.5 | 0.7 | 1.0 | Begin | Middle | End | Begin | Middle | End |
| 0.0 | Non-hier | 43.2 | 91.3 | 0.93 | 0.93 | 0.93 | 0.75 | 0.80 | 0.84 | 0.80 | 0.92 | 1.01 |
| | Static | 44.1 | 61.4 | 0.82 | 0.82 | 0.79 | 0.37 | 0.80 | 1.31 | 0.37 | 0.77 | 1.21 |
| | StaticUI | 44.6 | 57.5 | 0.79 | 0.76 | 0.76 | **0.32** | 0.75 | 1.32 | 0.30 | 0.75 | 1.22 |
| | Dynamic | 45.4 | 81.4 | 0.89 | 0.89 | 0.88 | 0.65 | 0.86 | 1.02 | 0.66 | 0.89 | 1.06 |
| | DynamicUI | 44.7 | 91.6 | 0.94 | 0.94 | 0.93 | 0.72 | 0.84 | 0.86 | 0.73 | 0.93 | 0.97 |
| 0.5 | Non-hier | 43.4 | 87.2 | 0.84 | 0.83 | 0.81 | 0.63 | 0.74 | **0.76** | 0.69 | 0.81 | 0.86 |
| | Static | 44.5 | 66.5 | 0.70 | 0.69 | 0.67 | 0.42 | 0.78 | 1.12 | 0.34 | 0.71 | 0.99 |
| | StaticUI | 44.3 | 47.7 | 0.74 | 0.74 | 0.70 | 0.39 | 0.71 | 1.08 | 0.40 | **0.69** | 0.96 |
| | Dynamic | 44.6 | 81.9 | 0.79 | 0.78 | 0.77 | 0.64 | 0.74 | 0.84 | 0.61 | 0.77 | 0.85 |
| | DynamicUI | 43.9 | 86.7 | 0.82 | 0.81 | 0.80 | 0.60 | 0.84 | 0.87 | 0.61 | 0.80 | 0.83 |
| | Non-atten-loss StaticUI | 44.7 | 71.1 | 0.73 | 0.73 | 0.72 | 0.39 | 0.68 | 0.93 | 0.40 | 0.80 | 1.11 |
| 0.7 | Non-hier | 43.2 | 86.9 | 0.84 | 0.82 | 0.80 | 0.72 | 0.82 | 0.82 | 0.71 | 0.85 | 0.87 |
| | Static | **44.0** | 57.6 | 0.73 | 0.72 | 0.69 | 0.40 | 0.70 | 1.08 | 0.41 | 0.70 | 0.98 |
| | StaticUI | 44.9 | 43.7 | **0.67** | **0.67** | **0.65** | 0.36 | **0.66** | 1.02 | 0.36 | 0.70 | 0.99 |
| | Dynamic | 44.3 | 82.0 | 0.76 | 0.75 | 0.73 | 0.58 | 0.71 | 0.86 | 0.58 | 0.73 | 0.83 |
| | DynamicUI | 44.8 | 85.3 | 0.93 | 0.93 | 0.93 | 0.45 | 0.78 | 0.80 | 0.60 | 0.80 | **0.81** |
| | Non-atten-loss StaticUI | 44.1 | 55.4 | 0.72 | 0.70 | 0.69 | 0.45 | 0.70 | 0.98 | 0.43 | 0.73 | 0.97 |
| 1.0 | Non-hier | 47.3 | 95.9 | 0.91 | 0.90 | 0.90 | 0.84 | 0.86 | 0.85 | 0.85 | 0.87 | 0.88 |
| | Static | **44.0** | 65.4 | 0.70 | 0.70 | 0.68 | 0.49 | 0.74 | 1.08 | 0.46 | 0.71 | 0.88 |
| | StaticUI | 49.6 | 73.5 | 0.96 | 0.95 | 0.94 | 0.66 | 0.86 | 1.53 | **0.21** | 0.86 | 1.50 |
| | Dynamic | 44.7 | 88.8 | 0.79 | 0.78 | 0.77 | 0.63 | 0.75 | 0.82 | 0.65 | 0.77 | 0.82 |
| | DynamicUI | 45.2 | 90.2 | 0.87 | 0.86 | 0.85 | 0.73 | 0.81 | 0.83 | 0.75 | 0.88 | 0.88 |
| | Non-atten-loss StaticUI | 44.1 | 76.5 | 0.72 | 0.71 | 0.69 | 0.49 | 0.74 | 0.98 | 0.49 | 0.77 | 0.98 |

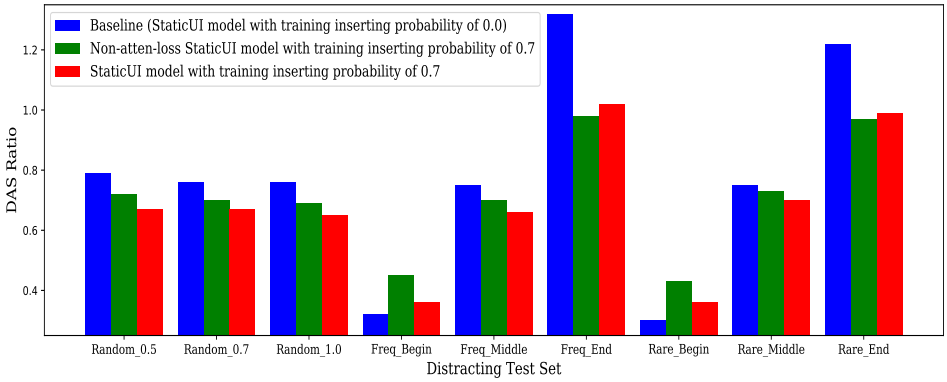



Figure 2: DAS ratios of 3 example model variants on 9 distracting test sets. The lower the DAS ratio, the better the performance.

The StaticUI model with a training inserting probability of 0.7 achieves the best overall performance on DAS ratio. As shown in Figure 2, on all the random distracting test sets (probabilities of 0.5, 0.7, and 1.0), the StaticUI model is better than the baseline StaticUI model and the Non-atten-loss StaticUI model. The baseline model is not trained with any self-contained distractions (training inserting probability is 0.0), and it gets the worst performance. The Non-atten-loss model is trained with self-contained distractions (with a training inserting probability of 0.7) while not knowing which utterances are distractions, and it achieves a better performance than the baseline. The StaticUI model with a training inserting probability of 0.7 is trained to minimize the attention loss of self-contained distractions and it achieves the best performance. Naturally since the optimization strategy minimizes the attention loss of distractions, the StaticUI model distributes less attention to *History* and more attention to *Query* (refer to the "Avg" column in Appendix 4 for more details); nevertheless, a lower DAS ratio shows that the model distributes even less attention to the distracting utterances compared to the original utterances in *History*.

Note that even if both our proposed strategy and the random distracting test use the same trick: insert random distracting utterances among original utterances in *History*, the random utterances inserted in the distracting test are different from those inserted in the training process, thus it is difficult for the test to be biased in favor of models with our proposed strategy. It is apparent that less attention is distributed to *History*, while DAS ratio calculates the ratio between the distracting utterances and the original utterances in *History*, so it shows the attention distributed to the distracting utterances regardless of the total attention distributed to *History*. Moreover, we adopt three testing inserting probability levels to ensure stable evaluation results for each model.

### 5.3. Distracting Test: Frequent and Rare

Results of the frequent and the rare distracting test are shown in the "Frequent" and "Rare" columns in Table 3. Different from the random distracting test, the inserting locations of these two tests are decided manually. As a nature of LSTM model, all models distribute more attention to utterances near *Query* and less attention to utterances far away from *Query*, as can be seen in Table 3 and Figure 2 that DAS ratios are higher for End test set (near *Query*) and lower for Begin test set (far away from *Query*). Since the results on Begin and End test sets are biased by the structure of LSTM, we mainly analyze the results on Middle test sets.

For the Middle test sets of both the frequent and rare distracting test, the best models are still those trained with our proposed optimization strategy. StaticUI models with training inserting probabilities of 0.5 and 0.7 achieve the best performance (lowest DAS ratios) on the Frequent Middle and Rare Middle test sets. The Non-atten-loss models can be better than the models trained with a wrong training inserting probability. Telling from similar DAS ratios, the frequent distracting

test set is as difficult for the trained models to distinguish as the rare distracting test set, although for humans, the rare distracting utterances are much easier to distinguish than the frequent ones.

Table 4: Results of perplexity (Perp.) and average AS of *History* (Avg.) on the original test set (%) are shown in the "Original" column. Besides, we show the results on the random distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), and average AS of original utterances in *History* (Avg.) (%).

| Model | | Original | | Distracting Test Set | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Random 0.5 | | | Random 0.7 | | | Random 1.0 | | | |
| Probability | Structure | Perp. | Avg. | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | |
| 0.0 | Non-hier | 43.2 | 91.3 | 0.93 | 86.8 | 93.1 | 0.93 | 87.1 | 93.8 | 0.93 | 87.6 | 94.5 | |
| | Static | 44.1 | 61.4 | 0.82 | 52.6 | 64.5 | 0.82 | 53.5 | 65.6 | 0.79 | 53.4 | 67.6 | |
| | StaticUI | 44.6 | 57.5 | 0.79 | 47.4 | 60.2 | 0.76 | 46.3 | 61.3 | 0.76 | 47.7 | 62.7 | |
| | Dynamic | 45.4 | 81.4 | 0.89 | 74.9 | 84.4 | 0.89 | 75.6 | 85.2 | 0.88 | 76.2 | 86.6 | |
| | DynamicUI | 44.7 | 91.6 | 0.94 | 87.5 | 93.4 | 0.94 | 88.0 | 93.7 | 0.93 | 88.2 | 94.5 | |
| 0.5 | Non-hier | 43.4 | 87.2 | 0.84 | 77.2 | 91.6 | 0.83 | 77.1 | 93.3 | 0.81 | 77.0 | 95.5 | |
| | Static | 44.5 | 66.5 | 0.70 | 50.3 | 71.5 | 0.69 | 50.5 | 73.5 | 0.67 | 51.1 | 76.5 | |
| | StaticUI | 44.3 | 47.7 | 0.74 | 38.1 | 51.2 | 0.74 | 39.2 | 53.1 | 0.70 | 39.1 | 55.5 | |
| | Dynamic | 44.6 | 81.9 | 0.79 | 68.3 | 86.6 | 0.78 | 69.1 | 88.2 | 0.77 | 69.4 | 90.8 | |
| | DynamicUI | 43.9 | 86.7 | 0.82 | 74.5 | 91.1 | 0.81 | 75.2 | 92.5 | 0.80 | 75.8 | 94.8 | |
| | Non-labled StaticUI | 44.7 | 71.1 | 0.73 | 55.5 | 75.6 | 0.73 | 56.6 | 77.1 | 0.72 | 57.4 | 79.6 | |
| 0.7 | Non-hier | 43.2 | 86.9 | 0.84 | 76.5 | 91.3 | 0.82 | 75.9 | 93.1 | 0.80 | 75.9 | 95.4 | |
| | Static | **44.0** | 57.6 | 0.73 | 45.5 | 62.2 | 0.72 | 45.8 | 64.0 | 0.69 | 46.4 | 66.9 | |
| | StaticUI | 44.9 | **43.7** | **0.67** | **32.4** | **48.1** | **0.67** | **33.2** | **49.9** | **0.65** | **34.0** | **52.1** | |
| | Dynamic | 44.3 | 82.0 | 0.76 | 66.3 | 87.2 | 0.75 | 66.6 | 89.1 | 0.73 | 67.2 | 91.8 | |
| | DynamicUI | 44.8 | 85.3 | 0.93 | 86.8 | 93.1 | 0.93 | 87.1 | 93.8 | 0.93 | 87.6 | 94.5 | |
| | Non-labled StaticUI | 44.1 | 55.4 | 0.72 | 43.3 | 59.9 | 0.70 | 43.4 | 62.0 | 0.69 | 44.3 | 64.4 | |
| 1.0 | Non-hier | 47.3 | 95.9 | 0.91 | 88.7 | 98.0 | 0.90 | 89.3 | 98.7 | 0.90 | 89.5 | 99.9 | |
| | Static | **44.0** | 65.4 | 0.70 | 49.7 | 71.1 | 0.70 | 51.3 | 73.1 | 0.68 | 51.8 | 76.4 | |
| | StaticUI | 49.6 | 73.5 | 0.96 | 74.8 | 77.8 | 0.95 | 75.2 | 79.4 | 0.94 | 76.7 | 81.2 | |
| | Dynamic | 44.7 | 88.8 | 0.79 | 74.2 | 93.4 | 0.78 | 74.4 | 95.2 | 0.77 | 75.4 | 97.4 | |
| | DynamicUI | 45.2 | 90.2 | 0.87 | 81.3 | 93.6 | 0.86 | 81.5 | 94.9 | 0.85 | 81.9 | 96.5 | |
| | Non-labled StaticUI | 44.1 | 76.5 | 0.72 | 59.5 | 82.1 | 0.71 | 59.7 | 84.4 | 0.69 | 60.3 | 87.6 | |

Table 5: Results on the frequent distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), average AS of original utterances in *History* (Avg.) (%), and AS of the first/last utterance in *History* (%).

| Model | | Distracting Test Set | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Frequent: Begin | | | | Frequent: Middle | | | Frequent: End | | | | |
| Probability | Structure | DAS ratio | DAS | Avg. | 1st | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | Last |
| 0.0 | Non-hier | 0.75 | 70.8 | 94.1 | 82.3 | 0.80 | 74.7 | 93.2 | 0.84 | 78.0 | 92.7 | 98.4 |
| | Static | 0.37 | 29.3 | 79.5 | 47.9 | 0.80 | 56.7 | 71.2 | 1.31 | 82.0 | 62.4 | 72.6 |
| | StaticUI | **0.32** | **24.3** | **76.1** | **42.2** | 0.75 | 50.6 | 67.7 | 1.32 | 77.2 | 58.7 | 69.5 |
| | Dynamic | 0.65 | 60.9 | 93.3 | 72.7 | 0.86 | 75.5 | 88.0 | 1.02 | 87.5 | 85.1 | 88.8 |
| | DynamicUI | 0.72 | 72.2 | 100.7 | 81.0 | 0.84 | 81.6 | 97.0 | 0.86 | 84.1 | 98.2 | 98.8 |
| 0.5 | Non-hier | 0.63 | 57.3 | 91.0 | 84.0 | 0.74 | 66.4 | 89.7 | **0.76** | 68.7 | 90.8 | 99.5 |
| | Static | 0.42 | 35.0 | 84.5 | 47.4 | 0.78 | 59.8 | 77.0 | 1.12 | 80.3 | 71.5 | 81.7 |
| | StaticUI | 0.39 | 24.6 | 62.9 | 44.3 | 0.71 | 41.8 | 58.9 | 1.08 | **54.7** | **50.6** | **55.2** |
| | Dynamic | 0.64 | 60.4 | 94.6 | 76.7 | 0.74 | 68.2 | 92.1 | 0.84 | 76.2 | 91.1 | 94.4 |
| | DynamicUI | 0.60 | 60.1 | 100.4 | 77.7 | 0.84 | 78.4 | 92.9 | 0.87 | 82.3 | 94.2 | 93.0 |
| | Non-labled StaticUI | 0.39 | 35.1 | 90.3 | 53.7 | 0.68 | 56.1 | 82.7 | 0.93 | 72.6 | 78.4 | 91.2 |
| 0.7 | Non-hier | 0.72 | 64.7 | 90.1 | 83.1 | 0.82 | 72.6 | 88.8 | 0.82 | 73.4 | 89.4 | 97.1 |
| | Static | 0.40 | 30.3 | 75.2 | 48.3 | 0.70 | 48.7 | 69.3 | 1.08 | 68.1 | 62.9 | 69.2 |
| | StaticUI | 0.36 | 21.0 | 57.6 | 37.4 | **0.66** | **36.0** | **54.7** | 1.02 | 51.7 | 50.6 | 56.4 |
| | Dynamic | 0.58 | 56.3 | 96.8 | 73.9 | 0.71 | 66.4 | 93.4 | 0.86 | 76.0 | 88.8 | 91.8 |
| | DynamicUI | 0.45 | 44.4 | 98.8 | 76.1 | 0.78 | 73.2 | 93.8 | 0.80 | 75.8 | 95.2 | 95.0 |
| | Non-labled StaticUI | 0.45 | 31.7 | 70.6 | 51.0 | 0.70 | 46.6 | 66.6 | 0.98 | 60.2 | 61.2 | 65.8 |
| 1.0 | Non-hier | 0.84 | 82.0 | 97.8 | 92.9 | 0.86 | 83.7 | 97.7 | 0.85 | 83.2 | 97.4 | 100.0 |
| | Static | 0.49 | 40.2 | 82.5 | 60.3 | 0.74 | 57.1 | 76.8 | 1.08 | 73.3 | 67.8 | 72.3 |
| | StaticUI | 0.66 | 73.3 | 110.4 | 24.9 | 0.86 | 71.5 | 82.9 | 1.53 | 104.8 | 68.3 | 88.6 |
| | Dynamic | 0.63 | 64.0 | 102.1 | 81.7 | 0.75 | 73.9 | 98.5 | 0.82 | 79.5 | 97.5 | 99.7 |
| | DynamicUI | 0.73 | 72.5 | 100.0 | 83.5 | 0.81 | 79.2 | 97.4 | 0.83 | 81.6 | 98.7 | 97.6 |
| | Non-labled StaticUI | 0.49 | 46.1 | 95.0 | 67.1 | 0.74 | 65.0 | 87.7 | 0.98 | 79.9 | 81.6 | 86.6 |

Table 6: Results on the rare distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), average AS of original utterances in *History* (Avg.) (%), and AS of the first/last utterance in *History* (%).

| Model | | Distracting Test Set | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Rare: Begin | | | | Rare: Middle | | | Rare: End | | | |
| Probability | Structure | DAS ratio | DAS | Avg. | 1st | DAS ratio | DAS | Avg. | DAS ratio | DAS | Avg. | Last |
| 0.0 | Non-hier | 0.80 | 74.7 | 93.8 | 82.3 | 0.92 | 84.3 | 92.1 | 1.01 | 92.0 | 91.2 | 96.7 |
| | Static | 0.37 | 29.3 | 79.6 | 47.9 | 0.77 | 55.5 | 72.2 | 1.21 | 77.8 | 64.3 | 75.2 |
| | StaticUI | 0.30 | 22.9 | 76.4 | 42.2 | 0.75 | 51.1 | 67.9 | 1.22 | 74.2 | 61.0 | 73.2 |
| | Dynamic | 0.66 | 61.4 | 93.6 | 72.7 | 0.89 | 77.4 | 87.2 | 1.06 | 87.0 | 82.3 | 85.4 |
| | DynamicUI | 0.73 | 73.8 | 100.5 | 81.0 | 0.93 | 87.9 | 94.2 | 0.97 | 90.1 | 93.0 | 93.0 |
| 0.5 | Non-hier | 0.69 | 62.9 | 90.8 | 84.0 | 0.81 | 72.6 | 89.2 | 0.86 | 77.7 | 90.3 | 98.6 |
| | Static | 0.34 | 29.4 | 86.1 | 47.4 | 0.71 | 55.4 | 78.4 | 0.99 | 72.2 | 72.5 | 82.6 |
| | StaticUI | 0.40 | 24.7 | 62.2 | 44.3 | **0.69** | **40.4** | **58.6** | 0.96 | 53.0 | 55.4 | 60.5 |
| | Dynamic | 0.61 | 58.1 | 95.4 | 76.7 | 0.77 | 70.3 | 91.0 | 0.85 | 76.0 | 89.4 | 92.2 |
| | DynamicUI | 0.61 | 60.9 | 100.5 | 77.7 | 0.80 | 75.5 | 94.9 | 0.83 | 78.6 | 94.4 | 93.6 |
| | Non-labled StaticUI | 0.40 | 36.4 | 90.4 | 53.7 | 0.80 | 64.4 | 80.7 | 1.11 | 82.7 | 74.6 | 87.4 |
| 0.7 | Non-hier | 0.71 | 64.1 | 90.5 | 83.1 | 0.85 | 75.0 | 88.8 | 0.87 | 78.1 | 89.6 | 97.3 |
| | Static | 0.41 | 30.0 | 73.5 | 48.3 | 0.70 | 48.7 | 69.4 | 0.98 | 63.6 | 64.6 | 71.7 |
| | StaticUI | 0.36 | 20.9 | 57.7 | 37.4 | 0.70 | 37.6 | 54.1 | 0.99 | **50.6** | **51.1** | **57.3** |
| | Dynamic | 0.58 | 55.8 | 96.6 | 73.9 | 0.73 | 67.8 | 92.4 | 0.83 | 74.6 | 89.8 | 92.5 |
| | DynamicUI | 0.60 | 59.7 | 98.8 | 76.1 | 0.80 | 74.4 | 93.2 | **0.81** | 75.9 | 93.9 | 93.8 |
| | Non-labled StaticUI | 0.43 | 30.8 | 70.9 | 51.0 | 0.73 | 48.1 | 66.3 | 0.97 | 60.5 | 62.5 | 67.3 |
| 1.0 | Non-hier | 0.85 | 82.8 | 97.8 | 92.9 | 0.87 | 84.8 | 97.5 | 0.88 | 85.6 | 97.3 | 100.2 |
| | Static | 0.46 | 37.5 | 81.6 | 60.3 | 0.71 | 55.0 | 77.4 | 0.88 | 65.2 | 74.4 | 79.8 |
| | StaticUI | **0.21** | **22.4** | **105.6** | **24.9** | 0.86 | 71.0 | 83.0 | 1.50 | 103.1 | 68.7 | 89.0 |
| | Dynamic | 0.65 | 65.6 | 101.4 | 81.7 | 0.77 | 75.1 | 98.1 | 0.82 | 79.6 | 96.9 | 98.5 |
| | DynamicUI | 0.75 | 74.3 | 99.3 | 83.5 | 0.88 | 83.9 | 95.4 | 0.88 | 84.2 | 96.0 | 94.3 |
| | Non-labled StaticUI | 0.49 | 45.5 | 93.7 | 67.1 | 0.77 | 67.4 | 87.3 | 0.98 | 80.6 | 82.5 | 87.4 |

### 5.4. Detailed Results on the Distracting Tests

In addition to DAS ratio, Table 4 shows the average AS of distracting utterances and of original utterances in *History*. Table 5 and Table 6 additionally show the AS of the first or last utterances in *History*. Note again that an attention score of $100\%$ for a utterance indicates that this utterance receives an average attention score, e.g. for a dialogue containing 10 utterances, an attention score of $100\%$ indicates that the utterance receives $10\%$ attention out of all.

From Table 4 it is clear that the average AS of the original utterances in *History* varies by model variants. A higher average AS for *History* indicates a lower AS for *Query*. Some models distribute most of the attention to *Query* while some models distribute the attention evenly to both *History* and Query. Normally, *Query* contains more relevant information, so we expect a lower average AS for *History*; however, the average AS for *History* is not the lower the better, since there are still some utterances in *History* that are important for the context. A lower average AS for *History* comes together with a lower average AS for distracting utterances (or a lower DAS), so DAS ratio is better suited for evaluating a model's capability on context attention distribution, since it takes the average AS for original utterances in *History* into account. In Table 4, the models with the lowest DAS ratio also have the lowest average AS for distracting utterances and original utterances, while in Table 5 and Table 6, it is not always the case.

In Table 5 and Table 6, for the distracting test sets where distracting utterances are put in the beginning/end of the context, we show AS for the first/last utterance in *History* to have a clearer comparison. We can see in columns of Frequent: Begin and Rare: Begin that the distracting utterances usually receive lower attention than the first utterance in *History*, while the other original utterances in *History* receive more attention than the first utterance. This indicates a good performance of the model variants. Utterances far away from *Query* are normally distributed lower attention, so in a normal case, it is natural that the utterances that come after the first utterance receive more attention; however, these distracting utterances receive less attention, regardless of the fact that they are placed after the first utterances. It can thus be inferred that most model variants can distinguish distracting utterances as unimportant and distribute less attention to them. Similarly, the last utterances in *History* usually get more attention, while as the columns of Frequent: End and Rare: End show, distracting utterances receive less attention compared to other original utterances in *History*, regardless of that the distracting utterances are placed closer to *Query*.

### 5.5. Summary of Results

DAS ratio can distinguish conversational agents with similar perplexity on their ability of context attention distribution. In general, models trained with our proposed optimization strategy focus less on distracting utterances and more on original utterances in *History*. For most models, DAS ratios decrease by about $10\%$ when trained with our proposed strategy with a $0.5$ or $0.7$ probability level. $0.7$ is generally the best option for a training inserting probability.

## 6. Conclusions and Future Works

We have studied context attention distribution, an essential component of multi-turn modelling for open-domain conversational agents. We have proposed an evaluation metric for context attention distribution based on the distracting test: DAS ratio. We have also improved the performance of context attention distribution for common multi-turn conversational agents through an optimization strategy via reducing the attention loss of self-contained distracting utterances. Extensive experiments show that our proposed strategy achieves improvements on most models, especially with a training inserting probability level of $0.7$. Future works can focus on adapting the proposed evaluation metric and optimization strategy to transformer-based conversational agents.

## ACKNOWLEDGEMENTS

This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assist on computing resources from the IDUN cluster of NTNU [20]. We would like to thank Benjamin Kille and Peng Liu for their helpful comments.

## REFERENCES

[1] Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan, "DIALOGPT : Large-scale generative pre-training for conversational response generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, Jul. 2020, pp. 270–278. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-demos.30

[2] X. Zhao, W. Wu, C. Xu, C. Tao, D. Zhao, and R. Yan, "Knowledge-grounded dialogue generation with pre-trained language models," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 3377–3390. [Online]. Available: https://aclanthology.org/2020.emnlp-main.272

[3] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11957

[4] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio, "A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14567

[5] Z. Li, J. Zhang, Z. Fei, Y. Feng, and J. Zhou, "Conversations are not flat: Modeling the dynamic information flow across dialogue utterances," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 128–138. [Online]. Available: https://aclanthology.org/2021.acl-long.11

[6] R. Lowe, N. Pow, I. Serban, and J. Pineau, "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems," in *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, 2015, pp. 285–294. [Online]. Available: http://aclweb.org/anthology/W15-4640

[7] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318. [Online]. Available: https://aclanthology.org/P02-1040

[8] C.-W. Liu, R. Lowe, I. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2016, pp. 2122–2132. [Online]. Available: http://aclweb.org/anthology/D16-1230

[9] E. Bruni and R. Fernandez, "Adversarial evaluation for open-domain dialogue generation," in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*. Association for Computational Linguistics, 2017, pp. 284–288. [Online]. Available: http://aclweb.org/anthology/W17-5534

[10] R. Lowe, M. Noseworthy, I. V. Serban, N. Angelard-Gontier, Y. Bengio, and J. Pineau, "Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017, pp. 1116–1126. [Online]. Available: http://aclweb.org/anthology/P17-1103

[11] Y. Zemlyanskiy and F. Sha, "Aiming to Know You Better Perhaps Makes Me a More Engaging Dialogue Partner," in *Proceedings of the 22nd Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2018, pp. 551–561. [Online]. Available: http://aclweb.org/anthology/K18-1053

[12] Z. Tian, R. Yan, L. Mou, Y. Song, Y. Feng, and D. Zhao, "How to Make Context More Useful? An Empirical Study on Context-Aware Neural Conversational Models," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2017, pp. 231–236. [Online]. Available: http://aclweb.org/anthology/P17-2036

[13] W. Zhang, Y. Cui, Y. Wang, Q. Zhu, L. Li, L. Zhou, and T. Liu, "Context-Sensitive Generation of Open-Domain Conversational Responses," in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2018, pp. 2437–2447. [Online]. Available: http://aclweb.org/anthology/C18-1206

[14] X. Gu, K. M. Yoo, and J.-W. Ha, "DialogBERT: Discourse-aware response generation via learning to recover and rank utterances," in *In Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021)*, 2021.

[15] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112. [Online]. Available: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[18] J. Li, M. Galley, C. Brockett, G. Spithourakis, J. Gao, and B. Dolan, "A Persona-Based Neural Conversation Model," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 994–1003. [Online]. Available: http://aclweb.org/anthology/P16-1094

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[20] M. Själander, M. Jahre, G. Tufte, and N. Reissmann, "EPIC: An energy-efficient, high-performance GPGPU computing research infrastructure," 2019.

# Paper III

# Improving Context-Awareness on Multi-Turn Dialogue Modeling with Extractive Summarization Techniques

Yujie Xing(✉) and Jon Atle Gulla

Norwegian University of Science and Technology, Trondheim, Norway
{yujie.xing,jon.atle.gulla}@ntnu.no

**Abstract.** The study of context-awareness in multi-turn generation-based dialogue modeling is an important but relatively underexplored topic. Prior research has employed hierarchical structures to enhance the context-awareness of dialogue models. This paper aims to address this issue by utilizing two extractive summarization techniques, namely the PMI topic model and the ORACLE algorithm, to filter out unimportant utterances within a given context. Our proposed approach is assessed on both non-hierarchical and hierarchical models using the *distracting test*, which evaluates the level of attention given to each utterance. Our proposed methods gain significant improvement over the baselines in the distracting test.

**Keywords:** Multi-Turn Response Generation · Conversational Agent · Summarization

## 1 Introduction

Although generation-based dialogue models have achieved much progress in recent years, multi-turn dialogue models are still facing challenges. Recent works deal with multi-turn using modified attention mechanisms and hierarchical structures. One focus of dealing with multi-turn is the ability of context-awareness on a dialogue model, which requires a model to pay more attention to important utterances while less attention to unimportant ones. An example of important/unimportant utterances is given by Table 1.


E. Métais et al. (Eds.): NLDB 2023, LNCS 13913, pp. 478–488, 2023.
https://doi.org/10.1007/978-3-031-35320-8_35

**Table 1.** An example of important utterances and unimportant utterances under the same context in the Ubuntu chatlog dataset [9]. Unimportant utterances are marked in red.

| User | Utterances |
|---|---|
| Taru | Haha sucker. |
| Kuja | ? |
| Taru | Anyways, you made the changes right? |
| Kuja | Yes. |
| Taru | Then from the terminal type: sudo apt-get update |
| Kuja | I did |

In Table 1, the first two utterances ("Haha sucker." and "?") are unimportant utterances that are irrelevant to the main topic of the context. A multi-turn dialogue model with good ability on context awareness should identify and ignore these unimportant utterances and focus only on the important ones. Thus, we propose that one way to improve the context awareness of a model is to **filter out** the unimportant utterances, which is a task similar to summarization: given a reference and a source, an extractive summarization algorithm extracts all utterances related to the reference and eliminate all others in the source. In the case of dialogue models, we do not have a reference for the context; nevertheless, the last utterance in the context, i.e., the *query*, plays a crucial role in generating the response. In most cases, responses aim to provide answers to the *query* while utilizing other utterances in the context as the source for answering. We denote all utterances in a context except for the last one as *source*. This paper investigates improving context awareness for multi-turn dialogue models by filtering out unimportant utterances from the *source* using extractive summarization techniques with the *query* as the reference.

There are a few works that combine summarization with dialogue models. One of the techniques used in these works is the topic model, where a keyword is predicted from the *query* and the entire corpus to help a model generate detailed responses. In our paper, we also use a PMI topic model to extract keywords from the context, while instead of using the keywords to support the generation task, we pass the keywords directly to the dialogue model. Additionally, we explore the ORACLE algorithm, a widely-used algorithm for generating gold labels for extractive summarization, to filter out utterances unrelated to *query* before passing them to the dialogue model.

For evaluation, we use an evaluation method tailored for multi-turn dialogue models. Since most multi-turn dialogue models have attention mechanisms and they rely on the mechanism to assign different extents of focus to each utterance in the context, we use the **distracting test** to measure if a model pays more attention to the important utterances and less to the unimportant ones. The test simply adds distracting utterances to each dialogue and compares the attention

scores on these distracting (unimportant) utterances with the original (important) utterances in the *source*, thus measures the ability of context awareness for a dialogue model.

This paper is organized as follows. In Sect. 2, we introduce related works. In Sect. 3 and Sect. 4, we introduce the model to be examined, the summarization techniques to be integrated, and the evaluation metrics. In Sect. 5, we describe our experiment settings, and we report the results in Sect. 6.

## 2 Related Work

Previous works try to improve context-awareness on dialogue modeling through the hierarchical structure. [13,14] first introduce the hierarchical structure to dialogue models. [17] evaluate different methods of integrating context utterances in hierarchical structures, and [21] further evaluate the effectiveness of static and dynamic attention mechanism. In our paper, we examine our context-summarization module with both different methods of integrating context utterances and two kinds of attention mechanisms.

A similar direction of combining summarization and multi-turn dialogue modeling is the integration of topic models, though current works in this direction are all on single-turn dialogues. [6] uses a classifier to select the keyword for a given query from a pre-generated keyword list. [10,20] use PMI to choose a keyword for a given query from a big corpus. Similarly, [2,18] uses a topic model to predict the keyword out from vocabulary words. In our paper, we also examine if a topic model can improve the context-awareness of dialogue models.

As mentioned in [19], a typical way to construct labeled data for extractive summarization is to set ROUGE. Most works including [5] construct gold label sequences by greedily optimizing ROUGE-1, which is the algorithm ORACLE. Further, although in this paper we stick to extractive summarization due to lack of suitable conversational datasets for abstractive summarization, we expect the very soon coming of this kind of dataset from [3].

## 3 Models to be Examined

We use an LSTM Seq2Seq model with attention [1,4,16] as the base model, since it is a common model for conversational systems [7,12].

The basic task of conversational agents is to predict the next word given all the past and current words of the context and response, and to make the generated response as similar to the original response as possible. Formally, the task can be described as follows. Probability of response $Y$ given context $X$ is predicted as:

$$P(Y|X) = \prod_{t=1}^{n} p(y_t|y_1, \ldots, y_{t-1}, X), \tag{1}$$

where $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ are a context-response pair.

### 3.1 LSTM Seq2Seq Model with Attention

We simplify an LSTM structure with attention mechanism as $LSTM^*$ since it is well introduced in previous work [7]. We calculate the hidden vector $h_t$ at step $t$ as:

$$h_t = LSTM^*(h_{t-1}, c_t, E(z_t)), \tag{2}$$

where $h_{t-1} \in \mathbb{R}^{dim}$ is the hidden vector at step $t-1$, $dim$ is the dimension of hidden vectors, and $E(z_t)$ is the word embedding for token $z_t \in \{x_1, ..., x_m, y_1, ..., y_{n-1}\}$. The context vector $c_t$ is inputted only to the decoder at step $t$.

### 3.2 Attention Mechanism and Utterance Integration

We examine both hierarchical and non-hierarchical structures. For hierarchical structures, following [21], we examine two attention mechanisms, namely static and dynamic attention mechanisms. Following [17], we examine hierarchical models with or without utterance integration LSTM units.

For the non-hierarchical structured model, there are no utterance vectors. Hidden vectors of all words in the encoder are concatenated and used in the attention mechanism. Denoting the concatenated vector as $\mathcal{H}$ ($\mathcal{H} = [h_1, h_2, ..., h_m]$), we calculate the context vector $c_t$ for each decoder step as

$$c_t = \mathcal{H} \cdot (softmax(\mathcal{H}^\top \cdot h_{t-1})) \ . \tag{3}$$

For the hierarchically structured models, we denote the last utterance of the context as the *query*, and the other utterances as the *source*. At each step where an utterance ends, we collect the hidden vector of its last word as the hidden vector of the utterance, thus compared to the non-hierarchical structured model, we have much fewer hidden vectors from the encoder. Denoting the hidden vector of $k$th utterance as $H_k$, the hidden vector of the *query* as $H_q$, and the concatenated vector of the *source* and the *query* as $\mathcal{H}_c$ ($\mathcal{H}_c = [H_1, H_2, ..., H_q]$), we calculate the context vector $c_t$ for static attention mechanism as

$$c_t = \mathcal{H}_c \cdot (softmax(\mathcal{H}_c^\top \cdot H_q)) \tag{4}$$

where it is easy to see that static attention does not change during steps in the decoder. And we calculate $c_t$ for dynamic attention mechanism as

$$c_t = \mathcal{H}_c \cdot (softmax(\mathcal{H}_c^\top \cdot h_{t-1})) \ . \tag{5}$$

In the decoder, $c_t$ is input to the next step $t$, and each token's hidden vector $h_{t-1}$ is combined with $c_t$ to predict the next token.

Finally, with the utterance integration LSTM unit, the hidden vector to be put into the first step of the decoder is different from the regular $h_m$; instead, the vector is calculated by integrating $H_1$, $H_2$, ... $H_q$ through a separate LSTM unit.

# 4 Proposed Methods

## 4.1 PMI-Context

The method PMI-context uses a Pointwise Mutual Information (PMI) to select the $k$ most relevant words in a *source* given a *query*. Given a word $x_c$ in a *source*, the total PMI of $x_c$ given a *query* $= x_{q1}, ..., x_{ql}$ is calculated following [20]:

$$\text{PMI}(x_{q1}, ..., x_{ql}, x_c) \approx \sum_{i}^{l} \text{PMI}(x_{qi}, x_c) \ . \tag{6}$$

The selected $k$ keywords $x_{c1}, ..., x_{ck}$ and the *query* are combined through the static attention mechanism described in Eq. (4) to calculate the context vector $c_t$. Note that here a *query* does not attend to itself, but only to the selected keywords. The context vector $c_t$, the selected $k$ keywords, and the *query* are then inputted into the LSTM unit as described in the following adapted version of Eq. (2):

$$h_t = LSTM^*(h_{t-1}, c_t, E(z'_t)) \ , \tag{7}$$

where $z'_t \in \{x_{c1}, ..., x_{ck}, x_{q1}, ..., x_{ql}, y_1, ..., y_{n-1}\}$.

## 4.2 ORACLE-Context

The method ORACLE-context is based on an extractive summarization algorithm named the ORACLE algorithm. It uses the ORACLE algorithm to extract relevant utterances from the *source* by greedily optimizing ROUGE-1 using the *query* as the summarization reference. The extracted $k$ most relevant utterances are then inputted into the LSTM unit as described in the following adapted version of Eq. (2):

$$h_t = LSTM^*(h_{t-1}, c_t, E(z''_t)) \ , \tag{8}$$

where $z''_t \in \{x^1_{c1}, x^1_{c2}, ..., x^k_{c1}, x^k_{c2}, ..., x_{q1}, ..., x_{ql}, y_1, ..., y_{n-1}\}$, and $X_i = x^i_{c1}, x^i_{c2}, ...$ $(i \in \{1, ..., k\})$ denotes for each of the extracted $k$ most relevant utterances.

This method intends to filter out irrelevant utterances from the *source* given the *query* and delete the utterances from the inputs to the dialogue model, which helps the model to pay attention correctly to the important utterances.

## 4.3 Evaluation

Since perplexity is considered not a good measure of how good a conversation is [8], besides perplexity, we examine whether the model pays attention to the correct utterance through a simple **distracting test**.

In the distracting test, for each dialogue, we insert several distracting utterances into the dialogue. The distracting utterance can be anything that does not belong to the original dialogue. Then we compare the attention scores of

the distracting utterances with the attention scores of the original utterances. A well-performing model should pay less attention to the distracting utterances but more attention to the original utterances. For an utterance $H_k$ in the context, the score is calculated as

$$\begin{cases} \frac{\exp(H_k^\top \cdot H_q)}{\sum_k \exp(H_k^\top \cdot H_q)} & \text{Static attention} \\ \texttt{mean}_t(\frac{\exp(H_k^\top \cdot h_t)}{\sum_k \exp(H_k^\top \cdot h_t)}) & \text{Dynamic attention} \end{cases} \tag{9}$$

To avoid bias, we weigh the attention score with the utterance amount, or the total word amount of *source* plus *query* divided by the word amount of the utterance to be examined. That gives us 100% for any utterance that is paid average attention among *source* plus *query*, i.e. $\frac{1}{k}$ attention for a total of $k$ utterances in *source* plus *query*.

## 5 Experiment Setup

### 5.1 Dataset

We use the Ubuntu chatlogs dataset [9], which contains dialogues about solving technical problems of Ubuntu, as the training and testing corpus. We have about $0.48M$ dialogues for training, $20K$ dialogues for validation, and $10K$ dialogues for testing. These are the original settings of the Ubuntu chatlogs dataset. We removed all single-turn dialogues, since single-turns do not have contexts that we need to study on. The last utterance in the context is treated as *query*, and the other utterances are treated as *source*.

For the distracting test, we set the amount of distracting utterances for each dialogue as 2. We have 3 distracting test datasets: 1) dataset distracted with utterances containing frequent words, which are "why should I help you" and "I have my right"; 2) dataset distracted with utterances containing rare words, which are "would you have lunch?" and "I should have lunch"; 3) dataset distracted with utterances randomly picked from the training set.

### 5.2 Training

Our methods are built on a basic LSTM Seq2Seq model. We used Pytorch [11] for implementation. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, a learning rate of 1.0, and a gradient clip threshold of 5. The vocabulary size is 25000 and the dropout rate is 0.2.

### 5.3 Models to Be Examined

For the method PMI-context, we examine the maximum keyword amounts of both 10-word level and 30-word level. For the method ORACLE-context, we examine the maximum extracted utterance amounts of both 5-utterance level

and 10-utterance level. Also, we examine ORACLE-context on 5 model variants, namely static attention with utterance integration LSTM unit, static attention without utterance integration LSTM unit, dynamic attention with utterance integration LSTM unit, and dynamic attention without utterance integration LSTM unit. Among these variants, one is non-hierarchical structured, and the other four are hierarchical structured.

## 6 Results

We show the perplexity and attention scores of the models to be examined. For comparison, we also show scores of non-hierarchical model trained on either the whole context (*source* and *query*) or only *query*. The results are shown in Table 2.

For the distracting test, besides the attention scores of the distracting utterances, we also show the average attention scores of the *source*. A lower score indicates that more attention is paid to the *query* instead of the *source*. In addition, we calculate the ratio between the attention scores of the distracting utterances and those of the original utterances in the *source*, to show how much attention is paid to the distracting utterances compared to the *source*. A lower ratio indicates that the model is less distracted by the distracting utterances.

Table 2 shows that the non-hierarchical model with the ORACLE-context method of 10-utterance level has the best perplexity and the lowest attention scores' ratio for the frequent and rare distracting datasets, which indicates that this model is the least distracted from frequent and rare distracting utterances. Among the four kinds of hierarchical models, the variant of static attention mechanism with utterance integration LSTM unit (Static+UttLSTM) gets the best performance on the random distracting dataset, and most of the other variants manage to exceed the non-hierarchical model on the random distracting dataset, from which we can infer that the hierarchical models are less distracted from random distracting utterances. PMI-context method of the 30-word level also gains a good perplexity, but since perplexity is not a good method for evaluating responses' quality, more evaluation is needed.

It is easy to notice that while the perplexity scores of the ORACLE-context models show marginal improvement over the baselines, they outperform the baselines in the distracting test, which is a better evaluation metric for the ability of context-awareness. To assess the efficacy of the ORACLE algorithm, we further investigated the filtered-out and extracted utterances. Results show that approximately 79%, 84%, and 82% of the distracting utterances were filtered out in each of the three distracting datasets, respectively. In contrast, the algorithm extracted a considerable portion of the first and second utterances closest to the *query*, which are typically regarded as important utterances in a *source*, and these make up 30% and 43% of the total extracted utterances, respectively. This means that the ORACLE algorithm does filter out unimportant utterances to some extent.

It is surprising to see that the models have the worst performance for the distracting dataset with rare utterances. It is obvious for humans to identify

**Table 2.** Perplexity (Perp), attention score of distracting utterances (Distract, %), attention score of average original utterances in the *source* (Avg., %), and their ratio (ratio). The best attention scores of distracting utterances and the best ratios are bolded.

(a) Results on the random distract testset

| Method | Model | Original | | Distract: random | | |
|---|---|---|---|---|---|---|
| | | Perp | Avg. | Perp | Distract (ratio) | Avg. |
| \ | Non-hier (*query* only) | 49.5 | 100 | \ | \ | |
| | Non-hier | 49.8 | 94.7 | 49.8 | 94.4 (0.99) | 95.4 |
| PMI | PMI-10 | 49.5 | \ | 49.5 | \ | |
| | PMI-30 | 47.8 | \ | 47.8 | \ | |
| ORACLE-5 | Non-hier | 48.1 | 86.2 | 48.7 | 82.4 (0.94) | 87.2 |
| | static | 49.0 | 68.0 | 49.3 | 56.8 (0.81) | 70.0 |
| | static+UttLSTM | 51.3 | 52.8 | 51.6 | **41.2 (0.76)** | 54.1 |
| | dynamic | 49.7 | 86.8 | 50.2 | 81.4 (0.93) | 88.0 |
| | dynamic+UttLSTM | 50.7 | 93.8 | 51.2 | 91.3 (0.97) | 94.4 |
| ORACLE-10 | Non-hier | **47.1** | 86.5 | **47.7** | 82.5 (0.94) | 87.4 |
| | static | 49.5 | 60.7 | 49.9 | **47.1 (0.75)** | 62.4 |
| | static+UttLSTM | 47.7 | 54.1 | 48.0 | 43.5 (0.79) | 55.3 |
| | dynamic | 49.9 | 85.5 | 50.3 | 80.0 (0.92) | 86.7 |
| | dynamic+UttLSTM | 49.6 | 95.0 | 49.9 | 93.4 (0.98) | 95.3 |

(b) Results on the frequent and rare distracting dataset

| Method | Model | Distract: frequent | | | Distract: rare | | |
|---|---|---|---|---|---|---|---|
| | | Perp | Distract (ratio) | Avg. | Perp | Distract (ratio) | Avg. |
| \ | Non-hier (*query only*) | \ | \ | | \ | \ | |
| | Non-hier | 49.7 | 94.3 (0.98) | 95.8 | 49.8 | 94.4 (0.99) | 95.5 |
| PMI | PMI-10 | 49.5 | \ | | 49.5 | \ | |
| | PMI-30 | 47.8 | \ | | 47.8 | \ | |
| ORACLE-5 | Non-hier | 48.3 | 74.8 (0.86) | 86.9 | 48.4 | 78.1 (0.90) | 86.3 |
| | static | 49.1 | 65.1 (0.95) | 68.7 | 49.2 | 63.0 (0.91) | 69.3 |
| | static+UttLSTM | 51.4 | 46.9 (0.88) | 53.4 | 51.4 | **48.3 (0.90)** | 53.5 |
| | dynamic | 49.9 | 79.3 (0.90) | 88.3 | 50.0 | 83.0 (0.95) | 87.5 |
| | dynamic+UttLSTM | 50.8 | 89.3 (0.95) | 94.6 | 50.9 | 94.3 (1.01) | 93.0 |
| ORACLE-10 | Non-hier | **47.3** | **69.9 (0.80)** | 87.3 | **47.3** | **74.3 (0.86)** | 86.8 |
| | static | 49.7 | 51.0 (0.83) | 61.7 | 49.7 | 55.3 (0.90) | 61.5 |
| | static+UttLSTM | 47.7 | **46.8 (0.86)** | 54.7 | 47.9 | 51.1 (0.95) | 54.1 |
| | dynamic | 50.1 | 79.3 (0.92) | 86.4 | 50.1 | 87.9 (1.03) | 85.0 |
| | dynamic+UttLSTM | 49.7 | 91.1 (0.95) | 95.9 | 49.8 | 94.6 (1.00) | 94.3 |

"Would you have lunch?" and "I should have lunch" as distracting utterances, while although the ORACLE algorithm only keeps 16% of these distracting utterances, the model still cannot learn to pay less attention to them.

## 7 Conclusions

We have integrated extractive summarization techniques with multi-turn dialogue models to improve their ability of context-awareness. The techniques that we have examined are PMI topic model and ORACLE algorithm; we have integrated them with both non-hierarchical and hierarchical dialogue models. For evaluation, we have employed the distracting test to evaluate the context-awareness of each model. With extractive summarization techniques integrated, we find significant improvements in distracting tests for the multi-turn conversational agents. For future works, more summarization techniques can be considered, and more evaluation metrics can be used.

**Acknowledgement.** This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We have also received assistance on computing resources from the IDUN cluster of NTNU [15]. We would like to thank Pinar Øzturk for her helpful comments.

## References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015, Conference Track Proceedings (2015). http://arxiv.org/abs/1409.0473
2. Baheti, A., Ritter, A., Li, J., Dolan, B.: Generating more interesting responses in neural conversation models with distributional constraints. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3970–3980. Association for Computational Linguistics (2018). http://aclweb.org/anthology/D18-1431
3. Gliwa, B., Mochol, I., Biesek, M., Wawer, A.: SAMSum corpus: a human-annotated dialogue dataset for abstractive summarization. In: Proceedings of the 2nd Workshop on New Frontiers in Summarization, pp. 70–79. Association for Computational Linguistics, Hong Kong, China (2019). https://doi.org/10.18653/v1/D19-5409, https://www.aclweb.org/anthology/D19-5409
4. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
5. Kedzie, C., McKeown, K.R., III, H.D.: Content selection in deep learning models of summarization. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October - 4 November 2018, pp. 1818–1828. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1208
6. Li, J., Sun, X.: A syntactically constrained bidirectional-asynchronous approach for emotional conversation generation. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 678–683. Association for Computational Linguistics, Brussels, Belgium (2018). http://www.aclweb.org/anthology/D18-1071
7. Li, J., Galley, M., Brockett, C., Spithourakis, G., Gao, J., Dolan, B.: A persona-based neural conversation model. In: Proceedings of the 54th Annual Meeting of

the Association for Computational Linguistics (Volume 1: Long Papers), pp. 994–1003. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/P16-1094, http://aclweb.org/anthology/P16-1094
8. Liu, C.W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., Pineau, J.: How not to evaluate your dialogue system: an empirical study of unsupervised evaluation metrics for dialogue response generation. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2122–2132. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/D16-1230, http://aclweb.org/anthology/D16-1230
9. Lowe, R., Pow, N., Serban, I., Pineau, J.: The ubuntu dialogue corpus: a large dataset for research in unstructured multi-turn dialogue systems. In: Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 285–294. Association for Computational Linguistics (2015). https://doi.org/10.18653/v1/W15-4640, http://aclweb.org/anthology/W15-4640
10. Mou, L., Song, Y., Yan, R., Li, G., Zhang, L., Jin, Z.: Sequence to backward and forward sequences: a content-introducing approach to generative short-text conversation. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 3349–3358. The COLING 2016 Organizing Committee (2016). http://aclweb.org/anthology/C16-1316
11. Paszke, A., et al.: Automatic differentiation in PyTorch. In: NIPS-W (2017)
12. See, A., Roller, S., Kiela, D., Weston, J.: What makes a good conversation? How controllable attributes affect human judgments. arXiv:1902.08654 [cs] (2019). http://arxiv.org/abs/1902.08654
13. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: 13th AAAI Conference on Artificial Intelligence (2016). https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11957
14. Serban, I.V., et al.: A hierarchical latent variable encoder-decoder model for generating dialogues. In: 31st AAAI Conference on Artificial Intelligence (2017). https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14567
15. Själander, M., Jahre, M., Tufte, G., Reissmann, N.: EPIC: an energy-efficient, high-performance GPGPU computing research infrastructure (2019)
16. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 3104–3112. Curran Associates, Inc. (2014). http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf
17. Tian, Z., Yan, R., Mou, L., Song, Y., Feng, Y., Zhao, D.: How to make context more useful? An empirical study on context-aware neural conversational models. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 231–236. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/P17-2036, http://aclweb.org/anthology/P17-2036
18. Xing, C., et al.: Topic aware neural response generation. In: 31st AAAI Conference on Artificial Intelligence (2017). https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14563
19. Yao, J., Wan, X., Xiao, J.: Recent advances in document summarization. Knowl. Inf. Syst. **53**(2), 297–336 (2017). https://doi.org/10.1007/s10115-017-1042-4
20. Yao, L., Zhang, Y., Feng, Y., Zhao, D., Yan, R.: Towards implicit content-introducing for generative short-text conversation systems. In: Proceedings of the

2017 Conference on Empirical Methods in Natural Language Processing, pp. 2190–2199. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/D17-1233, http://aclweb.org/anthology/D17-1233

21. Zhang, W., Cui, Y., Wang, Y., Zhu, Q., Li, L., Zhou, L., Liu, T.: Context-sensitive generation of open-domain conversational responses. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 2437–2447. Association for Computational Linguistics (2018). http://aclweb.org/anthology/C18-1206

# Paper IV

# Prompt and Instruction-Based Tuning for Response Generation in Conversational Question Answering

Yujie Xing(✉) and Peng Liu

Norwegian University of Science and Technology, Trondheim, Norway
{yujie.xing,peng.liu}@ntnu.no

**Abstract.** In recent years, prompt-based tuning and instruction-based tuning have emerged as popular approaches for natural language processing. In this paper, we investigate the application of prompt and instruction-based tuning approaches for response generation in conversational question answering. We approach this task from both extractive and generative angles, where we adopt prompt-based tuning for the extractive angle and instruction-based tuning for the generative angle. Additionally, we utilize multi-task learning to integrate these two angles. To evaluate the performance of our proposed approaches, we conduct experiments on the GPT-2 model. The results show that the approaches improve performance by 18% on F1 score over the baseline. We share our codes and data for reproducibility. (https://github.com/yujie-xing/Multi-Turn_QA_Prompt).

**Keywords:** Prompt · Instruction · Pre-Trained Language Model · Response Generation · Conversational Question Answering

## 1 Introduction

Conversational Question Answering (CQA) is a QA dialogue system that can answer user questions based on a given document. CQA is an extension of traditional QA systems to a conversational setting and engages in multi-turn conversation to satisfy a user's information needs. According to the types of QA, CQA is studied in two settings: extractive and generative. In the extractive setting, the answer is marked as a span in the text paragraph, whereas in the generative setting, i.e. response generation in CQA, the answer is free-form text generated by autoregressively predicting tokens.

With the rapid development of language modeling techniques, a lot of pre-trained language models have been successfully applied to extractive CQA [3, 20], generative CQA [7, 27] and unified systems that solve various CQA tasks through a single model [10, 24]. Recently, Gekhman et al. [5] have conducted a comprehensive robustness study of history modeling approaches for CQA and propose a prompt-based history highlighting method to improve robustness while


E. Métais et al. (Eds.): NLDB 2023, LNCS 13913, pp. 156–169, 2023.
https://doi.org/10.1007/978-3-031-35320-8_11

maintaining overall high performance. However, prompts are generally short and do not generalize well to reformulations and new tasks.

Instruction tuning is an emergent paradigm where models are trained on a variety of tasks with natural language instructions. Instructions of natural language formats are easy for questioners to ask questions, and are proven to achieve a good performance due to the nature of the language model [6]. To the best of our knowledge, we are the first to apply instruction tuning for response generation on conversational question answering. Our paper proposes approaches for enhancing the response generation of conversational question answering by integrating prompt-based and instruction-based tuning. We adopt the prompt-based tuning method introduced by Gekhman et al. [5] to improve from the extractive angle on the multi-turn scenario. Additionally, we propose an instruction-based tuning method to enhance from the generative angle, based on the work of Zhong et al. [29] and Gupta et al. [6]. Furthermore, we investigate the integration of these two angles through multi-task learning.

In our experiments, we verify the influence of prompt-based tuning, instruction-based tuning, and multi-task learning for the task. We evaluate the performance of various settings, including prompt-based tuning with or without multi-task learning, prompt-based with or without instruction-based tuning, and prompt-based tuning with both multi-task learning and instruction-based tuning. We conduct the experiments on GPT-2 and evaluate the results on F1 score with 2 modes: the decoding mode and the evaluation mode. Additionally, we assess the extractive question answering part of the settings with a GPT-2 fine-tuned on the extractive question answering task.

The results show that our prompt-based tuning together with other approaches has improved the performance by about 18% on F1 score over the baseline, and the instruction-based tuning and multi-task learning settings have improved further at about 1% compared to pure prompt-based tuning approach.

The main contributions of this work are:

- To the best of our knowledge, we are the first to incorporate instruction tuning in conversational question answering.
- We investigate tuning approaches based on prompt and instruction for the response generation task on conversational question answering. The approaches are simple and easy to be adapted to other models.
- We conduct comprehensive experiments on the influence of instruction-based tuning, prompt-based tuning and multi-task learning for this task. The results show that the best approach improves about 18% on F1 score than the baseline.

The paper is organized as follows: we summarize related works in Sect. 2. We define our task and introduce the approaches used in our research in Sect. 3. In Sect. 4 we describe the setups of our experiments, and in Sect. 5 we present our results. We conclude and describe future works in Sect. 6.

## 2 Related Work

### 2.1 Conversational Question Answering with Prompts

In earlier times, recurrent neural networks (RNN) and attention variations were used to model conversation histories of QA [21,30]. Modern approaches leverage transformer-based pre-trained language models for QA by fine-tuning the models on massive annotated data from downstream QA tasks [8,11]. Recently, some works proposed to effectively adapt the pre-trained LMs to the downstream QA with only a handful of annotated data [2,20]. For instance, Chada et al. [2] proposed to cast QA as a text-generation problem by designing a prompt of a concatenation of the question and a special mask token representing the answer span. Similarly, Chen et al. [3] proposed to use Masked Language Model on entities to enhance few-shot QA learning. However, none of the abovementioned research works adopt instructions in prompt tuning for QA tasks. Considering the various QA tasks, some works explore multi-task learning QA by jointly training a single encoder to enhance the sharing of knowledge across tasks [4,22]. However, these works may suffer from poor scalability and flexibility when facing new types of QA tasks due to the requirement of deploying distinct prediction heads for different tasks.

### 2.2 Response Generation on Question Answering Task

Generative QA models [7,10,12,19] have shown remarkable performance, where the goal is to generate answers by autoregressively predicting tokens. Generative methods are more often used in open-domain [7,12,19,27] and unified settings [10,24]. Roberts et al. [19] proposed to use large pre-trained generative models, without using additional knowledge, for open-domain question answering. Lewis et al. [12] introduced retrieval-augmented generative models for open-domain question answering. Khashabi et al. [10] and Tafjord et al. [24] proposed to learn various QA formats in a unified way to alleviate the manual effort of task-specific design. Different from them, our work focuses on conversational answer generation with passages from the given task and investigates the influence of instruction tuning, prompt tuning and multi-task learning for conversational QA.

### 2.3 Instruction Tuning

Instruction tuning is a paradigm where models are trained on a variety of tasks with natural language instructions. Recent literature has been motivated by building models that are generalizable across a variety of NLP tasks when prompted with a few examples [1,13,14] or language definitions and constraints [26,28] introduced natural language instructions to improve the performance of LMs such as BART and GPT-3 for cross-task. Followed by this, FLAN [25] has been proposed, which uses instructions to achieve generalization across unseen tasks. Recently, Mishra et al. [9] have shown reframing instructional prompts can

boost both few-shot and zero-shot model performance. The InstructGPT model is proposed, which is fine-tuned with human feedback [15]. Puri et al. [18] introduced instruction augmentation to improve model performance in task-specific, multi-task and cross-task learning paradigms. Prasad et al. [17] introduced Gradient-free Instructional Prompt Search (GrIPS) to improve task instructions for large language models. Motivated by the effectiveness of instruction tuning, in this work, we explore the potential application of instructional prompts for conversational question-answering response generation.

## 3 Methodology

In this section, we first define the tasks of conversational question answering and response generation, and we introduce how these tasks are realized under GPT-2. After that, we explain the proposed multi-task learning, prompt tuning, and instruction tuning in detail.

### 3.1 Conversational Question Answering

The task of conversational question answering is to predict the answer span (start position, end position) in a passage for the given question and the previous questions and answer spans. The question answering task can be transferred to two classification tasks: one for the start position, and the other for the end position. Given a question $Q$ and a passage $X$, the tasks are to calculate the probability of the $t$-th token in the passage $X$ is the start position $P_{x_t=\text{start}}$ and is the end position $P_{x_t=\text{end}}$:

$$P(x_t = \text{start} \mid Q, X) \tag{1}$$

$$P(x_t = \text{end} \mid Q, X), \tag{2}$$

where $Q = q_1, \ldots, q_k$, $X = x_1, \ldots, x_m$ are sequences of tokens.

The difference between the task of conversational question answering with regular question answering is that there are conversation histories, i.e. multiple turns of questions and answer spans.

The question answering task is dealt with the GPT-2 model as follows. First, a hidden vector that is to be input to the transformer block is calculated as:

$$h_0 = E(Q, X) + (E_0, E_1) + W_p, \tag{3}$$

where $E(Q, X)$ is the sub-word embedding for question $Q$ and passage $X$. $E_0$ and $E_1$ are state embeddings, where $E_0$ is assigned to the question, and $E_1$ is assigned to the passage. $W_p$ is a pre-trained position embedding. Then, the probability of the subword $t$ to be the start or end position is calculated as:

$$h_X = \mathit{transformer\_block}(h_0)[X] \tag{4}$$

$$P(x_t = \mathit{start}) = \mathit{softmax}(A \cdot h_X)[t] \tag{5}$$

$$P(x_t = \mathit{end}) = \mathit{softmax}(B \cdot h_X)[t], \tag{6}$$

where $A \in \mathbb{R}^{1\times\dim(h)}$ and $B \in \mathbb{R}^{1\times\dim(h)}$, $h_X$ denotes for slice of the passage $X$ part in the hidden vector, and $[t]$ denotes for the $t$-th subword token in the passage $X$. We simplify the structure of the transformer block as *transformer_block*. In the block, a mask bans past words from attending to future words. Equation 5 and Eq. 6 transfer $h_X \in \mathbb{R}^{\dim(h)\times|X|}$ into sequences of probabilities for each subword token in $X$, where the probability of a subword $t$ being the start position or the end position can be obtained.

### 3.2 Response Generation

The task of response generation is to predict the next token given the past and current tokens of the context and response, and to make the generated response as similar to the original response as possible. In the scale of the conversational question answering task, the response generation task can be described as follows. Probability of answer $Y$ given a question $Q$ and a passage $X$ is predicted as:

$$P(Y \mid Q, X) = \prod_{t=1}^{n} P(y_t|y_1, \ldots, y_{t-1}, Q, X), \tag{7}$$

where $Q = q_1, \ldots, q_k$, $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ are sequences of tokens. (Q,X,Y) is a question-passage-answer tuple.

The response generation task is dealt with the GPT-2 model as follows. First, a hidden vector that is to be input to the transformer block is calculated as:

$$h_{0[t]} = E(Q, X, Y_{[1:t]}) + (E_0, E_0, E_1) + W_p, \tag{8}$$

where $Y_{[1:t]}$ is $(y_1, \ldots, y_t)$, $E(Q, X, Y_{[1:t]})$ is the sub-word embedding for question $Q$, passage $X$ and answer $Y_{[1:t]}$. $E_0$ and $E_1$ are state embeddings, where $E_0$ is assigned to the question and passage, and $E_1$ is assigned to the answer. $W_p$ is a pre-trained position embedding. Then, the probability of the subword to generate is calculated as:

$$h_{[t]} = \textit{transformer\_block}(h_{0[t]}) \tag{9}$$

$$P(y)_{t+1} = \textit{softmax}(E^{\top}(h_{[t]})), \tag{10}$$

where $y \in V$, and $V$ stands for the sub-word vocabulary. We simplify the structure of the transformer block as *transformer_block*. The hidden vector of $t_{\text{th}}$ sub-word is used to generate the probability distribution for the vocabulary $(P(y),\ y \in V)$ for $(t+1)_{\text{th}}$ sub-word. $E^{\top}$ means that the model uses the sub-word embeddings in calculating sub-word probabilities for generation.

### 3.3 Prompt-Based Tuning

Following Gekhman et al. [5], we add prompts to the passage for the conversational question answering task, where the prompts indicate the answers to the previous questions. For any turn $i$, all the answer spans of the previous turns $(S_j, A_j)$ $(j \in [1, \ldots, i-1])$ are marked in the passage $X$ with the prompts $<j>$. Examples of prompt-based tuning can be found in the following Table 1:

**Table 1.** An example of prompt-based tuning

| Turn | Question | Text of Answer Span | Prompted Passage |
|---|---|---|---|
| 1 | What color was Cotton? | a little white kitten named Cotton | Once upon a time, in a barn near a farm house, there lived a little white kitten named Cotton. Cotton lived high up... |
| 2 | Where did she live? | in a barn near a farm house, there lived a little white kitten | Once upon a time, in a barn near a farm house, there lived <1> a little white kitten named Cotton <1>. Cotton lived high up... |
| 3 | Did she live alone? | Cotton wasn't alone | Once upon a time, <2> in a barn near a farm house, there lived <1> a little white kitten <2> named Cotton <1>. Cotton lived high up... |

Note that for any turn $j$ that does not have an answer span, there is not a prompt $<j>$ for it.

### 3.4 Instruction-Based Tuning

Furthermore, following Zhong et al. [29] and Gupta et al. [6], we add instructions to the inputs. We use two kinds of instructions: an *instruction* at the beginning of the input, and several *guidances* among the sections that constitute the input. The instruction at the beginning of the input is word-based, and it introduces what the task is about. The guidances are word-based with symbols, such as "[Instruction]:", "[Question]:", "[Passage]:" and "[Answer]:", which separate each section and clarify what each section is. We denote an instruction as a sequence of tokens: $I = I_1, \ldots, I_j$, and guidances for each section as $G_{\text{Section 1}}, G_{\text{Section 2}}, \ldots$. The instruction and the guidances are inserted into the original input as follows:

$$[G_{\text{instruction}}, I, G_{\text{question}}, Q, G_{\text{passage}}, X, G_{\text{answer}}, Y], \tag{11}$$

where $Q$ is the question, $X$ is the passage, and $Y$ is the answer. $Q$, $X$ and $Y$ are all sequences of tokens, and in Eq. 11 they are concatenated. We denote $X_I = [G_{\text{instruction}}, I, G_{\text{question}}, Q, G_{\text{passage}}, X, G_{\text{answer}}]$, then the hidden vector to be input to the transformer block is calculated as:

$$h_{0[t]} = E(X_I, Y_{[1:t]}) + (E_0, E_1) + W_p, \tag{12}$$

### 3.5 Multi-task Learning

To fully leverage the extractive question answering task, we employ a multi-task learning approach to integrate it with the response generation task. Specifically,

we use the same hidden vector as described in Eq. 7 as input to the transformer block, which is then used for calculating the probability distribution of the vocabulary for the next token, as well as the probability of the start and end position for each token in the passage. The multi-task learning approach optimizes both answer span extraction and response generation simultaneously. The loss is then integrated as:

$$\mathcal{L}_{\text{QA}} = \frac{\mathcal{L}_{\text{start position}} + \mathcal{L}_{\text{end position}}}{2} \tag{13}$$

$$\mathcal{L} = \mathcal{L}_{\text{QA}} + \mathcal{L}_{\text{response generation}}. \tag{14}$$

## 4 Experimental Setup

### 4.1 Dataset

We employ the CoQA (Conversational Question Answering) dataset [21] for our research. The CoQA dataset is a collection of conversational question answering instances spanning a broad range of domains, such as literature, news, and Wikipedia articles. The dataset is conversational because it includes conversational histories, i.e., the previous turns in a conversation leading up to the current question-answer pair. The answers in the dataset include both answer spans for extractive question answering and human-written free-form answers for generative question answering.

### 4.2 Model and Tuning

In the experiments, we will evaluate 5 models:

(1) Response generation (baseline)
(2) Response generation with prompt-based tuning (`prompt`)
(3) Response generation with prompt-based tuning & instruction-based tuning (`w instruct`)
(4) Response generation with prompt-based tuning & multi-task learning (`w multi-task`)
(5) Response generation with prompt-based tuning & instruction-based tuning & multi-task learning (`w multi-task & w instruct`)

We have excluded three other settings, namely response generation with instruction-based tuning, response generation with multi-task learning, and response generation with instruction-based tuning & multi-task learning, since prompts are necessary indicators for multi-turns. Our task–the conversational question answering–is based on multi-turns, so any model without prompt-based tuning, other than the baseline, is considered not relevant to the task.

The instructions and prompts that we used in the prompt-based tuning and instruction-based tuning are described in the following Table 2:

**Table 2.** An example for prompt and instruction based tuning

| | Prompt-Based Tuning | Instruction-Based Tuning |
|---|---|---|
| Instruction | \ | [Instruction]:<br>Answer the question based on the given passage |
| Question | Where did she live? | [Question]:<br>Where did she live? |
| Passage | Once upon a time, in a barn near a farm house, there lived <1> a little white kitten named Cotton <1>. Cotton lived high up... | [Passage]:<br>Once upon a time, in a barn near a farm house, there lived a little white kitten named Cotton. Cotton lived high up... |
| Answer | in a barn | [Answer]:<br>in a barn |

### 4.3 Training

Our implementation makes use of Pytorch [16] and the HuggingFace Transformers[1]. We adopted GPT-2 basic[2] which has 12 layers and 12 heads with a dimension of 768. The training procedure was with a batch size of 16, 10 epochs, a learning rate of $3 \cdot 10^{-5}$, a weight decay of 0.01, cross-entropy loss and AdamW. The input sequences are 1024 tokens.

### 4.4 Evaluation

We evaluate the similarity between the human input answers and the generated answers using the F1 score. We compare the performance of five models, namely the baseline, `prompt`, `w instruct`, `w multi-task`, and `w multi-task & w instruct`, using the official dev dataset for evaluation. We compare the latter 4 models with the baseline and the latter 3 models with the prompt model. To ensure consistency, we limit the maximum output length to 64 tokens. We use two different evaluation modes, decoding mode and evaluation mode, to assess the performance of the models.

In decoding mode, models are not provided with any information about the previous turns and are required to use the predicted answer spans from the previous turn as prompts for generating responses. Only models with multi-task learning can generate answers under this mode. In contrast, the evaluation mode provides the correct information on previous turns to the models. This mode enables pure generation models to handle multi-turns with prompts, thus making them more accurate in generating responses. We employ prompt-tuning in the evaluation mode, whereby the correct information on the previous answers is prompted in the same way as introduced in Sect. 3.3.

[1] https://huggingface.co/.
[2] https://huggingface.co/gpt2.

By default, the evaluation mode generates better results than the decoding mode, given the correct information on previous turns. We provide results for both the evaluation mode and decoding mode to ensure a comprehensive evaluation. In many real-life scenarios, we cannot assume that we have access to the correct answer spans for previous questions, which makes evaluation using the evaluation mode impractical. Therefore, by including decoding mode results, we can provide a more realistic evaluation of our approach that reflects the real-life scenarios.

We also evaluate the performance of the extractive QA part of the two models with multi-task learning (`w multi-task` and `w multi-task & w instruct`) and compare them with an GPT-2 model fine-tuned on extractive question answering task. We measure the similarity between the predicted answer span text and the original answer span text using the F1 score.

We show which mode is applied for each model in the following Table 3:

**Table 3.** Models and modes

| | Decoding Mode | Evaluation Mode |
|---|---|---|
| baseline | ✗ | ✓ |
| `prompt` | ✗ | ✓ |
| `w instruct` | ✗ | ✓ |
| `w multi-task` | ✓ | ✓ |
| `w multi-task & w instruct` | ✓ | ✓ |

## 5 Results

### 5.1 Automatic Results

Table 4 and Table 5 summarize the response generation performance of five models w.r.t. F1 score and its improvements. Since only models with multi-task learning can generate answers in the decoding mode, we use backslash '\' to denote this setting is not applicable to the first three models.

**Table 4.** F1 results for different models. Numbers in the brackets state F1 improvements compared to the baseline under evaluation mode.

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| baseline | \ | 53.8 |
| `prompt` | \ | 63.0 (+17.1) |
| `w instruct` | \ | 63.7 (+18.4) |
| `w multi-task` | 61.6 (+14.4) | 63.9 (+18.7) |
| `w multi-task & w instruct` | 56.5 (+5.0) | 57.8 (+7.4) |

**Table 5.** F1 improvement compared to `prompt` (evaluation mode)

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| `w instruct` | \ | +1.1 |
| `w multi-task` | −2.2 | +1.4 |
| `w multi-task & w instruct` | −10.3 | −8.2 |

From the results, we have the following observations:

1) As shown in all the tables, the performance of the evaluation mode is better than decoding mode. This is because the evaluation mode can provide the correct answer spans from previous turns to the models for prompt-tuning.
2) In Table 4, prompt-based tuning outperforms baseline by a large margin, demonstrating that prompt can encode valuable information about the answers from previous conversation turns for model tuning. Besides, instruction-based tuning can further improve the response generation performance, which proves the usefulness of injecting task-specific guidance during fine-tuning. Apart from that, compared with the "`prompt`" model and the "`w instruct`" model, the "`w multi-task`" model achieves the best performance, from which we can find the conversational question answering task can significantly facilitate the response generation task.
3) The brackets of Table 4 show the F1 score improvements compared to the baseline under evaluation mode. As expected, all the models have certain performance improvements compared to the baseline. In particular, the "`w multi-task`" model has the highest performance improvement, which is 18.7% and 14.4% in the evaluation and decoding modes, respectively.
4) Table 5 shows the F1 score improvement compared to the "`prompt`" model (evaluation mode). We find that the performance of the "`w multi-task`" model drops by 2.2% in the decoding mode, suggesting that answer prediction errors from previous conversation turns can accumulate to have a large impact on the response generation task. Another interesting observation is that the performance of the "`w multi-task & w instruct`" model drops 10.3% and 8.2% in the decoding and evaluation modes, respectively. This is probably because the optimization of the multi-task learning and instruction-based tuning are conflicting with each other.

**Table 6.** F1 results and improvement for the extractive question answering part. Answer span texts instead of human answers are used for evaluation.

| | F1 (decoding mode) | F1 (evaluation mode) |
|---|---|---|
| GPT-2 fine-tuned on extractive QA | 63.9 (\) | 64.7 (\) |
| `w multi-task` (QA part) | 60.2 (−5.7) | 65 (+0.4) |
| `w multi-task & w instruct` (QA part) | 64.9 (+1.6) | 70.1 (+8.3) |

Table 6 reports the evaluation results of the extractive question answering part of a GPT-2 model fine-tuned on extractive question answering task and the two models with multi-task learning. Compared with the baseline (GPT-2 fine-tuned on extractive question answering), both multi-task learning models can improve the performance of question answering task, which demonstrates the effectiveness of prompt-based and instruction-based tuning and the boosting effect of the response generation task on the question answering task. We can also observe that the performance of the "`w multi-task`" model drops by 5.7% in the decoding mode, which is due to the accumulated answer prediction errors from previous turns.

### 5.2 Qualitative Results

**Table 7.** An example of the difference between extractive question answering and generated answers

| Question | Gold Answer Span Text | Human | Extractive QA Answer | Generated |
|---|---|---|---|---|
| Is it a small city? | the most populated city in the state of Nevada | No | is the 28th-most populated city in the United States | No |
| Which state is it in? | Vegas, is the 28th-most populated city in the United States, the most populated city in the state of Nevada | Nevada | is the 28th-most populated city in the United States, the most populated city in the state of Nevada | Nevada |
| What is it famous for? | The city bills itself as The Entertainment Capital of the World, and is famous for its mega casino hotel | mega casino hotel | famous for its mega casino hotels and associated activities | gambling, shopping, fine dining, entertainment, and nightlife |

Table 7 presents a comparative analysis between answer spans predicted by the question answering module and generated answers. The first question demonstrates that for yes/no questions, the generated answer provides a more direct response, whereas the extractive QA answer only provides the information required to answer the question without a simple yes or no. The second question highlights that in cases where there is no direct answer in the passage, the

generated answer provides a better response as it directly addresses the question. However, the third question illustrates that in some cases, extractive QA answers are superior, as the given answer is fully grounded in the passage. The generated answer may be based on the passage and relevant to the question, but not necessarily grounded in the passage.

**Table 8.** An example of answers generated by different models

| Question | Baseline | `prompt` | `w instruct` | `w multi-task` | `w multi-task & w instruct` |
|---|---|---|---|---|---|
| What is it famous for? | its the largest city within the greater Mojave Desert | its real things | its gambling, shopping, fine dining, entertainment, and nightlife | gambling, shopping, fine dining, entertainment, and nightlife | a guitar hotels and associated activities |

Table 8 provides a comparative analysis of answers generated by different models. The baseline model generates answers that are not related to the question, while the "`prompt`" model generates answers that are related to the question but not grounded in the passage. In contrast, the "`w instruct`" and "`w multi-task`" models generate good quality answers that are grounded in the passage. The "`w multi-task & w instruct`" model generates an answer that is almost identical to the gold standard, however with a deviation in the form of "guitar hotels" instead of "mega casino hotels". Qualitatively, the "`w instruct`" and "`w multi-task`" models can generate better and more robust answers compared to the baseline and the "`prompt`" model.

## 6 Conclusion and Future Works

This study aimed to explore different tuning approaches for response generation in conversational question answering. Specifically, we experimented with the effectiveness of prompt tuning, instruction tuning, and multi-task learning on GPT-2, under both decoding mode and evaluation mode. The F1 results demonstrated that prompt-based tuning outperformed the baseline, while models with instruction-based tuning and multi-task learning yielded slightly better results than those with prompt-based tuning alone. In the future, we will explore more multi-task learning algorithms and test instruction-based tuning on a larger language model.

**Acknowledgements.** This paper is funded by the collaborative project of DNB ASA and Norwegian University of Science and Technology (NTNU). We also received assistance on computing resources from the IDUN cluster of NTNU [23]. We would like to thank Jon Atle Gulla for his helpful comments.

## References


1. Bragg, J., Cohan, A., Lo, K., Beltagy, I.: FLEX: unifying evaluation for few-shot NLP. Adv. Neural. Inf. Process. Syst. **34**, 15787–15800 (2021)
2. Chada, R., Natarajan, P.: FewshotQA: a simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 6081–6090 (2021)
3. Chen, X., Zhang, Y., Deng, J., Jiang, J.Y., Wang, W.: Gotta: generative few-shot question answering by prompt-based cloze data augmentation. In: Proceedings of the 2023 SIAM International Conference on Data Mining (SDM) (2023)
4. Deng, Y., et al.: Multi-task learning with multi-view attention for answer selection and knowledge base question answering. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 6318–6325 (2019)
5. Gekhman, Z., Oved, N., Keller, O., Szpektor, I., Reichart, R.: On the robustness of dialogue history representation in conversational question answering: a comprehensive study and a new prompt-based method. arXiv preprint arXiv:2206.14796 (2022)
6. Gupta, P., Jiao, C., Yeh, Y.T., Mehri, S., Eskenazi, M., Bigham, J.: InstructDial: improving zero and few-shot generalization in dialogue through instruction tuning. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, UAE, pp. 505–525. Association for Computational Linguistics (2022). https://aclanthology.org/2022.emnlp-main.33
7. Izacard, G., Grave, É.: Leveraging passage retrieval with generative models for open domain question answering. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pp. 874–880 (2021)
8. Joshi, M., Chen, D., Liu, Y., Weld, D.S., Zettlemoyer, L., Levy, O.: SpanBERT: improving pre-training by representing and predicting spans. Trans. Assoc. Comput. Linguist. **8**, 64–77 (2020)
9. Khashabi, D., Baral, C., Choi, Y., Hajishirzi, H.: Reframing instructional prompts to GPTk's language. In: Findings of the Association for Computational Linguistics: ACL 2022, pp. 589–612 (2022)
10. Khashabi, D., et al.: UnifiedQA: crossing format boundaries with a single QA system. In: Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 1896–1907 (2020)
11. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: a lite BERT for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942 (2019)
12. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. Adv. Neural. Inf. Process. Syst. **33**, 9459–9474 (2020)
13. Min, S., Lewis, M., Zettlemoyer, L., Hajishirzi, H.: MetaICL: learning to learn in context. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2791–2809 (2022)
14. Min, S., et al.: Rethinking the role of demonstrations: what makes in-context learning work? arXiv preprint arXiv:2202.12837 (2022)
15. Ouyang, L., et al.: Training language models to follow instructions with human feedback. Adv. Neural. Inf. Process. Syst. **35**, 27730–27744 (2022)

16. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. Advances Neural Inf. Process. Syst. **32** (2019)
17. Prasad, A., Hase, P., Zhou, X., Bansal, M.: GrIPS: gradient-free, edit-based instruction search for prompting large language models. arXiv preprint arXiv:2203.07281 (2022)
18. Puri, R.S., Mishra, S., Parmar, M., Baral, C.: How many data samples is an additional instruction worth? arXiv preprint arXiv:2203.09161 (2022)
19. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(1), 5485–5551 (2020)
20. Ram, O., Kirstain, Y., Berant, J., Globerson, A., Levy, O.: Few-shot question answering by pretraining span selection. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 3066–3079 (2021)
21. Reddy, S., Chen, D., Manning, C.D.: CoQA: a conversational question answering challenge. Trans. Assoc. Comput. Linguist. **7**, 249–266 (2019). https://aclanthology.org/Q19-1016
22. Shen, T., et al.: Multi-task learning for conversational question answering over a large-scale knowledge base. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 2442–2451 (2019)
23. Själander, M., Jahre, M., Tufte, G., Reissmann, N.: EPIC: an energy-efficient, high-performance GPGPU computing research infrastructure (2019)
24. Tafjord, O., Clark, P.: General-purpose question-answering with macaw. arXiv preprint arXiv:2109.02593 (2021)
25. Wei, J., et al.: Finetuned language models are zero-shot learners. In: International Conference on Learning Representations (2022)
26. Weller, O., Lourie, N., Gardner, M., Peters, M.E.: Learning from task descriptions. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1361–1375 (2020)
27. Xiong, W., et al.: Answering complex open-domain questions with multi-hop dense retrieval. In: International Conference on Learning Representations (2021)
28. Xu, H., et al.: ZeroPrompt: scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization. arXiv preprint arXiv:2201.06910 (2022)
29. Zhong, W., et al.: ProQA: structural prompt-based pre-training for unified question answering. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, USA, pp. 4230–4243. Association for Computational Linguistics (2022). https://aclanthology.org/2022.naacl-main.313
30. Zhu, C., Zeng, M., Huang, X.: SDNet: contextualized attention-based deep network for conversational question answering. arXiv preprint arXiv:1812.03593 (2018)

ISBN 978-82-326-7742-9 (printed ver.)
ISBN 978-82-326-7741-2 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (online ver.)





NTNU

Norwegian University of
Science and Technology