Towards Controllable and Data-Efficient Natural Language Generation

---

A

Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Wanyu Du

May 2024

APPROVAL SHEET

This

Dissertation

is submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Author: Wanyu Du

This Dissertation has been read and approved by the examing committee:

Advisor: Yangfeng Ji

Advisor:

Committee Member: Matthew Dwyer

Committee Member: Yen-Ling Kuo

Committee Member: Cong Shen

Committee Member: Xiaodong Liu

Committee Member: Sebastian Gehrmann

Committee Member:

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, School of Engineering and Applied Science

May 2024

ABSTRACT

Large Language Models (LLMs) have revolutionized artificial intelligence (AI) applications by significantly simplifying users' efforts to provide detailed and task-specific instructions to AI systems. The adoption of the transformer architecture has notably enhanced LLMs' ability to learn intricate linguistic patterns and world knowledge from extensive text datasets. Moreover, the large-scale pre-training on multiple natural language processing (NLP) tasks enables LLMs to address a wide range of NLP challenges with improved performance and efficiency. While LLMs excel in natural language understanding, they still face undeniable challenges in natural language generation (NLG). These challenges include generating undesired outputs (e.g., factually incorrect, harmful, biased contents), difficulty in adapting to low-resource domain-specific tasks, and misalignment with user intentions. To address the above challenges, this dissertation introduces methods to refine the generative capabilities of LLMs through controllable and data-efficient natural language generation techniques. The goal is to improve the alignment of generated content with user intentions using controllable and data-efficient LLMs. We design three major components to achieve the goal: (1) user intention identification to align models with human preferences; (2) controllable LLMs to produce outputs that meet specific user requirements; (3) data-efficient NLG to enhance LLM adaptability to low-resource domain-specific tasks. In summary, this dissertation provides a framework to understand user intentions and develop controllable generation methods to align LLMs with these intentions. At the end of the dissertation, an interactive text generation application is presented to demonstrate the benefits of leveraging user intentions and controllable LLMs for human-AI collaborative text generation.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Yangfeng Ji for his mentoring, support, and inspiration over the past six years. I wouldn't have started my academic research journey without him. He provided a lot of guidance for me in coding, problem analysis, paper writing and presentation, and most importantly, a good mindset to do research. I really enjoy working and discussing research with him.

I am grateful for having Matthew Dwyer, Yen-Ling Kuo, Cong Shen, Xiaodong Liu, and Sebastian Gehrmann as my dissertation committee members. Thank you for reviewing my work and providing valuable feedback and suggestions that help me improve my research.

I also want to show my gratitude to other amazing researchers I worked with. Thank Yanjun Qi for providing me with the precious job opportunity to work as an applied scientist at AWS AI after graduation. Thank Song Feng, James Gung, Yi Zhang, Saab Mansour, and Lijia Sun for mentoring and supporting me during my internship at AWS AI. Thank Donegyeop Kang, Vipul Raheja, Zae Myung Kim, and Dhruv Kumar for sharing an inspiring and fruitful collaboration with me during my internship at Grammarly. Thank Liwei Wang and Jianqiao Zhao for broadening my research horizons during my internship at Tencent AI lab.

I am fortunate to be a member of the Information and Language Processing (ILP) lab at UVa, working and studying with other amazing graduate students. I would also like to thank all my lab mates, Hanjie Chen, Aidan San, Stephanie Schoch, Hannah Chen, Dane Williamson, Minjae Kwon, for their kind help in my research and daily life. Besides, I want to thank my friends, Hanjie Chen, Danhua Zhang, Zhe Wang, Guangtao Zheng, Wenbo Pang, Jing Ma, ..., for their support and companionship during my PhD journey.

Finally, my deepest thanks to my grandma and dad, for their love that supports me going through the hard time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

With the fast development of deep learning and natural language processing (NLP) techniques, large language models (LLMs) [1, 2, 3, 4] have demonstrated remarkable contextual understanding capabilities, significantly improving user's efficiency in various writing tasks, including translation [5], summarization [6], creative writing [7], and text revisions [8]. By simply writing instructions in natural language, users can prompt LLMs to produce outputs that adaptively meet their specified requirements.

The success of LLMs could be attributed to four major factors: (1) the adoption of Transformer-based model architectures [9], which leverages the self-attention mechanism to capture long-range dependencies between words and model complex linguistic patterns, addressing the limitations faced by LSTM architectures [10] when stacking up to deeper layers; (2) the extensive pre-training on large-scale text datasets, which enables LLMs to have broad general world knowledge and learn various language patterns [11]; (3) the implementation of a unified text-to-text framework [12], which enables one single LLM to solve multiple NLP tasks; and (4) the alignment with human preferences with fine-tuning [1], which ensures LLMs more accurately adapt to users' requirements. The state-of-the-art LLMs [3, 4] are pre-trained on expansive internet-based text corpora, learning to predict subsequent tokens based on previous input. They are further fine-tuned through task-specific instructions and human preference scores using reinforcement learning techniques [13, 14], achieving human-level performance across a variety of NLP tasks [15].

1.1 Limitations of LLMs in Text Generation

Although current LLMs exhibit remarkable proficiency in understanding natural language, they nonetheless possess inherent limitations in natural language generation (NLG) and

human-AI collaboration. These constraints impede their integration into the development of user-centric AI applications.

For the limitations in NLG, current LLMs are susceptible to generating outputs that might be harmful, biased, or even factually incorrect [16, 17, 18], which motivates the development of effective methods to control the generation of undesired outputs. Furthermore, current LLMs still require a decent amount (e.g., a few thousand) of high-quality labeled data to be adapted to domain-specific tasks (e.g., task-oriented dialogue state tracking, medical document analysis, and legal document generation) [19, 20, 21]. However, collecting large-scale labeled data for domain-specific tasks not only demands in-depth expert knowledge, but also incurs significant costs due to the need for meticulous human annotations. Therefore, the development of data-efficient methods in NLG is imperative to better adapt LLMs to these domain-specific tasks.

From the perspective of human-AI collaboration, LLMs provide a convenient natural language interface for human users to interact with AI systems to perform challenging tasks with improved efficiency. Current LLMs interact with users through text-based conversational interfaces, and such interaction mode requires large amounts of user's cognitive load to formulate task instructions and read model outputs. Besides, LLMs sometimes misinterpret ambiguous and generic user instructions. The failure to identify user intentions may incur the loss of the user's trust and patience with the LLMs. Additionally, there is a risk of diminishing user engagement and satisfaction, if the system continues querying the user without resolving their issues expediently [22, 23, 24]. Users' patience may decay if the system fails to deliver solutions within a tolerable number of interactions. Therefore, it is important to conduct thorough analyses of user behaviors and preferences to inform the design of more intuitive user interfaces. Such enhancements are crucial to augment the accessibility and utility of LLMs, thereby extending their benefits to a broader spectrum of users.

1.2 Controllable Text Generation

Controllable text generation techniques aim at improving the alignment of LLMs with human preferences, including generating factually correct, contextually relevant, unbiased, and harmless outputs [25]. Previous methods control the generation of LLMs through supervised fine-tuning on millions of training data, which is data-intensive [26, 27], or attribute model that updates LLMs during inference, which is computation-intensive [28, 29]. Recent works design different prompting strategies to control LLMs generating desired outputs, which is task-dependent [30, 31].

The supervised fine-tuning methods train a conditional language model from scratch [26, 32, 33]. This conditional language model learns the control attributes as latent variables, and generates the desired outputs by conditioning on the target control attributes. While the direct training of conditional language models achieves a good performance in both controllability and text quality, it requires a large amount of training data to learn the latent distribution of the control attributes.

The attribute model methods train a smaller attribute network to guide the generation of LLMs [34, 35, 28, 36]. The attribute network, crucial for encoding control attribute information, can be implemented through various approaches, each with its own set of strengths and limitations. Some popular baseline methods are included as follows. First, the plug-and-play model [34], which takes the gradient of the attribute model to update the base LLM at each decoding time step. Although it exhibits excellent control over the output, the plug-and-play model significantly increases the computational cost during inference. Second, the scoring function for weighted decoding [36], which adjusts the probability distribution of the base LLM during inference using a scoring function. While it reduces the decoding cost compared to the plug-and-play model, its ability to control outputs is ultimately constrained by the capabilities of the base LLM. Third, the reward model for reinforcement learning applies a reward function to guide the generation policy to produce

high-reward outputs [37]. Despite its effectiveness in controlling generation, this method risks "hacking" the reward system, potentially compromising the quality of the generated text.

The prompt engineering methods leverage the general knowledge of LLMs to produce desired outputs [38]. Prompt engineering designs task instructions and samples a set of input-output training instances as the context information to guide LLMs generating outputs in a similar format. Although this approach significantly reduces the need for labeled data, it implicitly presupposes that LLMs already possess a reasonable level of proficiency in the target tasks, which may not hold true in some domain-specific tasks, including task-oriented dialogue state tracking, medical document analysis, and legal document generation.

This dissertation addresses these limitations by designing new loss functions and neural network modules along the data-driven pipelines. To enhance computation and data efficiency, I propose fine-tuning a small side network to encode control attributes and integrate it with pre-trained LLMs [39]. This is complemented by a specific control loss to prevent ambiguous and generic responses. Moreover, recognizing that some training data, especially LLM-generated data, can be very noisy and uninformative, I design a model-uncertainty-based data selection method to select training data that is beneficial for learning [40]. To address the generation of factually incorrect content, I devise a new reward function and fine-tuned LLMs with reinforcement learning to ensure the model's outputs remain faithful to the input knowledge texts [41].

1.3 Human-AI Collaborative Generation

Previous works on human-AI collaborative text generation have investigated the interactions with LLMs that support story generation [42, 43], text revision [44, 8], or creative writing [7]. However, there's a notable lack of focus on the alignment with user intentions during human-AI interactions.

User intentions reflect the goal of the user when interacting with the system, which

covers a wide range of objectives across different domains. In task-oriented dialogues, for instance, intentions can range from finding a flight, setting an alarm, to sending an invite. Within open-domain chit-chat, users might aim to ask questions, provide information, or offer feedback. Meanwhile, intentions in formal writing revisions could involve correcting grammatical mistakes, enhancing readability, or altering the writing style. Recognizing these intentions is critical in guiding the system's responses and functionalities to meet users' specific needs effectively.

Some works on text revision [45, 46, 47, 8] have proposed human-AI collaborative writing interfaces to collect human-AI interaction data for training better neural models. Other works [48, 42] on creative writing designed human-AI interaction interfaces to encourage new content generation. Another line of works on interactive text generation [49] builds user simulators to test the robustness of the interactive text generation system. As mentioned above, those human-AI interfaces do not focus on the alignment with user intentions during human-AI interactions.

This dissertation focuses on developing interactive text generation systems for the text revision task, which leverages LLMs to improve the quality of formal writing with less user effort. To achieve this goal, I propose a novel edit intention taxonomy to model user intentions in iterative text revisions, and train controllable LLMs to produce accurate, formal, and coherent text revisions [50]. To examine the capability of LLMs for making continuous document revisions and collaborating with human writers, I build a human-in-the-loop system [51]. The proposed system achieves high-quality text revisions with minimal human efforts by reading model-generated revisions and user feedback, revising documents, and repeating human-AI interactions.

1.4 Contributions

This dissertation aims to advance the field of human-AI collaborative text generation, focusing on enhancing LLMs to better align with user specifications. The detailed contributions

are summarized as follows.

Designing controllable text generation methods. To improve the diversity of generated outputs, I design a stochastic function to map contextual representations of LLMs into a set of random context variables, and control the model generating more diverse texts by sampling from the random context variables. The learned stochastic function introduces context-aware variations into the original representations, which makes the model generate high-quality and high-diversity texts.

To control the faithfulness and accuracy of the model outputs, I fine-tune LLMs using reinforcement learning with multiple rewards. Our data analysis found that reference answers typically share a high degree of overlap with ground-truth knowledge spans to maintain their accuracy and faithfulness. Motivated by this finding, we study a collection of reward functions and identify the most effective yet simple reward function for conversational information-seeking tasks.

Developing data-efficient learning algorithms for domain-specific NLG tasks. To guide the LLMs generating outputs that align with specific user persona profiles using a few training data, I propose to add a small side network, which encodes the user persona information, on top of the last hidden states of LLMs before producing the final probability distribution. Training a small side network requires much less labeled data and computational resources than fine-tuning the whole LLMs, and inserting it before producing the final probability distribution differentiates the side network with the scoring functions used in weighted decoding.

Besides, collecting large-scale labeled data in domain-specific NLG tasks can be prohibitively expensive, therefore, I propose to leverage the self-training algorithms to adapt LLMs to low-resource domains. The self-training algorithm is a semi-supervised learning algorithm that trains the model with a small set of labeled data and a large set of unlabeled data, assuming that the model can obtain better generalization capability by learning from

the unlabeled data. We leverage the model prediction uncertainty to select the informative unlabeled data, and conduct pseudo-label enhancement to reduce the prediction errors.

Building human-AI collaborative text generation applications. To build user-centric NLG applications, I collect real-world data and analyze human behaviors in the text revision task. By analyzing the text revision data from human-written texts, we better understand the text revision process, making vital connections between edit intentions and writing quality, and then train controllable LLMs to model human revision patterns. Next, I develop a human-in-the-loop text revision system based on the controllable LLM to provide high-quality text revisions with minimal human effort. Finally, we analyze the user-system interactions to better understand user preferences, which provides insights on how to further improve the system's performance.

For the rest of the dissertation, Chapter 2 discusses the background for controllable text generation, data-efficient text generation, user intention identification, and human-AI interaction applications. In Chapter 3, I present two works on controllable LLMs development, including controlling text diversity via variational encoder-decoders and controlling knowledge texts via reinforcement learning. In Chapter 4, I introduce two works on data-efficient controllable text generation, including personalized dialogue generation via additive side networks and task-oriented dialogue generation via a self-training algorithm. In Chapter 5, I demonstrate two works on user intention modeling for open-domain dialogue evaluation and iterative text revision respectively. In Chapter 6, I present one work on building a human-AI collaborative text generation application for the iterative text revision task. Finally, Chapter 7 summarizes the dissertation.

CHAPTER 2

BACKGROUND

This chapter discusses the recent literature on controllable text generation, data-efficient text generation, user intention identification, and the development of human-AI collaborative text generation applications.

2.1 Controllable Text Generation

Early works on controllable text generation train class-conditional language models from scratch, and guide the generation with explicit control codes provided in the training data. [32] trains a 1.63 billion-parameter Conditional Transformer Language (CTRL) model by prepending control codes in front of raw texts. However, training the CTRL model requires 140 GB of training data [32], which may not be affordable for some low-resource languages. [33] builds a controllable neural conversation model by leveraging an adversarial learning framework that alternatively trains between a class-conditional language model and a multi-class discriminator, where the discriminator is used to help the generative model produce responses with appropriate dialogue act. However, the control code is modeled as a discrete variable in this work, which limits the controllability capacity of the dialogue model.

Another popular approach is guiding generation with gradients from additional attribute models. [34] introduce a plug-and-play language model (PPLM) which combines the pre-trained language model $p(x)$ with attribute models $p(a|x)$ to approximate the conditional generative model $p(x|a)$. At each decoding timestep, all hidden representations of the pre-trained language model are shifted with gradients towards a higher $p(x|a) \propto p(a|x)p(x)$. The attribute models of PPLM are either in the form of bag-of-words or single-layer classifiers, which require much less training data than learning a conditional generative

model. The following works [52, 53, 54] further propose more fine-grained attribute models and generation strategies for specific tasks, such as emotional text generation [52], story generation [53] and conversation generation [54]. However, since the plug-and-play language models have to compute gradients from the attribute model and update hidden representations at each decoding timestep, the generation process is very time-consuming, which leads to high decoding costs.

Recent works on weighted decoding also demonstrate their impressive performance on controllable text generation tasks. Weighted decoding runs a more expensive beam search where the sampling probability distribution is altered by desired control attributes, such as topic, sentiment, etc. [55] design a set of style features on controlling topic, sentiment, and repetitive words, and re-compute the beam score of each token with a combination of the original beam score and the style feature score. A recent work [36] introduces a Future Discriminator for Generation (FUDGE) that trains a binary discriminator for the control attribute prediction and re-scores the probability distribution of the original pre-trained language model with the discriminator prediction via Bayesian factorization. The major limitation of weighted decoding methods is that, if the pre-trained language model is a high-bias estimator, which assigns a low probability for desired attribute words and a high probability for commonly observed but unrelated words, re-scoring or re-ranking such a "high-biased" distribution cannot guarantee the generation of desired attributes.

Finally, using reinforcement learning to control the generation of LLMs by designing specific reward functions is another popular approach. [56] propose to train a reward model from human preference feedback and use reinforcement learning to fine-tune LLMs that can generate outputs that can better align with human preferences. [57] optimize a reward function that quantifies an unwanted property, such as toxicity, negative sentiment, and repetition, and fine-tune LLMs to avoid generating those unwanted properties. [58] design LLM prompts to simulate human feedback and validate it against human instructions obtained on real-world interactions. [59] use fine-grained human feedback to control LLMs

generating desired outputs. The fine-grained rewards come from two aspects: density, providing a reward after every segment is generated; and incorporating multiple reward models associated with different feedback types (e.g., factual incorrectness, irrelevance, and information incompleteness). This line of work has been shown to bring substantial performance improvement than supervised fine-tuning when aligning LLMs to match user intentions.

2.2 Data-Efficient Text Generation

Early works on data-efficient text generation mainly focus on pre-training or adapting large language models. [19] develops a pre-trained language model which can be fine-tuned with only a few domain-specific labels to adapt to new domains, and presents the first few-shot NLG benchmark for task-oriented dialog systems. [60] applies the switch mechanism to combine the information from both input data and pre-trained language models, which achieves good performance in table-to-text generation tasks. [61] studies the training data selection strategies in few-shot NLG, and finds that clustering-based selection strategy consistently helps generative models get better performance than random sampling.

The self-training technique has shown its capability to improve the model's generalization ability in many NLG tasks. Some works [62, 63] leverage the self-training framework to pseudo-label the unlabeled data and select the training data based on the confidence score from a single student model. Other works [64, 65] show that noisy self-training is able to utilize unlabeled data and improve the performance of the supervised baseline. However, their observations come from large-scale training datasets, which may not necessarily hold in the few-shot data setting, because a single Transformer-based model may heavily overfit on the few-shot training data in the early iteration and consequently generate pseudo-label with high prediction errors.

To adopt the self-training algorithm into the few-shot NLG tasks, some works [66, 67, 68] train additional neural models to filter out the pseudo-labeled data with high prediction

errors. [66] and [67] use the reconstruction loss from a fine-tuned BART model [69] to select the pseudo-labeled data. Besides, [68] leverage a fine-tuned BLEURT model [70] with a selection threshold to select pseudo-responses for self-training. Intuitively, the pseudo-labeled data should bring new domain-specific knowledge to the model.

2.3 User Intention Modeling

User intentions reflect the objectives users have when interacting with a system, encompassing a broad spectrum of goals across multiple domains. In task-oriented dialogues, communicative intent [71, 72] and similar concepts, like dialog act [73, 74], have been widely studied in the past decades. [75] construct a dialog act tagging scheme and evaluate travel planning systems [76] based on standalone dialog acts, rather than dialog act sequences. This tagging scheme, while providing more detailed information compared with previous works, only focuses on the system side in a task-oriented setting and may need major modifications when applied to open-domain dialogues. After the initial flourish, recent works come with their own purposes and tagsets for dialog act, tailored for different scenarios or special needs [77, 78, 79]. Despite the potential benefits, the application of dialog acts for evaluating coherence in human-system conversations remains under-explored. Introducing dialog acts into the evaluation of multi-turn dialogues could be advantageous, as they distill the fundamental function of each utterance, offering insights into the overarching interaction patterns. However, directly using existing dialog act definitions [73, 71] seems undesirable, as an utterance can contain several segments that possess different conversational functions, using a single dialog act to express the core function of an utterance inevitably suffers from information loss.

Identification of edit intentions is an integral part of the iterative text revision task. Prior works have studied the categorization of different types of edit actions to help understand why editors do what they do and how effective their actions are [80, 81, 82]. However, these works do not further explore how to leverage edit intentions to generate better-revised

documents. Moreover, some of their proposed edit intention taxonomies are constructed with a focus on specific domains of writing, such as Wikipedia articles [83, 84, 85] or academic essays [81]. As a result, their ability to generalize to other domains remains an open question.

2.4 Human-AI Interaction Applications

This thesis narrows down the scope of the interactive text generation applications to the text revision systems, and leaves other applications (e.g. AI assistants, LLM agents) for future research.

Previous works on modeling text revision [86, 87, 82, 85] have ignored the iterative nature of the task, and simplified it into a one-shot "original-to-final" sentence-to-sentence generation task. However, in practice, at every revision step, multiple edits happen at the document-level which also plays an important role in text revision. For instance, reordering and deleting sentences to improve the coherence. More importantly, performing multiple high-quality edits at once is very challenging. Continuing the previous example, document readability can degrade after reordering sentences, and further adding transitional phrases is often required to make the document more coherent and readable. Therefore, one-shot sentence-to-sentence text revision formulation is not sufficient to deal with real-world challenges in text revision tasks.

While some prior works on text revision [45, 46, 47, 8] have proposed human-machine collaborative writing interfaces, they are mostly focused on collecting human-machine interaction data for training better neural models, rather than understanding the iterative nature of the text revision process, or the model's ability to adjust editing suggestions according to human feedback. Another line of work by [48, 42] on creative writing designed human-machine interaction interfaces to encourage new content generation. However, text revision focuses on improving the quality of existing writing and keeping the original content as much as possible.

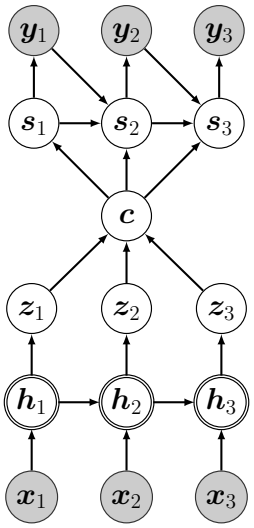
CHAPTER 3
CONTROLLABLE TEXT GENERATION

Controlling LLMs to generate outputs that align with user intentions is important in building user-centric text generation applications. In this chapter, I investigate the user intentions of generating semantically appropriate, diverse, and faithful texts. Specifically, I propose to control the generation of diverse outputs by introducing context-aware variations into the contextual representations of pre-trained LLMs in Section 3.1; and control the generation of faithful and accurate outputs in knowledge-grounded dialogue generation by investigating effective reward functions in reinforcement learning in Section 3.2.

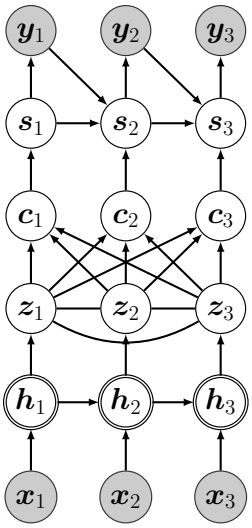
3.1 Controlling Text Diversity via Variational Encoder-Decoders

Generating high-quality texts with high diversity is an important requirement for many text generation applications, such as paraphrase generation [88, 89], style transfer [90, 91], dialog generation [92, 93], etc. The encoder-decoder framework [94, 10] is widely adopted [5, 95, 96, 6, 97] to generate high-quality texts, where an encoder is applied to learn contextual information from source texts and a decoder is used to generate texts for target tasks. To improve the diversity of generated texts, prior works propose to introduce variations into either the encoder [98, 99, 100, 101, 102, 103, 104, 105] or the decoder [106, 107, 108, 109]. However, it is difficult to incorporate meaningful variations into encoder-decoder models without hurting the quality of generated texts.

Promoting diversity on the encoder side mainly focuses on modeling the probabilistic distribution of contextual representations. Some prior works [99, 98] propose to model attention alignments between encoder and decoder hidden states as latent variables, and generate diverse texts by sampling from the latent attention variables. Other existing works [100, 110, 111, 105] directly apply conditional variational autoencoders to model encoder



(a) With a normal Gaussian prior



(b) With a Gaussian process prior

Figure 3.1: A simple illustration of a variational encoder-decoder model with (a) a normal Gaussian prior and (b) a Gaussian process prior. With a normal Gaussian prior, each hidden state $\boldsymbol{h}_i$ will be mapped into a random vector $\boldsymbol{z}_i$ independently; while a Gaussian process prior imposes dependency constraints among $\{\boldsymbol{z}_i\}$. Double circles on $\{\boldsymbol{h}_i\}$ indicate they are deterministic variables.

hidden states as latent variables, and generate high-diversity texts by sampling from the latent context variables. However, when modeling the latent variables, they treat each latent variable as independent of each other, which inevitably causes the loss of some contextual information during learning, as shown in Figure 3.1a.

Other works turn towards designing diversity-promoting decoding strategies at the decoder side, such as diverse beam search [106], top-k sampling [26], and nucleus sampling [107]. But for those decoding strategies, there is often a trade-off between quality and diversity, and the generation models have to sacrifice quality for a higher diversity. Another line of works suggests learning a mixture of expert encoders [101] or decoders [108, 109], and generating diverse texts by sampling from different encoders or decoders. While different expert encoders or decoders can introduce some diversity, the model capacities are limited within the pre-defined set of experts.

In this work, we propose a novel approach to introduce context-aware variations into the encoder in order to generate high-quality and high-diversity texts. For an encoder-decoder

model, we introduce a stochastic function to map deterministic encoder hidden states $\{\boldsymbol{h}_i\}$ into a set of random context variables $\{\boldsymbol{z}_i\}$. The advantage of this stochastic function is that it explicitly models the dependency between each context variable, as shown in Figure Figure 3.1b, which can help preserve more semantic information from source texts. During generation, the decoder generates diverse outputs conditioning on sampled different context variables. In other words, by learning a stochastic function on top of *one* deterministic encoder, the proposed approach offers *many* versions of random context variables for a decoder to generate diverse texts.

To learn the stochastic function over hidden states, we propose a Gaussian process prior (GP) [112] to model the joint distribution of all encoder hidden states. The major differences between GP priors and other priors used in previous works [98, 99, 100, 101, 103, 104, 111, 105] have two-folds: (1) GP priors explicitly model the dependency between latent variables of varying sizes as illustrated in Figure 3.1b, while previous works consider latent variables as independent with each other as shown in Figure 3.1a; (2) GP priors provide *infinite* a number of joint Gaussian distributions of latent variables as shown in Figure 3.3b, while previous works have to pre-define a fixed set of Gaussian distributions (e.g. a standard normal distribution, or a mixture of Gaussian distributions) with the risk of experiencing the posterior collapse problem [113, 114, 115]. Besides, the proposed random function only introduces variations into the encoder, and is orthogonal to diversity-promoting decoding strategies at the decoder side. Users can freely adopt different decoding strategies to further encourage diverse generation outputs.

The major contributions of this work are three-fold:

1. Proposing a novel method to introduce context-aware variations into encoder-decoder models, which can help the model learn rich contextual representations and also promote diversity in generation.

2. Proposing an efficient variational inference method to approximate the joint distribution of fully connected random context variables.

3. Testing our proposed method in both LSTM-based [6] and Transformers-based [97] encoder-decoder models on paraphrase generation and style transfer tasks. Empirical experimental results show that, on one hand, the proposed method can generate higher quality texts than deterministic encoder-decoder models and conditional variational auto-encoders; on the other hand, it also supports diverse generation by conditioning on different sets of sampled random context variables.

3.1.1 Gaussian Processes as Function Priors

In this work, we are interested in learning a non-linear mapping function from encoder hidden states to latent context variables. Gaussian process has the nice property which can represent complex non-linear functions and also allow uncertainty to account for noisy data observations. Considering a set of observed training data $\mathcal{D} = \{(\boldsymbol{h}_i, \boldsymbol{z}_i)\}_{i=1}^{N}$, a Gaussian process defines a probability distribution over possible functions $p(g)$. Given a Gaussian process prior $\mathcal{GP}(\mathbf{0}, \mathbf{K})$ on the function $g(\boldsymbol{h})$, we have:

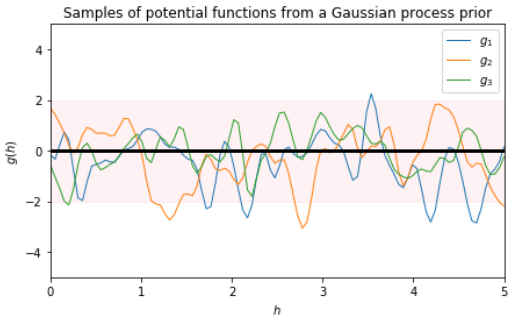
$$\boldsymbol{z}_i = g(\boldsymbol{h}_i) + \epsilon_i \tag{3.1}$$

$$g(\boldsymbol{h}_i) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}) \tag{3.2}$$

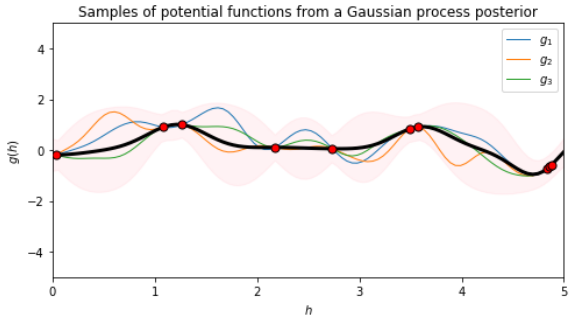
$$\epsilon_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}) \tag{3.3}$$

Note that ϵ_i is the noise of the observed data point $\boldsymbol{z}_i$, which is assumed to be an independent identically distributed Gaussian with variance σ^2. $\mathbf{K}$ is the covariance matrix which is constructed using a squared exponential covariance function $k(\boldsymbol{h}, \boldsymbol{h}') = \exp(-\frac{\|\boldsymbol{h}-\boldsymbol{h}'\|^2}{2})$. Now, we can sample different mapping functions $g(\boldsymbol{h})$ from this Gaussian process prior $\mathcal{GP}(\mathbf{0}, \mathbf{K})$. Figure 3.2a illustrates some possible mapping functions g_1, g_2 and g_3.

In the Gaussian process, each training and testing data point is treated as a random variable which follows Gaussian distribution. Therefore, we can apply the Bayesian inference to predict a testing data point $\boldsymbol{z}_*$ conditioning on observed training data points $\mathcal{D}$. To make con-



(a) Sample g from Gaussian process prior.



(b) Sample g from Gaussian process posterior.

Figure 3.2: Samples of potential functions g from (a) a Gaussian process prior $\mathcal{GP}(\mathbf{0}, \mathbf{K})$ which uses a squared exponential kernel $k(h, h') = \exp(-\frac{(h-h')^2}{2})$, (b) a Gaussian process posterior $p(g \mid \mathcal{D})$ conditioning on the training data $\mathcal{D}$.

cise notation, we let $\boldsymbol{h}_{1:N} = \{\boldsymbol{h}_i\}_{i=1}^N$, $\boldsymbol{z}_{1:N} = \{\boldsymbol{z}_i\}_{i=1}^N$, and $\mathbf{K}^{-1} = [k(\boldsymbol{h}_{1:N}, \boldsymbol{h}_{1:N}) + \sigma^2\mathbf{I}]^{-1}$. The probability distribution of the testing data point $\boldsymbol{z}_*$ can be computed by:

$$p(\boldsymbol{z}_* \mid \boldsymbol{h}_*, \mathcal{D}) = \int p(\boldsymbol{z}_* \mid \boldsymbol{h}_*, \boldsymbol{g}, \mathcal{D}) p(\boldsymbol{g} \mid \mathcal{D}) dg \tag{3.4}$$

where

$$\begin{aligned} &p(\boldsymbol{z}_* \mid \boldsymbol{h}_*, \mathcal{D}) \sim \mathcal{N}(\boldsymbol{\mu}_*, \mathbf{K}_*) \\ &\boldsymbol{\mu}_* = k(\boldsymbol{h}_*, \boldsymbol{h}_{1:N})\mathbf{K}^{-1}\boldsymbol{z} \\ &\mathbf{K}_* = k(\boldsymbol{h}_*, \boldsymbol{h}_*) - k(\boldsymbol{h}_*, \boldsymbol{h}_{1:N})\mathbf{K}^{-1}k(\boldsymbol{h}_{1:N}, \boldsymbol{h}_*) \end{aligned} \tag{3.5}$$

Intuitively, training data points $\mathcal{D}$ constrain the set of functions g to pass through them since the covariance becomes smaller when we have training data, as shown in Figure 3.2b.

Under our variational encoder-decoder framework, $\boldsymbol{h}_{1:N}$ are encoder hidden states and $\boldsymbol{z}_{1:N}$ are latent context variables. Since the Gaussian process induces a distribution over the mapping function $g(\boldsymbol{h})$, theoretically we could sample an infinite number of mapping functions, where each function gives us a different set of latent context representations $\boldsymbol{z}_{1:N}$. In this way, we managed to obtain diverse context representations in encoder-decoder models.

3.1.2 Variational Encoder-Decoder with Gaussian Process priors

This section discusses our novel latent structured variable model on learning rich context representations by transforming the hidden states from a deterministic encoder into random hidden states via stochastic functions.

Encoding with Stochastic Functions. Let $\boldsymbol{x}_{1:N} = \{\boldsymbol{x}_i\}_{i=1}^{N}$ be the source sentence of length N, and $\boldsymbol{y}_{1:T} = \{\boldsymbol{y}_t\}_{t=1}^{T}$ be the target sentence of length T. In encoder-decoder models, an encoder is used to obtain deterministic context representations of the source sentence, i.e. the encoder hidden states: $\boldsymbol{h}_{1:N} = f_{enc}(\boldsymbol{x}_{1:N})$, where $f_{enc}(\cdot)$ is a nonlinear transition function implemented by LSTM [10] or Transformer [116].

To introduce context-aware variations into the encoder, we propose to learn a stochastic function that maps the deterministic hidden states to variables. Specifically, after computing the hidden states $\boldsymbol{h}_{1:N}$ from the transition function $f_{enc}(\cdot)$, the proposed method employs a stochastic mapping function $g(\cdot)$ to model the deterministic context representations as a series of random context variables:

$$p(\boldsymbol{z}_{1:N} \mid \boldsymbol{h}_{1:N}) = g(\boldsymbol{h}_{1:N}) + \boldsymbol{\epsilon} \tag{3.6}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ is a Gaussian noise. Then, the decoder can generate diverse texts conditioning on different sets of context variables sampled from $p(\boldsymbol{z}_{1:N} \mid \boldsymbol{h}_{1:N})$, as shown in Figure Figure 3.3b. Considering that natural language texts are always context-dependent, we expect the random context variables $\boldsymbol{z}_{1:N}$ to encode the context dependency to some extent. In other words, the distribution of $\boldsymbol{z}_i$ will not only depend on $\boldsymbol{h}_i$, but also depend on other $\{\boldsymbol{z}_j\}_{j\neq i}$, as shown in Figure 3.1b.

Under this framework, variational encoder-decoder models [98, 99, 100] can be viewed as a special case, as illustrated in Figure 3.3a, where random context variables $\{\boldsymbol{z}_i\}_{i=1}^{N}$ are *independent* from each other. In this work, we consider this special case as generation with

normal priors.

Gaussian Process Priors for Stochastic Functions. The learning of stochastic function $g(\boldsymbol{h})$ is the key for the proposed method to be successful. Intuitively, we design $g(\boldsymbol{h})$ to satisfy two constraints simultaneously: (1) it can introduce some variation to the deterministic encoder hidden states; (2) it should preserve the contextual information in the deterministic encoder hidden states to be a faithful representation.

In this work, we propose to learn $g(\boldsymbol{h})$ with a functional prior defined by Gaussian processes. As shown in Figure 3.2a, we can sample very different functions $g(\boldsymbol{h})$ from the same GP prior, which ensures randomness when sampling $\boldsymbol{z}_{1:N}$.[1] We define the stochastic function $g(\boldsymbol{h})$ following a GP prior:

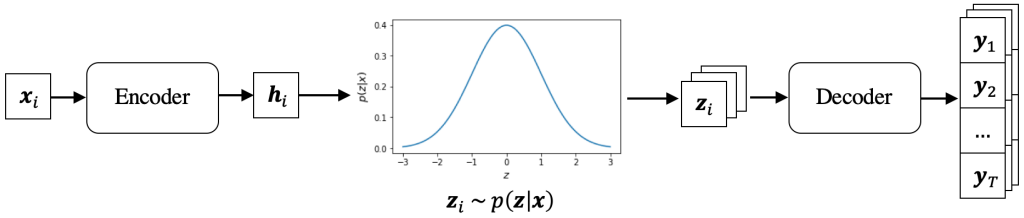
$$g(\boldsymbol{h}) \sim \mathcal{GP}(m(\boldsymbol{h}), k(\boldsymbol{h}, \boldsymbol{h}')) \tag{3.7}$$

with the mean function $m(\boldsymbol{h})$ and covariance function $k(\boldsymbol{h}, \boldsymbol{h}')$ as

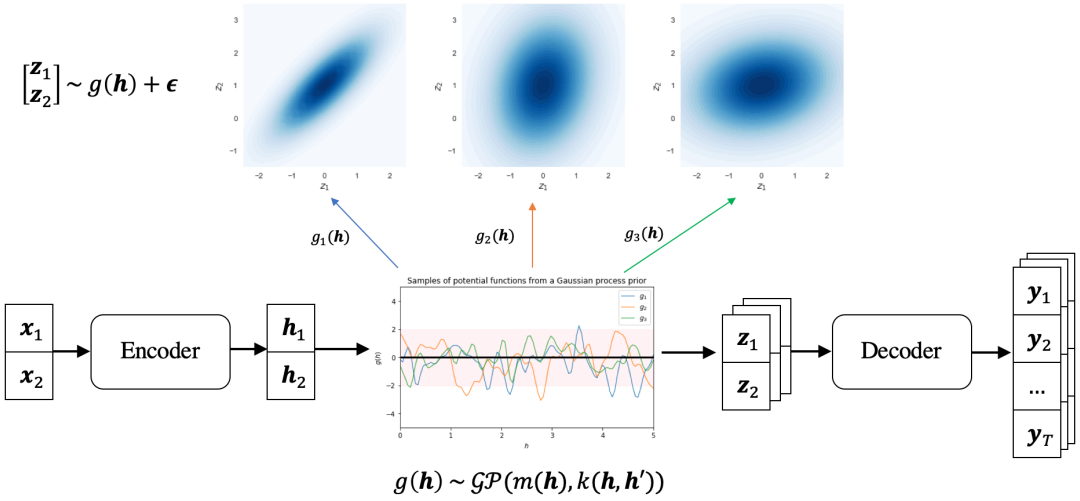
$$\begin{aligned} m(\boldsymbol{h}) &= \boldsymbol{h} \\ k(\boldsymbol{h}, \boldsymbol{h}') &= v^2 \exp\{-\frac{\|\boldsymbol{h} - \boldsymbol{h}'\|_2^2}{2r^2}\} \end{aligned} \tag{3.8}$$

where $\boldsymbol{h}$ indicates the current observed encoder hidden state, and $\boldsymbol{h}'$ indicates the other contextual encoder hidden states; v controls the average distance between a sampled function $g(\boldsymbol{h})$ and the mean function $m(\boldsymbol{h})$, and r controls the covariance between random variables, increasing r will make $\boldsymbol{z}$ and $\boldsymbol{z}'$ become more correlated. In this work, v and r are chosen based on the text generation performance on development sets. By setting $m(\boldsymbol{h}) = \boldsymbol{h}$, we actually define a semi-parametric GP prior [117] instead of a fully non-parameteric prior, since $\boldsymbol{h}$ as a hidden state is computed from the deterministic encoder with learnable parameters. The intuition behind this definition is that, although we want to introduce some

[1] Please refer to subsection 3.1.1 and [112] for detailed introduction of Gaussian processes.



(a) Variational encoder-decoder model with a normal Gaussian prior. Note that we simplify the latent variable z_i from a vector to a scalar in order to plot out the Gaussian distribution for better illustration.



(b) Variational encoder-decoder model with a GP prior. Note that we simplify the latent variable z_i from a vector to a scalar in order to plot out the joint Gaussian distribution for better illustration.

Figure 3.3: A simple illustration for comparison between our GP priors and the priors in conditional variational autoencoders [118] under the variational encoder-decoder framework.

variations, taking the expectation of the sampled random states $\boldsymbol{z}$ should still be $\boldsymbol{h}$.

The main advantage of applying GP priors is that we can sample an infinite number of random functions $g(\boldsymbol{h})$ thus obtaining infinite sets of random context variables $\boldsymbol{z}_{1:N}$, as illustrated in Figure 3.3b. In contrast, standard variational encoder-decoder models can only learn a fixed set of C joint distributions $p(\boldsymbol{z}_{1:N}|\boldsymbol{h}_{1:N})$, where $1 \leq C \ll \infty$.[2]

Generation with Random Context Variables. Now, we demonstrate how to incorporate $\boldsymbol{z}_{1:N}$ into two typical encoder-decoder models for text generation: an LSTM-based encoder-

[2]When $C = 1$, it represents a conventional variational autoencoder [113]; when $C = 5$, it represents a variational autoencoder with a mixture of Gaussians prior (component number = 5); when $C \to \infty$, it represents a variational autoencoder with a GP prior.

Algorithm 1: The generative story with a stochastic function $g(\cdot)$ sampled from the GP prior

Input: A source sentence $\boldsymbol{x}_{1:N}$
Output: A generated sentence $\boldsymbol{y}_{1:T}$
// Encode context
Initialize $\boldsymbol{h}_0 \leftarrow \mathbf{0}$
for $i = 1, \ldots, N$ **do**
    Compute $\boldsymbol{h}_i = f_{enc}(\boldsymbol{h}_{i-1}, \boldsymbol{x}_i)$
end for
// Sample random context variables
Draw $g(\boldsymbol{h}) \sim \mathcal{GP}(m(\boldsymbol{h}), k(\boldsymbol{h}, \boldsymbol{h}'))$
Draw $\boldsymbol{z}_{1:N} \sim g(\boldsymbol{h}_{1:N}) + \boldsymbol{\epsilon}$
// Generate a new sentence
Initialize $\boldsymbol{s}_0 \leftarrow \mathbf{0}$
for $t = 1, \ldots, T$ **do**
    Compute $\boldsymbol{s}_t = f_{dec}(\boldsymbol{s}_{t-1}, \boldsymbol{y}_{t-1}, \boldsymbol{z}_{1:N})$
    Draw $\boldsymbol{y}_t \sim \text{softmax}(\mathbf{W} \cdot \boldsymbol{s}_t)$
end for

decoder model [6] and a Transformer-based encoder-decoder model [97]. The performance of these two variational encoder-decoder models will be evaluated in subsection 3.1.4.

Given the deterministic encoder hidden states $\boldsymbol{h}_{1:N}$, we first sample a function $g(\boldsymbol{h})$ from the GP prior in Equation 3.7; then sample a set of random context variables $\boldsymbol{z}_{1:N}$ from $g(\boldsymbol{h})$; finally generate a output sentence $\boldsymbol{y}_{1:T}$ based on the sampled $\boldsymbol{z}_{1:N}$. The generative story with random context variables $\boldsymbol{z}_{1:N}$ is detailed in algorithm 1.

For a LSTM-based encoder-decoder model [6], we apply the attention mechanism [5] over the random context variables $\{\boldsymbol{z}_i\}_{i=1}^N$ to construct $\boldsymbol{c}_t$ for the decoder. At each decoding time step t, the decoder computes the attention vector $\boldsymbol{c}_t$ and decoder hidden state $\boldsymbol{s}_t$ as follows:

$$\alpha_{ti} = \frac{\exp\left(a(\boldsymbol{s}_{t-1}, \boldsymbol{z}_i)\right)}{\sum_{j=1}^{N} \exp\left(a(\boldsymbol{s}_{t-1}, \boldsymbol{z}_j)\right)} \tag{3.9}$$

$$\boldsymbol{c}_t = \sum_{i=1}^{N} \alpha_{ti} \cdot \boldsymbol{z}_i \tag{3.10}$$

$$\boldsymbol{s}_t = f_{dec}(\boldsymbol{s}_{t-1}, \boldsymbol{y}_{t-1}, \boldsymbol{c}_t) \tag{3.11}$$

where $a(\boldsymbol{s}_{t-1}, \boldsymbol{z}_i) = \boldsymbol{v}_a^\top \tanh(\mathbf{W}_a \boldsymbol{s}_{t-1} + \mathbf{U}_a \boldsymbol{z}_i)$, $\mathbf{W}_a$, $\mathbf{U}_a$ and $\boldsymbol{v}_a^\top$ are parameter matrices. Finally, the decoder outputs a word distribution based on the context representations and previous decoded words at each decoding time step t:

$$p(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \boldsymbol{z}_{1:N}) = \text{softmax}(\mathbf{W}_b \cdot \boldsymbol{s}_t) \tag{3.12}$$

where $\mathbf{W}_b$ is a parameter matrix.

For a Transformer-based encoder-decoder model [97], we take the output of the last layer in the encoder as $\boldsymbol{h}_{1:N}$, and feed them into $g(\boldsymbol{h})$ to get random context variables $\boldsymbol{z}_{1:N}$. For the decoder, the inputs $\mathbf{K}$ and $\mathbf{V}$ are the combination of $\boldsymbol{h}_{1:N}$ and $\boldsymbol{z}_{1:N}$:

$$\mathbf{K}^l = \mathbf{V}^l = \mathbf{W}_z[\boldsymbol{z}_{1:N}; \boldsymbol{h}_{1:N}] \tag{3.13}$$

$$\mathbf{A} = \text{MultiHead}(\mathbf{S}^{l-1}, \mathbf{K}^l, \mathbf{V}^l) \tag{3.14}$$

$$\mathbf{B} = \text{LayerNorm}(\mathbf{A} + \mathbf{S}^{l-1}) \tag{3.15}$$

$$\mathbf{S}^l = \text{LayerNorm}(\text{FFN}(\mathbf{B}) + \mathbf{B}) \tag{3.16}$$

where $\mathbf{S}^l = \{\boldsymbol{s}_t\}_{t=1}^T$ is the last layer of decoder hidden states, and MultiHead$(\cdot)$, LayerNorm$(\cdot)$ and FFN$(\cdot)$ follow the standard implementation in [116]. The word distribution $p(\boldsymbol{y}_t \mid \boldsymbol{y}_{t-1}, \boldsymbol{z}_{1:N})$ at each decoding time step t is computed the same way as in Equation 3.12.

3.1.3 Efficient Variational Inference

With the observed deterministic hidden states $\boldsymbol{h}_{1:N}$, we estimate the GP posterior to make the prediction of context variables $\boldsymbol{z}_{1:N}$ more accurate. Although the posterior estimation of Gaussian processes can be written in a closed form theoretically, the challenge in this work comes from learning with other parts of the model, such as the deterministic encoder producing $\boldsymbol{h}_{1:N}$ and the decoder generating $\boldsymbol{y}_{1:T}$. To simplify the inference procedure, we will focus on inferring the samples of the GP posterior regarding the hidden states *only* as

$p(g \mid \boldsymbol{h}_{1:N})$, which essentially is a Gaussian distribution with non-isotropic covariance. In this work, we apply variational inference to approximate the GP posterior $p(g \mid \boldsymbol{h}_{1:N})$ and learn other model parameters *jointly* with maximum likelihood estimation.

For notation simplicity, we let $\boldsymbol{h} = f_{enc}(\boldsymbol{x}_{1:N})$, $\boldsymbol{z} = \{\boldsymbol{z}_i\}_{i=1}^N$, $\boldsymbol{y} = \{\boldsymbol{y}_i\}_{i=1}^T$ in this section. With a sampled random function $g(\boldsymbol{h})$ from the GP prior as described in line 9 of algorithm 1, we will get the joint prior distribution $p(\boldsymbol{z} \mid \boldsymbol{h})$ according to Equation 3.6. Then we approximate the true posterior $p(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ with the variational posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ by maximizing the evidence lower bound of the marginal log-likelihood (ELBo):

$$\log p(\boldsymbol{y} \mid \boldsymbol{h}) \geq \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p(\boldsymbol{y} \mid \boldsymbol{z})] - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \| p(\boldsymbol{z} \mid \boldsymbol{h})] \tag{3.17}$$

where $\boldsymbol{\phi}$ denotes the variational parameters.

Derivations of ELBo. We follow conditional variational autoencoders [118] and assume that for given observation $\boldsymbol{h}$, $\boldsymbol{z}$ is drawn from the prior distribution $p(\boldsymbol{z} \mid \boldsymbol{h})$, and the output $\boldsymbol{y}$ is generated from the distribution $p(\boldsymbol{y} \mid \boldsymbol{h}, \boldsymbol{z})$. We learn the variational posterior by minimizing $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \| p(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}))$, which is equivalent to maximizing the evidence lower bound of the marginal log-likelihood (ELBo):

$$\begin{aligned}
&\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \| p(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})) \\
&= \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})}{p(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})} d\boldsymbol{\phi} \\
&= \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) p(\boldsymbol{y} \mid \boldsymbol{h}) p(\boldsymbol{h})}{p(\boldsymbol{z}, \boldsymbol{h}, \boldsymbol{y})} d\boldsymbol{\phi} \\
&= \log p(\boldsymbol{y} \mid \boldsymbol{h}) + \int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) p(\boldsymbol{h})}{p(\boldsymbol{y} \mid \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z} \mid \boldsymbol{h})} d\boldsymbol{\phi}
\end{aligned} \tag{3.18}$$

Since $\mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \| p(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})) \geq 0$, we have:

$$\begin{aligned}
\log p(\boldsymbol{y} \mid \boldsymbol{h}) &\geq -\int q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) p(\boldsymbol{h})}{p(\boldsymbol{y} \mid \boldsymbol{h}, \boldsymbol{z}) p(\boldsymbol{z} \mid \boldsymbol{h})} d\boldsymbol{\phi} \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{h}) + \log p(\boldsymbol{z} \mid \boldsymbol{x}) - q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{h})] - \mathbb{E}_{q_{\boldsymbol{\phi}}}[\frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})}{\log p(\boldsymbol{z} \mid \boldsymbol{h})}] \\
&= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log p(\boldsymbol{y} \mid \boldsymbol{z}, \boldsymbol{h})] - \mathrm{KL}[q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \| p(\boldsymbol{z} \mid \boldsymbol{h})]
\end{aligned} \tag{3.19}$$

where $\boldsymbol{\phi}$ are parameters for the variational inference networks.

Simplifying Inference. During generation, we propose a two-step approximation to simplify $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$. First, to maintain the generative property when using the variational distribution, we propose an approximation of the variational distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y}) \approx q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h})$. In this case, the random context vector $\boldsymbol{z}$ will only depend on $\boldsymbol{h}$ during inference. Second, we apply the mean-field amortized variational approximation [119] to approximate the parameters of $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h})$:

$$\begin{aligned}
q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{h}) &= \prod_{i=1}^{N} q_{\boldsymbol{\phi}}(\boldsymbol{z}_i \mid \boldsymbol{h}_i) \\
&= \prod_{i=1}^{N} \mathcal{N}(f_{\mu}(\boldsymbol{h}_i), f_{\sigma^2}(\boldsymbol{h}_i))
\end{aligned} \tag{3.20}$$

where $f_{\mu}(\cdot)$ and $f_{\sigma^2}(\cdot)$ are the mean and covariance in the amortized variational inference network. In this work, we use two simple feed-forward neural networks f_{μ} and f_{σ^2}. The implementation details are included in subsection 3.1.4.

3.1.4 Experimental Setup

We evaluate our method on two text generation tasks that require rich contextual representations: paraphrase generation and text style transfer. We compared our method with previous works on diverse text generation in terms of quality and diversity. Empirical experiment results show that our method is able to: (1) adapt to different encoder-decoder architectures,

such as the pointer-generator network (PG) [6] and the text-to-text transfer Transformer (T5) [97]; (2) generate higher quality texts compared with deterministic encoder-decoder models [6, 97] while also enabling diverse generation by conditioning on random context variables.

Evaluation Metrics. For quality evaluation, we use two commonly used automatic metrics in text generation: *METEOR* [120] and *BLEU* with up to bi-grams [121], which tell us how well the generated outputs match the reference sentences.

For diversity evaluation, we aim at examining how well different latent context variables $\boldsymbol{z}_{1:N}$ from $q_{\phi}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ can make the decoder generate diverse outputs. We use self-BLEU with up to bi-grams (*self-BLEU*) [122] to measure the mutual bi-gram overlap between the set of outputs per source sentence, lower self-BLEU indicates less bi-gram overlap between generated outputs. In addition, we use diverse 4-gram (*Div-4*) [123] to measure the ratio of distinct 4-grams in the set of outputs per source sentence, higher diverse 4-gram shows more unique 4-grams between generated outputs. Finally, we use uniqueness (*Uni.*) [123] to measure the ratio of unique generated sentences in the set of outputs per source sentence, higher uniqueness suggests different context variables $\boldsymbol{z}_{1:N}$ lead to very different output sentences.

Competitive Baselines. We compare our method with competitive deterministic encoder-decoder models and variational encoder-decoder models as follows.

1. *PG* [6]: the pointer-generator network, which is a strong deterministic LSTM-based encoder-decoder baseline. We refer to their model as PG.

2. *T5* [97]: the text-to-text transfer Transformer, which is a strong deterministic Transformer-based encoder-decoder baseline. We refer to their model as T5.

3. *Variation Attention* [99]: modeling the deterministic attention vectors as latent alignment variables to promote diverse text generation. We refer to their model as Variation Attention.

4. *Multi-Selectors* [101]: using a mixture of experts to sample different binary masks on the source texts for diverse content generation. We refer to their model as Multi-Selectors.

5. *T-CVAE* [100]: modeling deterministic encoder hidden states as latent context variables with a Transformer-based conditional variational autoencoder. We refer to their model as T-CVAE.

6. *PG/T5 + Normal prior*: PG or T5 with a normal prior $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$, which follows the conventional variational autoencoders [119, 113].

7. *PG/T5 + GP prior*: PG or T5 with our GP prior defined in Equation 3.7.

Model Configurations. The implementation details of PG, it is an LSTM-based encoder-decoder model with copying mechanism. The encoder is a 1-layer Bi-LSTM, and the decoder is a 1-layer uni-directional LSTM. We set the word embedding size to 300, and the hidden dimension for both encoder and decoder to 512. We let the encoder and decoder share the same vocabulary list and word embedding, and the vocabulary size is 20000. For the configuration of the posterior networks, both the mean and covariance networks are a single feed-forward neural network, and we set the dimension of the latent variable to 256.

For the implementation details of T5, we use the T5-base implementation from Huggingface [124] [3], and use their default model configuration. We load the pre-trained weights of T5-base, and fine-tune them on our target task datasets. For the configuration of the posterior networks, both the mean and covariance networks are a single feed-forward neural network, and we set the dimension of the latent variable to 512.

Training Configurations. For the training details of PG and T5, we do not apply KL annealing and the coefficient of the KL divergence is always 1. We use Adam optimizer [125] with a learning rate of 0.0001, and adopt early stopping if the validation loss does

[3] `https://huggingface.co/transformers/model_doc/t5.html`

not decrease after 10 epochs. For the hyper-parameters $\{v, r\}$ of the kernel function in Equation 3.8, we try a range of values where $v \in [0.01, 100]$ and $r \in [0.0001, 10]$, and do a grid search cross-validation on the validation set to select the best model. All experiments are independently conducted on a GPU server (RTX 2090 Ti) with 12GB Memory.

Generation Setups. For the decoding strategy, we use beam search with a beam size of 10. Note that our method is orthogonal to all diversity-promoting decoding strategies, such as top-k sampling [26] and nucleus sampling [107]. We choose beam search in order to make a fair comparison with other works which promote diversity on the encoder side.

For quality generation, we directly take the mean of $q_{\phi}(\boldsymbol{z}|\boldsymbol{h}, \boldsymbol{y})$, and generate one $\boldsymbol{y}_{1:T}$ based on the sampled context variables $\boldsymbol{z}_{1:N}$, since we want to examine how well the posterior network can encode contextual information and make the decoder generate high-quality texts.

For diverse generation, we sample different $\boldsymbol{z}_{1:N}$ (instead of directly taking the mean) from $q_{\phi}(\boldsymbol{z}|\boldsymbol{h}, \boldsymbol{y})$, and generate different $\boldsymbol{y}_{1:T}$ based on the sampled context variables $\boldsymbol{z}_{1:N}$, since we want to examine how well different latent context variables $\boldsymbol{z}_{1:N}$ from $q_{\phi}(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ can make the decoder generate diverse outputs. For experiment setups, we sample 10 different $\boldsymbol{z}_{1:N}$ and generate 10 different $\boldsymbol{y}_{1:T}$ correspondingly. We compute the diversity scores following the prior work [123]. To compute the self-BLEU and Div-4, we randomly sample 5 different $\boldsymbol{y}_{1:T}$ out of the 10 generated $\boldsymbol{y}_{1:T}$. To compute the Uni., we compute the unique number of sentences among the 10 generated $\boldsymbol{y}_{1:T}$.

In preliminary experiments, we found that sampling from the original variational distribution tends to make the decoder generate the same sentences. We hypothesize that $q_{\phi}(\boldsymbol{z}|\boldsymbol{h}, \boldsymbol{y})$ is a high-dimensional multivariate Gaussian, and sampling from a high-dimensional distribution is a fundamental challenging problem. Therefore, we applied simple heuristics to alleviate the sampling issue, where we scaled up the covariance matrix of the variational distribution by a numeric scalar. We find this simple heuristics can help the decoder generate

Methods	TWITTER URL		GYAFC (E&M)		GYAFC (F&R)	
	BLEU↑	METEOR↑	BLEU↑	METEOR↑	BLEU↑	METEOR↑
Seq2Seq baselines						
PG	0.291	0.471	0.683	0.817	0.717	0.845
T5	0.264	0.453	0.683	0.819	0.726	0.847
Related works						
Multi-Selectors	0.290	0.492	0.606	0.779	0.618	0.783
Variation Attention	0.294	**0.512**	0.632	0.804	0.675	0.833
T-CVAE	**0.339**	0.494	0.481	0.686	0.537	0.730
Our works						
PG + Normal prior	0.041	0.127	0.145	0.354	0.191	0.452
T5 + Normal prior	0.269	0.461	0.675	0.815	0.722	0.846
PG + GP prior	0.307	0.483	0.681	**0.828**	0.734	**0.849**
T5 + GP prior	0.281	0.474	**0.688**	0.815	**0.739**	0.847

Table 3.1: Model performance on text quality. To get the best performance of variational encoder-decoder models, we directly take the mean of $q_\phi(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ as the sampled context variables to generate the output texts. Results of the Multi-Selectors [101], Variation Attention [99] and T-CVAE [100] are collected based on the source code provided in the original paper.

more diverse sentences. For PG + Normal prior and PG + GP prior, we set the numeric scalar for the paraphrase generation task to 25, and for the style transfer task to 10. For T5 + Normal prior and T5 + GP prior, we set the numeric scalar for the paraphrase generation task to 7, and for the style transfer task to 4.

3.1.5 Result Analysis

Paraphrase Generation. We first evaluate the model's capability of generating paraphrases using the Twitter URL paraphrasing dataset [126]. In this task, we aim to compare the quality of generated texts between our method and other competitive baselines.

The Twitter URL paraphrasing dataset [126] contains both positive and negative examples of paraphrases. We filter out all negative examples from the 1-year 2,869,657 candidate pairs, and divided the remaining paraphrase pairs into 110K training pairs, 3K testing pairs, and 1K validation pairs.

As shown in Table 3.1, for the quality of generated texts, our method is able to well preserve the semantic information from source texts. For LSTM-based models, PG + GP

Methods	GYAFC (E&M)				GYAFC (F&R)			
	avg-BLEU↑	self-BLEU↓	Div-4↑	Uni.↑	avg-BLEU↑	self-BLEU↓	Div-4↑	Uni.↑
T-CVAE	0.481	0.986	0.221	0.130	0.537	0.990	0.220	0.126
T5 + Normal prior	0.419	0.522	0.524	0.791	0.347	0.415	0.484	0.845
T5 + GP prior	0.329	**0.395**	**0.727**	**0.898**	0.252	**0.295**	**0.748**	**0.910**

Table 3.2: Model performance on text diversity. Avg-BLEU measures the average quality of generated sentences compared with ground-truth references. Self-BLEU measures the token-level repetitiveness, and a lower self-BLEU indicates a higher token-level diversity. Div-4 measures the ratio of unique 4-grams, and a higher Div-4 means a higher token-level diversity. Uni. measures the ratio of unique generated sentences, and a higher Div-4 illustrates a higher sentence-level diversity.

prior generates better quality texts compared with both its deterministic baseline PG and other variational baselines, e.g. Multi-Selectors and Variation Attention. Note that PG + Normal prior experiences the posterior collapse problem [113, 114, 115] during training, which causes the context variables to preserve little semantic information in the source text and the model generating random tokens during inference. For Transformer-based models, T-CVAE generates better quality texts than T5, T5 + Normal prior, and T5 + GP prior. But T-CVAE lowercase all input and output tokens while the other models keep both lowercase and capital tokens, this text preprocessing step may bring an unfairly better performance of T-CVAE in quality scores. Note that the posterior collapse problem does not happen in T5 + Normal prior, and T5 + GP prior still outperforms T5 + Normal prior, which shows the advantage of GP priors in introducing context-aware variations.

Text Style Transfer. We evaluate our model's capability of generating stylistic texts using Grammarly's Yahoo Answers Formality Corpus (GYAFC) [91]. In this task, we first compare the quality of generated texts between our method and other competitive baselines, and then we test the diversity of generated texts between our GP prior and conditional variational autoencoders.

The GYAFC dataset covers two sub-domains: Entertainment & Music (E&M), which has 52,593 training pairs, 2,877 validation pairs, 1,416 testing pairs; and Family & Relationships

Informal Sentence: Your age... Dude that one is old.
Formal References: ["Your age. That one is old."; "You are quite old."; "Wow, that one is very old."; "How old are you?"]

T-CVAE	**T5 + Normal Prior**	**T5 + GP Prior**
you are older .	Your age is old.	Your age, that one is old.
you are older .	Your age, that one is old.	Your age, that one is old.
you are older .	You're your age. No, that one is old.	Your age, and that one is old.
you are older .	You are your age. Due, that one is old.	You are your age, and that one is old.
you are older .	Your age, that one is old.....	Your age, you are not the one who is old.
you are older .	Your age and i.....	You're a fool, that one is old.
you are older .	Your age is arbitrary to you.....	Your age doesn't matter, that one is old.
you are older .	You are a very, jo, you are a very, jo	Your age is due to the fact is very old.
you are older .	You are a ant / a ante / a ante / a ante /	Regardless of your age, he is a young person.
you are older .	Your count count count count count count	I am not sure your age, but that one is old.

Table 3.3: Sample outputs conditioned on different $\boldsymbol{z}$ sampled from $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ on GYAFC (E&M) test set.

(F&R), which has 51,967 training pairs, 2,788 validation pairs, 1,332 testing pairs.

For the quality of generated texts, GP prior makes the model more robust to generate accurate texts. As shown in Table 3.1, for LSTM-based models, PG + GP prior generates the most accurate texts compared with PG, Multi-Selectors, and Variation Attention. Note that PG + Normal prior also experiences the posterior collapse problem in GYAFC datasets, resulting in very low-quality scores on the test set. For Transformer-based models, T5 + GP prior achieves better performance than T5, T5 + Normal, and T-CVAE, which shows the superiority of GP priors in encoding contextual information.

For the diversity of generated texts, imposing context-aware variations into encoder hidden states is beneficial for generating diverse outputs. As demonstrated in Table 3.2, for transformer-based models, T5 + GP prior gives the best diversity performance in both token-level and sentence-level compared with T5 + Normal prior and T-CVAE. The model performance on the style transfer task verifies the capability of our GP prior to promoting generation diversity. Table 3.3 shows some diverse generation outputs of Transformer-based variational encoder-decoder models. However, we also notice that increasing diversity will inevitably cause degradation in quality, because $\boldsymbol{z}_{1:N}$ are i.i.d. sampled from a high-dimensional multivariate Gaussian $q_\phi(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$. As discussed in previous work [127], multivariate sampling in high-dimensional settings can become computationally demanding.

Computation Complexity Analysis. Our GP priors require more computation during training, where the major computation comes from calculating the full covariance matrix of context variables of the GP prior. However, during inference, we approximate the GP posterior with a variational posterior $q_\phi(\boldsymbol{z} \mid \boldsymbol{h}, \boldsymbol{y})$ and conducts i.i.d. sampling, which saves time for multivariate sampling and has the same computation complexity with other conditional variational autoencoder baselines at testing time.

3.2 Controlling Faithfulness via Reinforcement Learning

Recent large language models (LLMs) have enabled conversational information-seeking systems to exhibit remarkable proficiency in producing fluent and coherent responses [128, 1, 2, 3]. However, LLMs sometimes fail to generate faithful and accurate responses supported by verified knowledge texts, but instead generate content either not from verified knowledge texts or from an incorrect span of irrelevant texts. This undesirable model behavior stems from three distinct sources: the bias inherent in LLMs, the irrelevant information in knowledge texts, and the characteristics of the employed learning algorithms. Firstly, LLMs are likely to generate texts that are most frequently seen during pretraining [129, 130], and may either disregard the knowledge texts or generate additional information not provided in the knowledge texts, as illustrated in the first example of Table 3.4. This problem is also described as hallucination in other works [131, 132, 133]. Secondly, the knowledge texts often contain irrelevant information that diverts the model's attention away from the relevant text span [134], as demonstrated in the second example of Table 3.4. Lastly, the model is trained to maximize the likelihood of ground-truth responses, which is easy to suffer from exposure bias [135, 136], and sometimes fails to be faithful to the knowledge texts at test time [137, 138].

Aligning with the problem source analysis, we classify the related prior works into three categories. The first category of works proposes to use additional data annotations on the knowledge texts to filter out irrelevant information, such as learning a knowledge selection

History	User: Do you know anything about space elevators?
Knowledge Text	The design would permit vehicles to travel along the cable from a planetary surface, such as the Earth's, directly into space or orbit, without the use of large rockets.
Reference Answer	System: Yeah, they would allow vehicles to move along a cable into space, instead of on a rocket.
LLM Output	System: Yes, a space elevator is a proposed type of transportation system. It consists of a cable anchored to the surface of a planet, with the other end in space. *The cable would be held up by the centrifugal force generated by its orbital motion around. (unfaithful)*
History	User: How do i apply for Medicare?
Knowledge Text	Benefits Planner: Retirement — Applying for Medicare Only — Social Security Administration // Applying for Medicare Only. If you are within three months of age 65 or older and not ready to start your monthly Social Security benefits yet, you can use our online retirement application to sign up just for Medicare and wait to apply for your retirement or spouse's benefits later. Before you decide, you need to be sure that you understand how waiting until later will affect: the lifetime benefits we can pay on your account and your health insurance coverage.
Reference Answer	System: You can use our online retirement application to sign up just for Medicare and wait to apply for your retirement or spouse's benefits.
LLM Output	System: You can use the Social Security Administration's online retirement application to sign up for Medicare only. Be aware that if you wait to apply for retirement or spouse benefits later, it may affect the benefits you receive and your health insurance coverage. (irrelevant)

Table 3.4: Two example responses generated by an LLM (text-davinci-003) in knowledge-grounded information-seeking conversations. The green text highlights the ground-truth knowledge span. The *blue text* is the unfaithful model-generated response caused by the bias from the LLM. The red text is the irrelevant model-generated response misled by the text that contains redundant information.

module [139, 140, 141, 142], or leveraging cross-attention scores to identify important texts [143, 144]. While directly learning from additional data annotations can improve the accuracy of locating the intended text spans, high-quality annotated data is often expensive to collect. The second group of works modifies the model architecture [145] or refines the decoding process [146] to force the model generating responses more faithful to the knowledge texts. This kind of approach is typically model-dependent, which may not generalize well to different pre-trained LLMs. The third line of works switches to alternative learning algorithms to explore the optimal dialogue generation policy, such as unlikelihood training [147], imitation learning [148] and reinforcement learning [149, 14, 37]. However, previous learning algorithms focus on optimizing the coherence and fluency of generated responses, and it is still less studied in improving the faithfulness and accuracy of generated

responses in knowledge-grounded conversations.

In this work, we find that a specific segment of the knowledge text, i.e. the ground-truth knowledge span, shares a high degree of n-gram overlap with the reference answer, as illustrated in Figure 3.4. This observation leads us to posit that increasing the overlap between the model-generated outputs and the reference answers, while simultaneously penalizing the inclusion of irrelevant content in these outputs, could significantly enhance both the faithfulness and coherence of the generated content. Therefore, we apply reinforcement learning (RL) algorithms to learn faithful, accurate, and coherent dialogue generation policy. On one hand, fine-tuning LLMs with RL on the downstream datasets can help alleviate the bias obtained during pre-training; on the other hand, an appropriate reward function can guide LLMs to generate responses that align with the relevant knowledge text.

We investigate a set of reward functions that could potentially improve the faithfulness and accuracy of the generated outputs, including SacreBLEU [150], Knowledge-F1 [151], BERTScore [152], and a coherence discriminator. The SacreBLEU reward measures the geometric mean of the modified n-gram precision between generated outputs and reference answers, which aims at matching model outputs with reference answers. The coherence discriminator is trained to distinguish whether the output is coherent with the conversation history. The Knowledge-F1 reward and the BERTScore reward measure the similarity between generated outputs and knowledge texts, which aims at matching model outputs with knowledge texts. We also linearly combine some reward functions together to see if there is a further performance improvement. Finally, empirical experiments on two information-seeking conversation

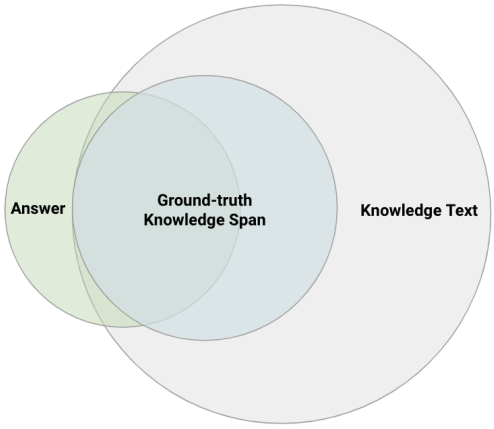


Figure 3.4: The relationship between the knowledge text, the ground-truth knowledge span, and the reference answer in knowledge-grounded dialogue generation tasks.

benchmark datasets suggest that the SacreBLEU reward is the most effective in improving both faithfulness and coherence of generated outputs. This is because the SacreBLEU reward computes the modified n-gram precision that penalizes the irrelevant contents generated by the model. As demonstrated by the relationship between knowledge text and reference answers in Figure 3.4, encouraging model outputs to align closely with reference answers suggests that such outputs can inherently adhere to the ground-truth knowledge span. This alignment occurs despite the model lacking direct access to this specific knowledge during the fine-tuning process.

We summarize the contributions of this work as follows: (1) identifying the relationship between knowledge texts and reference answers in knowledge-grounded dialogue generation tasks; (2) applying reinforcement learning with a simple reward function to improve both faithfulness and accuracy of generated outputs; (3) conducting empirical experiments to demonstrate the effectiveness of our method compared with strong supervised fine-tuning baselines.

3.2.1 Learning Faithful and Accurate Generation Policy with RL

Problem Definition. Given the knowledge text K_n and the conversation history $X = (\boldsymbol{u}_0, \cdots, \boldsymbol{u}_{n-1})$, the task is to generate a response $\boldsymbol{u}_n$ that is faithful to K_n and matches the reference answer $\boldsymbol{y}_n$. Following [153, 37], we formulate the response generation $\boldsymbol{u}_n = (a_0, \cdots, a_T)$ as a Markov Decision Process (MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. $\mathcal{S}$ is a finite set of states, where the initial state $s_0 \in \mathcal{S}$ is a concatenation of input conversation history X and knowledge text K_n. $\mathcal{A}$ is a finite set of actions, where an action $a_t \in \mathcal{A}$ is a token from our vocabulary $\mathcal{V}$. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is a transition function that determines the next state s_{t+1} given the current state action pair (s_t, a_t). $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function that returns a real number given the current state action pair (s_t, a_t). $\gamma \in [0, 1]$ is a discount factor. Each episode in the MDP begins by sampling a datapoint $(\boldsymbol{u}_0, \cdots, \boldsymbol{u}_{n-1}, \boldsymbol{y}_n, K_n)$ from the dataset, and ends when the current time step t exceeds the horizon T or an end of

sentence token is generated.

Proximal Policy Optimization (PPO). The policy $\pi_\theta : \mathcal{S} \rightarrow \mathcal{A}$ is a function that selects an action in a given state in order to maximize the long-term discounted rewards over a trajectory $\mathbb{E}_\pi[\sum_{t=0}^{T} \gamma^t R_t]$. In this work, we initialize the policy π_θ with a pre-trained language model π_0. We learn the policy using the Proximal Policy Optimization (PPO) algorithm [13], which is an effective actor-critic algorithm in many text generation tasks [14, 37]. The advantage is approximated using Generalized Advantage Estimation [154]: $A_t = \sum_{\tau=t}^{T} (\gamma\lambda)^{\tau-t}(R_\tau + \gamma V_\phi(s_{\tau+1}) - V_\phi(s_\tau))$, where λ is a hyper-parameter, γ is the discount factor, R_t is the reward assigned to a_t, and $V_\phi(s_t)$ is the value of state s_t given by the value network V_ϕ. Note that the reward R_t is regularized using a token-level KL penalty [155], in order to prevent the updated policy π_θ deviating too far from the pre-trained language model π_0:

$$R_t = R_t - \beta \text{KL}(\pi_\theta(a_t|s_t)||\pi_0(a_t|s_t)) \tag{3.21}$$

where β is a dynamically adapted coefficient [156]. During training, the policy network π_θ and the value network V_ϕ will be optimized jointly. [4]

Reward Functions Selection. In this section, we discuss the selection of different reward functions R_t. Note that R_t is only assigned to the final token in the generated response, and will be regularized with the token-level KL penalty the same way as in Equation 3.21.

Accuracy Reward Functions: The goal of the accuracy reward is to match generated response $\boldsymbol{u}_n$ and reference answer $\boldsymbol{y}_n$. We select the SacreBLEU as one reward function which calculates the n-gram precision between generated responses and reference answers, in order to discourage the model from generating irrelevant content. In addition, we train a coherence discriminator as another reward function to distinguish whether the generated response is coherent with the given conversation history. We provide implementation details

[4] In this work, we use the RL4LMs library to learn the response generation policy, so please refer to [37] for more algorithm implementation details.

of the coherence discriminator in paragraph 3.2.2. Both reward functions will output a scalar value between 0 and 1, where 1 means a perfect match between the generated response and the reference answer.

Faithfulness Reward Functions: The goal of the faithfulness reward is to ensure that the generated responses $\boldsymbol{u}_n$ accurately reflect the content of the knowledge text K_n, thereby preventing the generation of irrelevant information not present in the knowledge text. We select the Knowledge-F1 as one reward function which measures the surface-level token overlap between generated responses and knowledge texts, in order to encourage the model to generate content from the knowledge texts. Besides, we select BERTScore as another reward function which computes the sentence embedding similarity between generated responses and knowledge texts, with the goal of encouraging the model to generate information semantically relevant to the knowledge text. Both reward functions will output a scalar value between 0 and 1, where 1 means a complete copy of the knowledge text to the generated response.

Combined Reward Functions: We hypothesize that a combined reward function may contribute to the learning of a better policy which improves both accuracy and faithfulness. We propose to linearly blend the accuracy reward and faithfulness reward. Formally, the combined reward function is defined as:

$$R_t = \alpha R_t^{\text{acc}}(\boldsymbol{u}, \boldsymbol{y}) + (1 - \alpha) R_t^{\text{faith}}(\boldsymbol{u}, K) \tag{3.22}$$

where R_t^{acc} is the accuracy reward measuring the similarity between the generated response $\boldsymbol{u}$ and the ground-truth reference $\boldsymbol{y}$, R_t^{faith} is the faithfulness reward evaluating the factual consistency between the generated response $\boldsymbol{u}$ and the knowledge text K, and $\alpha \in [0, 1]$ is a coefficient used to balance the accuracy and faithfulness of generated responses. In this work, we choose SacreBLEU as R_t^{acc} and Knowledge-F1 as R_t^{faith}, as they are recognized as effective rewards in our preliminary experiments.

In-Context Example
Instruction: The following is a conversation with an AI assistant. The assistant is providing an answer based on a knowledge passage. **Human**: I recently discovered rap music and I'm so intrigued by it! Do you listen to rap music? **Knowledge Text**: Rapping is a musical form of vocal delivery that incorporates "rhyme, rhythmic speech, and street vernacular", which is performed or chanted in a variety of ways. **AI**: No, I'm a bot and can't hear. I know that it's a form of music that involves chanting and rhythmic speech.
Input
Human: Chevrolet is my all time favorite car brand, have you heard of it? **Knowledge Text**: The Chevrolet Chevy II/Nova was a small automobile manufactured by Chevrolet, and produced in five generations for the 1962 through 1979, and 1985 through 1988 model years.
Output
Model Output 1: Yes, I'm familiar with Chevrolet. It's a car brand *that has been around since the early 1900s and* has produced popular models like the Chevy II/Nova, *Corvette, Camaro, and Impala*.
Model Output 2: Yes, I heard of Chevrolet. It's a popular car brand that has been manufacturing cars for several decades. Chevy II/Nova was produced in five generations between 1962 and 1988.

Table 3.5: An example from the validation set of FaithDial for human expert to do pair-wise comparison between the outputs of GPT-3.5-turbo and t5-base. The *blue text* is the unfaithful model-generated response caused by the bias from the LLM.

The learning of the coefficient α is non-trivial, and we propose to learn it from human pair-wise comparison data. Since the primary requirement of the reward function is to differentiate faithful and accurate responses, leveraging human comparisons can provide valuable insights for determining an appropriate value for α. Specifically, we leverage one state-of-the-art model GPT-3.5-turbo[5] and one baseline model T5-base [157] fine-tuned on the target dataset, to generate 50 responses respectively. Then we shuffle the presentation order and ask an NLP expert to do pair-wise comparisons between the two model outputs. An illustration example is provided in Table 3.5. Next, we compute the reward using Equation 3.22 for both models' outputs, and align our reward comparison results with the human pair-wise comparison results. The alignment is done by iterating values of α and finding the optimal value that maximizes the Pearson correlation coefficient [158] between the human pair-wise comparison results and our reward pair-wise comparison results. By learning α from the human preference data, we can effectively calibrate the balance between faithfulness and accuracy in the generated responses.

[5] `https://platform.openai.com/docs/api-reference`

Algorithm 2: Learning α from human pair-wise comparisons

Input: Models' output pairs $\{(\boldsymbol{u}_n^1, \boldsymbol{u}_n^2\}_{n=1}^N$. Human pair-wise comparisons on models' outputs $\{\hat{p}_n\}_{n=1}^N$.

Output: The optimal blending coefficient α

1: **for** $\alpha = 0.00, \ldots, 1.00$ **do**
2: Compute our reward on two models' outputs using Equation 3.22: $\{(r_n^1, r_n^2)\}_{n=1}^N$.
3: Get the pair-wise comparison of our reward $p_n^\alpha = \text{argmax}(r_n^1, r_n^2)$, for $n = 1, \ldots, N$
4: Compute the Pearson correlation coefficient r between $\{\hat{p}_n\}_{n=1}^N$ and $\{p_n^\alpha\}_{n=1}^N$
5: **end for**
6: Save the optimal α which achieves the highest Pearson correlation coefficient

3.2.2 Experimental Setups

Benchmark Datasets. We choose two information-seeking conversation datasets as our benchmarks: MultiDoc2Dial [140] and FaithDial [133]. Both datasets contain two participants in each conversation: a user (or seeker) who initiates the conversation with a question, and a system (or wizard) who answers the user's question by referring to a piece of knowledge text. Each conversation contains several turns and probably topic shifts. The datasets statistics are demonstrated in Table 3.6.

Dataset	Training	Validation	Test	Avg. Length of Knowledge Text
MultiDoc2Dial	21,453	4,201	4,094	106
FaithDial	18,357	3,417	3,539	27

Table 3.6: Benchmark datasets statistics.

Competitive Methods. We compare with the following knowledge-grounded dialogue generation methods:

- **R3** [159]: a retriever-reranker-reader system that achieves state-of-the-art performance on MultiDoc2Dial. The system uses a bi-encoder DistilSPLADE [160] retriever to fetch top-100 relevant knowledge passages from the corpus, then applies a RoBERTa-based [161] cross-encoder to rerank the top-100 knowledge passages, finally passes the top-10 reranked knowledge passages to a T5-based FiD [162] to generate the response.
- **T5-CTRL** [133]: a controlled generation method that achieves state-of-the-art perfor-

mance on FaithDial. Following [163], it sets control feature tokens based on measures of entailment, lexical precision, and objective voice of the ground-truth response, to steer a T5-base model [12] generating responses faithful to the input knowledge texts.

- **T5-SFT**: a supervised fine-tuning baseline for both datasets. This method directly fine-tunes the T5-base model with maximum likelihood estimation on ground-truth responses in the full training set.

Implementation Details. We choose T5-base [157] as our backbone language model for all experiments. For the input knowledge text in MultiDoc2Dial, we use the ground-truth knowledge passage preprocessed by the official code[6], because we focus on the agent response generation subtask. For **T5-SFT** baseline, we fine-tune the model on the ground-truth response from the full training set. We fine-tune the model for 10 epochs using the AdamW optimizer [164] with a linear decaying learning rate starting from 1×10^{-5}. During inference, we use beam search with a beam size of 4 to generate the final response.

For the coherence discriminator in paragraph 3.2.1, we implement it with a RoBERTa-base model [161] and train it to discriminate the **T5-SFT** generated responses and the ground-truth responses. We train the reward model for 10 epochs using the AdamW optimizer with a constant learning rate of 1×10^{-6}. The reward model achieves 89% accuracy on the test set of MultiDoc2Dial and 96% accuracy on the test set of FaithDial.

For the combined coefficient α in paragraph 3.2.1, we manually collect 50 pair-wise human comparison data between **T5-SFT** and **GPT-3.5-turbo** on each dataset respectively, and perform grid search to find the optimal value of α that produces the highest Pearson correlation with the human judgments. We find the optimal value of α on MultiDoc2Dial is 0.51 with a Pearson correlation coefficient of 0.1653, and the optimal value of α on FaithDial is 0.05 with a Pearson correlation coefficient of 0.3381.

For all PPO experiments, the policy network and value network share the same base

[6] https://github.com/IBM/multidoc2dial/blob/main/scripts/run_data_preprocessing.sh

model initialized from **T5-SFT** but separate the last output layer. The output layer of the value network is a linear network that maps the last hidden state to a scalar value. We update the parameters for 10,000 iterations using the AdamW optimizer [164] with a learning rate of 5×10^{-7}. The policy is evaluated on the full validation set every 100 iterations, and the final policy is the one that achieves the highest total scores in accuracy and faithfulness on the validation set. We use top-k ($k = 50$) sampling to generate trajectories during training, and use beam search with beam size of 4 to generate the final response during testing.

Evaluation Metrics. We follow the prior works [140, 133] to evaluate the accuracy and faithfulness of generated responses. Specifically, we use SacreBLEU [150], ROUGE-L [165], and METEOR [166] to evaluate the accuracy, which measures the similarity between generated responses and reference answers. For the faithfulness evaluation, we use the token-level F1 scores and the F1 measure of BERTScore [152], which computes the similarity between generated responses and ground-truth knowledge spans. Additionally, we use GPT-4 to evaluate the faithfulness of generated responses following [167], since they find GPT-4 evaluation results correlate best with human judgments in conversational QA tasks. The prompt template for faithfulness evaluation is provided in Table 3.7. Note that we only sample 100 test data for GPT-4 evaluation due to limited budgets.

System prompt: You are CompareGPT, a machine to verify the groundedness of predictions. Answer with only yes/no. You are given a question, the corresponding evidence and a prediction from a model. Compare the "Prediction" and the "Evidence" to determine whether all the information of the prediction in present in the evidence or can be inferred from the evidence. You must answer "no" if there are any specific details in the prediction that are not mentioned in the evidence or cannot be inferred from the evidence. **Question**: {Question} **Prediction**: {Model response} **Evidence**: {Knowledge text} **CompareGPT response**:

Table 3.7: The prompt template used for GPT-4 faithfulness evaluation.

	Accuracy (u, y)			Faithfulness (u, K_{span})		
Method	**BLEU**	**ROUGE-L**	**METEOR**	**F1**	**BERTScore**	**GPT4**
Baselines						
R3	31.10	41.40	-	-	-	-
T5-SFT	25.38	41.13	**38.86**	51.61	91.10	95.00
Accuracy Rewards						
T5-PPO-BLEU	**30.60**	**43.04**	35.19	**54.80**	**91.67**	**96.00**
T5-PPO-RoBERTa	30.51	42.56	32.67	47.05	90.66	91.00
Faithfulness Rewards						
T5-PPO-F1	22.82	38.36	37.83	51.19	90.99	95.00
T5-PPO-BERTScore	27.45	42.42	32.10	51.12	91.37	94.00
Combined Rewards						
T5-PPO-BLEU&F1	23.72	39.48	37.69	52.85	91.26	97.00

Table 3.8: Automatic evaluation results on the test set of MultiDoc2Dial. The results of **R3** are reprinted from the original paper [159]. The other results come from our implementation. Note that K_{span} is the ground-truth knowledge span, and is not presented during training.

3.2.3 Result Analysis

The experiments in this section are designed to answer the following research questions:

RQ1 Does there exist a trade-off between faithfulness and accuracy in the knowledge-grounded dialogue generation task?

RQ2 Which reward function is most effective in improving both faithfulness and accuracy?

RQ3 Can the combined reward function help to learn a better policy?

RQ1: Faithfulness v.s. Accuracy. We first investigate whether there exists a trade-off between faithfulness and accuracy in the knowledge-grounded dialogue generation task. The empirical results in Table 3.8 suggest that RL with BLEU reward can improve both faithfulness and accuracy. For the MultiDoc2Dial dataset, the lengthy grounding knowledge texts often contain redundant information, and the reference answer in the BLEU reward can help locate the relevant text span in a long knowledge text.

On the other hand, the empirical results in Table 3.9 show that increasing the faithfulness performance will lead to a decrease in accuracy performance. For the FaithDial dataset, the grounding knowledge texts only contain the precise information needed for the answer,

	Accuracy (u, y)			Faithfulness (u, K_{span})		
Method	**BLEU**	**ROUGE-L**	**METEOR**	**F1**	**BERTScore**	**GPT4**
Baselines						
T5-CTRL	13.75	38.57	35.49	70.91	94.42	62.00
T5-SFT	13.69	39.58	**37.71**	75.49	95.13	70.00
Accuracy Rewards						
T5-PPO-BLEU	**14.19**	**39.61**	35.48	76.90	95.45	71.00
T5-PPO-RoBERTa	11.87	36.75	28.98	52.05	92.06	66.00
Faithfulness Rewards						
T5-PPO-F1	12.47	36.64	33.77	96.46	98.95	72.00
T5-PPO-BERTScore	12.59	36.95	30.12	54.47	91.57	70.00
Combined Rewards						
T5-PPO-BLEU&F1	12.42	36.56	33.86	**97.12**	**99.04**	**75.00**

Table 3.9: Automatic evaluation results on the test set of FaithDial. The results of **T5-CTRL** come from the code and data released by [133]. The other results come from our implementation. Note that K_{span} is the ground-truth knowledge span, and is not presented during training.

and further RL fine-tuning makes the model directly copy from the knowledge text. This approach, while effective for faithfulness, inadvertently compromises the accuracy of the generated responses, leading to a performance trade-off.

RQ2: Most Effective Reward. Generally, we find RL with BLEU reward achieves the best overall performance compared with other competitive baselines in both datasets, as shown in Table 3.8 and Table 3.9. In Table 3.8, **T5-PPO-BLEU** achieves the best performance in both accuracy and faithfulness evaluation, indicating that BLEU reward has the advantage of locating the relevant information from a long knowledge document. **T5-PPO-RoBERTa** achieves improved performance in accuracy but degraded performance in faithfulness compared with **T5-SFT**, because its reward function is designed to only encourage the generated response matching the reference answer. **T5-PPO-F1** and **T5-PPO-BERTScore** could not improve the accuracy and faithfulness performances, probably because the faithfulness rewards encourage the generated response to match the entire knowledge text, thus distracting the model to attend to the irrelevant information in the knowledge text.

History	User: I'd like to ask more information on what to expect when I receive my new Empire Gold plates?
Knowledge Text	Information about Empire Gold plates // What to do with your old Empire Blue and White plates When you receive your new Empire Gold plates, do not return your old plate to the DMV. Recycle your old plates based on local guidelines. For a statewide list of local recycling coordinators, visit the website of the NYS Department of Environmental Conservation at http://www.dec.ny.gov/chemical/8511.html [6]. Before you recycle your old plates, use a permanent ink marker to cross out the plate number or otherwise deface the plates. For your own protection, separate the plates and put them in the recycle bin during different weeks.
Reference Answer	System: Do you want to know what to do when you receive your new Empire Gold plates?
T5-SFT	System: Do not return your old plate to the DMV. Recycle your old plates based on local guidelines. (incorrect span)
T5-PPO-BLEU	System: When you receive your new Empire Gold plates, do not return your old plate to the DMV.
T5-PPO-F1	System: Do not return your old plate to the DMV. Recycle your old plates based on local guidelines. For a statewide list of local recycling coordinators, visit the web site of the NYS Department of Environmental Conservation at http://www.dec.ny.gov/chemical/8511.html. (incorrect span)
T5-PPO-BLEU&F1	System: Do not return your old plate to the DMV. Recycle your old plates based on local guidelines. (incorrect span)

Table 3.10: Different model-generated responses sampled from the test set of MultiDoc2Dial. The green text highlights the ground-truth knowledge span. The red text is the incorrect model-generated response misled by the text that contains redundant information.

In Table 3.9, **T5-PPO-BLEU** achieves improved performance in both accuracy and faithfulness evaluation compared with **T5-SFT**. However, **T5-PPO-RoBERTa** achieves the worst performance even if its reward model has 96% test accuracy, which indicates the accuracy discriminator is not a robust and effective reward function. Because the FaithDial dataset has short grounding knowledge texts that contain the precise information needed for the answer, we find **T5-PPO-F1** improves the faithfulness scores by a large margin compared with **T5-SFT**. After manual inspections on some randomly sampled outputs, we find the model just simply copies the entire knowledge text as its output. This explains why **T5-PPO-F1** leads to degraded performance in accuracy evaluation. **T5-PPO-BERTScore** is not as effective as **T5-PPO-F1**, which indicates the sentence embedding similarity scores may not provide a strong reward signal for the model to learn a good policy.

RQ3: Combined Reward. We didn't find the combined reward significantly outperforms the BLEU reward in Table 3.8. Compared with **T5-BLEU**, **T5-PPO-BLEU&F1** in the MultiDoc2Dial dataset has decreased performance in both accuracy and faithfulness evaluation, suggesting that the Knowledge-F1 reward is not helpful in this dataset. Compared with **T5-F1**, **T5-PPO-BLEU&F1** in the MultiDoc2Dial dataset has increased performance in both accuracy and faithfulness evaluation, indicating that the BLEU reward is very effective in locating the ground-truth knowledge span.

On the other hand, we find the combined reward achieves the best faithfulness performance in Table 3.9. However, we find the model simply learns to copy from the knowledge text as the output, which is not a good policy and does not demonstrate the effectiveness of the combined reward. Future research may be conducted to find a more effective way to combine two types of rewards that could improve performance in both faithfulness and accuracy.

3.3 Summary

In this chapter, we design controllable text generation methods to steer LLMs to produce desired outputs.

In Section 3.1, we investigate the problem of generating high-quality texts for variational encoder-decoder models. We propose a novel stochastic function to introduce context-aware variations into encoder hidden states, which provides the decoder with more diverse contextual representations. To learn this stochastic function, we propose a GP prior to modeling the dependency between random context variables, and apply an efficient amortized variational inference method to approximate the GP posterior. Experimental results demonstrate that our method can learn a better contextual representation that leads to higher generation quality compared with deterministic encoder-decoder models and conditional variational autoencoders.

In Section 3.2, we investigate how to improve faithfulness and accuracy in knowledge-

grounded dialogue generation tasks. Firstly, we identify the relationship between the grounding knowledge text, the ground-truth knowledge span, and the reference answer. Then, we solve the problem by applying a reinforcement learning algorithm with a suitable reward function. The selected reward function can improve both the faithfulness and accuracy of generated outputs. Finally, we validate the effectiveness of our method in two information-seeking conversation datasets. The empirical experiment results show that our method can outperform other strong supervised learning baselines.

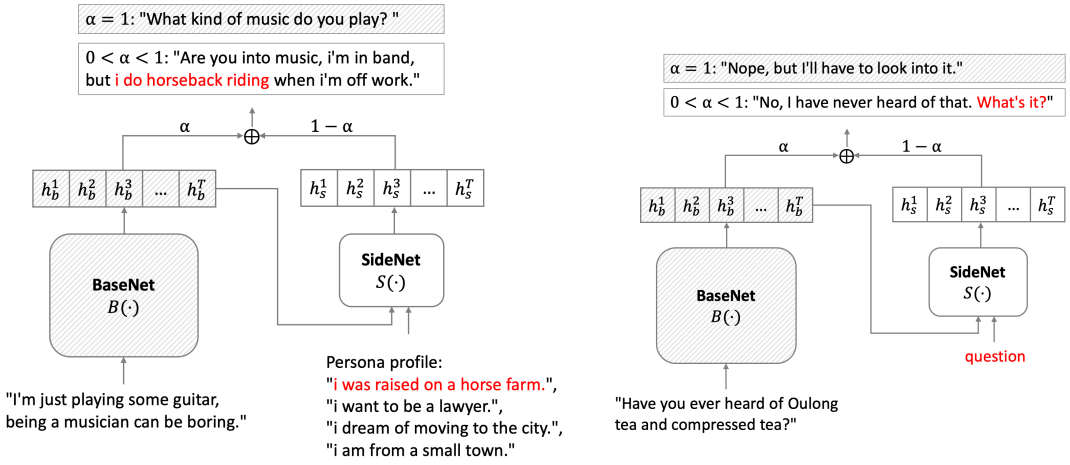
CHAPTER 4
DATA-EFFICIENT TEXT GENERATION

LLMs pre-trained on open-domain data may not generalize well in domain-specific tasks (e.g., task-oriented dialogue generation, medical report generation, legal document generation), however, it is often prohibitively expensive to collect large-scale, high-quality data in domain-specific NLG tasks. In this chapter, I introduce two data-efficient text generation methods to better adapt pre-trained LLMs to specific NLG tasks. In Section 4.1, I introduce an additive side network for the personalized dialogue generation task, which effectively encodes persona profile information to pre-trained LLMs with a few training data. In Section 4.2, I devise a novel self-training algorithm for the task-oriented dialogue generation task, which reduces the model prediction errors by fine-tuning with pseudo-labeled data generated by the model itself.

4.1 Personalized Dialogue Generation via Additive Side Networks

With the advance of Transformer-based pre-trained language models [168, 157, 11, 169], many dialogue systems [169, 170, 171] have shown promising performance in challenging open-domain conversations with humans. However, for controlled dialogue generation, prior works mainly focus on building LSTM-based class-conditional generative model on specific datasets with task-specific design on model architecture [172, 173, 174, 175] or policy learning strategy [33, 176, 177, 178]. In this work, we explore effective methods for controlled generation on Transformer-based dialogue systems, with the goal of adding controllability functionality into state-of-the-art Transformer-based dialogue systems with lower computation cost, less training data, and a more flexible control mechanism.

Prior works on controlled text generation for Transformer-based pre-trained language models can be categorized into two general approaches: (1) gradient-based methods and



(a) SideNet for Knowledge Document Control (b) SideNet for Semantic Label Control

Figure 4.1: General architecture of the SIDECONTROL framework.

(2) weighted-decoding methods. The gradient-based methods [34, 52, 53] propose a plug-and-play language model following $p(x|a) \propto p(a|x)p(x)$, which plugs an attribute model $p(a|x)$ with a pre-trained language model $p(x)$ to control generation. The gradients from $p(a|x)$ are used to guide the latent representations of pre-trained models encoding more control attribute information. The weighted-decoding methods [55, 179, 180, 36] modify the sampling weights with attribute functions in beam search at each decoding timestep to control generation. Essentially, the attribute functions are used to re-rank the original beam candidates generated by the pre-trained language models. The main idea of both gradient-based methods and weighted decoding methods is flexibility: users can design any attribute models or functions for different controlled generation tasks and apply the attribute model or function to any state-of-the-art pre-trained language models for generating high-quality texts.

However, weighted decoding methods [55, 179, 180, 36] are limited by the low-variance high-biased pre-trained language models, since they do not update the pre-trained language models. If the pre-trained model yields commonly observed words rather than target attribute words in the beam candidates list, it is difficult for the attribute functions to re-rank and find the target words during generation. Although gradient-based methods [34, 52, 53] do not

have this limitation since they update the latent representations of pre-trained models during inference, the gradient propagation at each decoding timestep involves heavy computation, which results in slow response speed to users. In addition, the controllability performance of gradient-based methods relies on the attribute model. If the attribute model gets overfitted on a small training set, the gradient from this attribute model will just lead to meaningless updates.

To build an effective and efficient controlled open-domain dialogue system, we propose the SIDECONTROL framework, which treats the pre-trained language model as a feature extractor and trains light-weight side networks to encode complementary information from control attributes. In addition, we introduce a novel control attribute loss to guide the side network during training. As shown in Figure 4.1, the final output representation is a mixture of a base representation from the pre-trained language model and a side representation from the side network. The mixture coefficient α is learned during training, and is used to balance the prior knowledge from the base network and the task-specific control attributes signals from the side network. From the encoding perspective, the SIDECONTROL framework not only can be applied to any pre-trained language models, but also supports diverse format attributes control (e.g. dialogue act, external knowledge document). From the decoding perspective, the SIDECONTROL framework has low computation cost, since it directly samples from its optimized class-conditional language model $p(x|a)$ without additionally updating latent representations during generation. From the sample-efficiency perspective, the SIDECONTROL framework achieves good performance with a few thousand training samples by leveraging the control loss.

We summarize the contributions of this work as follows:

1. Proposing a new controlled dialogue generation framework with novel control attributes losses to support different forms of attributes control (e.g. dialogue act, external knowledge document);

2. Conducting empirical experiments to show the sample-efficiency of the SIDECON-

TROL framework, which can achieve good performance with only $100 \sim 1000$ training samples;

3. Conducting empirical experiments to validate that the SIDECONTROL framework has better controllability, better text quality, and lower decoding cost compared to gradient-based methods and weighted-decoding methods.

4.1.1 The SideControl Framework

Firstly, we introduce the SIDECONTROL framework, which presents the general idea of using a small side network to coordinate the generation process based on large-scale pre-trained language models [169, 170, 171]. Then we provide two realizations of side networks for two types of control attributes: (1) external knowledge document, and (2) semantic label.

General Framework. Given a dialogue context which contains a fixed number of previous utterances $\mathbf{X} = \{\boldsymbol{x}_i\}_{i=1}^{N}$, where N is the total number of tokens in the given dialogue context, and a control attribute $\boldsymbol{a}$ which represents the desired controllable attributes, the goal is to build a model conditioned on $\mathbf{X}$ and $\boldsymbol{a}$ that can generate a response which best approximates the ground-truth human response $\mathbf{Y} = \{\boldsymbol{y}_t\}_{t=1}^{T}$:

$$\begin{aligned} p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{a}) &= \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{y}_{1:t-1}, \boldsymbol{x}_{1:N}, \boldsymbol{a}) \\ &= \prod_{t=1}^{T} p(\boldsymbol{y}_t \mid \boldsymbol{h}_t) \end{aligned} \tag{4.1}$$

where $\boldsymbol{h}_t$ is the last hidden state of the generative model at decoding timestep t.

The SIDECONTROL framework consists of a large base network $B(\cdot)$ providing rich feature representations and a small side network $S(\cdot)$ encoding control attribute(s), as illustrated in Figure 4.1. The base network $B(\cdot)$ can be any pre-trained language models [169, 170, 171]. Given dialogue context $\boldsymbol{x}_{1:N}$ as the input to the base network, we just take last hidden states $\{\boldsymbol{h}_b^t\}_{t=1}^{T}$ for the response $\{\boldsymbol{y}_t\}_{t=1}^{T}$ from the base network as our base

representations:

$$\boldsymbol{h}_b^1, \ldots, \boldsymbol{h}_b^T = B(\boldsymbol{x}_{1:N}) \tag{4.2}$$

The side network $S(\cdot)$ is a light-weight neural network, which encodes the control attribute $\boldsymbol{a}$ into base representations $\{\boldsymbol{h}_b^t\}_{t=1}^T$:

$$\boldsymbol{h}_s^1, \ldots, \boldsymbol{h}_s^T = S(\boldsymbol{a}, \boldsymbol{h}_b^1, \ldots, \boldsymbol{h}_b^T) \tag{4.3}$$

Finally, we keep the base representation $\boldsymbol{h}_b^t$ fixed, and add the side representation $\boldsymbol{h}_s^t$ upon it to obtain the final combined representation $\boldsymbol{h}_t$ for the current token $\boldsymbol{y}_t$:

$$\boldsymbol{h}_t = \alpha \cdot \boldsymbol{h}_b^t + (1 - \alpha) \cdot \boldsymbol{h}_s^t \tag{4.4}$$

$$p(\boldsymbol{y}_t \mid \boldsymbol{h}_t) = \text{softmax}(\mathbf{W}_{\text{vocab}} \boldsymbol{h}_t) \tag{4.5}$$

where $\mathbf{W}_{\text{vocab}}$ is learnable parameters, and the mixture coefficient α is also learned during training, which aims to encode both useful prior knowledge from pre-trained language models and important attribute information from target dataset for controlled generation.

The *main challenge* in this framework is to teach the side network $S(\cdot)$, such that it can provide complementary information of control signals via $\boldsymbol{h}_s^t$ during generation, since the pre-trained language models can already generate fluent responses. To address this challenge, we intentionally freeze the parameters of the base network $B(\cdot)$ when training the side network. Otherwise, it is essentially training a large neural network model even deeper than $B(\cdot)$. Second, we introduce the control attribute loss $\mathcal{L}_{control}$, which is designed to teach the side network explicitly encoding control signals to improve the controllability of the model. The final objective is a combination of class-conditional language modeling loss $\mathcal{L}_{cclm}$ and task-specific control attributes loss $\mathcal{L}_{control}$:

$$\mathcal{L} = \mathcal{L}_{cclm} + \lambda \cdot \mathcal{L}_{control} \tag{4.6}$$

where λ is a task-specific hyper-parameter, and $\mathcal{L}_{control}$ has different implementation when controlling different forms of attributes.

Knowledge Document Control. When having external knowledge documents as the control attributes, such as persona profile [181], Wikipedia articles [139], etc., the format of control attribute is sequences of tokens $\boldsymbol{a} = \{\boldsymbol{k}_i\}_{i=1}^{K}$, where K is the total number of tokens in the external knowledge document. In this case, we model the knowledge document representation with a single-layer bi-directional LSTM:

$$\boldsymbol{h}_k^1, ..., \boldsymbol{h}_k^K = \text{BiLSTM}(\boldsymbol{a}) \tag{4.7}$$

The side network is designed to align the controlled knowledge document representation $\{\boldsymbol{h}_k^i\}_{i=1}^{K}$ with the base representation $\boldsymbol{h}_b^t$ at each decoding timestep. We compute the cross-attention between $\{\boldsymbol{h}_k^i\}_{i=1}^{K}$ and $\boldsymbol{h}_b^t$ following [5]:

$$e_i^t = v^T \cdot \tanh\left(\mathbf{W}_k \boldsymbol{h}_k^i + \mathbf{W}_b \boldsymbol{h}_b^t + b_{kb}\right) \tag{4.8}$$

$$a_i^t = \text{softmax}(e_i^t) \tag{4.9}$$

$$\boldsymbol{c}_k^t = \sum_{i=1}^{K} a_i^t \cdot \boldsymbol{h}_k^i \tag{4.10}$$

where $\mathbf{W}_k \in \mathbb{R}^{D \times D}$, $\mathbf{W}_b \in \mathbb{R}^{D \times D}$ and $b_{kb} \in \mathbb{R}^D$ are learnable parameters. The attention a^t is a probability distribution over the controlled knowledge document that tells the decoder where to look when generating the next word, and the context vector $\boldsymbol{c}_k^t$ represents what has been read from the controlled knowledge document representation at decoding timestep t. The final side representation $\boldsymbol{h}_s^t$ incorporates the context vector $\boldsymbol{c}_k^t$ into the base representation $\boldsymbol{h}_b^t$:

$$\boldsymbol{h}_s^t = \tanh\left(\mathbf{W}_c[\boldsymbol{c}_k^t; \boldsymbol{h}_b^t] + b_c\right) \tag{4.11}$$

where we concatenate $\boldsymbol{c}_k^t$ and $\boldsymbol{h}_b^t$, and $\mathbf{W}_c \in \mathbb{R}^{2D \times D}$ and $b_c \in \mathbb{R}^D$ are learnable parameters.

Since the controlled knowledge document is different per utterance, we implement the mixture coefficient α based on the side representation $\boldsymbol{h}_s^t$ and base representation $\boldsymbol{h}_b^t$ at decoding timestep t:

$$\alpha_t = \sigma(\mathbf{W}_\alpha[\boldsymbol{h}_s^t; \boldsymbol{h}_b^t] + b_\alpha) \tag{4.12}$$

$$\boldsymbol{h}_t = \alpha_t \cdot \boldsymbol{h}_b^t + (1 - \alpha_t) \cdot \boldsymbol{h}_s^t \tag{4.13}$$

where we concatenate $\boldsymbol{h}_s^t$ and $\boldsymbol{h}_b^t$, and $\mathbf{W}_\alpha \in \mathbb{R}^{2D \times 1}$ and $b_\alpha \in \mathbb{R}$ are learnable parameters.

In order to encourage the decoder to generate more words from the knowledge document, we adopt the copy mechanism from [6] to formulate $\mathcal{L}_{cclm}$:

$$\beta = \sigma(\mathbf{W}_\beta[\boldsymbol{c}_k^t; \boldsymbol{h}_b^t] + b_\beta) \tag{4.14}$$

$$p(\boldsymbol{y}_t \mid \boldsymbol{h}_t) = \beta p(\boldsymbol{y}_t|\boldsymbol{h}_t) + (1 - \beta) \sum_{i=1}^{K} a_i^t \tag{4.15}$$

$$\mathcal{L}_{cclm} = -\sum_{t=1}^{T} \log p(\boldsymbol{y}_t^* \mid \boldsymbol{h}_t) \tag{4.16}$$

where we concatenate $\boldsymbol{c}_k^t$ and $\boldsymbol{h}_b^t$, and $\mathbf{W}_\beta \in \mathbb{R}^{2D \times 1}$ and $b_\beta \in \mathbb{R}$ are learnable parameters. $\boldsymbol{h}_t$ comes from Equation 4.13. $\boldsymbol{y}_t^*$ is the ground-truth word at decoding timestep t. $\sum_{i=1}^{K} a_i^t$ is the summation of attention distribution over the knowledge document at current decoding timestep t, which will assign a higher probability for attended knowledge document words in the final word probability distribution.

The control attribute loss for this task is used to encourage generating more non-repetitive words from the knowledge document. We adopt the coverage mechanism from [6] to formulate $\mathcal{L}_{control}$:

$$\mathcal{L}_{control} = \sum_{t=1}^{T} \sum_{i=1}^{K} \min(a_i^t, \sum_{t'=0}^{t-1} a_i^{t'}) \tag{4.17}$$

where $a_i^{t'}$ is the attention weight of knowledge document word $\boldsymbol{k}_i$ at previous decoding time step t'. $\mathcal{L}_{control}$ penalizes the overlap between current attention distribution and previous

attention distributions, which prevents the model from repeatedly attending to the same word in the knowledge document. For more details about the copy mechanism and coverage mechanism, please refer to the original paper [6].

Semantic Label Control. When having a semantic label as the control attribute, such as dialogue act [182], emotion [183], etc., we implement the side network as a simple feed-forward neural network:

$$\boldsymbol{h}_s^t = \tanh\left(\mathbf{W}_d[\mathbf{W}_a\boldsymbol{a}; \boldsymbol{h}_b^t] + b_d\right) \tag{4.18}$$

$$\boldsymbol{h}_t = \alpha \cdot \boldsymbol{h}_b^t + (1-\alpha) \cdot \boldsymbol{h}_s^t \tag{4.19}$$

$$\mathcal{L}_{cclm} = -\sum_{t=1}^{T} \log p(\boldsymbol{y}_t^* \mid \boldsymbol{h}_t) \tag{4.20}$$

where we concatenate $\mathbf{W}_a\boldsymbol{a}$ and $\boldsymbol{h}_b^t$, $\mathbf{W}_a \in \mathbb{R}^{1\times D}$ is an embedding matrix that maps the discrete label $\boldsymbol{a}$ to a continuous representation, $\mathbf{W}_d \in \mathbb{R}^{2D\times D}$ and $b_d \in \mathbb{R}^D$ are learnable parameters. The mixture coefficient $\alpha \in [0, 1]$ is a global parameter which is learned during training, in order to encode both useful prior knowledge from pre-trained language models and control signals from semantic labels. $\boldsymbol{y}_t^*$ is the ground-truth word at the decoding time step t.

The control attributes loss $\mathcal{L}_{control}$ for this task is used to modify the final latent representations so that the model can generate responses with the target control attribute. However, it is difficult to directly measure how much control attribute information has been encoded into the side representation. Therefore, we approximate it using an independent attribute classifier $p(\boldsymbol{a}|\boldsymbol{h}_{1:T})$. When training the side network, we keep the attribute classifier fixed and feed the side representations $\{\boldsymbol{h}_s^t\}_{t=1}^T$ into the classifier. The classifier will return a loss between the current side representation and the target control attribute $\boldsymbol{a}^*$, and optimizing

this loss will update the side representation $\boldsymbol{h}_s^t$ towards obtaining a higher $p(\boldsymbol{a}^*|\boldsymbol{h}_{1:T})$:

$$p(\boldsymbol{a} \mid \boldsymbol{h}_{1:T}) = \text{softmax}(\mathbf{W}_{\text{clf}} \frac{\sum_{t=1}^{T} \boldsymbol{h}_s^t}{T}) \tag{4.21}$$

$$\mathcal{L}_{control} = -\log p(\boldsymbol{a}^* \mid \boldsymbol{h}_{1:T}) \tag{4.22}$$

Note that $\mathbf{W}_{\text{clf}} \in \mathbb{R}^{D \times K}$ is independently learned on the same training set based on the base representation $\{\boldsymbol{h}_b^t\}_{t=1}^T$, but is fixed when we update the side network.

4.1.2 Experimental Setup

Evaluation Metrics. In this work, we focus on evaluating the controllability and text quality of different controlled generation methods. Additionally, we prefer to have lower decoding costs and better modularity in order to apply the proposed method to more possible applications. Therefore, we consider the following dimensions to evaluate the performance: controllability, text quality, decoding cost, and modularity.

Controllability: This is our main metric. It aims to evaluate whether the proposed method can successfully generate the target controlling attributes. For the semantic label control task, we use the *classification accuracy* computed by an independently trained BERT classifier [184]. We train an independent dialogue act classifier to evaluate whether the current generated response matches its conditioning dialogue act. The input to the evaluation dialogue act classifier is a single response, and the output is a prediction of one of the 4 dialogues in DailyDialog, i.e. *inform*, *questions*, *directives* and *commissive*.

We construct the training corpus following the standard splition of the original DailyDialog dataset, and obtain 87,170 training samples, 8,069 validation samples, and 7,740 testing samples. We leverage the BERT model to provide a sequence of word representations and add a single-layer feed-forward neural network to predict the dialogue act of the current sentence. We use AdamW [185] with a learning rate of 0.0001 to train this classifier. We set the batch size to 16, the total training epoch to 10, and automatically evaluate the model

on the validation set very 5000 iterations. We save the model checkpoint with the lowest validation loss as the optimal model. This dialogue act classifier achieves 0.79 accuracy on

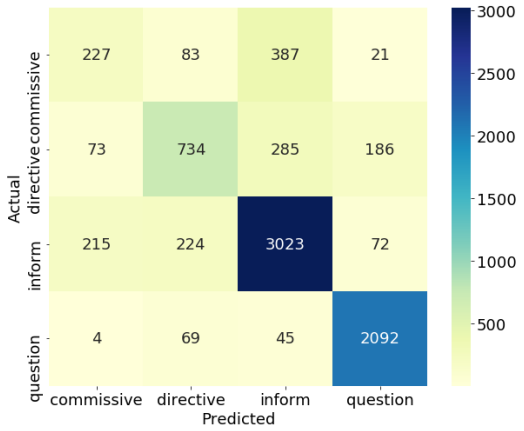


Figure 4.2: Confusion matrix of the evaluation dialogue act classifier.

the test set. Figure 4.2 shows the confusion matrix of this dialogue act classifier.

For the knowledge document control task, we use the *cosine similarity* between the word vectors of external knowledge document and model generated response. We use the pre-trained GloVe embedding [186] to model the word vectors. The word embeddings are GloVe embeddings [186] pre-trained on Wikipedia 2014 and Gigaword 5, which are 100-dimension vectors and have 6 billion tokens [1]. We use the NLTK word tokenizer [2] to tokenize the texts into a set of tokens, and remove stop words based on a pre-defined stop words list in [187]. Finally, we compute the cosine similarity between the two sets of word vectors.

Text Quality: It aims at evaluating how well the model learns to generate responses that match the ground-truth references, where we use model perplexity (PPL) computed on the test set [3], BLEU [121] and METEOR [166] to approximate it.

Decoding Cost: It evaluates the generation efficiency of the proposed method. Given the same set of 10 dialogue contexts, we compute the decoding time per token across different

[1] `https://nlp.stanford.edu/projects/glove/`

[2] `https://www.nltk.org/_modules/nltk/tokenize.html`

[3] Note that PPLM and FUDGE do not update the generative model and are applied only during generation, which means their model perplexity will be the same with their base network, i.e. DialoGPT-Ori, therefore we do not report their model perplexity in performance results.

methods, a faster decoding time indicates the method is more efficient during generation.

Modularity: It evaluates how well the side network can be applied to different base networks. We compare model performance under two different types of pre-trained language models: DialoGPT [169] and BlenderBot [170]. Ideally, we expect a good or even better performance when switching the base network from DialoGPT to BlenderBot, since BlenderBot has been trained on a larger dialogue corpus that is likely to provide more informative base representations.

Competitive Baselines. We compare the SIDECONTROL framework with the following competitive baselines:

1. *DialoGPT-Ori*: the original pre-trained language model for open-domain dialogue generation, DialoGPT [169]. DialoGPT is a Transformer-based language model. It is the baseline for all other controlled generation methods.

2. *DialoGPT-FT*: direct fine-tuning the DialoGPT on the target dialogue dataset. It is used as a strong baseline for evaluating the generation quality of the generative model.

3. *DialoGPT-PPLM*: the Plug-and-Play Language Model (PPLM) [34] with DialoGPT as the base pre-trained language model. It is a strong gradient-based baseline.

4. *DialoGPT-FUDGE*: the Future Discriminators for Generation (FUDGE) [36] with DialoGPT as the base pre-trained language model. It is a strong weighted decoding baseline.

5. *DialoGPT-SideControl*: our SIDECONTROL framework with DialoGPT as the base pre-trained language model. It is used to validate the effectiveness of our side network compared with other controlled generation baselines.

6. *BlenderBot-Ori*: the original BlenderBot [170], which is a Transformer-based sequence-to-sequence model showing state-of-the-art performance on some challenging open-domain dialogue datasets.

7. *BlenderBot-SideControl*: our SIDECONTROL framework with BlenderBot as the base pre-trained language model. It is used to show the high modularity of our side network.

Knowledge Document Control. In this task, given the previous dialogue context and the external knowledge document for the current speaker, the model will generate one utterance that is relevant both to the context and to the knowledge document.

We use the ConvAI2 dataset [181] for the knowledge document control task. We set the previous 4 utterances as the dialogue context. Each utterance is linked to its corresponding persona profile. Since the test set of ConvAI2 has not been made public, we use the original training set to construct our training set, and split the first 80% original validation set as our validation set and the remaining 20% original validation set as our testing set. In total, we have 153,082 training samples, 38,271 validation samples, and 11,590 testing samples.

We conduct all of our experiments on a single GeForce RTX 2080Ti GPU server with 12GB memory. The experiment setup details are as follows.

Direct Fine-tuning: We directly update all parameters of the pre-trained language model on the ConvAI2 training set without having any side network or control attribute loss. For the training of the pre-trained language model, we use AdamW [185] with a learning rate of 0.0001. We set the batch size to 2, the total training epoch to 10, and automatically evaluate the model on the validation set every 1000 iterations. We save the model checkpoint which achieves the lowest validation loss as the final optimal model. For generation, we follow the setup of FUDGE, which uses top-k sampling with $k = 10$.

PPLM: For the implementation of the attribute model, we use the bag-of-words attribute model proposed in the original paper [34] to encode external knowledge documents. We run the model on the ConvAI2 dataset using the code provided by the original paper.[4] We set the maximum generation length to 50, the number of gradient update steps to 3, the step size to 0.03, the window length to 5, the number of generated sentences to 1, $\gamma_{gm} = 0.99$,

[4] `https://github.com/uber-research/PPLM`

$\lambda_{KL} = 0.01$.

FUDGE: For the implementation of the attribute model, we use the bag-of-words attribute model proposed in the original paper [36] to encode external knowledge documents. We run the model on the ConvAI2 dataset using the code provided by the original paper.[5] We set the maximum generation length to 80, the weight on the conditioning model to 4.0, consider the top 200 outputs from DialoGPT at each decoding timestep before conditioning, and sample from the top 10 outputs from DialoGPT at each decoding timestep.

SideControl: For the implementation of the side network, we use a single-layer bi-LSTM which shares the same hidden dimension with the final hidden states of the base network. We tokenize the knowledge document using the same tokenizer with the base network, and share the same word embedding with the base network as well. For the training of the side network, we use AdamW [185] with a learning rate of 0.0001. We set the batch size to 4, the total training epoch to 10, and automatically evaluate the model on the validation set every 100 iterations. For the hyper-parameter λ of the coverage loss in Equation 4.17, we use grid search on the validation set to obtain the optimal number. We search from the set $\lambda = \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 0.01, 0.1\}$ and find $\lambda = 10^{-5}$ yields best performance. For generation, we follow the setup of FUDGE, which uses top-k sampling with $k = 10$.

Semantic Label Control. In this task, given the previous dialogue context and the current dialogue act, the model will generate one utterance that is relevant to the context and also satisfies the current dialogue act.

We use the DailyDialog dataset [182] for the semantic label control task. We set the previous 5 utterances as the dialogue context and follow the standard train/validation/test split of the original dataset to construct our generation dataset. In total, we obtain 35,781 training samples, 3,388 validation samples, and 3,123 testing samples.

We conduct all of our experiments on a single GeForce RTX 2080Ti GPU server with 11019 MB memory. The experimental setup details are described below.

[5] `https://github.com/yangkevin2/naacl-2021-fudge-controlled-generation`

Direct Fine-tuning: We directly update all parameters of the pre-trained language model on the DailyDialog training set without having any side network or control attribute loss. For the training of the pre-trained language model, we use AdamW [185] with a learning rate of 0.0001. We set the batch size to 2, the total training epoch to 10, and automatically evaluate the model on the validation set every 1000 iterations. We save the model checkpoint which achieves the lowest validation loss as the final optimal model. For generation, we follow the setup of FUDGE, which uses top-k sampling with $k = 10$.

PPLM: For the implementation of the attribute model, we follow the generic discriminator implementation in the original paper [34]. We run the model on the DailyDialog dataset using the code provided by the original paper. We train a dialogue act classifier which takes a single response as input and produces a prediction on one of the four dialogue acts. For the training of the classifier, we use Adam [188] with a learning rate of 0.0001. We set the batch size to 64, and the total training epoch to 10. For the generation of PPLM, we set the maximum generation length to 50, the number of gradient update steps to 10, the step size to 0.2, the number of generated sentences to 1, $\gamma_{gm} = 0.95$, $\lambda_{KL} = 0.01$.

FUDGE: For the implementation of the attribute model, we follow the attribute discriminator implementation in the original paper [36]. We run the model on the DailyDialog dataset using the code provided by the original paper. We train a dialogue act discriminator which takes the dialogue context and the current response as input and produces a prediction on one of the four dialogue acts. For the training of the discriminator, we use Adam [188] with a learning rate 2×10^{-5}. We set the batch size to 16, and the total training epoch to 10. For the generation of FUDGE, we set the maximum generation length to 60, the weight on the conditioning model to 1.0, consider the top 200 outputs from DialoGPT at each decoding timestep before conditioning, and sample from the top 10 outputs from DialoGPT at each decoding timestep.

SideControl: For the implementation of the side network, we use a single-layer feed-forward neural network which shares the same hidden dimension with the final hidden

	Controllability	Text Quality				Decoding Cost
METHOD	SIMILARITY ↑	PPL ↓	BLEU-1 ↑	BLEU-2 ↑	METEOR ↑	TIME ↓
DialoGPT-Ori	0.6382	68.63	12.95	1.22	0.0526	0.0786 s/tok
DialoGPT-FT	0.6732	15.22	17.27	2.05	0.0675	0.0721 s/tok
DialoGPT-FUDGE	0.6684	-	10.26	0.60	0.0514	0.0510 s/tok
DialoGPT-PPLM	0.6858	-	11.30	0.94	0.0646	0.5208 s/tok
DialoGPT-SideControl	**0.7526**	**14.34**	13.46	1.96	0.0988	0.0824 s/tok
BlenderBot-Ori	0.7455	90.89	9.38	0.54	0.0908	0.0384 s/tok
BlenderBot-SideControl	**0.7841**	**14.34**	10.10	1.20	0.0964	0.0608 s/tok

Table 4.1: Knowledge document control performances under full training set of ConvAI2, where $\lambda = 10^{-5}$ for $\mathcal{L}_{control}$ in DialoGPT-SideControl and BlenderBot-SideControl.

states of the base network. Besides, we pre-trained a dialogue act classifier to compute the control loss in Equation 4.22. We emphasize that this dialogue act classifier is different from the evaluation classifier. It models the sentence representation from the base network, i.e. DialoGPT, and adds a single-layer feed-forward neural network to predict the dialogue act of the current response. We train this classifier using AdamW [185] with a learning rate of 0.0001 for 10 epochs. Then, we fix this classifier and begin to train the side network using AdamW [185] with a learning rate of 0.0001 for another 10 epochs. We evaluate the model on the validation set every 1000 iterations, and save the model checkpoint which has the lowest validation loss. For the hyper-parameter λ of the control loss in Equation 4.22, we use grid search on the validation set to obtain the optimal number. We search from the set $\lambda = \{1, 10, 100, 10^3, 10^4, 10^5, 10^6\}$ and find $\lambda = 10^5$ yields best performance on the full training set. For generation, we follow the setup of FUDGE, which uses top-k sampling with $k = 10$.

4.1.3 Result Analysis

Knowledge Document Control. Table 4.1 shows that DialoGPT+SideControl outperforms all other baselines in controllability, which validates the effectiveness of the SIDECONTROL framework. For the quality of the generated texts, we find that both FUDGE and PPLM perform worse than the original pre-trained language model, while the SIDE-

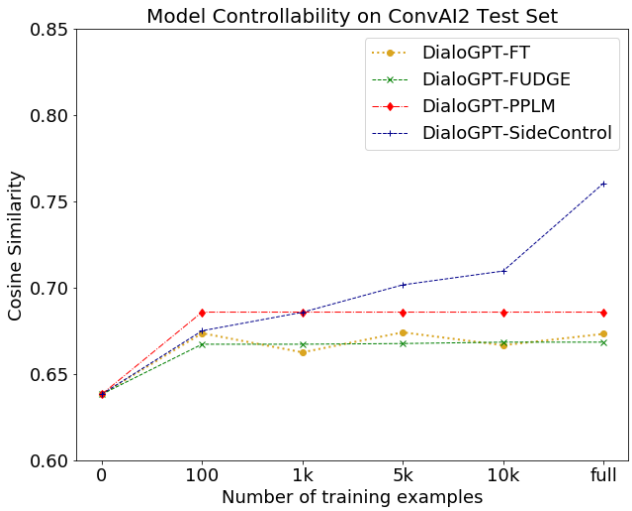


Figure 4.3: Controllability under a different number of training examples in ConvAI2 dataset.

CONTROL shows improved quality because of the $\mathcal{L}_{cclm}$ during training. We also notice that direct fine-tuning gives the best performance in BLEU-1 and BLEU-2, but worse controllability compared with the SIDECONTROL. This is because direct fine-tuning only focuses on optimizing the language modeling loss, and does not take the control attributes information into account. For the decoding cost, our SIDECONTROL is around 6x faster than PPLM during generation, which shows its efficiency during inference. Finally, we find that the performance improvement in controllability and text quality also holds when we apply the SIDECONTROL to BlenderBot, which shows the flexible modularity of the side network.

With the goal of testing the sample-efficiency of the SIDECONTROL framework, we train all baselines under smaller datasets, where we randomly sample 100, 1000, 5000 and 10000 training samples from the original training set to train the model, and evaluate the model performance using the full testing test. Figure 4.3 shows the controllability performance under different training sizes. We find that SIDECONTROL only underperforms PPLM in 100 training samples, since PPLM uses non-parametric bag-of-words features as its attribute model while SIDECONTROL uses a BiLSTM as its attribute model. And 1000 training samples are sufficient enough for SIDECONTROL to achieve comparable performance with PPLM. In addition, SIDECONTROL constantly achieves performance improvement when

	Controllability	Text Quality			
	SIMILARITY ↑	PERPLEXITY ↓	BLEU-1 ↑	BLEU-2 ↑	METEOR ↑
$\lambda = 0$	0.7273	**14.24**	15.72	2.16	0.0858
$\lambda = 10^{-6}$	0.7284	14.30	16.08	2.29	0.0800
$\lambda = 10^{-5}$	**0.7526**	14.34	13.46	1.96	0.0988
$\lambda = 10^{-4}$	0.7306	14.65	15.87	2.32	0.0846
$\lambda = 10^{-3}$	0.7259	15.65	15.72	2.09	0.0802
$\lambda = 10^{-2}$	0.7217	30.29	15.30	2.05	0.0803
$\lambda = 10^{-1}$	0.7137	22481.68	15.50	2.01	0.0774

Table 4.2: Knowledge document control performances of DialoGPT-SideControl with different λ.

	Controllability	Text Quality				Decoding Cost
METHOD	ACCURACY ↑	PPL ↓	BLEU-1 ↑	BLEU-2 ↑	METEOR ↑	TIME ↓
DialoGPT-Ori	0.4307	55.09	7.78	0.66	0.0333	0.0921 s/tok
DialoGPT-FT	0.4358	8.95	14.35	2.30	0.0523	0.0786 s/tok
DialoGPT-FUDGE	0.4701	-	14.40	1.59	0.0411	0.0535 s/tok
DialoGPT-PPLM	0.5994	-	14.22	1.25	0.0506	2.4171 s/tok
DialoGPT-SideControl	0.5376	12.79	16.37	1.90	0.0526	0.0990 s/tok
BlenderBot-Ori	0.4605	110.05	12.21	1.10	0.0775	0.0603 s/tok
BlenderBot-SideControl	**0.6865**	**8.16**	14.49	1.36	0.0680	0.0995 s/tok

Table 4.3: Semantic label control performances under full training set of DailyDialog, where $\lambda = 10^5$ for $\mathcal{L}_{control}$ in DialoGPT-SideControl and BlenderBot-SideControl.

increasing the training size.

To verify the effectiveness of the control loss $\mathcal{L}_{control}$, we conduct an ablation study by trying out different values of λ in Equation 4.6. As shown in Table 4.2, when $\lambda = 0$, the model becomes a vanilla language model and takes no information from the side network, which leads to low performance in controllability. When $\lambda \neq 0$, the model incorporates control attributes information from the side network, which leads to improved performance in controllability. However, incorporating side information will lead to a slight increase in model perplexity.

Semantic Label Control. Table 4.3 demonstrates that SIDECONTROL has better text quality than FUDGE and PPLM, since we explicitly optimize $\mathcal{L}_{cclm}$ during training. For controllability, PPLM achieves the best performance with a sacrifice of inference efficiency, while SIDECONTROL can achieve comparable performance in controllability with around

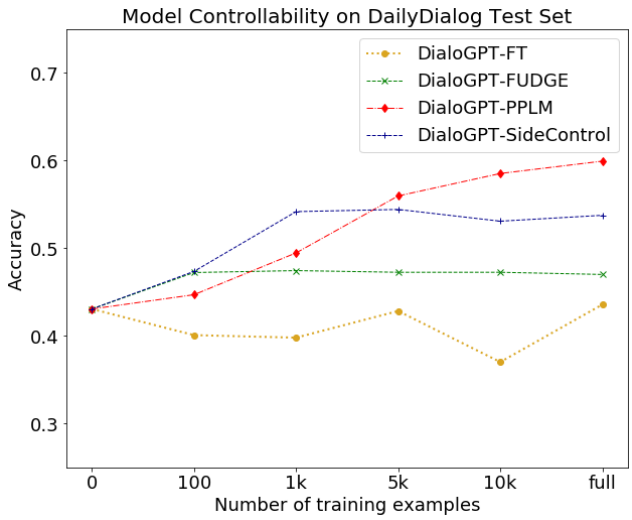


Figure 4.4: Controllability under a different number of training examples in DailyDialog dataset.

24x faster decoding time. Finally, the performance improvements in controllability and text quality still hold when we switch the base network from DialoGPT to BlenderBot, which demonstrates that the side network is flexible to be applied to different types of pre-trained language models. And surprisingly, BlenderBot can even provide state-of-the-art performance in controllability.

We also compare the model performance under different training sizes following the same setup with the knowledge document control task. Figure 4.4 illustrates that SIDECONTROL achieves better controllability than PPLM when the training size is under 1000. This is because PPLM uses a data-driven classifier as its attribute model in this task, and its attribute model gets overfitted on the 100 training samples, which results in poor controllability performance. Similarly, FUDGE has the same overfitting issue for its attribute discriminator on these small training sets, and gets unsatisfied controllability performance. Although SIDECONTROL also pre-trains a classifier on the 100 training samples to guide the update of side representation, its final representation is a combination of base and side representation. We believe incorporating prior knowledge from the base representation helps SIDECONTROL alleviate the overfitting issue on a small training set.

	Controllability	Text Quality			
	ACCURACY ↑	PERPLEXITY ↓	BLEU-1 ↑	BLEU-2 ↑	METEOR ↑
$\lambda = 0$	0.4950	**12.37**	16.19	1.95	0.0534
$\lambda = 10^1$	0.5229	12.48	15.06	1.76	0.0525
$\lambda = 10^2$	0.5366	12.51	15.59	1.76	0.0517
$\lambda = 10^3$	0.5232	12.59	15.59	1.75	0.0512
$\lambda = 10^5$	**0.5376**	12.79	16.37	1.90	0.0526
$\lambda = 10^6$	0.5357	13.10	15.29	1.67	0.0485

Table 4.4: Semantic label control performances of DialoGPT-SideControl with different λ.

METHOD	FLUENCY ↑	RELEVANCY ↑
DialoGPT-PPLM	3.832	3.188
DialoGPT-FUDGE	4.016	3.348
DialoGPT-SideControl	**4.108**	**3.816**

Table 4.5: Human evaluation of fluency and context relevancy on semantic label control task.

	Wins %		
	PPLM	FUDGE	SideControl
PPLM	-	55.25%	**57.61%**
FUDGE	44.76%	-	**54.46%**
SideControl	42.39%	45.54%	-

Table 4.6: Human evaluation of attribute relevancy on semantic label control task.

We also try out different values of λ to study the effect of control loss $\mathcal{L}_{control}$. As shown in Table 4.4, when $\lambda = 0$, the model takes no control attribute signals from the side network during training, which results in a low controllability performance. When $\lambda \neq 0$, the controllability performance of the model is improved but with a slight increase in model perplexity. Both Table 4.2 and Table 4.4 verify the effectiveness of control loss $\mathcal{L}_{control}$ in improving the controllability of pre-trained language models.

Human Evaluation. To validate the good performance of SIDECONTROL, we follow prior works [34, 189] and deploy a set of human evaluations to compare the text quality and controllability between several methods. For the text quality, we ask human annotators to evaluate the fluency and context relevancy of the generated responses on a scale of 1-5, where a higher score indicates better quality. For the controllability, we use A/B testing following [189] and compare all model pairs (e.g. PPLM vs. SIDECONTROL) [6]. For all human evaluations, we randomly sample 50 dialogue contexts, and collect the corresponding model-generated responses. Human annotators are recruited using Amazon Mechanical

[6]We show the same dialogue context, current dialog act and two responses generated by model A and model B respectively, and ask human annotators to select the response which is more related to the current dialog act among: model A, model B, both and neither.

Turk and each response has 5 annotations. In total, we collect 2250 human annotations. Table 4.5 shows the results of text quality evaluation, and SIDECONTROL achieves the best fluency and context relevancy than PPLM and FUDGE. Table 4.6 shows the results of controllability evaluation, and SIDECONTROL wins over PPLM and FUDGE in 57% and 54% respectively. Both text quality and controllability evaluation show that SIDECONTROL can generate more fluent, context-relevant, and attribute-relevant responses than PPLM and FUDGE.

4.2 Task-Oriented Dialogue Generation via Self-training Algorithms

In task-oriented dialogue systems, a natural language generation (NLG) module is an essential component: it maps structured dialogue meaning representations (MRs) into natural language responses. The NLG module has a great impact on users' experience because it directly interacts with users using text responses [172, 190, 191, 19]. However, in real-world applications, developers often only have a few well-annotated data and confront a high data collection cost in specific domains. This real-world challenge makes building an NLG module in the low-data setting a valuable research problem [191, 60, 19].

While language models have been widely adopted to build the NLG module in task-oriented dialogue systems, they usually require thousands of MR-to-Text pairs for learning the domain-specific knowledge [192, 193, 194, 195]. To collect more training data under a feasible budget, previous works propose three general approaches: (1) designing handcraft rules to augment new data, which is hard to scale up [196, 197]; (2) building task-specific data retriever to search related data, which may overfit on the few training data [63]; or (3) leveraging pre-trained language models to generate new data, which may generate "too noisy" data [198, 199, 67].

Ideally, the augmented data should help the model better learn the domain-specific knowledge. However, some augmented data can be "too noisy" that leads the model to learn irrelevant or inappropriate data patterns. This phenomenon is also described as negative

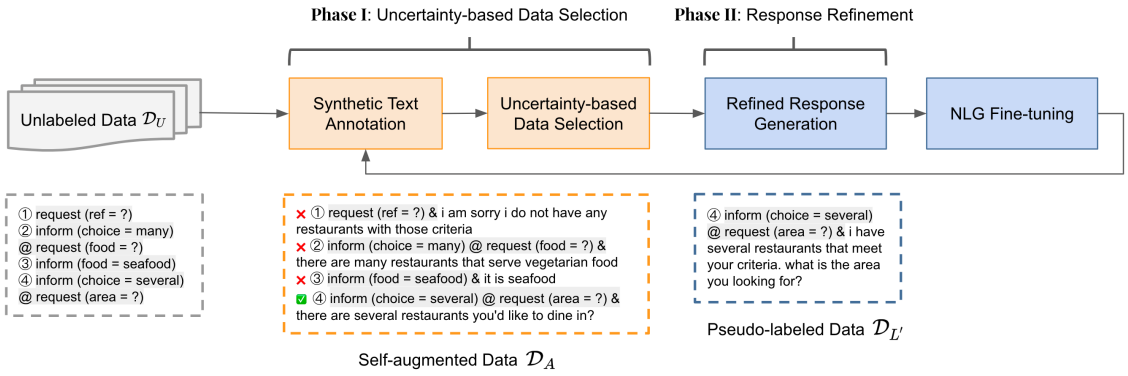


Figure 4.5: Our two-phase self-augmentation (SA^2) self-training framework for few-shot MR-to-Text generation.

transfer in other works [200, 201, 202, 203]. To address this challenge, some works leverage human judgments to filter out the "too noisy" augmented data, which are difficult to scale up across different domains and tasks [204, 205]. Other works train task-specific discriminators to pick up the valid augmented data, which are likely to overfit in the low-data setting [62, 63, 66, 67, 68].

In this work, we propose to address the issue of selecting high-quality self-augmented examples with a two-phase procedure, where each phase will take care of selecting inputs and generating outputs independently. As illustrated in Figure 4.5, the first phase evaluates input MRs with the model's prediction uncertainty, aiming at selecting input examples that are informative to the current model. Specifically, for each input MR, we let the current model generate a response, and then apply the Monte Carlo Dropout method [206] to estimate the predictive mean $\mathbb{E}[p_\theta]$ and predictive variance $Var[p_\theta]$ of the generated response. In uncertainty quantification [207], a high predictive mean indicates that the model is familiar with this input (i.e. in-domain data) and high predictive variance reflects that the model is sensitive to this input (i.e. informative data). Hence, we propose to select input MRs with high predictive mean and variance. Note that our uncertainty-based data selection strategy neither requires training additional neural models to select the valid data [66, 67, 68], nor need to calculate the data statistics across all training epochs and re-train the model overall again [208]. The second phase aims to further improve the quality of the

	Self-augmented Data	$\mathbb{E}[p_\theta]$	$Var[p_\theta]$
1	request (ref = ?) & i am sorry i do not have any restaurants with those criteria	**low**	**low**
2	inform (choice = many) @ request (food = ?) & there are many restaurants that serve vegetarian food	**low**	**high**
3	inform (food = seafood) & it is seafood	**high**	**low**
4 ✓	inform (choice = several) @ request (area = ?) & there are several restaurants you'd like to dine in?	**high**	**high**

Table 4.7: Examples of our self-augmented data and data selection strategy. text is the input MR (e.g. *request* is the dialogue intent, and *(ref = ?)* is the slot-value pair of the current intent). The model p_θ generates synthetic dialogue response conditioning on the text. For each self-augmented data, a **low** predictive mean $\mathbb{E}[p_\theta]$ indicates that the model finds the augmented data "too noisy" (e.g. out-of-domain or invalid response), and a **low** predictive variance $Var[p_\theta]$ indicates that the model finds the augmented data "too certain" (e.g. uninformative response). In this work, we propose to select examples with **high** $\mathbb{E}[p_\theta]$ and **high** $Var[p_\theta]$.

selected data. We adopt an idea from contrastive representation learning [209] and use the aggregation of randomly perturbed latent representations to help the model produce more accurate responses. The combination of these two phases guarantees the proposed method selects more informative MR inputs and generates less noisy responses for further model fine-tuning.

In summary, the contributions of this work are as follows:

1. Proposing a novel self-training algorithm for the few-shot MR-to-Text generation problem in task-oriented dialogue systems, which applies a two-phase self-augmentation strategy to identify informative MRs and generate accurate responses for further fine-tuning.
2. Showing that the proposed method generally outperforms other few-shot NLG baselines on two benchmark datasets, FEWSHOTWOZ [19] and FEWSHOTSGD [63] in both automatic and human evaluations.
3. Conducting in-depth empirical analysis on key components of the proposed few-shot self-training framework: the pre-trained language model, the data selection strategy,

and the model training configurations.

4.2.1 Self-training with Two-phase Self-augmentation (SA^2)

MR-to-Text Generation. In task-oriented dialogue systems, the NLG module translates a structured dialogue meaning representation $\mathcal{A}$ into a natural language response $\boldsymbol{x} = \{x_1, ..., x_T\}$. One structured dialogue meaning representation $\mathcal{A}$ consists of K dialogue intents and a list of slot-value pairs for each intent:

$$\mathcal{A} = \{\mathcal{I}_k, (s_{k,1}, v_{k,1}), ..., (s_{k,P_k}, v_{k,P_k})\}_{k=1}^{K} \tag{4.23}$$

where the dialogue intent $\mathcal{I}_k$ indicates different types of system actions and the slot-value pairs $\{(s_{k,i}, v_{k,i})\}_{i=1}^{P_k}$ shows the category names and their content information to be expressed in the response. For example, *inform (area = west; choice = many)*, where *inform* is the dialogue intent, *area* and *choice* are the slot names, *west* and *many* are the slot values.

We define $p_\theta(\boldsymbol{x} \mid \mathcal{A})$ as the generation model that generates the response $\boldsymbol{x}$ in an auto-regressive way conditioning on $\mathcal{A}$:

$$p_\theta(\boldsymbol{x} \mid \mathcal{A}) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{1:t-1}, \mathcal{A}) \tag{4.24}$$

where θ is the model parameter. A typical way of learning θ is by maximizing the log-likelihood of the conditional probabilities in Equation 4.24 over the original training set $\mathcal{D}_L$:

$$\mathcal{L}_\theta(\mathcal{D}_L) = \sum_{n=1}^{|D_L|} \sum_{t=1}^{T_n} \log p_\theta(x_{t,n} \mid x_{1:t-1,n}, \mathcal{A}_n) \tag{4.25}$$

In the few-shot MR-to-Text generation setting, the size of training data $|D_L|$ is a small number (e.g. ≤ 50).

SA^2 Self-training Algorithm Overview. The SA^2 self-training algorithm starts from a warm-up stage, where a base generation model is trained on the original training set $\mathcal{D}_L$ for

Algorithm 3: SA^2 Self-training Algorithm

Input: The original training set $\mathcal{D}_L$, in-domain MRs $\mathcal{D}_U$, base generation model p_θ, number of self-training iterations S

Output: A fine-tuned generation model p_θ

Load p_θ and train p_θ on $\mathcal{D}_L$
for $s = 1, \ldots, S$ **do**
  Initialize $\mathcal{D}_A = \emptyset$ and $\mathcal{D}_{L'} = \emptyset$
  // Synthetic Text Annotation
  for $\mathcal{A}_n \in \mathcal{D}_U$ **do**
    Generate $\boldsymbol{x}_n \sim p_\theta(\boldsymbol{x}_n \mid \mathcal{A}_n)$
    $\mathcal{D}_A \cup \{(\boldsymbol{x}_n, \mathcal{A}_n)\}$
  end for
  // Data Selection
  Compute threshold $\bar{\mu}$ and $\bar{s}$ using Eq. (4.28)
  for $(\boldsymbol{x}_n, \mathcal{A}_n) \in \mathcal{D}_A$ **do**
    if $\mathbb{E}[p_\theta] > \bar{\mu}$ and $Var[p_\theta] > \bar{s}$ **then**
      // Response Refinement
      Generate $\bar{\boldsymbol{x}}_n$ using Eq.(4.29)
      $\mathcal{D}_{L'} \cup \{(\bar{\boldsymbol{x}}_n, \mathcal{A}_n)\}$
    end if
  end for
  Fine-tune p_θ on $\mathcal{D}_L \cup \mathcal{D}_{L'}$
end for

a few epochs. Then, in each iteration of self-training, the algorithm consists of four steps: synthetic text annotation, uncertainty-based data selection, response refinement, and model fine-tuning.

The synthetic text annotation uses the current model to generate synthetic text responses based on input MRs and constructs a preliminary version of self-augmented data $\mathcal{D}_A$. Next, the data selection uses the prediction uncertainty of the current model on the synthetic responses to select informative MRs in $\mathcal{D}_A$, which is the *first phase* of self-augmentation. Given the selected MRs, the *second phase* of self-augmentation is to generate more accurate text responses via aggregating multiple latent representations from model parameters with different dropout masks, which produces the pseudo-labeled data $\mathcal{D}_{L'}$. Finally, the current model is fine-tuned with both the original training set $\mathcal{D}_L$ and the pseudo-labeled dataset $\mathcal{D}_{L'}$.

The detailed procedure of SA^2 self-training algorithm is demonstrated in algorithm 3. We choose the pre-trained language model SC-GPT [19] as our base generation model p_θ. We collect in-domain MRs from the training set of existing task-oriented dialogue datasets, such as MultiWOZ corpus [210] and Schema-Guided Dialog corpus [190]. We use nucleus sampling [211] with the threshold $p = 0.9$ to generate the output tokens for both synthetic text annotation and refined response generation.

Phase I: Uncertainty-based Data Selection. We hypothesize that the generation model is likely to gain little by learning from the data, if (1) it finds "too noisy", which may be out-of-domain or invalid; (2) it finds "too certain", which may be uninformative to learn from. Therefore, we propose to select the data which the current model finds "less noisy" and "more uncertain". Intuitively, data with "less noise" may provide helpful domain-specific knowledge to the model, meanwhile "more uncertainty" indicates the model has not learned well from the data yet, thus may produce incoherent responses.

We use the Monte Carlo Dropout method [206, 212] to estimate the "noise" and "uncertainty" of each self-augmented data regarding the current model. For each self-augmented data $(\boldsymbol{x}, \mathcal{A})$, we enable dropouts before every hidden layer in the generation model, perform M forward passes through the model, and get M i.i.d. model likelihood estimations $\{p_{\theta_i}(\boldsymbol{x} \mid \mathcal{A})\}_{i=1}^{M}$. These M outputs are empirical samples of an approximated posterior distribution $p(\boldsymbol{x} \mid \mathcal{A})$ [207]. Then, we compute the predictive mean $\mathbb{E}[p_\theta]$ of the approximated distribution $p(\boldsymbol{x} \mid \mathcal{A})$ and predictive variance $Var[p_\theta]$ of the empirical samples:

$$\mathbb{E}[p_\theta] \approx \frac{1}{M}\sum_{i=1}^{M} p_{\theta_i}(\boldsymbol{x} \mid \mathcal{A}) \tag{4.26}$$

$$Var[p_\theta] \approx \frac{1}{M}\sum_{i=1}^{M} (p_{\theta_i}(\boldsymbol{x} \mid \mathcal{A}) - \mathbb{E}[p_\theta])^2 \tag{4.27}$$

A low predictive mean $\mathbb{E}[p_\theta]$ means the model finds the current data "too noisy", because it has a low likelihood estimation of the current data, which indicates the current data may be

out-of-domain or invalid; while a low predictive variance $Var[p_\theta]$ means the model finds the current data "too certain", because all empirical samples have a similar likelihood estimation of the current data, which indicates the current data may be uninformative for the model to learn from. Therefore, we consider self-augmented data with both high predictive means and variances are examples of interest.

The next question is *what are the thresholds for high predictive means and variances?* First, we calculate the corpus-level predictive mean μ_A of the self-augmented $\mathcal{D}_A$, and filter out the augmented data which have a lower predictive mean than μ_A, because we observe that such data are often very noisy and contain many redundant slots. Then, we combine and sort the original training data $\mathcal{D}_L$ and the remaining self-augmented data, and further remove the outliers (i.e. first and last 1% of data points). Assume that the collection of predictive mean scores $\mathbb{E}[p_\theta]$ and variance scores $Var[p_\theta]$ of the selected data follow a Gaussian distribution respectively, then the data selection threshold is defined as

$$\bar{\mu} = \frac{1}{N}\sum_{n=1}^{N} p_n, \quad \bar{s} = \frac{1}{N}\sum_{n=1}^{N} v_n \tag{4.28}$$

where p_n is the predictive mean and v_n is the predictive variance of the n-th selected data, N is the total number of original training data and remaining self-augmented data (after removing the outliers).

We select the self-augmented data with high $\mathbb{E}[p_\theta]$ (above the average predictive mean $\bar{\mu}$) and high $Var[p_\theta]$ (above the average predictive variance $\bar{s}$). We also explored other data selection strategies (detailed in §4.2.3), and find that selecting high $\mathbb{E}[p_\theta]$ and high $Var[p_\theta]$ data empirically brings more performance improvements than other strategies.

Phase II: Response Refinement. Since the large generation model is trained on a small training set, it is very likely to overfit and produce high-biased latent representations that cause the generation of inaccurate text responses. To reduce the risk of producing high-biased latent representations, we adopt dropout noise proposed in contrastive learning [209]

into the latent representation during inference.

Specifically, for each selected input MR from **Phase I**, we enable the dropout masks of the model (placed on fully connected layers as well as attention probabilities) at the decoding timestamp t, and compute R latent representations $\{\boldsymbol{h}_{\theta_i}^t\}_{i=1}^R$, then take an average over all latent representations to obtain the final latent representation for the current probability distribution:

$$p(\bar{x}_t \mid \bar{x}_{1:t-1}, \mathcal{A}) = \text{softmax}(\frac{1}{R}\sum_{i=1}^{R} \boldsymbol{h}_{\theta_i}^t) \tag{4.29}$$

Then, we generate the text response $\bar{\boldsymbol{x}}$ according to the probability distribution $p(\bar{x}_t \mid \bar{x}_{1:t-1}, \mathcal{A})$ and add the data $(\bar{\boldsymbol{x}}, \mathcal{A})$ into the pseudo-labeled dataset $\mathcal{D}_{L'}$. We fine-tune the generation model on both the original training set $\mathcal{D}_L$ and the pseudo-labeled dataset $\mathcal{D}_{L'}$. Fine-tuning the refined responses is shown to improve the model's final performances (detailed in §4.2.3).

4.2.2 Experimental Setup

We conduct evaluation experiments to answer three research questions: (1) Is SA^2 self-training algorithm a helpful method to deal with the few-shot dialogue generation problem? (2) Can our data selection strategy effectively filter out the "too noisy" and "uninformative" augmented data? (3) Can our response refinement method help improve the performance of the NLG model?

Benchmark Datasets. We evaluate our method on two few-shot dialogue generation benchmark datasets: FEWSHOTWOZ [19] and FEWSHOTSGD [63]. FEWSHOTWOZ has 7 domains and an average number of 50 training examples per domain. FEWSHOTSGD has 16 domains and an average number of 35 training examples per domain. However, both datasets do not provide the development sets for hyperparameter tuning. To create the standard training/dev/test data splits, we randomly sampled 10% data from the original test set as the dev set, and kept the training set unchanged. For fair comparisons across different

	Restaurant	Laptop	Hotel	TV	Attraction	Train	Taxi
# Training Pairs	51	51	51	51	50	50	40
# Dev Pairs	12	137	7	68	34	65	4
# Test Pairs	117	1242	71	612	306	592	43
# Unlabeled Data	10,000	10,000	10,000	7,035	10,000	10,000	6,527

Table 4.8: Data statistics for the original manual-labeled data $\mathcal{D}_L$ and the unlabeled data $\mathcal{D}_U$ on FEWSHOTWOZ.

	Restaurants	Hotels	Flights	Buses	Events	Rentalcars	Services	Ridesharing
# Training Pairs	50	50	50	50	50	50	50	48
# Dev Pairs	961	401	272	427	836	287	793	819
# Test Pairs	8,657	3,615	2,453	3,845	7,526	2,592	7,146	7,378
# Unlabeled Data	10,000	10,000	10,000	10,000	10,000	10,000	10,000	8,259
	Movies	**Calendar**	**Banks**	**Music**	**Homes**	**Media**	**Travel**	**Weather**
# Training Pairs	30	25	23	21	21	14	14	11
# Dev Pairs	737	532	332	732	563	568	528	193
# Test Pairs	6,634	4,793	2,988	6,594	5,073	5,121	4,753	1,742
# Unlabeled Data	7,604	5,355	3,343	7,347	5,657	5,703	5,299	1,947

Table 4.9: Data statistics for the original manual-labeled data $\mathcal{D}_L$ and the unlabeled data $\mathcal{D}_U$ on FEWSHOTSGD.

methods, we evaluated all methods on the new split test set. The detailed data statistics of FEWSHOTWOZ is presented in Table 4.8. The detailed data statistics of FEWSHOTSGD is demonstrated in Table 4.9.

Unlabeled Data. The two benchmark datasets are sampled and constructed based on the three datasets: RNNLG [192], MultiWOZ [210] and SGD [213]. To ensure the input MRs are within the same domain of the original training set $\mathcal{D}_L$, we collect all augmented MRs from the training set of RNNLG, MultiWOZ, and SGD. For FEWSHOTWOZ, we collect an average number of 9,080 unlabeled MRs per domain. For FEWSHOTSGD, we collect an average number of 7,532 unlabeled MRs per domain.

Evaluation Metrics. We follow the prior works [172, 19, 63] and use BLEU score and Slot Error Rate (ERR) for automatic evaluation. ERR is computed by exactly matching the slot tokens in the generated responses as $ERR = (p + q)/N$, where N is the total number of slots in the MR, and p,q is the number of missing and redundant slots in the generated

response. For each MR, we generate five responses and select the top one with the lowest ERR as the final output. Note that we only compute ERR on the FEWSHOTWOZ dataset, because the FEWSHOTSGD dataset does not release its evaluation script.

We also follow the prior works [19, 191] and use Amazon Mechanical Turk to conduct the human evaluation. We recruited master-level workers with over 90% approval rate to compare and rate the responses generated by different methods and the the ground truth response. The workers are asked to rate the response on a scale of 1 (bad) to 3 (good) in terms of *informativeness* and *naturalness*. Informativeness indicates how much information from the input MR has been covered in the response, and naturalness measures whether the response looks coherent, grammatical, and natural. Each data pair is rated by 3 workers. We randomly sample 120 examples from each dataset, and collect a total of 2880 ratings.

Competitive Baselines. We compare our method with four baselines. (1) *SC-GPT* [19] is the state-of-the-art pre-trained language model for NLG in task-oriented dialogue systems, which is further fine-tuned on each specific domain using the original training data $\mathcal{D}_L$; (2) *AUG-NLG* [63] leverages the pre-trained SC-GPT model, first trains it on its automatically retrieved augmented data, then fine-tunes it on each few-shot domain; (3) *ST-ALL* is the traditional self-training baseline which learns from all self-augmented data without any data selection and text refinement; (4) *ST-NLL* adopts the traditional self-training baseline but learns from the self-augmented data which has a lower than the average reconstruction loss according to the current generation model; (5) *ST*-SA^2 is our method, in addition to our proposed data selection strategy and response refinement method, we apply a rule-based parser [64] to heuristically filter out invalid responses that do not match the slot-value pairs in the input MRs on the FEWSHOTWOZ dataset in order to achieve lower ERR.

	Restaurant		Laptop		Hotel		TV		Attraction		Train		Taxi	
	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR
SC-GPT	34.62	**1.95**	33.31	3.01	40.74	3.55	33.72	1.72	23.77	1.40	25.09	1.90	18.22	0.00
AUG-NLG	29.94	2.28	30.02	4.29	38.30	4.73	32.41	3.34	21.76	3.95	24.06	3.81	17.99	0.00
ST-ALL	33.84	6.51	34.40	4.28	39.68	1.78	34.88	1.76	24.32	3.19	24.47	3.87	17.89	0.00
ST-NLL	33.07	9.44	34.99	3.37	41.40	5.92	35.98	2.26	24.87	4.85	23.53	5.27	20.21	0.00
ST-SA^2 (ours)	**36.48**	2.60	**35.42**	**2.04**	**42.63**	**1.77**	**36.39**	**1.63**	**25.63**	**1.40**	**25.34**	**1.62**	**20.95**	**0.00**

Table 4.10: Automatic evaluation results on the test set of FEWSHOTWOZ (BLEU↑, ERR↓). The results of **AUG-NLG** come from the data and code released by [63], the other results come from our implementation.

4.2.3 Result Analysis

On FEWSHOTWOZ. The automatic evaluation results in Table 4.10 show that *ST*-SA^2 outperforms other baselines across all domains in both BLEU and ERR. Besides, we observe that *SC-GPT* is a strong baseline, and *ST-NLL* can bring more performance improvements than *AUG-NLG* and *ST-ALL* in 5 out of 7 domains, which shows the effectiveness of data selection in self-training. The human evaluation results in Table 4.13 indicate that *ST*-SA^2 can generate more natural and informative responses than *SC-GPT* and *ST-NLL*.

On FEWSHOTSGD. The automatic evaluation results in Table 4.11 illustrates that *ST*-SA^2 outperforms other baselines in 14 out of 16 domains in BLEU score. Additionally, we find that *ST-ALL* generally outperforms *AUG-NLG*, which indicates that additional pre-training on the retrieved task-relevant data does not necessarily help the model generate better responses. In contrast, the self-training method *ST-ALL* generally improves the model performances in 10 out of 16 domains, which shows the benefit of learning from self-augmented data. The human evaluation results in Table 4.12 demonstrate that *ST*-SA^2 is capable of generating more informative and natural responses than *SC-GPT* and *ST-ALL*.

Ablation Study on Response Refinement. To validate the effectiveness of the proposed response refinement method, we conduct an ablation study on *ST*-SA^2 by removing the representation aggregation in Equation 4.29 and the rule-based filter [64] respectively. We

	Restaurants	Hotels	Flights	Buses	Events	Rentalcars	Services	Ridesharing
SC-GPT	19.86	22.21	26.63	19.87	**26.41**	20.21	27.32	22.03
AUG-NLG	19.73	12.38	23.20	16.81	19.62	16.64	20.18	17.20
ST-ALL	19.71	21.45	26.90	19.76	25.68	20.22	27.59	21.14
ST-NLL	14.52	21.29	**27.59**	20.27	25.81	20.07	26.54	19.84
ST-SA^2 (ours)	**20.42**	**22.90**	27.12	**21.16**	25.32	**20.70**	**28.34**	**23.28**
	Movies	**Calendar**	**Banks**	**Music**	**Homes**	**Media**	**Travel**	**Weather**
SC-GPT	25.71	23.53	25.99	24.01	24.90	26.24	24.97	27.89
AUG-NLG	16.93	13.60	12.89	9.56	18.06	10.51	15.77	10.74
ST-ALL	26.19	24.86	25.03	24.62	24.97	26.56	25.28	28.06
ST-NLL	23.98	23.67	25.70	18.88	24.82	26.99	24.95	28.64
ST-SA^2 (ours)	**28.95**	**25.24**	**28.14**	**27.23**	**25.03**	**28.76**	**25.34**	**29.27**

Table 4.11: Automatic evaluation results of BLEU scores on the test set of FEWSHOTSGD. The results of **AUG-NLG** come from the data and code released by [63], the other results come from our implementation.

	Informativeness ↑	Naturalness ↑
SC-GPT	2.53	2.31
ST-ALL	2.55	2.40
ST-SA^2 (ours)	**2.69**	**2.42**
Human	2.69	2.56

Table 4.12: Human evaluation results on the sampled test set of FEWSHOTSGD.

	Informativeness ↑	Naturalness ↑
SC-GPT	2.62	2.32
ST-NLL	2.69	2.31
ST-SA^2 (ours)	**2.69**	**2.41**
Human	2.71	2.49

Table 4.13: Human evaluation results on the sampled test set of FEWSHOTWOZ.

observe from Table 4.14 that removing the representation aggregation during response refinement will lead to degraded performances in both BLEU and ERR across all domains, which indicates the importance of obtaining lower-biased latent representations during self-augmentation. Besides, we find that removing the rule-based filter will lead to worse performances in ERR across all domains, which reveals that the model is still likely to generate incorrect responses, and those incorrect pseudo-labeled data will cause the model to learn irrelevant patterns and perform worse on the unseen test set.

Analysis of Other Components in SA^2 Self-training Algorithm. We provide additional empirical analysis on other components that will affect the performance of the SA^2 self-training algorithm, in order to gain more insights about the self-training technique in solving the few-shot NLG problem.

	Restaurant		Laptop		Hotel		TV		Attraction		Train		Taxi	
	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR	BLEU	ERR
ST-SA^2 (ours)	**36.48**	**2.60**	**35.42**	**2.04**	**42.63**	**1.77**	**36.39**	**1.63**	**25.63**	**1.40**	**25.34**	**1.62**	**20.95**	**0.00**
w/o aggregation	35.30	3.25	34.30	3.57	39.08	2.96	36.24	5.25	24.44	2.55	24.15	2.01	20.17	1.69
w/o filter	36.17	3.90	34.19	5.85	39.52	3.55	35.45	2.76	25.51	2.42	24.89	2.24	20.60	0.00

Table 4.14: Ablation study results on the test set of FEWSHOTWOZ (BLEU↑, ERR↓).

	$\mathbb{E}[p_\theta]$	$Var[p_\theta]$	**BLEU** ↑	**ERR** ↓
1	**low**	**low**	32.72	1.62
2	**low**	**high**	32.24	1.62
3	**high**	**low**	33.18	2.28
4	**high**	**high**	**36.48**	2.60

Table 4.15: Different data selection strategy comparison of **ST-SA**^{2} in the **Restaurant** domain on the test set of FEWSHOTWOZ.

	Base Model	**BLEU** ↑	**ERR** ↓
1	GPT2	24.22	13.68
2	DialoGPT	14.77	20.84
3	SC-GPT	**36.48**	2.60

Table 4.16: Different base generation model comparison of **ST-SA**^{2} in the **Restaurant** domain on the test set of FEWSHOTWOZ.

Data Selection Strategies: Table 4.15 compares different data selection strategies of *ST*-SA^2 in the restaurant domain of FEWSHOTWOZ. We find that selecting low $\mathbb{E}[p_\theta]$ data will lead to degraded performance in BLEU score, because low $\mathbb{E}[p_\theta]$ data often contains more redundant tokens compared with the ground-truth response. Although low $\mathbb{E}[p_\theta]$ data gives lower ERR, the generated texts are not very natural and fluent. Selecting high $\mathbb{E}[p_\theta]$ and low $Var[p_\theta]$ data will also lead to degraded performance in the BLEU score, which is probably because the model overfits on the uninformative data.

Base Generation Models: For the base generation model selection, we compare different pre-trained language models, including GPT2 [168], DialoGPT [169], and SC-GPT. GPT2 is an open-end text generation model, and DialoGPT is an open-domain dialogue generation model. In contrast, SC-GPT is trained on around 400K MR-to-Text pairs in task-oriented dialogue generation datasets. As can be seen in Table 4.16, SC-GPT gives much better performance than GPT2 and DialoGPT, which indicates that selecting a suitable base generation model is critical for self-training.

Training Hyper-parameters: Table 4.17 compares different training hyper-parameters of *ST*-SA^2 in the attraction domain of FEWSHOTWOZ dataset. We observe that the learning

	Epoch	LR	BLEU$_{dev}$ ↑	BLEU$_{test}$ ↑	ERR$_{test}$ ↓
1	10	1e-6	23.22	24.75	2.93
2	20	1e-6	22.96	24.63	2.04
3	20	5e-7	**23.43**	**25.63**	**1.40**
4	20	5e-8	23.29	24.82	1.91

Table 4.17: Different training hyper-parameters comparison of **ST-SA**^{2} in the **Attraction** domain of FEWSHOTWOZ, where **Epoch** is the number of training epochs within a self-training iteration, and **LR** is the initial learning rate at the beginning of each training epoch. We select the best model which has the highest **BLEU**$_{dev}$.

rate plays an essential role in training NLG models under the low-data setting. If the learning rate is too large, the development loss may not converge because the training set is too small; if the learning rate is too small, the model may get stuck into the local optimal. Finally, we find a good combination of learning rate and training epoch can help the model achieve the best performance, but the specific values vary across different domains.

4.3 Summary

In this chapter, we design data-efficient text generation methods to better adapt pre-trained LLMs to domain-specific NLG tasks.

In Section 4.1, we propose a new method for controlled dialogue generation: adding a small side network to incorporate useful control signals into the pre-trained language models. We design control attribute loss to teach the side network learning useful control signals. Empirical experiments show that our method is effective even with $100 \sim 1000$ training samples. Besides, our side network supports diverse forms of attribute control and can be flexibly applied to any pre-trained language models, which extends its possible application to other general controlled text generation tasks.

In Section 4.2, we present a two-phase self-augmentation self-training algorithm to deal with the few-shot dialogue generation problem in task-oriented dialogue systems. We propose to select informative input MRs based on the model's prediction uncertainty, and improve the pseudo-response generation by aggregating randomly perturbed latent

representations. Empirical experiments on two few-shot NLG datasets show that our proposed method achieves the best performance among other baselines in both automatic and human evaluations.

CHAPTER 5
USER INTENTION MODELING

Accurately recognizing user intentions is important for controlling the model's responses, ensuring that they align closely with users' specific needs. This alignment will facilitate more effective and satisfying interactions between users and the system, enhancing user experience and engagement. In Section 5.1, I propose to model the communicative intents in open-domain conversations from the sentence-level to segment-level as the dialog flow for multi-turn dialogue evaluation. In Section 5.2, I design a novel edit intention taxonomy to better understand human editing behaviors, and use the edit intention labels to guide LLMs generating text revisions that better align with human preferences.

5.1 Identifying Segment Act Flows for Consensus-Based Dialogue Evaluation

Dialogue evaluation plays a crucial role in the recent advancement of dialogue research. While human evaluation is often considered a universal and reliable method by the community [214], automatic dialogue evaluation metrics draw growing attention as they can assess dialogues with faster speed and lower cost [215, 216, 217].

Traditional word-overlap metrics, like BLEU [218] and METEOR [166], lose some of their effectiveness in the dialogue setting as reliable references are hard to obtain [219]. Recent works tackle this problem by leveraging more sophisticated architectures [220, 221] and harnessing the power of large models [217]. Although these recent metrics claim to show some progress towards higher correlation with humans, the gap between automatic metrics and human evaluation is still noticeable [222]. Automatic open-domain dialogue evaluation is still an open question, and extensive efforts have been made to improve performance from different angles [223, 224, 225, 226].

Among those newly released metrics [220, 217, 221, 223, 215, 225, 226], hardly

Dialogue Section	Segment Act Flow
Speaker1: How are you? May I have a cup of coffee?	greeting , directive
Speaker2: Hmm. Certainly. What kind of coffee do you like? We have espresso and latte.	backchannel- success , commissive , question , inform

Table 5.1: A snippet of an open-domain dialogue and its segment act flow. Each segment is marked with the same color as its corresponding segment act label.

any explicitly employs dialog act, one of the pillars of dialogue study, in their methods. Intuitively, introducing dialog act into open-domain dialogue evaluation should be beneficial: a sequence of dialog acts distills the core function of each utterance and can potentially reveal how speakers interact in general. However, directly using preexisting dialog act definitions [73, 71] seems undesirable, as an utterance can contain several segments that possess different conversational functions. We show our observation in Table 5.1. By saying "*Hmm. Certainly. What kind of coffee do you like? We have espresso and latte.*", the participant first acknowledges the conversation with backchannel, then commits to finishing the request by saying "*Certainly*". Later, the speaker asks for a more concrete order and offers all the options. In human conversations, it is common to have more than one function for each utterance, which means using a single dialog act to express the core function of an utterance inevitably suffers from information loss. To solve this issue, we extend the concept of dialog act from the utterance level to the segment level. We name these segment-level dialog act *segment act* and a sequence of segment acts a *segment act flow*.

One difficulty of using segment act for open-domain dialogue evaluation is the lack of related data. Since there is no dataset for the segment act, we follow the ISO 24617-2 annotation criteria [227] and propose a simplified ISO-format segment act tagset. We crowdsource large-scale segment act annotations on two popular open-domain dialogue datasets: ConvAI2 [181] and DailyDialog [228]. We name our dataset *ActDial*.

Another challenge of incorporating segment act into open-domain dialogue evaluation lies in finding a suitable way to assess dialogues with the segment act feature. Modeling

segment act flow is not trivial. On the one hand, dialogues have different numbers of turns and, thus, have varying lengths of segment act sequences. On the other hand, defining and finding the ground-truth segment act flow for a dialogue is almost infeasible, discouraging the development of any reference-based methods. To overcome this challenge, we design the first consensus-based reference-free open-domain dialogue evaluation framework, *FlowEval*.

For a dialogue to be evaluated, our *FlowEval* first obtains the segment act flow, e.g., from a trained classifier. Then, we harvest segment act features, from a dedicated BERT-like [184] masked segment act model, and content features, from RoBERTa-large [229]. We retrieve pseudo-references from the training set, according to the segment act features as well as content features. Last, we evaluate the dialogue with the consensus of the pseudo-references, fusing metrics from both segment act and word-overlap perspectives. The essence of our consensus-based framework lies in retrieving pseudo-references and using the consensus of pseudo-references to assess a new dialogue. This process can be regarded as reference-free, since no additional dialogue evaluation label is required. Not limited to segment act features, our proposed consensus-based framework is compatible with a wide range of features and metrics, such as sentiment features, engagingness features, etc.

Extensive experiments are carried out against the state-of-the-art baselines on Controllable Dialogue dataset [230], FED dataset [217], and DSTC9 dataset [231]. The result supports that segment act flow is effective in dialogue evaluation: our consensus-based method achieves the best or comparable correlation with human evaluation. Additionally, segment act flow can bring complementary information to metrics that heavily focus on the raw text of dialogues.

In summary, the contributions of this work are three-fold:

1. We propose to model the segment level act as the dialog flow information for open-domain dialogue evaluation.
2. We are the first to propose a consensus-based framework for open-domain dialogue evaluation. Our studies show that the consensus approach can work efficiently even

when the size of the search set, i.e., the number of dialogues in the training set, is around ten thousand. This attainable size shows the promise of our consensus approach for dialogue evaluation and other natural language evaluation tasks.

3. Our method can reach the best or comparable performance when compared with state-of-the-art baselines. Additional experiments are conducted to examine the detailed properties of our method and consensus process.

5.1.1 ActDial: A Segment Act Dataset on Open-Domain Dialogues

We propose the new concept of segment act, extracting the core function of each segment in an utterance. We then crowdsource a large-scale open-domain dialogue dataset with our proposed segment act labels, called *ActDial*.

Segment Act Tagset. We design an open-domain segment act tagset based on the ISO 24617-2 annotation criteria [227]. We define a segment act as a functional label that expresses the communicative goal of participants in a conversation, which is irrelevant to syntactic or sentiment details. Based on this definition, we conduct combination operations, like merging Choice-Question, Check Question, etc. into question, on the original 56 labels proposed by [227] and eventually obtain 11 labels as our tagset. These combination operations guarantee robust coverage of diverse dialogue expressions and mutual exclusiveness between different segment act labels. From our later experiments, these 11 labels capture key information from dialogues while remaining simple enough to enable large-scale accurate annotations.

For the formal definitions and examples of segment act, please refer to Table 5.2. The eleven segment act labels cover three major communication activities: (i) general task, which includes information-transfer activities and action-discussion activities; (ii) social obligation management, which includes typical social conventions in communication; and (iii) simple feedback, which includes simple non-informative feedback about the processing

General Dimension	Segment Act	Definition	Examples	Distribution
General Task	inform	The sender makes the addressee know some information which he assumes to be correct.	"The train is leaving.", "The meeting starts in 5 minutes."	65.702%
	question	The sender asks the addressee to provide some information which he assumes the addressee knows.	"What time is it?", "Where is the nearest bank?"	16.529%
	directive	The sender asks the addressee to perform an action.	"Please don't do this ever again."	2.880%
	commissive	The sender considers to perform an action which he believes would be in addressee's interest, or he has been requested/suggested to perform by the addressee.	"I will not do that any more.", "May I offer you an upgrade?"	0.517%
Social Obligation Management	greeting	The speakers inform the presence of each other.	"how are you?", "I'm fine."	6.023%
	goodbye	The speakers inform the end of the dialog.	"Bye", "See you."	0.172%
	apology	The speakers express or mitigate the feelings of regret.	"Sorry.", "No problem."	0.542%
	thanking	The speakers express or mitigate the feelings of gratitude.	"Thanks.", "You are welcome."	1.049%
Simple Feedback	backchannel-success	The speakers succeed in processing the previous dialog.	"Okay", "Uh-huh"	6.543%
	backchannel-failure	The speakers fail in processing the previous dialog.	"Sorry?", "Excuse me?"	0.030%
	check-understanding	The sender wants to check whether the addressee succeed in processing the previous dialog.	"Do you get what I just said?"	0.013%

Table 5.2: Our ISO-format open-domain segment act tagset: the definition, examples, and distribution

of previous utterances. We segmented all the dialogue utterances using the NLTK sentence punctuation tokenizer [232] that mainly consists of a set of rule-based regular expressions on punctuation.

Datasets and Segmentation. We crowdsourced segment act annotations on ConvAI2 [181] and DailyDialog [228]. We crowdsourced segment act annotations from annotators whose native language is Mandarin Chinese (zh-cmn), but more importantly, they are proficient in English (en-US). More than 50 annotators participated after rigorous training to ensure data quality. Each segment is annotated by three different annotators. If the initial three annotations are all different, further round(s) of annotation on this segment would be

conducted until it got a majority vote (at least two annotations are the same).

The **ConvAI2** dataset is based on the PersonaChat dataset [233], where all dialogues are constructed by asking two crowd-workers to chat with each other based on randomly assigned persona profiles. ConvAI2 is a widely-used benchmark for many state-of-the-art dialogue systems [234, 187, 235, 170].

The **DailyDialog** dataset [228] is constructed by crawling raw data from various English-learning websites. Note that DailyDialog already has 4 dialog act labels: question, inform, directive, and commissive. Our finer-grained annotation which takes social chit-chat and simple feedback into account can better cover diverse dialogue scenarios and provide extra information.

Following our definition of segment acts, we split each utterance into multiple segments using NLTK [236] sentence punctuation tokenizer [232]. The resulting segments will have their own segment act labels during annotation. Each segment is annotated by three different crowd-workers. With our special tagset design and the segmentation process, annotators can easily reach substantial agreement and deliver a high-quality dataset: Fleiss' kappa [237] achieves 0.754 for DailyDialog and 0.790 for ConvAI2. Note that the majority of the segments are labeled as question and inform. This is common in dialog act datasets [73, 78] as most of the dialogues consist of asking for information and stating facts or opinions.

Dataset Statistics and Distributions. For the ConvAI2 dataset, we collected 481,937 segment acts on the training set, and 29,232 segment acts on the validation set. Since the testing set is not publicly available, we did not annotate it.

For the DailyDialog dataset, we gathered 178,604 segment acts on the training set, 16,500 segment acts on the validation set, and 16,028 segment acts on the testing set.

Note that even though ConvAI2 and DailyDialog split their data for training, validation, and testing purposes, it is not always necessary to mechanically follow the splits. Our annotations on ConvAI2 and DailyDialog can be used as a unity, *ActDial*, depending on the

research problems.

Table 5.2 shows the distribution of all segment acts on our dataset. The segment act distribution is unbalanced. Specifically, the distribution is highly skewed to `inform` and `question`, which is not surprising because ConvAI2 and DailyDialog are chitchat datasets and the majority of communication activities are exchanging information. In addition, few written dialogues between two strangers, the setting of ConvAI2 and DailyDialog, involve an apology or encounter communicative difficulties, which results in the rare occurrences of `apology`, `backchennel-failure`, and `check-understanding` segment acts. However, it is still essential to include these segment acts as they take place more commonly in spoken dialogues in the real world.

5.1.2 FlowEval: A Segment-Act-Flow Aware Evaluation Metric

In this section, we describe the details of our proposed dialogue evaluation framework, *FlowEval*. FlowEval is implemented in three stages: segment act harvesting, retrieval, and assessment.

Segment Act Harvesting. In order to utilize the segment act flow, we first need to harvest the segment act labels for an unseen raw dialogue U. In our experiments unless specified, the segment act labels are acquired by a text classification model, based on RoBERTa-large [229] and fine-tuned on ActDial. The accuracy of this classifier is 90% on unseen data. In the end, we will have the annotated segment act flow $A_U = \{a_1, \cdots, a_i, \cdots, a_n\}$ for the dialogue U, where a_i is the segment act label for i-th segment and n is the number of segments in U.

Retrieval. For the retrieval process, FlowEval retrieves two sets of dialogues based on segment act features and content features respectively. The search space for FlowEval is our ActDial dataset and the unseen raw dialogue U serves as a query. FlowEval first extracts segment act features from a masked segment act model, and retrieves k^a nearest

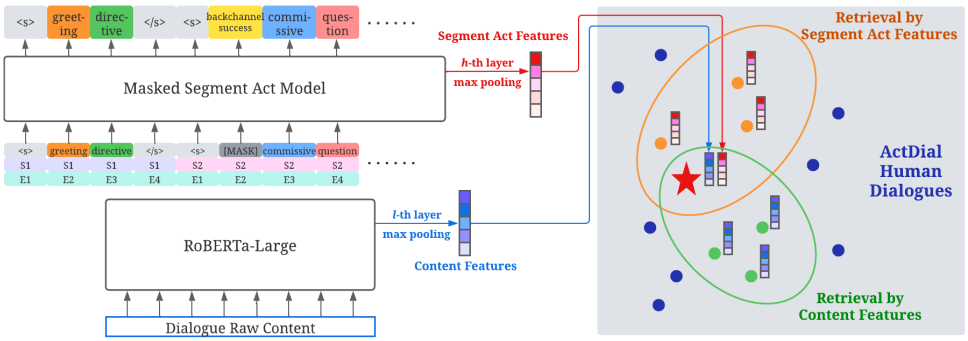


Figure 5.1: Extract segment act and content features. Retrieve closest human dialogues from ActDial dataset.

neighbors for U based on our defined similarity function. Then, FlowEval extracts content features from a RoBERTa-large model, and retrieves k^c nearest neighbours for U based on another similarity function. The final outcome of this retrieval stage is $k = k^a + k^c$ relevant dialogues for the unseen dialogue U. Figure 5.1 illustrates this process in detail.

Segment Act Flow Features. To extract segment act flow features, we treat every segment act label as a word and a segment act flow of a dialogue as a sequence. We then train a masked language model [184] called *ActBERT* on all segment act flows in our ActDial datasets. ActBERT has an accuracy of 81% for predicting the masked segment act on unseen data, which is significantly higher than guessing the majority segment act label (67%). This means that our ActBERT indeed captures reliable features from the segment act flow. ActBERT will be used to extract segment act features for any dialogue that has segment act flow.

Given a dialogue D, we first pass D's segment act flow A_D into ActBERT. The output of h-th intermediate layer of ActBERT, $H_D^h \in \mathbb{R}^{n \times d}$, will be chosen, where h is a hyper-parameter, n is the number of segments in D and d is the hidden size of ActBERT. H_D^h is then max-pooled along the n dimension to construct a fixed length vector $\bar{H}_D^h \in \mathbb{R}^d$ as the segment act feature of D.

We further employ TF-IDF features to constrain the retrieved dialogues to have a similar

topic as U. We collect the word count statistics from our ActDial dataset and compute the TF-IDF feature vector $T_D \in \mathbb{R}^v$ for any dialogue D, where v is the vocabulary size.

Having the feature set $\{\bar{H}^h_U, T_U\}$ of U and $\{\bar{H}^h_R, T_R\}$ of a human dialogue R in ActDial, we define an segment-act-based similarity metric S^a to retrieve k^a nearest neighbors $\{R_i\}_{k^a}$:

$$S^a(U, R) = (1 + \cos(\bar{H}^h_U, \bar{H}^h_R))(1 + \cos(T_U, T_R)) \tag{5.1}$$

where $\cos$ is the cosine similarity. S^a in Eq. 5.1 will only score high if R has a segment act flow as well as a topic close to U.

Content Features. Retrieval with segment act features only might miss dialogues that discussed similar contents as U but speakers communicated in a different way to U. Therefore, we retrieve from ActDial again but using features with regard to the content of U.

We use RoBERTa-large [229], a pre-trained language model, to extract the content feature of any dialogue D. We first feed the raw text of D into RoBERTa and take the l-th layer representation $L^l_D \in \mathbb{R}^{m \times d}$ of RoBERTa, where l is a hyper-parameter, m is the number of tokens in D and d is the hidden size of RoBERTa. L^l_D is then max-pooled along the m dimension to obtain a fixed-length content feature vector $\bar{L}^l_D \in \mathbb{R}^d$ for D. Having the content feature L^l_U of U and L^l_R of R in ActDial, we define a content-based similarity metric S^c for the second-round retrieval to retrieve k_c nearest neighbors $\{R_i\}_{k^c}$:

$$S^c(U, R) = \cos(L^l_U, L^l_R) \tag{5.2}$$

S^c in Eq. 5.2 will output a high score if R's content is closed to U. The final retrieved set of dialogues will be $\{R_i\}_k = \{R_i\}_{k^a} \bigcup \{R_i\}_{k^c}$.

Assessment. We define a metric to find the closest $R^* \in \{R_i\}_k$ to U by treating this small retrieved set $\{R_i\}_k$ as pseudo-references. The distance between R^* and U will be the final

score of U. Concretely, we have the following scoring function F:

$$F(U) = \max_{R \in \{R_i\}_k} wF^a(U,R) + (1-w)F^c(U,R) \tag{5.3}$$

$$F^a(U,R) = S^a(U,R) \cdot \text{BLEU}(A_U, A_R) \tag{5.4}$$

$$F^c(U,R) = \text{BERTScore}(U,R) \tag{5.5}$$

where w is a hyper-parameter between 0 and 1. Eq. 5.3 assess U from two aspects: F^a, computed by Eq. 5.4, indicates whether speakers in U interact naturally and is evaluated by ActBERT in Eq. 5.1 and BLEU score [218] of the raw segment act flow A_U; F^c, on the other hand, measures how natural sentences in U are using BERTScore [238] in Eq. 5.5.

5.1.3 Experimental Setup

In this section, we introduce evaluation datasets, baseline evaluation methods, and detailed experimental setups.

Evaluation Datasets. We evaluate the effectiveness of our metric on three benchmark datasets.

1. **Controllable Dialogue Dataset** contains the human-to-bot conversation data collected by [230]. These conversations are based on the ConvAI2 dataset [181]. We extend the original dataset by crowdsourcing segment act labels and human evaluation scores. There are 278 dialogues coming from 3 generative models. 28 dialogues are sampled randomly to form a validation set for hyperparameter tuning, while the rest make up the test set.

2. **FED Dataset** [217] contains 125 human-to-bot conversations coming from three systems. We take the mean of the 5 overall scores for each dialogue as the human

evaluation score in our experiments. We annotate all the segment act labels using the trained classifier described in Section 5.1.2.

3. **DSTC9 Dataset** [231] contains 2200 human-to-bot conversations from eleven chatbots. We take the mean of the 3 human ratings as the final score. All the segment act labels are predicted by a trained classifier.

Human Evaluation for Controllable Dialogue. We collected human judgments from Amazon Mechanical Turk (AMT). The crowd-workers are provided with the full multi-turn conversation for evaluation. We ask crowd-workers to evaluate the *relevancy*, *avoiding contradiction*, *avoiding repetition*, *persona consistency*, and *overall quality* of the conversation. The reason for designing the human evaluation on different aspects is that we assume a good conversation between humans and a dialogue system should satisfy the following properties: (1) generating relevant and non-repetitive responses (*relevancy* and *avoiding repetition*), (2) memorizing the dialogue history and generating non-contradictory information (*avoiding contradiction*), (3) maintaining a consistent persona/topic (*persona/topic consistency*), (4) formulating a natural conversation (*overall quality*).

The first four aspects are formulated as binary-choice questions, and the overall quality is formulated as a Likert question on a 1-5 scale, where higher is better. During evaluation, we did not distinguish whether an utterance is generated by a human or by the dialogue model, because we want the evaluation is about the full conversation, rather than just utterances generated by the dialogue model.

To ensure better data quality, Turkers are selected by their job success rate and geographic location (only admits turkers from English-speaking countries). Before starting our evaluation job, turkers must read through our detailed guidelines. For each dialogue, a turker is asked to evaluate the dialogue from the following perspectives:

1. **Irrelevant response (binary)**: Whether or not the speaker generates a response which seems to come out of nowhere according to the conversation history. Binary score.

2. **Contradictory information (binary)**: Whether or not the speaker generates a response which contradicts common sense or what he just said in the previous conversation. Binary score.

3. **Repetitive response (binary)**: Whether or not the speaker generates a response which has the same meaning as his previous utterance(s). Binary score.

4. **Inconsistent with persona (binary)**: Whether or not the speaker generates a response which is not consistent with his persona profile. **Only used if the dialogues-to-evaluate follow ConvAI2 setting and are generated with personas.** Binary score.

5. **Topic shifts (binary)**: Whether or not the speaker generates a response which belongs to a completely different topic compared with the previous conversation history. **Only used if the dialogues-to-evaluate follow the Daily Dialogue setting and are not generated with personas.** Binary score.

6. **Overall score (1-5)**: An overall impression of the dialogue quality, not necessary to have any relationship with the aspects above. The score is between 1 to 5 inclusive, all integers. The higher the better.

The evaluation results are examined by ourselves. Incorrect annotation would be rejected and re-evaluated by another turker. The final evaluation results are shown as Table 5.3.

	Controllable Dialogue Dataset				
	RELEVANCY	NO CONTRADICTION	NO REPETITION	CONSISTENCY	OVERALL
Seq2Seq	0.7802	0.8791	0.3846	0.9010	3.4505
Seq2Seq + Repetition	**0.8437**	**0.9062**	0.7812	0.8541	3.6250
Seq2Seq + Specificity	0.8351	0.8681	**0.8351**	**0.8681**	**3.8791**

Table 5.3: Controllable Dialogue [230] evaluation results by AMT crowd-workers.

Baseline Methods. Our method mainly compares with the following competitive baselines.

1. **FED metric** [217], leveraging the ability of DialoGPT-large [239] and the use of follow-up utterances, is an automatic and training-free evaluation method widely used by the community [231].

2. **DynaEval** [220] adopts the graph convolutional network to model dialogues, where the graph nodes are dialogue utterances and graph edges represent the relationships between utterances. DynaEval_emp and DynaEval_daily denote two variants trained on Empathetic Dialogues [183] and DailyDialog [228] respectively. DynaEval_emp reaches the best correlation on the FED dataset.

3. **Flow score** [221], considering the semantic influence of each utterance and modeling the dynamic information flow in dialogues, becomes the best evaluation method on the DSTC9 dataset.

4. **BLEU** [218] and **BERTScore** [238] are two popular reference-based metrics. The performance of BLEU and BERTScore are tested on the Controllable Dialogue dataset only, as finding suitable references is unfeasible on the FED and DSTC9 datasets.

5. **FlowEval (our method)** tune its hyperparameters on the validation set of the Controllable Dialogue dataset and directly apply to the test set of Controllable Dialogue, FED, and DSTC9. Besides, since the Controllable Dialogue dataset is constructed on top of ConvAI2 [230], we only use the DailyDialog part of ActDial for all the training and retrieval to prevent any data leakage.

Implementation of ActBERT. ActBERT follows the architecture of RoBERTa [229]. The vocabulary size is relatively small as it only contains 11 segment acts and other special tokens. It has 4 hidden layers, 4 attention heads, and a hidden dimension size of 256. Speaker information is included using different input token types. Similar to the masked language model task, we use a masked segment act task during the training.

Metric	Controllable Dialogue			FED Dataset			DSTC9 Dataset		
	Pearson	Spearman	Kendall	Pearson	Spearman	Kendall	Pearson	Spearman	Kendall
BLEU	0.132	0.136	0.104	N/A					
BERTSCORE	0.282	0.214	0.162						
CONSENSUS-BERTSCORE	0.284	0.240	0.183	0.0874*	-0.037*	-0.023*	0.060	0.054	0.039
FED Metric	-0.025*	0.010*	0.007*	0.134*	0.126*	0.088*	0.117	0.108	0.078
DynaEval_daily	0.050*	0.051*	0.039*	0.390	0.396	0.278	0.084	0.085	0.061
DynaEval_emp	0.026*	-0.007*	-0.005*	0.464	0.489	0.341	*0.029**	0.046	0.033
Flow	-0.065*	-0.029*	-0.020*	-0.073*	-0.003*	-0.002*	0.154	0.148	0.106
FlowEval (Our)	0.301	**0.256**	**0.193**	0.246	0.212	0.152	0.088	0.096	0.070
FlowEval (Our) + SOTA	**0.327**	0.250	0.190	**0.468**	**0.493**	**0.342**	**0.165**	**0.161**	**0.116**
FED Metric + SOTA	0.032*	0.058*	0.042*	0.403	0.411	0.284	0.103	0.093	0.067
DynaEval + SOTA	0.117*	0.109*	0.084*		N/A		0.054	0.059	0.042
Flow + SOTA	0.207	0.140	0.107	0.460	0.471	0.327		N/A	

Table 5.4: Correlations between different metrics and human evaluation on Controllable Dialogue (test set), FED and DSTC9 datasets. All values are statistically significant to $p < 0.05$, unless that are marked by *. SOTA refers to the previous best performing methods (except our FlowEval) in each dataset and is underlined.

Implementation of BLEU and BERTScore. Controllable Dialogue [230] are trained on the ConvAI2 dataset whose setting is two participants talking about their own personas. These unique characteristics make it feasible to find references for BLEU, BERTScore, or other reference-based metrics. We take dialogues, from the testing set of ConvAI2, that have the most overlapping personas as the references for a dialogue. Although not as convincing as references in machine translation tasks, references obtained in this way prove to be helpful to dialogue evaluation. Both BLEU and BERTScore reach relatively high correlations on Controllable Dialogue. The smooth function of the BLEU score is NIST geometric sequence smoothing [236]. BERTScore is calculated by using the package from its authors [238].

5.1.4 Result Analysis

The common practice to show the effectiveness of a dialogue evaluation metric is to calculate the Pearson, Spearman's, and Kendall correlation between human evaluation and the automatic evaluation [217, 220, 221, 222], as shown in Table 5.4. From these results, the following four conclusions can be drawn.

FlowEval Reaches Comparable Performance. Across three datasets, our FlowEval achieves the best or comparable correlations with human evaluation. On the Controllable Dialogue dataset, all baseline metrics fail to reach meaningful correlation, while FlowEval becomes the top performer. On the other two datasets, the results of FlowEval are comparable with most baselines, though the gap to the best method is obvious.

Ablation Study on Controllable Dialogues. We perform an ablation study on Controllable Dialogues and obtained positive results. This experiment is designed to reveal the effectiveness of the segment act, so content-related information and features are excluded from the whole process. Specifically, we remove the content feature and only used the segment act flow feature during the retrieval (Section 5.1.2). We later assessed each dialogue on this shrunk retrieval set. The Pearson, Spearman's, and Kendall correlations in this setting are 0.298, 0.252, and 0.189 respectively. These results decrease slightly from our full version of FlowEval (0.301, 0.256, and 0.193) but remain higher than the previous SOTA (0.282, 0.214, and 0.162). This ablation study strengthens our claim on the effectiveness of segment acts and our consensus-based framework.

Automatic Evaluation Metrics Lack Transferability. We can observe that the best method on one dataset becomes mediocre on the other datasets, including our FlowEval. FlowEval outperforms all other methods on the Controllable Dialogue dataset, but can only get to the second tier on the other two datasets. DynaEval, the best method on the FED dataset, loses its advantage when tested on other datasets. The same story also happens for the Flow score, a state-of-the-art metric in the DSTC9 dataset. This observation is consistent with a study from previous work [222]. One reason for the brittleness of these methods is that their calculations rely on large models. The data used to train these large models plays a decisive role, as we can see from the performance difference between DynaEval_emp and DynaEval_daily. In addition, FlowEval depends on the segment act labels, and these labels on the FED dataset and DSTC9 dataset are annotated by a trained classifier. Even though

the classifier has relatively high accuracy (90%), it still injects some errors into the segment act flow, which hinders the application of FlowEval on new datasets. These observations indicate that how to construct a robust dialogue evaluation metric remains a problem for the community.

FlowEval Can Provide Complementary Information to Other Methods. Similar to [222], we test different combinations of metrics by directly averaging one metric with the previous best metrics on the three datasets, which are BERTScore on the Controllable Dialogue dataset, DynaEval_emp on the FED dataset, and Flow score on DSTC9 dataset. The last 4 rows of Table 5.4 show that FlowEval can consistently push the current correlation ceiling to a new level the most, while many other combinations improve little or even hurt performance. These results imply that segment act is an important missing aspect in dialogue evaluation that is worth even further exploration in the future.

Our Consensus-Based Framework Shows Potential. In our consensus-based framework, the retrieval step of FlowEval could find pseudo-references for other reference-based metrics like BLEU [218] and BERTScore [238] and make them reference-free. Here we experiment with BERTScore, as it is the best-performing reference-based metric on Controllable Dialogue. The reference-free form of BERTScore, called *Consensus BERTScore*, is similar to our FlowEval, except that we do not employ segment act features in the retrieval step and we exclude the segment act score, i.e., Eq. 5.4, in the assessment step. As shown in the third row of Table 5.4, Consensus BERTScore slightly outperforms BERTScore in all three correlations (0.284 vs. 0.282, 0.240 vs. 0.214, 0.183 vs. 0.162). This promising result shows the potential of our consensus-based framework. It leads a new way to rethink the usability of reference-based metrics in dialogue evaluation.

What Does Segment Act Bring to Dialogue Evaluation? Compared with semantic-meaning-focused metrics, what does segment act bring to dialogue evaluation? We hypothe-

Metric 1	Metric 2	Pearson	Spearman	Kendall
FlowEval_seg	Human	0.191	0.151	0.113
BLEU	Human	0.132	0.136	0.104
BERTScore	Human	0.282	0.214	0.162
FlowEval_seg	BERTScore	0.014*	-0.030*	-0.022*
FlowEval_seg	BLEU	0.067*	0.042*	0.026*
BLEU	BERTScore	0.576	0.637	0.460

Table 5.5: Inter-correlations between FlowEval_seg, BERTScore, BLEU, and human evaluation. FlowEval_seg is a version of FlowEval using segment act flow only for assessment. All values are statistically significant to $p < 0.05$, unless that are marked by *.

size the explicit involvement of segment acts can bring useful information, complementary to semantic-meaning-focused metrics.

We illustrate our hypothesis in Figure 5.2. If segment act is useful, the segment-act-based evaluation $\boldsymbol{v}_p$ should be positively correlated to human evaluation $\boldsymbol{v}_o$, i.e., $\boldsymbol{v}_p$ has roughly the same direction as $\boldsymbol{v}_o$ but with a small angle θ_2. If segment act is complementary to semantic-meaning-focused metrics, the segment-act-based evaluation $\boldsymbol{v}_p$ should be almost orthogonal to the semantic-meaning-focused evaluation $\boldsymbol{v}_m$, i.e., $\boldsymbol{v}_m$ falls into the other side of $\boldsymbol{v}_o$ so that $\boldsymbol{v}_m$ is also positively correlated to $\boldsymbol{v}_o$ with a small angle θ_1 but almost orthogonal to $\boldsymbol{v}_p$ with a large angle $\theta_3 = \theta_1 + \theta_2$. These angles θ_1, θ_2 and θ_3 could be characterized by the correlation of two evaluation results. A higher correlation implies a smaller angle.

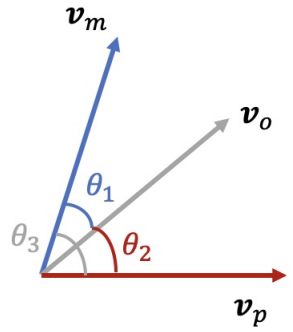


Figure 5.2: The relationships, in our hypothesis, between human evaluation $\boldsymbol{v}_o$, semantic-meaning-focused evaluation $\boldsymbol{v}_m$, and segment-act-based evaluation $\boldsymbol{v}_p$.

We conduct experiments on the test set of the Controllable Dialogue dataset to validate our hypothesis. Two of the popular semantic-meaning-focused metrics are BERTScore [238] and BLEU [218]. We modify the retrieval and assessment parts of our FlowEval, so that only segment act information is utilized. We denote this variant as *FlowEval_seg*.

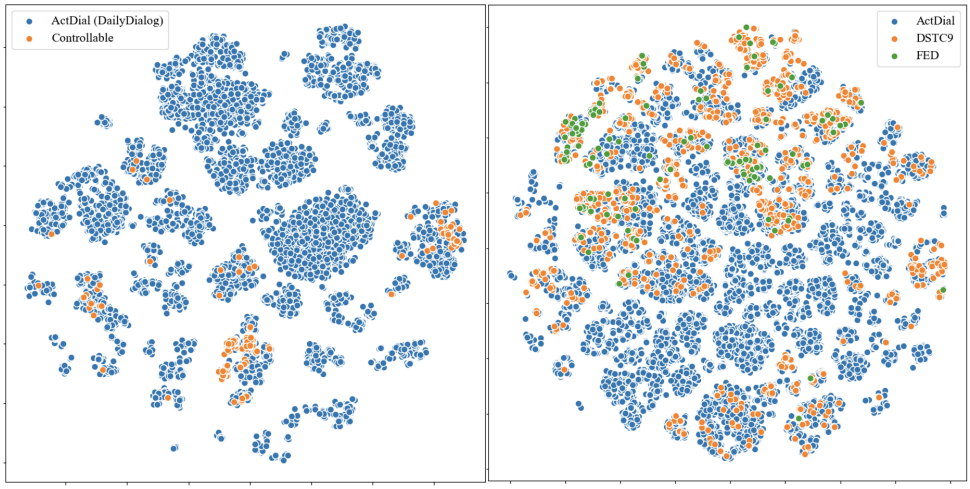


Figure 5.3: Segment act feature space of Controllable Dialogue, FED, DSTC9 dataset and the retrieval set ActDial. We have a separate plot for Controllable Dialogue because the ActDial we used are different (See Section 5.1.3).

As we can observe from the first three rows of Table 5.5 the FlowEval_seg, BLEU, and BERTScore all exhibit strong correlation to human evaluation. Unsurprisingly, BLEU and BERTScore are highly correlated (the last row of Table 5.5), since both of them focus on the semantic meaning of dialogues. In line with our hypothesis, the BLEU-FlowEval_seg correlation and BERTScore-FlowEval_seg correlation are far smaller (rows 4-5 of Table 5.5), which indirectly shows that segment act can evaluate dialogues from a complementary perspective. These findings resonate with the theory from [240], where the *meaning* and the *communicative intent*, i.e., segment act here, are considered to be two decoupled and complementary dimensions.

Why Does Consensus Work? We investigate why the consensus-based framework can perform well in dialogue evaluation by visualizing the segment act feature space, an essential aspect in the retrieval process of FlowEval. We compare the segment act feature distribution between the three test sets and their corresponding retrieval sets, projecting these features to 2-dimensional space by t-SNE [241] as shown in Figure 5.3. We did not tune any hyperparameter to obtain these results, in consideration of the sensitivity of t-SNE plots.

The core idea of consensus lies in using the nearest neighbors as references to measure

a newcomer. Only if the suitable nearest neighbors consistently exist, will the consensus of them have a meaningful indication to evaluate a new subject. We can observe from Figure 5.3 that, even though dialogues in three test sets are diverse, every datapoint from the test sets is surrounded by data points from the retrieval sets. We can always reliably find good references for a new dialogue, which explains why using consensus in dialog evaluation is promising. Moreover, this desirable coverage is achieved by an attainable amount of data points. It only needs 10,494 and 31,993 dialogues as retrieval sets in our experiments to get good results. The power of the consensus may become stronger and more reliable if the size of the retrieval set grows, which could be a favorable property in many of industrial applications.

5.2 Identifying Edit Intentions for Iterative Text Revisions

Writing is a complex and effortful cognitive task, where writers balance and orchestrate three distinct cognitive processes: planning, translation, and revising [242]. These processes can be hierarchical and recursive and can occur at any moment during writing. This work focuses on text revision as an essential part of writing [243]. Revising text is a strategic, and adaptive process. It enables writers to deliberate over and organize their thoughts, find a better line of argument, learn afresh, and discover what was not known before [244]. Specifically, text revision involves identifying discrepancies between intended and instantiated text, deciding what edits to make, and how to make those desired edits [245, 246, 247].

Text revision is an *iterative* process. Human writers are unable to simultaneously comprehend multiple demands and constraints of the task when producing well-written texts [248, 249, 250] – for instance, expressing ideas, covering the content, following linguistic norms and discourse conventions of written prose, etc. Thus, they turn towards making *successive iterations of revisions* to reduce the number of considerations at each time.

Previous works on iterative text revision have three major limitations: (1) simplifying the task to an noniterative ”original-to-final” text paraphrasing; (2) focusing largely on

103.14972v2	Each comment was annotated by three different annotators, which achieved high inter-annotator agreement. The proposed annotation {~~process~~ approach} CLARITY is also language and domain independent{, nevertheless, it was currently applied for Brazilian Portuguese} MEANING-CHANGED .
103.14972v3	Each comment was annotated by three different annotators, {~~which~~ and} COHERENCE achieved high inter-annotator agreement. The {new} MEANING-CHANGED proposed annotation approach is also language and {~~domain independent, nevertheless, it was currently~~domain-independent (although it has been} CLARITY applied for Brazilian Portuguese{)} FLUENCY .
103.14972v4	Each comment was annotated by three different annotators {~~,~~} FLUENCY and achieved high inter-annotator agreement. The {~~new~~} COHERENCE proposed annotation approach is also language and domain-independent {~~(although it has been applied~~ nevertheless it is currently customized} COHERENCE for Brazilian Portuguese {~~)~~} FLUENCY .

Table 5.6: An iteratively revised ArXiv abstract snippet (2103.14972, version 2, 3, and 4) with our annotated edit-intention in ITERATER.

sentence-level editing [86, 87, 82, 85]; (3) developing editing taxonomies within individual domains (e.g. Wikipedia articles, academic writings) [80, 81, 83]. These limitations make their proposed text editing taxonomies, datasets, and models lose their generalizability and practicality.

We present ITERATER— an annotated dataset for ITERAtive TExt Revision that consists of 31,631 iterative document revisions with sentence-level and paragraph-level edits across multiple domains of formally human-written text, including Wikipedia[1], ArXiv[2] and Wikinews.[3] Table 5.6 shows a sample ArXiv document in ITERATER, that underwent iterative revisions. Our dataset includes 4K manually annotated and 196K automatically annotated edit intentions based on a sound taxonomy we developed, and is generally applicable across multiple domains and granularities (presented in Table 5.7). Note that ITERATER is currently only intended to support formal writing revisions, as iterative revisions are more prevalent in formal rather than informal writings (e.g. tweets, chit-chats).

Our contributions are as follows:

[1] https://www.wikipedia.org/
[2] https://arxiv.org/
[3] https://www.wikinews.org/

Dataset	Size	Domain	Granularity	Revision History	Annotations
WikiRevisions [80]	5K	Wiki	Paragraph-level	×	√
WikiHowToImprove [83]	2.7M	Wiki	Sentence-level	√	×
ArgumentWriting [81]	180	Academic	Sentence-level	√	√
NewsEdits [251]	4.6M	News	Sentence-level	√	×
ITERATER (Ours)	31K	All above	Sentence-level&Paragraph-level	√	√

Table 5.7: Comparisons with previous related works.

1. Formulating the iterative text revision task in a more comprehensive way, capturing greater real-world challenges such as successive revisions, multi-granularity edits, and domain shifts.

2. Collecting and releasing a large, multi-domain Iterative Text Revision dataset: ITERATER, which contains 31K document revisions from Wikipedia, ArXiv and Wikinews, and 4K edit actions with high-quality edit intention annotations.

3. Analyzing how text quality evolves across iterations and how it is affected by different kinds of edits.

4. Showing that incorporating the annotated edit-intentions is advantageous for text revision systems to generate better-revised documents.

5.2.1 Task Formulation

We provide formal definitions of the Iterative Text Revision task, and its building blocks.

Edit Action. An edit action $\boldsymbol{a}_k$ is a local change applied to a certain text object, where k is the index of the current edit action. The local changes include: insert, delete and modify. The text objects include: token, phrase[4], sentence, and paragraph. This work defines local changes applied to tokens or phrases as *sentence-level edits*, local changes applied to sentences as *paragraph-level edits* and local changes applied to paragraphs as *document-level edits*.

[4]In this work, we define phrase as text pieces which contain more than one token and only appears within a sentence.

Edit Intention. An edit intention $\boldsymbol{e}_k$ reflects the revising goal of the editor when making a certain edit action. In this work, we assume each edit action $\boldsymbol{a}_k$ will only be labeled with one edit intention $\boldsymbol{e}_k$. We further describe our edit intention taxonomy in Table 5.9 and §5.2.2.

Document Revision. A document revision is created when an editor saves changes for the current document [252, 80]. One revision $\mathcal{R}^t$ is aligned with a pair of documents $(\mathcal{D}^{t-1}, \mathcal{D}^t)$ and contains K^t edit actions, where t indicates the version of the document and $K^t \geq 1$. A revision with K^t edit actions will correspondingly have K^t edit intentions:

$$(\mathcal{D}^{t-1}, \mathcal{D}^t) \rightarrow \mathcal{R}^t = \{(\boldsymbol{a}_k^t, \boldsymbol{e}_k^t)\}_{k=1}^{K^t} \tag{5.6}$$

We define t as the revision depth.

Iterative Text Revision. Given a source text $\mathcal{D}^{t-1}$, iterative text revision is the task of generating revisions of text $\mathcal{D}^t$ at depth t until the quality of the text in the final revision satisfies a set of pre-defined stopping criteria $\{s_0, ..., s_M\}$:

$$\mathcal{D}^{t-1} \xrightarrow{g(\mathcal{D})} \mathcal{D}^t, \text{if } f(\mathcal{D}^t) < \{s_0, ..., s_M\} \tag{5.7}$$

where $g(\mathcal{D})$ is a text revision system and $f(\mathcal{D})$ is a quality evaluator of the revised text. The quality evaluator $f(\mathcal{D})$ can be automatic systems or manual judgments which measure the quality of the revised text. The stop criteria $\{s_i\}$ is a set of conditions that determine whether to continue revising or not. In this work, we simply set them as revision depth equal to 10, and edit distance between $\mathcal{D}^{t-1}$ and $\mathcal{D}^t$ equal to 0 (§5.2.4). We will include other criteria which measures the overall quality, content preservation, fluency, coherence, and readability of the revised text in future works.

5.2.2 ITERATER Dataset Construction

Raw Data Collection. We select three domains – Wikipedia articles, academic papers, and news articles – to cover different human writing goals, formats, revision patterns, and quality standards. The three domains consist of formally written texts, typically edited by multiple authors. We describe why and how we collect text revision from each domain below:

- *Scientific Papers:* Scientific articles are written in a rigorous, logical manner. Authors generally highlight and revise their hypotheses, experimental results, and research insights in this domain. We collect paper abstracts submitted at different timestamps (i.e., version labels) from ArXiv.
- *Wikipedia Articles:* Encyclopedic articles are written in a formal, coherent manner, where editors typically focus on improving the clarity and structure of articles to make people easily understand all kinds of factual and abstract encyclopedic information. We collect revision histories of the main contents of Wikipedia articles.
- *News Articles:* News articles are generally written in a precise and condensed way. News editors emphasize improving the clarity and readability of news articles to keep people updated on rapidly changing news events. We collect revision histories of news content from Wikinews.

Raw Data Processing. We first collect all raw documents, then sort each document version according to its timestamp in ascending order. For each document $\mathcal{D}$, we pair two consecutive versions as one revision $(\mathcal{D}^{t-1}, \mathcal{D}^t) \rightarrow \mathcal{R}^t$, where t is the revision depth. For each sampled document-revision $\mathcal{R}^t$, we extract its full edit actions using *latexdiff*.[5] We provide both the paragraph-level and sentence-level revisions where the latter is constructed by applying a sentence segmentation tool,[6] and aligning each sentence to each revision. For each revision pair, we have: the revision type, the document id, the revision depth, an

[5] https://www.ctan.org/pkg/latexdiff
[6] https://github.com/zaemyung/sentsplit

	ITERATER-FULL						ITERATER-HUMAN					
	ArXiv		Wikipedia		Wikinews		ArXiv		Wikipedia		Wikinews	
Depth	#D	#E	#D	#E	#D	#E	#D	#E	#D	#E	#D	#E
1	9,446	65,450	8,195	51,290	7,878	39891	95	618	130	1,072	173	1,227
2	1,615	11,391	1,991	12,868	1,455	8,116	76	499	38	250	25	155
3	301	2,076	415	2,786	161	1,704	6	47	10	98	4	27
4	66	444	64	723	16	71	1	13	1	12	0	0
5	15	107	9	52	4	18	0	0	0	0	0	0
Total	11,443	79,468	10,674	67,719	9,514	49,800	178	1,177	179	1,432	202	1,409

Table 5.8: Statistics of the ITERATER dataset, where #D indicate the number of document revisions ($\mathcal{R}^t$), and #E indicate the number of annotated edit actions.

original phrase, and a revised phrase, respectively.[7]

For Wikipedia and Wikinews, we use the MediaWiki Action API[8] to retrieve raw pages updated at different timestamps. For each article, we start from July 2021 and trace back to its five most recent updated versions. Then, we parse[9] plain texts from raw wiki-texts and filter out all references and external links.For Wikipedia, we retrieve pages under the categories listed on the main category page [10]. For Wikinews, we retrieve pages listed on the published articles page[11].

For ArXiv, we use the ArXiv API[12] to retrieve paper abstracts. Note that we do not retrieve the full paper for two reasons: (1) some paper reserved their copyright for distribution, (2) parsing and aligning editing actions in different document types (e.g. pdf, tex) is challenging. For each paper, we start from July 2021 and retrieve all its previous submissions. We collect papers in the fields of Computer Science, Quantitative Biology, Quantitative Finance, and Economics.

In summary, we collect 31,631 document revisions with 196,987 edit actions, and maintain a relatively balanced distribution across three domains, as shown in Table 5.8. We call this large-scale dataset as ITERATER-FULL-RAW.

[7]We also record character-level indices of their positions within the original sentence and the paragraph.
[8]https://www.mediawiki.org/wiki/API:Main_page
[9]https://github.com/earwig/mwparserfromhell
[10]https://en.wikipedia.org/wiki/Wikipedia:Contents/Categories
[11]https://en.wikinews.org/wiki/Category:Published
[12]https://arxiv.org/help/api/

Edit-Intention	Description	Example	Counts (Ratio)
FLUENCY	Fix grammatical errors in the text.	She went to the ~~markt~~market.	942 (23.44%)
COHERENCE	Make the text more cohesive, logically linked and consistent as a whole.	She works hard~~. She~~; therefore, she is successful.	393 (9.78%)
CLARITY	Make the text more formal, concise, readable and understandable.	The changes ~~made the paper better than before~~improved the paper.	1,601 (39.85%)
STYLE	Convey the writer's writing preferences, including emotions, tone, voice, etc..	Everything was awfully rotten.	128 (3.19%)
MEANING-CHANGED	Update or add new information to the text.	This method improves the model accuracy from 64% to ~~78~~83%.	896 (22.30%)
OTHER	Edits that are not recognizable and do not belong to the above intentions.	This method is also named as ~~CITATION1~~.	58 (1.44%)

Table 5.9: A taxonomy of edit intentions in ITERATER, where Fluency, Coherence, Clarity and Style belong to Non-Meaning-changed edits.

Data Annotation Overview. To better understand the human revision process, we sample 559 document revisions from ITERATER-FULL-RAW, consisting of 4,018 human edit actions. We refer to this small-scale unannotated dataset as ITERATER-HUMAN-RAW. Then, we use Amazon Mechanical Turk (AMT) to crowdsource edit intention annotations for each edit action according to our proposed edit-intention taxonomy. We refer to this small-scale annotated dataset as ITERATER-HUMAN.We then scale these manual annotations to ITERATER-FULL-RAW by training edit intention prediction models on ITERATER-HUMAN, and automatically label ITERATER-FULL-RAW to construct ITERATER-FULL.

Edit Intention Taxonomy. For manual annotations, we propose a new edit intention taxonomy in ITERATER (Table 5.9), in order to comprehensively model the iterative text revision process. Our taxonomy builds on prior literature [253, 254]. At the highest level, we categorize the edit intentions into ones that change the meaning or the information contained in the text (MEANING-CHANGED), and ones that preserve these characteristics (NON-MEANING-CHANGED). Since our goal is to understand edit intentions to improve the quality of writing, we focus on categorizing edits in the latter category further into four sub-categories: FLUENCY, CLARITY, COHERENCE and STYLE. Our proposed taxonomy

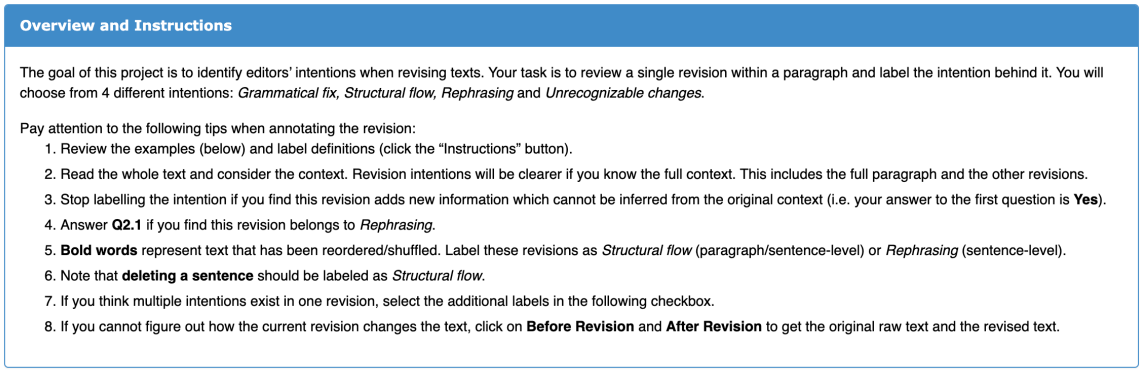


Figure 5.4: A screenshot of the annotation instruction for human annotators.

of edit intentions is generally applicable to multiple domains, edit-action granularities (sentence-level and paragraph-level), and revision depths. We also propose the OTHER category for edits that cannot be labeled using the above taxonomy.

Manual Annotation Instruction and Interface. To guide human annotators to making accurate edit-intention annotations, we provide them with a short task instruction (Figure 5.4) followed by some concrete edit-intention examples (Figure 5.5). Then, we highlight the edit-action within the document-revision and ask human annotators three questions to obtain the accurate edit-intention of the current edit-action, as illustrated in Figure 5.6. Note that in our previous test runs on AMT, we find that AMT workers can hardly reach a consensus on Clarity and Style edits, which give a very low IAA score. Therefore, in the annotation interface, we include Clarity and Style edits under the category of "Rephrasing", and further ask the annotators to judge whether the current "Rephrasing" edit is making the text more clearer and understandable. If yes, we convert this edit to Clarity, otherwise, we convert this edit to Style. This interface configuration gives us the best IAA score among our 5 test runs.

Manual Annotation Procedure. Since edit intention annotation is a challenging task, we design strict qualification tests to select 11 qualified AMT annotators. First, we prepare a small test set with 67 edit-actions and deploy parallel test runs on AMT to get more workers to participate in this task. Before starting the annotation, workers are required to pass a

Examples

Intention	Definition	Explanation	Example
Grammatical fix	Sentence-level revisions which fix spelling and grammatical errors.	Fix spelling	She went to the {markt=>market}.
		Fix grammar error	She is planning to study ~~at~~ tonight.
		Fix grammar error	She {catched=>caught} the ball.
Structural flow	Sentence/Paragraph/Document-level revisions which make sentences logically connected and make the document more organized, including modifying transition words, splitting sentences, merging sentences, reordering sentences, etc.	Add transitions	Humpty Dumpty sat on a wall. He, therefore, had a great fall.
		Delete transitions	Smith (2019) argues for the former hypothesis. ~~Moreover,~~ Baldwin (2018) argues for the former hypothesis.
		Modify transitions	It started out quite chaotic. {Moreover=>Finally}, it all fell into place.
		Split a sentence	She worked hard throughout her early life {and, through=>.Through} patience and perseverance, became a successful lawyer.
		Merge two or more sentences	He works hard {. H=>; therefore, h}e is successful.
		Delete a sentence	~~I will not be able to meet you.~~ Sorry for missing the party. I have been busy this week.
Rephrasing	Sentence-level revisions which rewrite the text *without* adding new information.	Use less diverse vocabulary, and simpler grammar and structure	Editors help {optimize=>improve} people's communication. The changes {made the paper better than before=>improved the paper}.
		Be concise and direct, state something in the least number of words	This task can be the building block of applications ~~that requires a deep understanding of the language~~ such as fake news detection and medical claim verification.
		Change tense from past to present and vice versa	In our work, we {tried=>try} to prove a new theory. We {conducted=>conduct} massive experiments on benchmark datasets.
		Change tone from formal to informal and vice versa	Let's {meet=>chill} on Sunday afternoon.
		Change tone from neutral to emotional and vice versa	They {focus=>exclusively focus} on text-level manipulation. The food in this restaurant {sucks=>is not delicious}.

Figure 5.5: A screenshot of the provided examples for human annotators.

qualification test which has 5 test questions to get familiar with our edit-intention taxonomy. Second, we compare workers' annotations with our golden annotations, and select workers who have an accuracy over 0.4. After 5 test runs, we select 11 AMT workers who are qualified to participate in this task. Then, we deploy the full 4K edit-actions on AMT, and collect 3 human annotations per edit-action.

To further improve the annotation quality, we ask another group of expert linguists (English L1, bachelor's or higher degree in Linguistics) to re-annotate the edits which do not have a majority vote among the AMT workers. Finally, we take the majority vote among 3 human annotations (either from AMT workers or from expert linguists) as the final edit intention labels. This represents the ITERATER-HUMAN dataset. We release both the final majority vote and the three raw human annotations per edit action as part of the dataset.

Automatic Annotation. To scale up the annotation, we train an edit-intention classifier to annotate ITERATER-FULL-RAW and construct the ITERATER-FULL dataset. We split the ITERATER-HUMAN dataset into 3,254/400/364 training, validation and test pairs. The edit

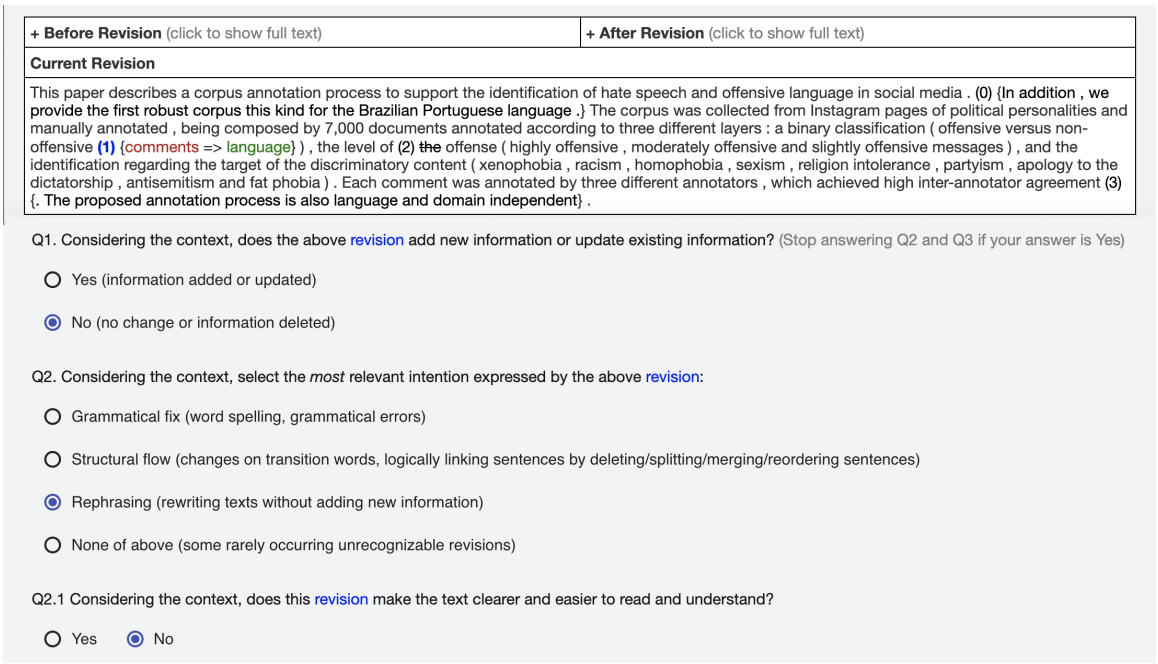


Figure 5.6: A screenshot of the annotation interface for human annotators.

Edit-Intention	Precision	Recall	F1
CLARITY	0.75	0.63	0.69
FLUENCY	0.74	0.86	0.80
COHERENCE	0.29	0.36	0.32
STYLE	1.00	0.07	0.13
MEANING-CHANGED	0.44	0.69	0.53

Table 5.10: Edit intention classifier performance on the test split of ITERATER-HUMAN.

intention classifier is a RoBERTa-based [161] multi-class classifier that predicts an intent given the original and the revised text for each edit action. We leverage the RoBERTa-large model from Huggingface transformers [124], which has 354 million parameters. For training, we set the total training epoch to 15 and batch size to 4. We use the Adam optimizer with weight decay [164], and set the learning rate to 10^{-5} which decreases linearly to 0 at the last training iteration.

For evaluation, we report descriptive statistics with a single run. We use the sklearn package [255] to calculate the precision, recall, and f1 score. Table 5.10 shows its performance on the test set. The Fluency and Clarity edit intentions are easy to predict with F1 scores of 0.8 and 0.69, respectively, while Style and Coherence edit intentions are harder to

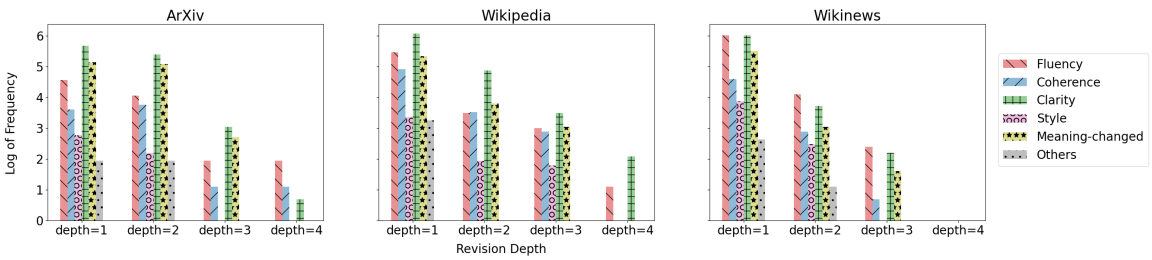


Figure 5.7: Logarithm (base e) of frequency for edit-intentions in each revision depth for the three dataset domains.

predict with F1 scores of 0.13 and 0.32, respectively, largely due to the limited occurrence of Style and Coherence intents in the training data.

Edit Intention Distributions. The iterative edit intention distributions in three domains are demonstrated in Figure 5.7. Across all three domains, authors tend to make the majority of edits at revision depth 1. However, the number of edits rapidly decreases at revision depth 2, and few edits are made at revision depth 3 and 4.

We find that CLARITY is one of the most frequent edit intentions across all domains, indicating that authors focus on improving readability across all domains. For ArXiv, MEANING-CHANGED edits are also among the most frequent edits, which indicates that authors also focus on updating the contents of their abstracts to share new research insights or update existing ones. Meanwhile, ArXiv also covers many FLUENCY and COHERENCE edits, collecting edits from scientific papers and suggesting meaningful revisions would be an important future application of our dataset. For Wikipedia, we find that FLUENCY, COHERENCE, and MEANING-CHANGED edits roughly share a similar frequency, which indicates Wikipedia articles have more complex revision patterns than ArXiv and news articles. For Wikinews, FLUENCY edits are equally emphasized, indicating that improving the grammatical correctness of the news articles is just as important.

Inter-Annotator Agreement. We measure inter-annotator agreement (IAA) using Fleiss' κ [237]. Table 5.11 shows the IAA across three domains. After the second round of re-

	ArXiv	Wikipedia	Wikinews	All
1st-round	0.3369	0.3630	0.3886	0.3628
2nd-round	0.4983	0.4274	0.5601	0.5014

Table 5.11: Inter-annotator agreement (Fleiss' κ [237]) across two rounds of annotations, where the 1st-round only contains annotations from qualified AMT workers, and the 2nd-round contains annotations from both qualified AMT workers and expert linguists.

annotation by proficient linguists, Fleiss' κ increases to 0.5014, which indicates moderate agreement among annotators.

We further look at the raw annotations where at least 1 out of 3 annotators assigns a different edit intention label. We find that the COHERENCE intention is the one that is the most likely to have a disagreement: 312 out of 393 COHERENCE annotations do not have consensus. Within those disagreements of the COHERENCE intention, 68.77% are considered to be CLARITY, and 11.96% are considered to be the FLUENCY intention. Annotators also often disagree on the CLARITY intention, where 1023 out of 1601 CLARITY intentions do not have a consensus. Among those disagreements of the CLARITY intention, 30.33% are considered to be COHERENCE, and 30.23% are considered to be STYLE.

The above findings explain why the inter-annotator agreement scores are lower in Wikipedia and ArXiv. As shown in Figure 5.7, Wikipedia has many COHERENCE edits while ArXiv has many CLARITY edits. This explains the difficulty of the edit intention annotation task: it not only asks annotators to infer the edit intention from the full document context, but also requires annotators to have a wide range of domain-specific knowledge in scientific writings.

5.2.3 Understanding Iterative Text Revisions

To better understand how text revisions affect the overall quality of documents, we conduct both manual and automatic evaluations on a sampled set of document revisions.

Evaluation Data. We sample two sets of text revisions for different evaluation purposes. The first set contains 21 iterative document revisions, consisting of 7 unique documents, each

document having 3 document revisions from revision depth 1 to 3. The second set contains 120 text pairs, each associated with exactly one edit intention of FLUENCY, COHERENCE, CLARITY, or STYLE. We validate the following research questions:

RQ1 How do human revisions affect the text quality across revision depths?

RQ2 How does text quality vary across edit intentions?

Manual Evaluation Configuration. We hire a group of proficient linguists to evaluate the overall quality of the documents/sentences, where each revision is annotated by 3 linguists. For each revision, we randomly shuffle the original and revised texts, and ask the evaluators to select which one has better overall quality. They can choose one of the two texts, or neither. Then, we calculate the score for the overall quality of the human revisions as follows: -1 means the revised text has worse overall quality than the original text; 0 means the revised text do not show a better overall quality than the original text, or cannot reach agreement among 3 annotators; 1 means the revised text has better overall quality than the original text.

Automatic Evaluation Configuration. We select four automatic metrics to measure the document quality on four different aspects: fluency, coherence readability, and content preservation. For **Fluency**, we use the Syntactic Log-Odds Ratio (SLOR) [256] to evaluate the naturalness and grammaticality of the current revised document, where a higher SLOR score indicates a more fluent document. Prior works [257, 256] found word-piece log-probability correlates well with human fluency ratings. For **Coherence**, we use the Entity Grid (EG) score [258] to evaluate the local coherence of the current revised document, where a higher EG score indicates a more coherent document. EG is a widely adopted [259, 260, 261] metric for measuring document coherence. For **Readability**, we use the Flesch–Kincaid Grade Level (FKGL) [262] to evaluate how easy the current revised document is for the readers to understand, where a lower FKGL indicates a more readable document. FKGL is a popular metric that has been used by many prior works [263, 264, 265, 266, 267] to

t	**Overall** ↑	**BLEURT**↑	**ΔSLOR** ↑	**ΔEG** ↑	**ΔFKGL** ↓
1	0.4285	0.1982	-0.0985	-0.0132	-1.0718
2	0.4285	0.1368	-0.1025	-0.0295	-2.4973
3	0.1428	-0.0224	-0.0792	0.0278	1.8131

Table 5.12: Evaluation results for 21 iterative document revisions, where t indicates the revision depth. Note that ΔSLOR, ΔEG and ΔFKGL are computed by subtracting the scores of original documents from the scores of revised documents. Overall is the manual evaluation of overall quality of the revised documents.

FLUENCY	COHERENCE	CLARITY	STYLE
0.3673	0.1500	0.2800	-0.0385

Table 5.13: Manually evaluated text quality of 120 single sentence-level edits for different edit intentions.

measure the readability of documents. For **Content Preservation**, we use the BLEURT score [70] to measure how much content has been changed from the previous document to the current revised one, where a higher BLEURT score indicates more content has been preserved. BLEURT has been shown to correlate better with human judgments than other metrics that take semantic information into account, e.g. METEOR [166] or BERTScore [268]. However, in our following experiments, we find these existing automatic metrics are poorly correlated with manual evaluations.

RQ1: Iterative Revisions vs. Quality. Table 5.12 shows the document quality changes at different revision depths. Generally, human revisions improve the overall quality of original documents, as indicated by the overall score at each revision depth.However, the overall quality keeps decreasing as the revision depth increases from 1 to 3, likely because it is more difficult for evaluators to grasp the overall quality in the deeper revision depths in the pair-wise comparisons between the original and revised documents, because less NON-MEANING-CHANGED edits have been conducted in deeper revision depths. For automatic metrics, we find ΔSLOR and ΔEG are not well-aligned with the human overall score, we further examine whether human revisions make original documents less fluent and less coherent in the analysis of RQ2.

RQ2: Edit Intentions vs. Quality. Table 5.13 shows how text quality varies across edit intentions. We find that FLUENCY and COHERENCE edits indeed improve the overall quality of original sentences according to human judgments. This finding suggests that ΔSLOR and ΔEG are not well-aligned with human judgments, and calls for the need to explore other effective automatic metrics to evaluate the fluency and coherence of revised texts. Besides, we observe that STYLE edits degrade the overall quality of original sentences. This observation also makes sense since STYLE edits reflect the writer's personal writing preferences (according to our edit intention taxonomy in Table 5.9), which is not necessarily improve the readability, fluency or coherence of the text.

5.2.4 Modeling Iterative Text Revisions

To better understand the challenges of modeling the task of iterative text revisions, we train different types of text revision models using ITERATER.

Text Revision Models. For training the text revision models, we experiment with both edit-based and generative models. For the edit-based model, we use FELIX [269], and for the generative models, we use BART [69] and PEGASUS [270]. FELIX decomposes text revision into two sub-tasks: Tagging, which uses a pointer mechanism to select the subset of input tokens and their order; and Insertion, which uses a masked language model to fill in missing tokens in the output not present in the input. BART and PEGASUS are Transformer-based encoder-decoder models which are used in a wide range of downstream tasks such as natural language inference, question answering, and summarization.

Training. We use four training configurations to evaluate whether edit intention information can help better model text revisions. The first configuration uses the pure revision pairs without edit intention annotations (ITERATER-HUMAN-RAW dataset). In the second configuration, we include the manually annotated edit intentions to the source text (ITERATER-HUMAN dataset). Similarly, for the third and fourth training configurations, we

Model	Dataset	SARI	BLEU	ROUGE-L	Avg.
FELIX	HUMAN-RAW	29.23	49.48	63.43	47.38
FELIX	HUMAN	30.65	54.35	59.06	48.02
FELIX	FULL-RAW	30.34	55.10	56.49	47.31
FELIX	FULL	**33.48**	**61.90**	**63.72**	**53.03**
BART	HUMAN-RAW	33.20	**78.59**	85.20	65.66
BART	HUMAN	34.77	74.43	84.45	64.55
BART	FULL-RAW	33.88	78.55	86.05	66.16
BART	FULL	**37.28**	77.50	**86.14**	**66.97**
PEGASUS	HUMAN-RAW	33.09	**79.09**	86.77	66.32
PEGASUS	HUMAN	34.43	78.85	86.84	66.71
PEGASUS	FULL-RAW	34.67	78.21	**87.06**	66.65
PEGASUS	FULL	**37.11**	77.60	86.84	**67.18**
Baseline	-	29.47	81.25	88.04	66.25

Table 5.14: Model performances on the test set of ITERATER-HUMAN. Baseline refers to a no-edit baseline, where we simply use the input text as the output. **Avg.** is the average score of **SARI**, **BLEU** and **ROUGE-L**.

use ITERATER-FULL-RAW dataset (no edit intention information) and ITERATER-FULL dataset (automatically annotated labels, as described in §5.2.2, simply appended to the input text). We use these four configurations for all model architectures.

We leverage the BART-large (with 400 million parameters) and PEGASUS-large (with 568 million parameters) from Huggingface transformers [124]. We set the total training epoch to 5 and batch size to 16. We use the Adam optimizer with weight decay [164], and set the learning rate to 3×10^{-5} which decreases linearly to 0 at the last training iteration. We report descriptive statistics with a single run. We use the metrics package from Huggingface transformers to calculate the SARI, BLEU, and ROUGE-1/2/L score.

Automatic Evaluation. Table 5.14 shows the results of the three models for our different training configurations. Following prior works [271, 272, 269], we report SARI, BLEU, and ROUGE-L metrics. It is noteworthy that the SARI score on the no-edit baseline is the lowest, which indicates the positive impact of revisions on document quality, as also corroborated by the human evaluations in §5.2.3. For both ITERATER-HUMAN and ITERATER-FULL datasets, we see that edit intention annotations help to improve the performance of both

	Human Revision	Tie	Model Revision
Overall	83.33%	10.00%	6.67%
Content	13.33%	70.00%	16.67%
Fluency	50.00%	50.00%	0.00%
Coherence	40.00%	56.67%	3.33%
Readability	86.67%	10.00%	3.33%

Table 5.15: Manual pair-wise comparison for 30 single document revisions without Meaning-changed edits.

t	Human Revisions	Tie	Model Revisions
1	57.14%	14.28%	28.58%
2	57.14%	14.28%	28.58%
3	42.85%	57.15%	0.00%

Table 5.16: Manual pair-wise comparison for overall quality of 21 iterative document-revisions, where t indicates the revision depth.

FELIX and PEGASUS. Also, both models perform better on the larger ITERATER-FULL dataset compared to the ITERATER-HUMAN dataset, showing that the additional data (and automatically-annotated annotations) are helpful.

Manual Evaluation. Table 5.15 shows how the model revision affects the quality of the original document. We choose PEGASUS trained on ITERATER-FULL to generate revisions and compare with human revisions, as the model produces the best overall results.

First, we evaluate how model revisions affect the quality of the document. We randomly sample 30 single document-revisions which do not contain Meaning-changed edits, and input the original documents to the best-performing model to get the model-revised documents. Then, for each data pair, we randomly shuffle model revisions and human revisions, and ask human evaluators to select which revision leads to better document quality in terms of:

- *Content Preservation*: keeping more content information unchanged;
- *Fluency*: fixing more grammatical errors or syntactic errors;
- *Coherence*: making the sentences more logically linked and organized;
- *Readability*: making the text easier to read and understand;

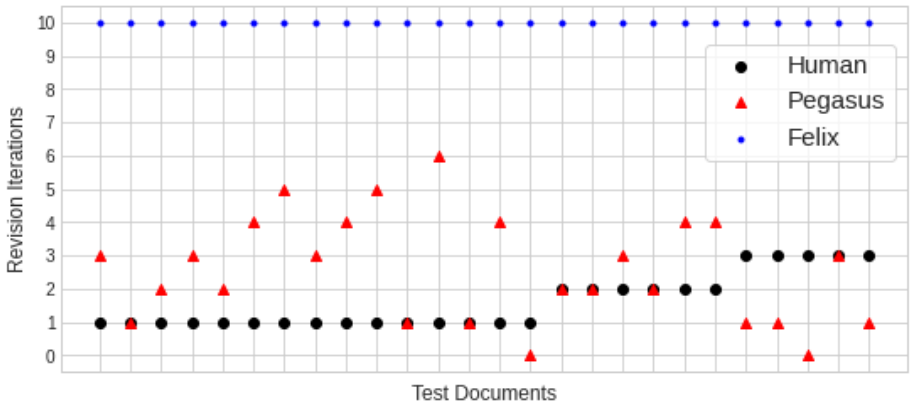


Figure 5.8: Number of iterations made by humans and different text revision models.

- *Overall Quality*: better improving the overall quality of the document.

Secondly, we evaluate how model generated text quality varies across revision depths. We use the same set of 21 iterative document-revisions in §5.2.3. We feed the original documents into the best-performing model to obtain the model's revised documents at each revision depth. For each data pair, we randomly shuffle model revisions and human revisions, and ask human evaluators to judge which one gives better overall text quality.

There exists a big gap between the best-performing model revisions and human revisions, indicating the challenging nature of the modeling problem. Thus, while model revisions can achieve comparable performance with human revisions on fluency, coherence, and meaning preservation, human revisions still outperform in terms of readability and overall quality.

Table 5.16 demonstrates how model-generated text quality varies across revision depths. In the first two depths, human revisions win over model revisions with a ratio of 57.14%. However, in the last depth, model revisions stay similar with human revisions in a ratio of 57.15%. Upon reviewing revisions in the last depth, we find a lot of MEANING-CHANGED edits in human revisions. At the same time, the model revisions only made a few FLUENCY or CLARITY edits, which the human evaluators tend to judge as "tie".

Iterativeness. We also compare the iterative ability between the two kinds of text revision models (best performing versions of both FELIX and PEGASUS: trained on ITERATER-

FULL), against human's iterative revisions. Figure 5.8 shows that while PEGASUS is able to finish iterating after 2.57 revisions on average, FELIX continues to make iterations until the maximum cutoff of 10 that we set for the experiment. In contrast, humans on average make 1.61 iterations per document. While FELIX is able to make meaningful revisions, it lacks the ability to effectively evaluate the text quality at a given revision, and decide whether or not to make further changes. PEGASUS, on the other hand, is able to pick up on these nuances of iterative revision, and learns to stop revising after a certain level of quality has been reached.

5.2.5 Discussion

Our work is a step toward understanding the complex process of iterative text revision from human-written texts. We collect, annotate, and release ITERATER: a novel, large-scale, domain-diverse, annotated dataset of human edit actions. Despite the deliberate design of our dataset collection, ITERATER only includes *formally* written texts. We plan to extend it to diverse sets of revision texts, such as informally written blogs and less informal but communicative texts like emails, as well as increase the size of the current dataset. For future research, we believe ITERATER can serve as a basis for future corpus development and computationally modeling iterative text revision.

5.3 Summary

In this chapter, we model two types of user intentions in open-domain conversations and iterative text revision tasks respectively, and apply the proposed intention taxonomy for model performance evaluation and controllable text generation.

In Section 5.1, we propose a consensus-based reference-free framework for open-domain dialog evaluation with segment act flows. From extensive experiments against state-of-the-art baselines, our method can reach the best or comparable correlation with human evaluation. Our segment-act-based methods complement well to previous semantic-meaning-focused

methods, pushing the ceiling of correlations. Moreover, the promise of our consensus-based framework encourages us to step further in the direction of dialog evaluation.

In Section 5.2, we construct a large-scale, annotated dataset of human edit actions, and train controllable LLMs to model the human revision process. Our research shows that different domains of text have different distributions of edit intentions, and the general quality of the text has improved over time. Computationally modeling the human revision process is still under-explored, yet our results indicate some interesting findings and potential directions.

CHAPTER 6
HUMAN-AI INTERACTION APPLICATIONS

Another important dimension of building text generation applications is the user-system interaction, where a good system not only successfully fulfills user requirements but also reduces user effort in interacting with the system. In this chapter, I present a human-AI interactive text generation system for the text revision task. Specifically, I develop a human-in-the-loop text revision system based on the controllable LLMs to provide high-quality text revisions with minimal human effort in Section 6.1.

6.1 A System Demonstration for Human-in-the-loop Iterative Text Revision

Text revision is a crucial part of writing. Specifically, text revision involves identifying discrepancies between intended and instantiated text, deciding what edits to make, and how to make those desired edits [273, 245, 274]. It enables writers to deliberate over and organize their thoughts, find a better line of argument, learn afresh, and discover what was not known before [244, 243]. Previous studies [248, 249, 250] have shown that text revision is an *iterative* process since human writers are unable to simultaneously comprehend multiple demands and constraints of the task when producing well-written texts – for instance, covering the content, following linguistic norms and discourse conventions of written prose, etc. Therefore, writers resort to performing text revisions on their drafts iteratively to reduce the number of considerations at each time.

Computational modeling of the iterative text revision process is essential for building intelligent and interactive writing assistants. Most prior works on the development of neural text revision systems [86, 87, 82, 85] do not take the iterative nature of text revision and human feedback on suggested revisions into consideration. The direct application of such revision systems in an iterative way, however, could generate some "noisy" edits and require

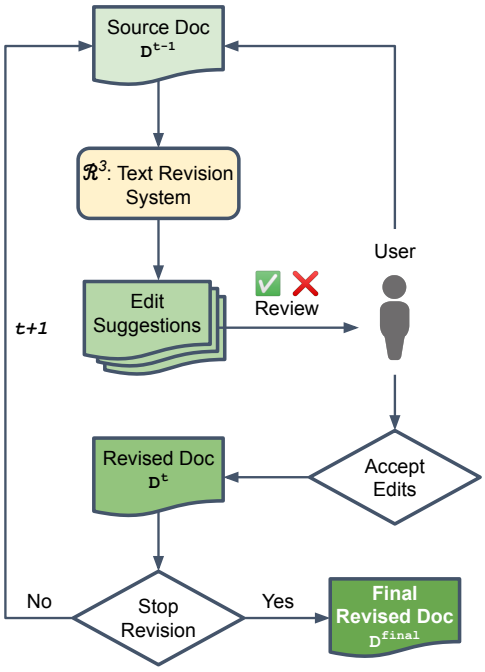


Figure 6.1: System overview for $\mathcal{R}3$ human-in-the-loop iterative text revision.

much burden on human writers to fix the noise. Therefore, we propose to collect human feedback at each iteration of revision to filter out those harmful noisy edits and produce revised documents of higher quality.

In this work, we present a novel human-in-the-loop iterative text revision system, $\mathcal{R}$ead, $\mathcal{R}$evise, $\mathcal{R}$epeat ($\mathcal{R}3$), which reads model-generated revisions and user feedbacks, revises documents, and repeats human-machine interactions in an iterative way, as depicted in Figure 6.1. First, users write a document as input to the system or choose one from a candidate document set to edit. Then, the text revision system provides multiple editing suggestions with their edits and intents. Users can accept or reject the editing suggestions in an iterative way and stop revision when no editing suggestions are provided or the model reaches the maximum revision limit. The overall model performance can be estimated by calculating the acceptance rate throughout all editing suggestions.

$\mathcal{R}3$ provides numerous benefits over existing writing assistants for text revision. First, $\mathcal{R}3$ improves the overall writing experience for writers by making it more interpretable, controllable, and productive: on the one hand, writers don't have to (re-)read the parts of

the text that are already high quality, and this, in turn, helps them focus on larger writing goals; on the other hand, by showing edit intentions for every suggested edit, which users can further decide to accept or reject, $\mathcal{R}3$ provides them with more fine-grained control over the text revision process compared to other one-shot based text revision systems [8], and are limited in both interpretability and controllability. Second, $\mathcal{R}3$ improves the revision efficiency. The human-machine interaction can help the system produce higher-quality revisions with fewer iterations and edits, and the empirical experiments validate this claim. To the best of our knowledge, $\mathcal{R}3$ is the first text revision system in literature that can perform *iterative* text revision in collaboration with human writers and revision models.

In this paper, we make three major contributions:

1. Presenting a novel human-in-the-loop text revision system $\mathcal{R}3$ to make text revision models more accessible; and to make the process of iterative text revision efficient, productive, and cognitively less challenging.

2. From an HCI perspective, conducting experiments to measure the effectiveness of the proposed system for the iterative text revision task. Empirical experiments show that $\mathcal{R}3$ can generate edits with a comparable acceptance rate to human writers at early revision depths.

3. Analyzing the data collected from human-model interactions for text revision and providing insights and future directions for building high-quality and efficient human-in-the-loop text revision systems. We release our code, revision interface, and collected human-model interaction dataset to promote future research on collaborative text revision.

6.1.1 $\mathcal{R}3$ Human-in-the-loop Iterative Text Revision System

Figure 6.1 shows the general pipeline of $\mathcal{R}3$ human-in-the-loop iterative text revision system. In this section, we will describe the development details of the text revision models and

demonstrate our user interfaces.

Iterative Text Revision Task. We first formulate an iterative text revision process: given a source document[1] $\mathcal{D}^{t-1}$, at each revision depth t, a text revision system will apply a set of edits to get the revised document $\mathcal{D}^t$. The system will continue iterating revision until the revised document $\mathcal{D}^t$ satisfies a set of predefined stopping criteria, such as reaching a predefined maximum revision depth t_{max}, or making no edits between $\mathcal{D}^{t-1}$ and $\mathcal{D}^t$.

Text Revision System Overview. We follow the prior work of [50] to build our text revision system. The system is composed of edit intention identification models and a text revision generation model. We follow the same data collection procedure in [50] to collect the iterative revision data.Then, we train the three models on the collected revision dataset.

Edit Intention Identification Models. Following [50], our edit intentions have four categories: FLUENCY, COHERENCE, CLARITY, and STYLE. We build our edit intention identification models at each sentence of the source document $\mathcal{D}^{t-1}$ to capture the more fine-grained edits. Specifically, given a source sentence, the system will make two-step predictions: (1) whether or not to edit, and (2) which edit intention to apply. The decision whether or not to edit is taken by an edit-prediction classifier that predicts a binary label of whether to edit a sentence or not. The second model, called the edit-intention classifier, predicts which edit intention to apply to the sentence. If the edit-prediction model predicts "not to edit" in the first step, the source sentence will be kept unchanged at the current revision depth.

Text Revision Generation Model. We fine-tune a large pre-trained language model like PEGASUS [275] on our collected revision dataset to build the text revision generation model. Given a source sentence and its predicted edit intention, the model will generate a revised

[1]The source document can be chosen by a user in the candidate set of documents or written from scratch by a user.

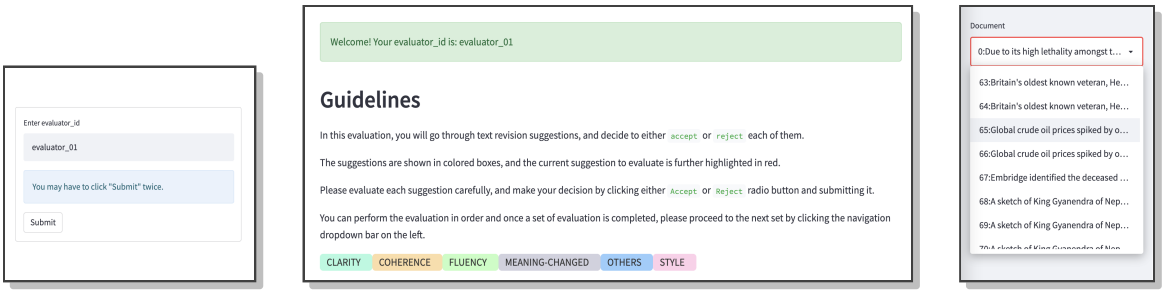
sentence, conditioned on the predicted edit intention. Then, we concatenate all un-revised and revised sentences to get the model-revised document $\mathcal{D}^t$, and extract all its edits using *latexdiff*[2] and *difflib*.[3]

In summary, at each revision depth t, given a source document $\mathcal{D}^{t-1}$, the text revision system first predicts the need for revising a sentence, and for the ones that need revision, it predicts the corresponding fine-grained edit intentions – thus, generating the revised document $\mathcal{D}^t$ based on the source document and the predicted edit decisions and intentions.

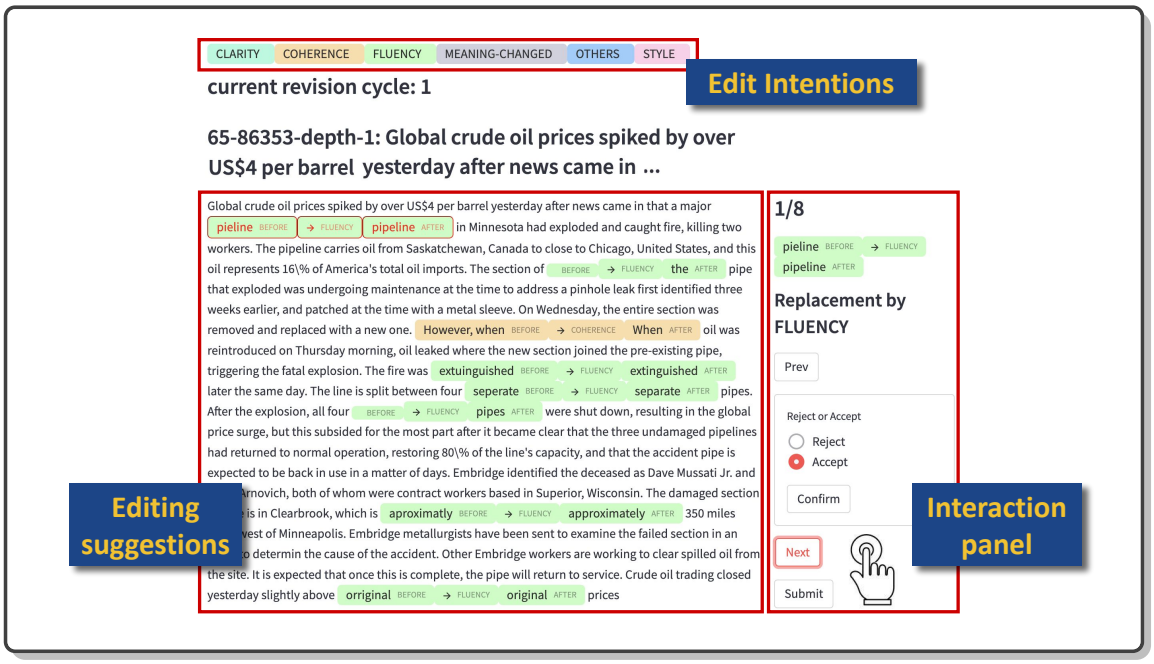
Human-in-the-loop Revision. In practice, not all model-generated edits are equally impactful towards improving the document quality [50]. Therefore, we enable user interaction in the iterative text revision process to achieve high-quality of text revisions along with a productive writing experience. At each revision depth t, our system provides the user with suggested edits, and their corresponding edit intentions. The user can interact with the system by choosing to accept or reject the suggested edits.

Figure 6.2 illustrates the details of $\mathcal{R}3$'s user interface. First, a user enters their id to log in to the web interface as shown in Figure 6.2a. Then, the user is instructed with a few guidelines on how to operate the revision as demonstrated in Figure 6.2b. After getting familiar with the interface, the user can select a source document from the left drop-down menu in Figure 6.2c. By clicking the source document, all the edits predicted by the text revision model, as well as their corresponding edit intentions will show up on the main page as illustrated in Figure 6.2d (left panel). The user is guided to go through each suggested edit, and choose to accept or reject the current edit by clicking the *Confirm* button in Figure 6.2d (right panel). After going through all the suggested edits, the user is guided to click the *Submit* button to save their decisions on the edits. Then, the user is guided to click the *Next Iteration!* button to proceed to the next revision depth and check the next round of edits suggested by the system. This interactive process continues until the system does not

[2] `https://ctan.org/pkg/latexdiff`
[3] `https://docs.python.org/3/library/difflib.html`



(a) Login (b) Read guidelines (c) Select doc



(d) Editing suggestions and interaction panel

Figure 6.2: User interface demonstration for $\mathcal{R}3$. The anonymized version is available at `https://youtu.be/lK08tIpEoaE`.

generate further edits or reaches the maximum revision depth t_{max}.

6.1.2 Experimental Setup

We conduct experiments to answer the following research questions:

RQ1 How likely are users to accept the editing suggestions predicted by our text revision system? This question is designed to evaluate whether our text revision system can generate high-quality edits.

RQ2 Which types of edit intentions are more likely to be accepted by users? This question

is aimed to identify which types of edits are more favored by users.

RQ3 Does user feedback in $\mathcal{R}3$ help produce higher quality of revised documents? This question is proposed to validate the effectiveness of human-in-the-loop component in $\mathcal{R}3$.

System Setups. We prepare three types of iterative revision systems to answer the above questions:

1. HUMAN-HUMAN: We ask users to accept or reject text revisions made by human writers, which are directly sampled from our collected iterative revision dataset. This serves as the baseline to measure the gap between our text revision system and human writers.
2. SYSTEM-HUMAN: We ask users to accept or reject text revisions made by our system. Then, we incorporate user accepted edits to the system to generate the next iteration of revision. This is the standard human-in-the-loop process of $\mathcal{R}3$.
3. SYSTEM-ONLY: We conduct an ablation study by removing user interaction in reviewing the model-generated edits. Then, we compare the overall quality of the final revised documents with and without the human-in-the-loop component.

In both HUMAN-HUMAN and SYSTEM-HUMAN setups where users interacted with the system, they were not informed whether the revisions were sampled from our collected iterative revision dataset, or generated by the underlying text revision models.

User Study Design. We hired three linguistic experts (English L1, bachelor's or higher degree in Linguistics) to interact with our text revision system. Each user was presented with a text revision (as shown in Figure 6.2d) and asked to accept or reject each edit in the current revision (users were informed which revision depth they were looking at). For a fair comparison, users were not informed about the source of the edits (human-written vs. model-generated), and the experiments were conducted separately one after the other. Note that the users were only asked to accept or reject edits, and they had control neither over the number of iterations, nor over the stopping criteria. The stopping criteria for the experiment

	# Docs	Avg. Depths	# Edits
Training	44,270	6.63	292,929
Validation	5,152	6.60	34,026
Test	6,226	6.34	39,511

Table 6.1: Statistics for our collected revision data which has been used to train the edit intention identification model and the text revision generation model. **# Docs** means the total number of unique documents, **Avg. Depths** indicates the average revision depth per document (for the human-generated training data), and **# Edits** stands for the total number of edits (sentence pairs) across the corpus.

were set by us and designed as: (1) no new edits were made at the following revision depth, or (2) the maximum revision depth $t_{max} = 3$ was reached.

Data Details. We followed the prior work [50] to collect the text revision data across three domains: ArXiv, Wikipedia, and Wikinews. This data was then used to train both the edit intention identification models and the text revision generation model. We split the data into training, validation, and test sets according to their document ids with a ratio of 8:1:1. The detailed data statistics are included in Table 6.1. Note that our newly collected revision dataset is larger than the previously proposed dataset in [50] with around 24K more unique documents and 170K more edits (sentence pairs).

For the human evaluation data, we randomly sampled 10 documents with a maximum revision depth of 3 from each domain in the test set in Table 6.1. For the evaluation of text revisions made by human writers (HUMAN-HUMAN), we presented the existing ground-truth references from our collected dataset to users. Since we do not hire additional human writers to perform continuous revisions, we just presented the static human revisions from the original test set to users at each revision depth, and collected the user acceptance statistics as a baseline for our system.

For the evaluation of text revisions made by our system (SYSTEM-HUMAN), we only presented the original source document at the initial revision depth ($\mathcal{D}^0$) to our system, and let the system generate edits in the following revision depths, while incorporating the

accept/reject decisions on model-generated edit suggestions by the users. Note that at each revision depth, the system will only incorporate the edits accepted by users and pass them to the next revision iteration.

For text revisions made by our system without human-in-the-loop (SYSTEM-ONLY), we let the system generate edits in an iterative way and accepted all model-generated edits at each revision depth.

Model Details. For both edit intention identification models, we fine-tuned the RoBERTa-large [276] pre-trained checkpoint from HuggingFace [124] for 2 epochs with a learning rate of 1×10^{-5} and batch size of 16. The edit-prediction classifier is a binary classification model that predicts whether to edit a given sentence or not. It achieves an F1 score of 67.33 for the edit label and 79.67 for the not-edit label. The edit-intention classifier predicts the specific intent of a sentence that requires editing. It achieves F1 scores of 67.14, 70.27, 57.0, and 3.21[4] for CLARITY, FLUENCY, COHERENCE and STYLE intent labels respectively.

For the text revision generation model, we fine-tuned the PEGASUS-LARGE [275] pre-trained checkpoint from HuggingFace. We set the edit intentions as new special tokens (e.g., <STYLE>, <FLUENCY>), and concatenated the edit intention and source sentence together as the input to the model. The output of the model is the revised sentence, and we trained the model with cross-entropy loss. We fine-tuned the model for 5 epochs with a learning rate of 3×10^{-5} and batch size of 4. Finally, our text revision generation model achieves 41.78 SARI score [277], 81.11 BLEU score [218] and 89.08 ROUGE-L score [165] on the test set.

6.1.3 Result Analysis

Iterativeness. The human-in-the-loop iterative text revision evaluation results are reported in Table 6.2. Each document is evaluated by at least 2 users. **We find that $\mathcal{R}3$ achieves comparable performances with ground-truth human revisions at revision depth 1 and**

[4]We note that the F1 score for STYLE is low as the number of training samples for that intent is particularly small.

	HUMAN-HUMAN				SYSTEM-HUMAN (ours)			
t	**# Docs**	**Avg. Edits**	**Avg. Accepts**	**% Accepts**	**# Docs**	**Avg. Edits**	**Avg. Accepts**	**% Accepts**
1	30	5.37	2.77	51.66	30	**5.90**	**2.90**	49.15
2	30	4.83	3.00	62.06	24	3.83	2.57	**67.02**
3	20	3.80	2.67	70.39	20	3.43	1.94	56.71

Table 6.2: Human-in-the-loop iterative text revision evaluation results. t stands for the revision depth, **# Docs** shows the total number of revised documents at the current revision depth, **Avg. Edits** indicates the average number of applied edits per document, **Avg. Accepts** means the average number of edits accepted by users per document, and **% Accepts** is calculated by dividing the total accepted edits with the total applied edits.

	HUMAN-HUMAN			SYSTEM-HUMAN (ours)		
	# Edits	**# Accepts**	**% Accepts**	**# Edits**	**# Accepts**	**% Accepts**
CLARITY	197	119	60.40	**332**	195	58.73
FLUENCY	178	146	**82.02**	91	41	45.05
COHERENCE	103	41	39.80	141	68	48.22
STYLE	6	2	33.33	113	73	**64.60**

Table 6.3: The distribution of different edit intentions. **# Edits** indicates the total number of applied edits under the current edit intention, **# Accepts** means the total number of edits accepted by users under the current edit intention, and **% Accepts** is calculated by dividing the total accepted edits with the total applied edits.

2, and tends to generate less favorable edits at revision depth 3. At revision depth 1, $\mathcal{R}3$ is able to generate more edits than ground-truth human edits for each document, and gets more edits accepted by users on average. This shows the potential of $\mathcal{R}3$ in generating appropriate text revisions that are more favorable to users.

At revision depth 2, while $\mathcal{R}3$ generates fewer edits than human writers on average, it gets a higher acceptance rate than human writers. This result suggests that for the end users, more edits may not necessarily lead to a higher acceptance ratio, and shows that $\mathcal{R}3$ is able to make high-quality edits for effective iterative text revisions. At revision depth 3, $\mathcal{R}3$ generates even fewer edits compared both to human writers and its previous revision depths. This result can be attributed to the fact that our models are only trained on static human revision data, while at testing time they have to make predictions conditioned on their revisions generated at the previous depth, which may have a very different distribution of edits than the training data.

Edit Intention	Edit Suggestion
CLARITY	Emerging new test procedures ~~, such as antigen or RT-LAMP tests,~~ might enable us to protect nursing home residents.
FLUENCY	For Radar tracking, we show how a model can reduce the tracking errors.
COHERENCE	However, ~~we show that~~ even a small violation can significantly modify the effective noise.
STYLE	There has been ~~numerous~~extensive research focusing on neural coding.

Table 6.4: Edit suggestion examples generated by $\mathcal{R}3$.

	Avg. Depths	# Edits	Quality
SYSTEM-HUMAN (ours)	2.5	148	**0.68**
SYSTEM-ONLY	2.8	175	0.28

Table 6.5: Quality comparison results of final revised documents with and without human-in-the-loop. **Avg. Depths** indicates the average number of iterations conducted by the system, **# Edits** means the total number of accepted edits by the system, and **Quality** represents the human judgements of the overall quality of system-revised final documents.

Edit Intentions. Table 6.3 demonstrates the distribution of different edit intentions, which can help us further analyze which type of edits are more likely to be accepted by end users. For human-generated revisions, we find that FLUENCY edits are most likely to be accepted since they are mainly fixing grammatical errors.

For system-generated revisions, we observe that CLARITY edits are the most frequent edits but end users only accept 58.73% of them, which suggests that our system needs further improvements in learning CLARITY edits. Another interesting observation is that STYLE edits are rarely generated by human writers (1.2%) and also get the lowest acceptance rate (33.33%) than other intentions, while they are frequently generated by our system (16.7%) and surprisingly gets the highest acceptance rate (64.6%) than other intentions. This observation indicates that $\mathcal{R}3$ is capable of generating favorable stylistic edits.

Role of Human Feedback in Revision Quality. Table 6.5 illustrates the quality comparison results of final revised documents with and without human-in-the-loop for $\mathcal{R}3$. We asked

Criterion	Avg. Score	Std. Deviation
Convenience	3.66	0.58
Satisfaction	2.33	0.58
Productivity	3.00	1.00
Retention	2.66	0.58

Table 6.6: User feedback survey ratings. Ratings are on 5-point Likert scale with 5 being strongly positive experience, 3 being neutral, and 1 being strongly negative. However, we'd like to point out that as the number of users (linguists) who participated in the study is small, the statistical significance of the results should be taken lightly.

another group of three annotators (English L2, bachelor's or higher degree in Computer Science) to judge whether the overall quality of the system-generated final document is better than the ground-truth reference final document. The quality score ranges between 0 and 1. We evaluated 10 unique documents in the ArXiv domain, and took the average score from all 3 annotators. As shown in Table 6.5, **SYSTEM-HUMAN produces a better overall quality score for the final system-generated documents with fewer iterations of revision and fewer edits**, which validates the effectiveness of the human-machine interaction proposed in $\mathcal{R}3$.

User Feedback. We also collected qualitative feedback about $\mathcal{R}3$ from the linguistic experts through a questionnaire. The first part of our questionnaire asks participants to recall their experience with the system, and evaluate various aspects of the system (in Table 6.6). They were asked to rate how easy it was to get onboarded and use the system (*convenience*), whether they were satisfied with the system (revision quality and usage experience) (*satisfaction*), whether they felt it improved their productivity for text revision (*productivity*), and whether they would like to use the system again (*retention*) for performing revisions on their documents.

In general, the users gave positive feedback towards the ease of use of the system. However, they were neutral on the potential productivity impact, owing to the lack of domain knowledge of the documents they were evaluating. This issue could be mitigated by asking users to revise their own documents of interest. The retention and satisfaction scores

were leaning slightly negative, which was explained as primarily attributed to gaps in the user interface design (eg. improperly aligned diffs, suboptimal presentation of word-level edits, etc.).

We also asked them to provide detailed comments on their experience, and the potential impact of the system on their text revision experience. Specifically, upon asking the users whether using the system to evaluate the model-suggested edits would be more time-efficient compared to actually revising the document themselves, we received many useful insights that help better design better interfaces and features of our system in future work, as some users noted:

> *I think it would be faster using the system, but I would still be checking the text myself in case edits were missed. The system made some edits where there were letters and parts of words being added/removed/replaced, which sometimes took some time to figure out. That wouldn't be the case if I were editing a document.*

> *Ultimately, I would use the system for grammar/coherence/clarity edits, and then still research (a lot) to ensure that meaning was preserved throughout the document. For topics that I was more familiar with/more general topics, using the system would probably reduce my time by a third or so. For topics that require more in-depth research, the time saved by using the system might be minimal.*

6.1.4 Discussion

When $\mathcal{R}3$ generates revisions at deeper depths, we observe a decrease in the acceptance ratio by human users. It is crucial to create a text revision system that can learn different revision strategies at each iteration and generate high-quality edits at deeper revision levels.

Editing suggestions provided by our text revision generation models could be improved. Particularly, FLUENCY edits show a huge gap between human and system revisions (45.05%

and 82.02%). Future work could focus on developing more powerful text revision generation models.

In our human-machine interaction, we restrict the users' role to accept or reject the model's predictions. Even with minimal human interaction, our experiment shows comparable or even better revision quality as compared to human writers at early revision depths. A potential future direction for human-machine collaborative text revision would be to develop advanced human-machine interaction interfaces, such as asking users to rewrite the machine-revised text.

Also, a larger-scale user study could be carried out to derive more meaningful statistics (e.g. optimal number of revision depths and edit suggestions) and investigate if there is any intriguing user behavior in the iterative revision process. For example, as mentioned in the users' feedback, it would be interesting to check if users behave differently when they are asked to accept/reject edit suggestions provided for their own texts as opposed to the texts written by a third party.

6.2 Summary

In this chapter, we analyze real-world human revision behaviors and develop an interactive text generation system for the text revision task. In Section 6.1, we develop an interactive iterative text revision system $\mathcal{R}3$ that is able to effectively assist users to make revisions and improve the quality of existing documents. Empirical results show that $\mathcal{R}3$ can generate iterative text revisions with acceptance rates comparable to or even better than human writers at early revision depths.

CHAPTER 7
CONCLUSION

My research aims to advance the field of human-AI collaborative text generation, focusing on enhancing large language models (LLMs) to better align with user specifications. Specifically, I achieve the goal through: (1) designing controllable and data-efficient text generation methods, and (2) building human-AI collaborative text generation applications guided by user intentions.

To enhance controllability and data efficiency, I propose fine-tuning a small side network to encode control attributes and integrate them with pre-trained LLMs. This is complemented by a specific control loss to prevent ambiguous and generic responses. Moreover, recognizing that some training data, especially LLM-generated data, can be very noisy and uninformative, I design a model-uncertainty-based data selection method to select training data that is beneficial for learning. To address the generation of factually incorrect content, I devise a new reward function and fine-tuned LLMs with reinforcement learning to ensure the model's outputs remain faithful to the input knowledge texts. To promote output diversity, I apply a stochastic function to introduce context-aware variations into encoder hidden states, which helps the model learn rich contextual representations.

For human-AI collaborative text generation, I apply LLMs for a document revision task that requires producing accurate, formal, and coherent text revisions. To examine the capability of LLMs for making continuous revisions and collaborating with human writers, I collect a large-scale iterative text revision dataset with fine-grained edit intention labels, train controllable text revision models, and build a human-in-the-loop system to incorporate user feedback into model revisions. The proposed system achieved high-quality text revisions with minimal human efforts by reading model-generated revisions and user feedback, revising documents, and repeating user-system interactions.

For future research directions, my primary objective is to further refine the interaction between users and AI systems. The goal is to develop more efficient writing systems that not only reduce user's cognitive load but also effectively meet domain-specific requirements. I plan to achieve the goal based on the retrieval-augmented generation framework [278] via: (1) collaborative text generation: understanding and predicting user requirements in real-time, and adaptively controlling LLMs to respond in various writing contexts; (2) knowledge-grounded text generation: equipping LLMs with domain-specific knowledge and controlling LLMs to generate factual and helpful content. Beyond these areas, I am also interested in educational AI applications. This includes the development of knowledge-grounded question-answering systems and academic writing assistants, leveraging LLMs to provide educational support, tailored guidance, and assistance in academic research and writing.

REFERENCES

[1] R. Nakano *et al.*, "Webgpt: Browser-assisted question-answering with human feedback," *arXiv*, 2022.

[2] J. Menick *et al.*, "Teaching language models to support answers with verified quotes," *arXiv*, 2022.

[3] L. Ouyang *et al.*, "Training language models to follow instructions with human feedback," *arXiv*, 2022.

[4] H. Touvron *et al.*, "Llama: Open and efficient foundation language models," *arXiv*, 2023.

[5] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[6] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1073–1083.

[7] E. Clark, A. S. Ross, C. Tan, Y. Ji, and N. A. Smith, "Creative writing with a machine in the loop: Case studies on slogans and stories," in *23rd International Conference on Intelligent User Interfaces*, ser. IUI '18, Tokyo, Japan: Association for Computing Machinery, 2018, 329–340, ISBN: 9781450349451.

[8] M. Lee, P. Liang, and Q. Yang, "Coauthor: Designing a human-ai collaborative writing dataset for exploring language model capabilities," *arXiv preprint arXiv:2201.06796*, 2022.

[9] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[11] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901.

[12] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, 2020.

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, 2017.

[14] N. Stiennon *et al.*, "Learning to summarize with human feedback," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 3008–3021.

[15] Y. Zhou *et al.*, "Large language models are human-level prompt engineers," *arXiv preprint arXiv:2211.01910*, 2022.

[16] E. M. Bender and B. Friedman, "Data statements for natural language processing: Toward mitigating system bias and enabling better science," *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 587–604, 2018.

[17] S. L. Blodgett, S. Barocas, H. Daumé III, and H. Wallach, "Language (technology) is power: A critical survey of "bias" in NLP," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 5454–5476.

[18] S. Harrer, "Attention is not all you need: The complicated case of ethically using large language models in healthcare and medicine," *EBioMedicine*, vol. 90, 2023.

[19] B. Peng *et al.*, "Few-shot natural language generation for task-oriented dialog," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 172–182.

[20] M. Cascella, J. Montomoli, V. Bellini, and E. Bignami, "Evaluating the feasibility of chatgpt in healthcare: An analysis of multiple clinical and research scenarios," *Journal of Medical Systems*, vol. 47, no. 1, p. 33, 2023.

[21] A. Deroy, K. Ghosh, and S. Ghosh, "How ready are pre-trained abstractive models and llms for legal case judgement summarization?" *arXiv preprint arXiv:2306.01248*, 2023.

[22] J. Schatzmann, K. Georgila, and S. Young, "Quantitative evaluation of user simulation techniques for spoken dialogue systems," in *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, Lisbon, Portugal: Special Interest Group on Discourse and Dialogue (SIGdial), Sep. 2005, pp. 45–54.

[23] J. Gao, M. Galley, and L. Li, "Neural approaches to conversational ai," in *The 41st International ACM SIGIR Conference on Research & Development in Informa-*

tion Retrieval, ser. SIGIR '18, Ann Arbor, MI, USA: Association for Computing Machinery, 2018, 1371–1374, ISBN: 9781450356572.

[24] S. Zhang and K. Balog, "Evaluating conversational recommender systems via user simulation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '20, Virtual Event, CA, USA: Association for Computing Machinery, 2020, 1512–1520, ISBN: 9781450379984.

[25] Y. Wolf, N. Wies, Y. Levine, and A. Shashua, "Fundamental limitations of alignment in large language models," *arXiv preprint arXiv:2304.11082*, 2023.

[26] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 889–898.

[27] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, "Ctrl: A conditional transformer language model for controllable generation," *arXiv preprint arXiv:1909.05858*, 2019.

[28] B. Krause *et al.*, *Gedi: Generative discriminator guided sequence generation*, 2020. arXiv: 2009.06367 [cs.CL].

[29] S. Dathathri *et al.*, "Plug and play language models: A simple approach to controlled text generation," in *International Conference on Learning Representations*, 2019.

[30] J. Wei *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[31] S. Min *et al.*, "Rethinking the role of demonstrations: What makes in-context learning work?" In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11 048–11 064.

[32] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, *arXiv preprint arXiv:1909.05858*, 2019.

[33] S. Kawano, K. Yoshino, and S. Nakamura, "Neural conversation model controllable by given dialogue act based on adversarial learning and label-aware objective," in *Proceedings of the 12th International Conference on Natural Language Generation*, Tokyo, Japan: Association for Computational Linguistics, 2019, pp. 198–207.

[34] S. Dathathri *et al.*, "Plug and play language models: A simple approach to controlled text generation," *CoRR*, vol. abs/1912.02164, 2019. arXiv: 1912.02164.

[35] J. O. Zhang, A. Sax, A. R. Zamir, L. J. Guibas, and J. Malik, "Side-tuning: Network adaptation via additive side networks," *CoRR*, vol. abs/1912.13503, 2019. arXiv: 1912.13503.

[36] K. Yang and D. Klein, "FUDGE: Controlled text generation with future discriminators," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 3511–3535.

[37] R. Ramamurthy *et al.*, "Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization," in *The Eleventh International Conference on Learning Representations*, 2023.

[38] E. Akyürek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou, "What learning algorithm is in-context learning? investigations with linear models," in *The Eleventh International Conference on Learning Representations*, 2023.

[39] W. Du and Y. Ji, "SideControl: Controlled open-domain dialogue generation via additive side networks," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2175–2194.

[40] W. Du, H. Chen, and Y. Ji, "Self-training with two-phase self-augmentation for few-shot dialogue generation," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 2770–2784.

[41] **Du, Wanyu** and Y. Ji, "Blending reward functions via few expert demonstrations for faithful and accurate knowledge-grounded dialogue generation," *arXiv preprint arXiv:2311.00953*, 2023.

[42] N. Singh, G. Bernal, D. Savchenko, and E. L. Glassman, "Where to hide a stolen elephant: Leaps in creative writing with multimodal machine intelligence," *ACM Trans. Comput.-Hum. Interact.*, vol. 30, no. 5, 2023.

[43] A. Yuan, A. Coenen, E. Reif, and D. Ippolito, "Wordcraft: Story writing with large language models," in *27th International Conference on Intelligent User Interfaces*, ser. IUI '22, New York, NY, USA: Association for Computing Machinery, 2022, 841–852, ISBN: 9781450391443.

[44] W. Cui, S. Zhu, M. R. Zhang, H. A. Schwartz, J. O. Wobbrock, and X. Bi, "Justcorrect: Intelligent post hoc text correction techniques on smartphones," in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*,

ser. UIST '20, Virtual Event, USA: Association for Computing Machinery, 2020, 487–499, ISBN: 9781450375146.

[45] A. Coenen, L. Davis, D. Ippolito, E. Reif, and A. Yuan, "Wordcraft: A human-ai collaborative editor for story writing," *arXiv preprint arXiv:2107.07430*, 2021.

[46] V. Padmakumar and H. He, "Machine-in-the-loop rewriting for creative image captioning," *arXiv preprint arXiv:2111.04193*, 2021.

[47] K. I. Gero, V. Liu, and L. B. Chilton, "Sparks: Inspiration for science writing using language models," *arXiv preprint arXiv:2110.07640*, 2021.

[48] S. Sun *et al.*, "IGA: An intent-guided authoring assistant," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 5972–5985.

[49] F. Faltings *et al.*, "Interactive text generation," *arXiv preprint arXiv:2303.00908*, 2023.

[50] W. Du, V. Raheja, D. Kumar, Z. M. Kim, M. Lopez, and D. Kang, "Understanding iterative revision from human-written text," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3573–3590.

[51] W. Du, Z. M. Kim, V. Raheja, D. Kumar, and D. Kang, "Read, revise, repeat: A system demonstration for human-in-the-loop iterative text revision," in *Proceedings of the First Workshop on Intelligent and Interactive Writing Assistants (In2Writing 2022)*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 96–108.

[52] T. Goswamy, I. Singh, A. Barkati, and A. Modi, "Adapting a language model for controlled affective text generation," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 2787–2801.

[53] Z. Lin and M. Riedl, "Plug-and-blend: A framework for controllable story generation with blended control codes," N. Akoury, F. Brahman, S. Chaturvedi, E. Clark, M. Iyyer, and L. J. Martin, Eds., pp. 62–71, Jun. 2021.

[54] A. Madotto, E. Ishii, Z. Lin, S. Dathathri, and P. Fung, "Plug-and-play conversational models," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 2422–2433.

[55] M. Ghazvininejad, X. Shi, J. Priyadarshi, and K. Knight, "Hafez: An interactive poetry generation system," in *Proceedings of ACL 2017, System Demonstrations*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 43–48.

[56] N. Stiennon *et al.*, "Learning to summarize from human feedback," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20, Vancouver, BC, Canada: Curran Associates Inc., 2020, ISBN: 9781713829546.

[57] X. Lu *et al.*, "QUARK: Controllable text generation with reinforced unlearning," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.

[58] Y. Dubois *et al.*, "Alpacafarm: A simulation framework for methods that learn from human feedback," *ArXiv*, vol. abs/2305.14387, 2023.

[59] Z. Wu *et al.*, "Fine-grained human feedback gives better rewards for language model training," *ArXiv*, vol. abs/2306.01693, 2023.

[60] Z. Chen, H. Eavani, W. Chen, Y. Liu, and W. Y. Wang, "Few-shot NLG with pre-trained language model," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 183–190.

[61] E. Chang, X. Shen, H.-S. Yeh, and V. Demberg, "On training instance selection for few-shot neural text generation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Online: Association for Computational Linguistics, Aug. 2021, pp. 8–13.

[62] F. Mi, W. Zhou, L. Kong, F. Cai, M. Huang, and B. Faltings, "Self-training improves pre-training for few-shot learning in task-oriented dialog systems," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1887–1898.

[63] X. Xu, G. Wang, Y.-B. Kim, and S. Lee, "AugNLG: Few-shot natural language generation using self-trained data augmentation," 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. 1183–1195.

[64] C. Kedzie and K. McKeown, "A good sample is hard to find: Noise injection sampling and self-training for neural language generation models," in *Proceedings*

of the 12th International Conference on Natural Language Generation, Tokyo, Japan: Association for Computational Linguistics, 2019, pp. 584–593.

[65] J. He, J. Gu, J. Shen, and M. Ranzato, "Revisiting self-training for neural sequence generation," in *International Conference on Learning Representations*, 2020.

[66] S. Bakshi, S. Batra, P. Heidari, A. Arun, S. Jain, and M. White, "Structure-to-text generation with self-training, acceptability classifiers and context-conditioning for the GEM shared task," in *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, Online: Association for Computational Linguistics, Aug. 2021, pp. 136–147.

[67] P. Heidari *et al.*, "Getting to production with few-shot natural language generation models," in *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Singapore and Online: Association for Computational Linguistics, Jul. 2021, pp. 66–76.

[68] S. V. Mehta, J. Rao, Y. Tay, M. Kale, A. P. Parikh, and E. Strubell, *Improving compositional generalization with self-training for data-to-text generation*, 2022. arXiv: 2110.08467 [cs.CL].

[69] M. Lewis *et al.*, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880.

[70] T. Sellam, D. Das, and A. Parikh, "BLEURT: Learning robust metrics for text generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7881–7892.

[71] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*, 1990.

[72] S. Larson *et al.*, "An evaluation dataset for intent classification and out-of-scope prediction," 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)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 1311–1316.

[73] A. Stolcke *et al.*, "Dialogue act modeling for automatic tagging and recognition of conversational speech," *CoRR*, vol. cs.CL/0006023, 2000.

[74] E. Shriberg, R. Dhillon, S. Bhagat, J. Ang, and H. Carvey, "The ICSI meeting recorder dialog act (MRDA) corpus," in *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004*, Cambridge, Massachusetts, USA: Association for Computational Linguistics, 2004, pp. 97–100.

[75] M. Walker and R. Passonneau, "DATE: A dialogue act tagging scheme for evaluation of spoken dialogue systems," in *Proceedings of the First International Conference on Human Language Technology Research*, 2001.

[76] M. A. Walker *et al.*, "Darpa communicator dialog travel planning systems: The june 2000 data collection," in *INTERSPEECH*, 2001.

[77] P. Budzianowski *et al.*, "Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling," *CoRR*, vol. abs/1810.00278, 2018. arXiv: 1810.00278.

[78] D. Yu and Z. Yu, "MIDAS: A dialog act annotation scheme for open domain human machine spoken conversations," *CoRR*, vol. abs/1908.10023, 2019. arXiv: 1908.10023.

[79] A. Cervone and G. Riccardi, "Is this dialogue coherent? learning from dialogue acts and entities," *CoRR*, vol. abs/2006.10157, 2020. arXiv: 2006.10157.

[80] D. Yang, A. Halfaker, R. Kraut, and E. Hovy, "Identifying semantic edit intentions from revisions in Wikipedia," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 2000–2010.

[81] F. Zhang, H. B. Hashemi, R. Hwa, and D. Litman, "A corpus of annotated revisions for studying argumentative writing," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1568–1578.

[82] T. Ito *et al.*, "Diamonds in the rough: Generating fluent sentences from early-stage drafts for academic writing assistance," in *Proceedings of the 12th International Conference on Natural Language Generation*, Tokyo, Japan: Association for Computational Linguistics, 2019, pp. 40–53.

[83] T. Anthonio, I. Bhat, and M. Roth, "WikiHowToImprove: A resource and analyses on edits in instructional texts," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, Marseille, France: European Language Resources Association, May 2020, pp. 5721–5729, ISBN: 979-10-95546-34-4.

[84] I. Bhat, T. Anthonio, and M. Roth, "Towards modeling revision requirements in wikiHow instructions," in *Proceedings of the 2020 Conference on Empirical Methods*

in Natural Language Processing (EMNLP), Online: Association for Computational Linguistics, Nov. 2020, pp. 8407–8414.

[85] F. Faltings *et al.*, "Text editing by command," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 5259–5274.

[86] M. Faruqui, E. Pavlick, I. Tenney, and D. Das, "WikiAtomicEdits: A multilingual corpus of Wikipedia edits for modeling language and discourse," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 305–315.

[87] J. A. Botha, M. Faruqui, J. Alex, J. Baldridge, and D. Das, "Learning to split and rephrase from Wikipedia edit history," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 732–737.

[88] A. Prakash *et al.*, "Neural paraphrase generation with stacked residual lstm networks," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2923–2934.

[89] Z. Li, X. Jiang, L. Shang, and H. Li, "Paraphrase generation with deep reinforcement learning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 3865–3878.

[90] H. Jhamtani, V. Gangal, E. Hovy, and E. Nyberg, "Shakespearizing modern language using copy-enriched sequence to sequence models," in *Proceedings of the Workshop on Stylistic Variation*, 2017, pp. 10–19.

[91] S. Rao and J. Tetreault, "Dear sir or madam, may I introduce the GYAFC dataset: Corpus, benchmarks and metrics for formality style transfer," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 129–140.

[92] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie, "A hierarchical recurrent encoder-decoder for generative context-aware query suggestion," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, 2015, pp. 553–562.

[93] 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.

[94] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734.

[95] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1412–1421.

[96] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1631–1640.

[97] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *CoRR*, vol. abs/1910.10683, 2019. arXiv: 1910.10683.

[98] H. Bahuleyan, L. Mou, O. Vechtomova, and P. Poupart, "Variational attention for sequence-to-sequence models," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 1672–1682.

[99] Y. Deng, Y. Kim, J. Chiu, D. Guo, and A. Rush, "Latent alignment and variational attention," in *Advances in Neural Information Processing Systems*, 2018, pp. 9712–9724.

[100] T. Wang and X. Wan, "T-cvae: Transformer-based conditioned variational autoencoder for story completion," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, International Joint Conferences on Artificial Intelligence Organization, Jul. 2019, pp. 5233–5239.

[101] J. Cho, M. Seo, and H. Hajishirzi, "Mixture content selection for diverse sequence generation," 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)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3121–3131.

[102] D. Qian and W. K. Cheung, "Enhancing variational autoencoders with mutual information neural estimation for text generation," 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)*,

Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 4047–4057.

[103] C. Wu, P. Z. Wang, and W. Y. Wang, "On the encoder-decoder incompatibility in variational text modeling and beyond," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 3449–3464.

[104] Y. Duan, C. Xu, J. Pei, J. Han, and C. Li, "Pre-train and plug-in: Flexible conditional text generation with variational auto-encoders," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 253–262.

[105] B. Sun, S. Feng, Y. Li, J. Liu, and K. Li, "Generating relevant and coherent dialogue responses using self-separated conditional variational AutoEncoders," 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. 5624–5637.

[106] A. K. Vijayakumar *et al.*, "Diverse beam search: Decoding diverse solutions from neural sequence models," *arXiv preprint arXiv:1610.02424*, 2016.

[107] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv preprint arXiv:1904.09751*, 2019.

[108] X. He, G. Haffari, and M. Norouzi, "Sequence to sequence mixture model for diverse machine translation," in *Proceedings of the 22nd Conference on Computational Natural Language Learning*, 2018, pp. 583–592.

[109] T. Shen, M. Ott, M. Auli, and M. Ranzato, "Mixture models for diverse machine translation: Tricks of the trade," *arXiv preprint arXiv:1902.07816*, 2019.

[110] D. Liu and G. Liu, "A transformer-based variational autoencoder for sentence generation," in *2019 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2019, pp. 1–7.

[111] K. Shinoda, S. Sugawara, and A. Aizawa, "Improving the robustness of QA models to challenge sets with variational question-answer pair generation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, Online: Association for Computational Linguistics, Aug. 2021, pp. 197–214.

[112] C. E. Rasmussen and C. K. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[113] S. R. Bowman, L. Vilnis, O. Vinyals, A. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 10–21.

[114] Y. Kim, S. Wiseman, A. Miller, D. Sontag, and A. Rush, "Semi-amortized variational autoencoders," in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 2678–2687.

[115] A. B. Dieng, Y. Kim, A. M. Rush, and D. M. Blei, "Avoiding latent variable collapse with generative skip models," in *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, K. Chaudhuri and M. Sugiyama, Eds., ser. Proceedings of Machine Learning Research, vol. 89, PMLR, 2019, pp. 2397–2405.

[116] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[117] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[118] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc., 2015, pp. 3483–3491.

[119] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proceedings of the International Conference on Representation Learning*, 2014.

[120] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, 2005, pp. 65–72.

[121] 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*, 2002, pp. 311–318.

[122] Y. Zhu *et al.*, "Texygen: A benchmarking platform for text generation models," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 1097–1100.

[123] A. Deshpande, J. Aneja, L. Wang, A. G. Schwing, and D. Forsyth, "Fast, diverse and accurate image captioning guided by part-of-speech," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 695–10 704.

[124] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45.

[125] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[126] W. Lan, S. Qiu, H. He, and W. Xu, "A continuously growing dataset of sentential paraphrases," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1224–1234.

[127] M. Vono, N. Dobigeon, and P. Chainais, "High-dimensional gaussian sampling: A review and a unifying approach based on a stochastic proximal point algorithm," *SIAM Review*, vol. 64, no. 1, pp. 3–56, 2022.

[128] R. Thoppilan *et al.*, "Lamda: Language models for dialog applications," *arXiv*, 2022.

[129] N. Kandpal, H. Deng, A. Roberts, E. Wallace, and C. Raffel, "Large language models struggle to learn long-tail knowledge," *arXiv*, 2022.

[130] Y. Wang *et al.*, "Self-instruct: Aligning language models with self-generated instructions," *arXiv*, 2023.

[131] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, "On faithfulness and factuality in abstractive summarization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 1906–1919.

[132] M. Cao, Y. Dong, and J. Cheung, "Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3340–3354.

[133] N. Dziri *et al.*, "FaithDial: A Faithful Benchmark for Information-Seeking Dialogue," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 1473–1490, Dec. 2022. eprint: `https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl\_a\_00529/2065956/tacl\_a\_00529.pdf`.

[134] X. Chen, F. Chen, F. Meng, P. Li, and J. Zhou, "Unsupervised knowledge selection for dialogue generation," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, Online: Association for Computational Linguistics, Aug. 2021, pp. 1230–1244.

[135] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," *arXiv*, 2016.

[136] W. Du and Y. Ji, "An empirical comparison on imitation learning and reinforcement learning for paraphrase generation," 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)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6012–6018.

[137] C. Wang and R. Sennrich, "On exposure bias, hallucination and domain shift in neural machine translation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 3544–3552.

[138] W. Li, W. Wu, M. Chen, J. Liu, X. Xiao, and H. Wu, "Faithfulness in natural language generation: A systematic survey of analysis, evaluation and optimization methods," *arXiv*, 2022.

[139] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, "Wizard of wikipedia: Knowledge-powered conversational agents," in *International Conference on Learning Representations*, 2018.

[140] S. Feng, S. S. Patel, H. Wan, and S. Joshi, "MultiDoc2Dial: Modeling dialogues grounded in multiple documents," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6162–6176.

[141] B. Kim, J. Ahn, and G. Kim, "Sequential latent knowledge selection for knowledge-grounded dialogue," in *International Conference on Learning Representations*, 2020.

[142] Z. Wu, B.-R. Lu, H. Hajishirzi, and M. Ostendorf, "DIALKI: Knowledge identification in conversational systems through dialogue-document contextualization," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1852–1863.

[143] X. Zhao, W. Wu, C. Tao, C. Xu, D. Zhao, and R. Yan, "Low-resource knowledge-grounded dialogue generation," in *International Conference on Learning Representations*, 2020.

[144] P. Xu, D. Liang, Z. Huang, and B. Xiang, "Attention-guided generative models for extractive question answering," *arXiv*, 2021.

[145] S. Prabhumoye, K. Hashimoto, Y. Zhou, A. W. Black, and R. Salakhutdinov, "Focused attention improves document-grounded generation," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 4274–4287.

[146] R. Tian, S. Narayan, T. Sellam, and A. P. Parikh, "Sticking to the facts: Confident decoding for faithful data-to-text generation," *arXiv*, 2020.

[147] M. Li *et al.*, "Don't say that! making inconsistent dialogue unlikely with unlikelihood training," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 4715–4728.

[148] B. Liu, G. Tür, D. Hakkani-Tür, P. Shah, and L. Heck, "Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2060–2069.

[149] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep reinforcement learning for dialogue generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 1192–1202.

[150] M. Post, "A call for clarity in reporting BLEU scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 186–191.

[151] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, "Retrieval augmentation reduces hallucination in conversation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3784–3803.

[152] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020.

[153] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018, ISBN: 0262039249.

[154] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, *High-dimensional continuous control using generalized advantage estimation*, 2018.

[155] J. Wu *et al.*, "Recursively summarizing books with human feedback," *arXiv*, 2021.

[156] D. M. Ziegler *et al.*, "Fine-tuning language models from human preferences," *arXiv*, 2020.

[157] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[158] K. Pearson, "Note on regression and inheritance in the case of two parents," *proceedings of the royal society of London*, vol. 58, no. 347-352, pp. 240–242, 1895.

[159] S. Bansal *et al.*, "R3 : Refined retriever-reader pipeline for multidoc2dial," in *Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 148–154.

[160] T. Formal, C. Lassance, B. Piwowarski, and S. Clinchant, "Splade v2: Sparse lexical and expansion model for information retrieval," *arXiv*, 2021.

[161] Y. Liu *et al.*, "RoBERTa: A robustly optimized BERT pretraining approach," in *International Conference on Learning Representations*, 2020.

[162] G. Izacard and E. Grave, "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*, Online: Association for Computational Linguistics, Apr. 2021, pp. 874–880.

[163] H. Rashkin, D. Reitter, G. S. Tomar, and D. Das, "Increasing faithfulness in knowledge-grounded dialogue with controllable features," 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. 704–718.

[164] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2018.

[165] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81.

[166] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–72.

[167] V. Adlakha, P. BehnamGhader, X. H. Lu, N. Meade, and S. Reddy, "Evaluating correctness and faithfulness of instruction-following models for question answering," *arXiv*, 2023.

[168] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[169] Y. Zhang *et al.*, "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.

[170] S. Roller *et al.*, "Recipes for building an open-domain chatbot," *arXiv preprint arXiv:2004.13637*, 2020.

[171] K. Shuster, D. Ju, S. Roller, E. Dinan, Y.-L. Boureau, and J. Weston, "The dialogue dodecathlon: Open-domain knowledge and image grounded conversational agents," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 2453–2470.

[172] T.-H. Wen, M. Gašić, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young, "Semantically conditioned LSTM-based natural language generation for spoken dialogue systems," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 1711–1721.

[173] P. Ke, J. Guan, M. Huang, and X. Zhu, "Generating informative responses with controlled sentence function," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 1499–1508.

[174] W. Chen, J. Chen, P. Qin, X. Yan, and W. Y. Wang, "Semantically conditioned dialog response generation via hierarchical disentangled self-attention," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3696–3709.

[175] A. See, S. Roller, D. Kiela, and J. Weston, "What makes a good conversation? how controllable attributes affect human judgments," in *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 1702–1723.

[176] C.-H. Hsueh and W.-Y. Ma, "Semantic guidance of dialogue generation with reinforcement learning," in *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 1st virtual meeting: Association for Computational Linguistics, Jul. 2020, pp. 1–9.

[177] J. Takayama and Y. Arase, "Consistent response generation with controlled specificity," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 4418–4427.

[178] D. Varshney, A. Ekbal, and P. Bhattacharyya, "Modelling context emotions using multi-task learning for emotion controlled dialog generation," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, Online: Association for Computational Linguistics, Apr. 2021, pp. 2919–2931.

[179] A. Baheti, A. Ritter, J. Li, and B. Dolan, "Generating more interesting responses in neural conversation models with distributional constraints," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 3970–3980.

[180] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi, "Learning to write with cooperative discriminators," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 1638–1649.

[181] E. Dinan *et al.*, "The second conversational intelligence challenge (convai2)," in *The NeurIPS'18 Competition*, Springer, 2020, pp. 187–208.

[182] Y. Li, H. Su, X. Shen, W. Li, Z. Cao, and S. Niu, "DailyDialog: A manually labelled multi-turn dialogue dataset," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 986–995.

[183] H. Rashkin, E. M. Smith, M. Li, and Y.-L. Boureau, "Towards empathetic open-domain conversation models: A new benchmark and dataset," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 5370–5381.

[184] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[185] I. Loshchilov and F. Hutter, *Decoupled weight decay regularization*, 2019. arXiv: 1711.05101 [cs.LG].

[186] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543.

[187] S. Bao, H. He, F. Wang, H. Wu, and H. Wang, "PLATO: Pre-trained dialogue generation model with discrete latent variable," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 85–96.

[188] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].

[189] M. Li, J. Weston, and S. Roller, "ACUTE-EVAL: improved dialogue evaluation with optimized questions and multi-turn comparisons," *CoRR*, vol. abs/1909.03087, 2019. arXiv: 1909.03087.

[190] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, pp. 8689–8696, 2020.

[191] M. Kale and A. Rastogi, "Template guided text generation for task-oriented dialogue," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 6505–6520.

[192] T.-H. Wen *et al.*, "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*, San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 120–129.

[193] C. Zhu, M. Zeng, and X. Huang, "Multi-task learning for natural language generation in task-oriented dialogue," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Con-*

ference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 1261–1266.

[194] Y. Yang, Y. Li, and X. Quan, "Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 16, pp. 14 230–14 238, 2021.

[195] Y. Lee, "Improving end-to-end task-oriented dialog system with a simple auxiliary task," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1296–1303.

[196] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," 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)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6382–6388.

[197] S. Y. Feng, V. Gangal, D. Kang, T. Mitamura, and E. Hovy, "GenAug: Data augmentation for finetuning text generators," in *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, Online: Association for Computational Linguistics, Nov. 2020, pp. 29–42.

[198] B. Peng, C. Zhu, M. Zeng, and J. Gao, "Data Augmentation for Spoken Language Understanding via Pretrained Language Models," in *Proc. Interspeech 2021*, 2021, pp. 1219–1223.

[199] A. Fabbri *et al.*, "Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 704–717.

[200] M. Chen, K. Q. Weinberger, and J. Blitzer, "Co-training for domain adaptation," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24, Curran Associates, Inc., 2011.

[201] Z. Wang, Z. Dai, B. Poczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[202] S. Meftah, N. Semmar, Y. Tamaazousti, H. Essafi, and F. Sadat, "On the hidden negative transfer in sequential transfer learning for domain adaptation from news to

tweets," in *Proceedings of the Second Workshop on Domain Adaptation for NLP*, Kyiv, Ukraine: Association for Computational Linguistics, Apr. 2021, pp. 140–145.

[203] L. Feng, M. Qiu, Y. Li, H. Zheng, and Y. Shen, "Wasserstein selective transfer learning for cross-domain text mining," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 9772–9783.

[204] Á. Peris and F. Casacuberta, "Active learning for interactive neural machine translation of data streams," in *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 151–160.

[205] A. P.V.S and C. M. Meyer, "Data-efficient neural text compression with interactive learning," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 2543–2554.

[206] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, 2016, pp. 1050–1059.

[207] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[208] S. Swayamdipta *et al.*, "Dataset cartography: Mapping and diagnosing datasets with training dynamics," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online: Association for Computational Linguistics, Nov. 2020, pp. 9275–9293.

[209] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910.

[210] P. Budzianowski *et al.*, "MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 5016–5026.

[211] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," in *International Conference on Learning Representations*, 2020.

[212] S. Mukherjee and A. Awadallah, "Uncertainty-aware self-training for few-shot text classification," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 21 199–21 212.

[213] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 8689–8696.

[214] E. M. Smith, O. Hsu, R. Qian, S. Roller, Y. Boureau, and J. Weston, "Human evaluation of conversations is an open problem: Comparing the sensitivity of various methods for evaluating dialogue agents," *CoRR*, vol. abs/2201.04723, 2022. arXiv: 2201.04723.

[215] C. Tao, L. Mou, D. Zhao, and R. Yan, "RUBER: an unsupervised method for automatic evaluation of open-domain dialog systems," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, S. A. McIlraith and K. Q. Weinberger, Eds., AAAI Press, 2018, pp. 722–729.

[216] L. Huang, Z. Ye, J. Qin, L. Lin, and X. Liang, "GRADE: automatic graph-enhanced coherence metric for evaluating open-domain dialogue systems," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds., Association for Computational Linguistics, 2020, pp. 9230–9240.

[217] S. Mehri and M. Eskénazi, "Unsupervised evaluation of interactive dialog with dialogpt," *CoRR*, vol. abs/2006.12719, 2020. arXiv: 2006.12719.

[218] 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.

[219] 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*, Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2122–2132.

[220] C. Zhang *et al.*, "Dynaeval: Unifying turn and dialogue level evaluation," *CoRR*, vol. abs/2106.01112, 2021. arXiv: 2106.01112.

[221] 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.

[222] Y. Yeh, M. Eskénazi, and S. Mehri, "A comprehensive assessment of dialog evaluation metrics," *CoRR*, vol. abs/2106.03706, 2021. arXiv: 2106.03706.

[223] B. Pang, E. Nijkamp, W. Han, L. Zhou, Y. Liu, and K. Tu, "Towards holistic and automatic evaluation of open-domain dialogue generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 3619–3629.

[224] S. Ghazarian, R. M. Weischedel, A. Galstyan, and N. Peng, "Predictive engagement: An efficient metric for automatic evaluation of open-domain dialogue systems," *CoRR*, vol. abs/1911.01456, 2019. arXiv: 1911.01456.

[225] S. Mehri and M. Eskenazi, "USR: An unsupervised and reference free evaluation metric for dialog generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 681–707.

[226] V. Phy, Y. Zhao, and A. Aizawa, "Deconstruct to reconstruct a configurable evaluation metric for open-domain dialogue systems," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 4164–4178.

[227] H. Bunt, V. Petukhova, A. Malchanau, A. Fang, and K. Wijnhoven, "The dialogbank: Dialogues with interoperable annotations," *Language Resources and Evaluation*, vol. 53, no. 2, pp. 213–249, 2019.

[228] Y. Li, H. Su, X. Shen, W. Li, Z. Cao, and S. Niu, "Dailydialog: A manually labelled multi-turn dialogue dataset," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2017, pp. 986–995.

[229] Y. Liu *et al.*, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. arXiv: 1907.11692.

[230] A. See, S. Roller, D. Kiela, and J. Weston, "What makes a good conversation? how controllable attributes affect human judgments," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 1702–1723.

[231] R. C. Gunasekara *et al.*, "Overview of the ninth dialog system technology challenge: DSTC9," *CoRR*, vol. abs/2011.06486, 2020. arXiv: 2011.06486.

[232] T. Kiss and J. Strunk, "Unsupervised multilingual sentence boundary detection," *Computational Linguistics*, vol. 32, no. 4, pp. 485–525, 2006.

[233] S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela, and J. Weston, "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)*, Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 2204–2213.

[234] S. Golovanov, R. Kurbanov, S. Nikolenko, K. Truskovskyi, A. Tselousov, and T. Wolf, "Large-scale transfer learning for natural language generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 6053–6058.

[235] K. Shuster, D. Ju, S. Roller, E. Dinan, Y.-L. Boureau, and J. Weston, "The dialogue dodecathlon: Open-domain knowledge and image grounded conversational agents," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 2453–2470.

[236] S. Bird and E. Loper, "NLTK: The natural language toolkit," in *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 214–217.

[237] J. L. Fleiss, "Measuring nominal scale agreement among many raters.," *Psychological bulletin*, vol. 76, no. 5, p. 378, 1971.

[238] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with BERT," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020.

[239] Y. Zhang *et al.*, "Dialogpt: Large-scale generative pre-training for conversational response generation," *CoRR*, vol. abs/1911.00536, 2019. arXiv: 1911.00536.

[240] E. M. Bender and A. Koller, "Climbing towards NLU: On meaning, form, and understanding in the age of data," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 5185–5198.

[241] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[242] L. Flower and J. R. Hayes, "The cognition of discovery: Defining a rhetorical problem," *College Composition and Communication*, vol. 31, no. 1, pp. 21–32, 1980.

[243] M. Scardamalia, "Research on written composition," *Handbook of reserch on teaching*, 1986.

[244] N. Sommers, "Revision strategies of student writers and experienced adult writers," *College composition and communication*, vol. 31, no. 4, pp. 378–388, 1980.

[245] L. Faigley and S. Witte, "Analyzing revision," *College composition and communication*, vol. 32, no. 4, pp. 400–414, 1981.

[246] J. Fitzgerald, "Research on revision in writing," *Review of educational research*, vol. 57, no. 4, pp. 481–506, 1987.

[247] L. S. Bridwell, "Revising strategies in twelfth grade students' transactional writing," *Research in the Teaching of English*, vol. 14, no. 3, pp. 197–222, 1980.

[248] L. Flower, "The dynamics of composing: Making plans and juggling constraints," *Cognitive processes in writing*, pp. 31–50, 1980.

[249] A. Collins and D. Gentner, "A framework for a cognitive theory of writing," in *Cognitive processes in writing*, Erlbaum, 1980, pp. 51–72.

[250] M. M. Vaughan and D. D. McDonald, "A model of revision in natural language generation," in *24th Annual Meeting of the Association for Computational Linguistics*, New York, New York, USA: Association for Computational Linguistics, Jul. 1986, pp. 90–96.

[251] A. Spangher, X. Ren, J. May, and N. Peng, "NewsEdits: A news article revision dataset and a novel document-level reasoning challenge," M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, Eds., pp. 127–157, Jul. 2022.

[252] D. Yang, A. Halfaker, R. Kraut, and E. Hovy, "Who did what: Editor role identification in wikipedia," in *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 10, 2016.

[253] D. Rathjens, "The seven components of clarity in technical writing," *IEEE Transactions on Professional Communication*, vol. PC-28, no. 4, pp. 42–46, 1985.

[254] R. A. Harris, *Writing with clarity and style: A guide to rhetorical devices for contemporary writers*. Routledge, 2017.

[255] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[256] K. Kann, S. Rothe, and K. Filippova, "Sentence-level fluency evaluation: References help, but can be spared!" In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 313–323.

[257] A. Pauls and D. Klein, "Large-scale syntactic language modeling with treelets," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 959–968.

[258] M. Lapata and R. Barzilay, "Automatic evaluation of text coherence: Models and representations," in *IJCAI*, 2005, pp. 1085–1090.

[259] R. Soricut and D. Marcu, "Discourse generation using utility-trained coherence models," in *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, Sydney, Australia: Association for Computational Linguistics, Jul. 2006, pp. 803–810.

[260] M. Elsner and E. Charniak, "Coreference-inspired coherence modeling," in *Proceedings of ACL-08: HLT, Short Papers*, Columbus, Ohio: Association for Computational Linguistics, Jun. 2008, pp. 41–44.

[261] A. Louis and A. Nenkova, "A coherence model based on syntactic patterns," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 1157–1168.

[262] J. P. Kincaid, R. P. Fishburne Jr, R. L. Rogers, and B. S. Chissom, "Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel," Naval Technical Training Command Millington TN Research Branch, Tech. Rep., 1975.

[263] M. Solnyshkina, R. Zamaletdinov, L. Gorodetskaya, and A. Gabitov, "Evaluating text complexity and flesch-kincaid grade level," *Journal of Social Studies Education Research*, vol. 8, no. 3, pp. 238–248, 2017.

[264] W. Xu, C. Napoles, E. Pavlick, Q. Chen, and C. Callison-Burch, "Optimizing Statistical Machine Translation for Text Simplification," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 401–415, Jul. 2016. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00107/1567406/tacl_a_00107.pdf.

[265] H. Guo, R. Pasunuru, and M. Bansal, "Dynamic multi-level multi-task learning for sentence simplification," in *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 462–476.

[266] I. Nassar, M. Ananda-Rajah, and G. Haffari, "Neural versus non-neural text simplification: A case study," in *Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association*, Sydney, Australia: Australasian Language Technology Association, 2019, pp. 172–177.

[267] D. Nishihara, T. Kajiwara, and Y. Arase, "Controllable text simplification with lexical constraint loss," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 260–266.

[268] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020.

[269] J. Mallinson, A. Severyn, E. Malmi, and G. Garrido, "FELIX: Flexible text editing through tagging and insertion," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 1244–1255.

[270] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "Pegasus: Pre-training with extracted gap-sentences for abstractive summarization," in *International Conference on Machine Learning*, PMLR, 2020, pp. 11 328–11 339.

[271] E. Malmi, S. Krause, S. Rothe, D. Mirylenka, and A. Severyn, "Encode, tag, realize: High-precision text editing," 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)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 5054–5065.

[272] Y. Dong, Z. Li, M. Rezagholizadeh, and J. C. K. Cheung, "EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3393–3402.

[273] L. Flower and J. R. Hayes, "A cognitive process theory of writing," *College Composition and Communication*, vol. 32, no. 4, pp. 365–387, 1981.

[274] J. Fitzgerald, "Research on revision in writing," *Review of Educational Research*, vol. 57, no. 4, pp. 481–506, 1987.

[275] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization," in *Proceedings of the 37th International Conference on Machine Learning*, H. D. III and A. Singh, Eds., ser. Proceedings of Machine Learning Research, vol. 119, PMLR, 2020, pp. 11 328–11 339.

[276] Y. Liu *et al.*, "RoBERTa: A robustly optimized BERT pretraining approach," in *International Conference on Learning Representations*, 2020.

[277] W. Xu, C. Napoles, E. Pavlick, Q. Chen, and C. Callison-Burch, "Optimizing statistical machine translation for text simplification," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 401–415, 2016.

[278] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.