Prompt-tuning in Controlled Dialogue Generation

by

Runcheng Liu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2022

Abstract

Recent years have witnessed a prosperous development of dialogue response generation since the advent of Transformer [66]. Fine-tuning pretrained language models for different downstream tasks has become the dominant paradigm in Natural Language Processing (NLP). However, fine-tuning requires storing a full copy of parameter states for every task, which is memory-consuming and expensive to serve when working with large-scale models with billions of parameters like GPT-3 [8].

Meanwhile, prompt-tuning has become an increasingly popular parameter-efficient method for steering large pretrained language models to various tasks. Most of the prompting techniques are applied in language understanding and assuming fixed prompts for all data samples within a task. Therefore, there arises an urgent need to exploit the ability of prompt-tuning in open-domain dialogue generation where data samples may vary greatly within a task.

In this thesis, we present a novel, instance-specific prompt-tuning algorithm for dialogue generation. Specifically, we generate prompts based on instance-level control code, rather than the conversation context, to explore their impact on controlled dialogue generation. Experiments on popular open-domain dialogue datasets, evaluated with both automated metrics and human evaluation, demonstrate that our method is superior to prompting baselines as well as other lightweight controlled generation methods, and comparable to fine-tuning with less than 10% of total parameters.

Acknowledgements

I would like to thank my supervisor, Professor Pascal Poupart for his great support and supervision throughout my graduate study. He always provides me with the flexibility and constant guidance to pursue my research interests. I would not have been able to finish my thesis without his invaluable assistance.

I would like to thank Professor Jimmy Lin and Professor Wenhu Chen for serving as my committee members and providing valuable and informative feedback.

I would also like to thank my colleagues at Huawei Noah's Ark Lab, Mehdi Rezagholizadeh, Ivan Kobyzev, Ahmad Rashid, and many others for their assistance in my thesis research and the internship at the company. It is always a pleasure to work with them.

Finally, I would like to thank my family members for their unconditional love and support during my study abroad and my boyfriend Peijia Luo for his companionship and encouragement.

Dedication

This is dedicated to my beloved parents.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The release of transformer-based models [66, 56, 11] and extensive training have boosted the development of natural language generation. Due to the enormous potential stored in millions of or billions of parameters, models can generate fluent and consistent responses during various dialogue generation tasks [76, 59]. In light of this development, researchers are interested in developing personalized conversation agents for different users and the concept of controlled dialogue generation has rapidly gained popularity [62]. Models are trained to create responses depending on conversation context, with additional control attributes taken into account. The control attribute can either be a label attribute such as the dialogue intent/action and the desired topic, or a series of sentences describing the user's persona or background knowledge.

Previous work on this topic proposes different methods, including "controllable training" [30, 59]: fine-tuning all model parameters for each task; or "post-processing" [17, 10]: changing the model's activation states or modifying the output conditional distribution based on an independent classifier. Despite their outstanding performance, these methods either require storing a full copy of model parameters ("controllable training") or suffer from low-variance and high-biased frozen pretrained models ("post processing"). Recently, there has been a surge of interest in "additive networks" which add new parameters for each new attribute. For instance, SideControl [14] trains a residual (side net) on top of the pretrained base model to flexibly adapt the model to the new task. Because the control attribute combination occurs after the base model's computation and does not inject directly into the model, we notice that the model's performance degrades greatly when applied to different downstream datasets in our preliminary experiments. To this end, we are interested in developing a method that is more expressive.

As of recently, we derive our inspiration from the continuous prompts / prompt-tuning [33, 38, 45, 44], that interleave a sequence of key-value pairs and impact the model's computation accordingly. However, most of the prompt-tuning methods are applied in language understanding tasks with relatively little work on language generation such as dialogue generation, and they typically assume a task-level prompt for all samples within a task, neglecting the fact that samples might vary greatly, especially in the field of conversation generation.

1.2 Problem Statement

Motivated by previous studies, we cast the controlled dialogue generation as a prompt-tuning task. Given a large pretrained model that consists of a function $f_\theta(x) = y$ that takes as input some text x and computes some output y based on a large number of parameters θ. Controlled dialogue generation additionally introduces control attribute a that frames the function as $f_\theta(x, a) = y$. In practice, people are interested in training specialized functions $g_{\theta_i}(x, a) = y_i$ for downstream tasks (indexed by i). In that process, θ_i for each task i is tuned and stored which includes as many parameters as the entire pretrained model for each downstream task.

The problem that this thesis explores is how to avoid creating or modifying as many parameters as the full pretrained model. In other words, we intend to create and/or modify only a small number of parameters in comparison to the full pretraind model. Inspired by prompt-tuning, We present a method for steering the pretrained model θ with an additional small set of parameters ϕ_i modifying the control attribute a such that $f_\theta([x, h_{\phi_i(a)}]) \approx g_{\theta_i}(x, a)$. Here h_{ϕ_i} is a small prompting module that takes as input some control attribute a and encodes it into a prompt that is concatenated with x as input to the pretrained language model. The pretrained model f_θ is frozen throughout the fine-tuning. Since the prompting module is a fraction of the size of the pretrained dialogue model, this allows many controlled dialogue systems to be stored on a device without too much overhead.

1.3 Contributions

In this work, we propose Controlled DialogPrompt which applies a finer-grained prompt-tuning in controlled dialogue generation. It optimizes prompts based on provided control codes rather than the previous conversation history and we further explore the controllability of prompts at the instance level. By encoding control attributes into hidden vectors and injecting them at every layer of the base model, we find our method can dynamically modulate the base model in content-rich and challenging datasets with specified attributes. The size of the prompt encoder is strictly limited and we freeze the pretrained model during training in order to preserve memory efficiency.

We test our technique on label control and document control scenarios with two different open-domain dialogue datasets, showing that our method outperforms prompting and other lightweight controlled generation baselines and matches fine-tuning with less than 10% of total parameters. Besides, we carry out an ablation study of the expressiveness of different prompts depth with pair-wise human evaluations and provide generated samples for reference.

1.4 Thesis Outline

This thesis is organized as follows.

- In Chapter 2, we introduce basic components in the generation model and the development of text generation structures including Feedforward neural networks, Recurrent neural networks (RNN), Sequence-to-Sequence models, and Transformers.
- In Chapter 3, we describe more related work in both controlled dialogue generation and prompt-tuning. These works serve as an important foundation on which we propose our approach in the intersection of these two fields.
- In Chapter 4, we propose the method of Controlled DialogPrompt to adapt pretrained models effectively and efficiently, introduce experimental design from datasets to evaluation metrics, and report comparative results with detailed explanations.
- In Chapter 5, we summarize our work and discuss possible future work.

Chapter 2

Background

In this chapter, we first introduce Feedforward neural networks which are the basic elements in Deep Learning. We then cover the concept of Recurrent neural networks which are proficient in solving sequential data and their inefficiencies. Besides, we cover the structure of Sequence-to-Sequence model that is popular in natural language generation and is widely applied in dialogue tasks. Finally, we describe the state-of-the-art Transformer architecture and the prevalent paradigm of pretraining as well as several advanced pretrained transformer-based dialogue generation models.

2.1 Feedforward Neural Networks

Feedforward neural networks are among the most flexible and powerful biologically inspired learning algorithms. The network consists of neurons, called units or nodes, which are connected to one another and arranged into three layers: input layer, hidden layer, and output layer. Nodes in one layer are fully connected to nodes in adjacent layers, and input information flows through the net from the input layer to hidden layers and finally to the output layer to generate the output. The connections do not form a loop, i.e. the output will not be fed back as the input and that's why these networks are called "feedforward". We provide one example of a simple feedforward neural network in Figure 2.1. Connections between nodes are called edges, and they are responsible for passing values from one layer to another. Normally, edges do not have the same strengths or weights.

In the forward pass, each node in the hidden layers would take input values from the previous layer, multiply them by the weights of the connecting edges, add a constant called

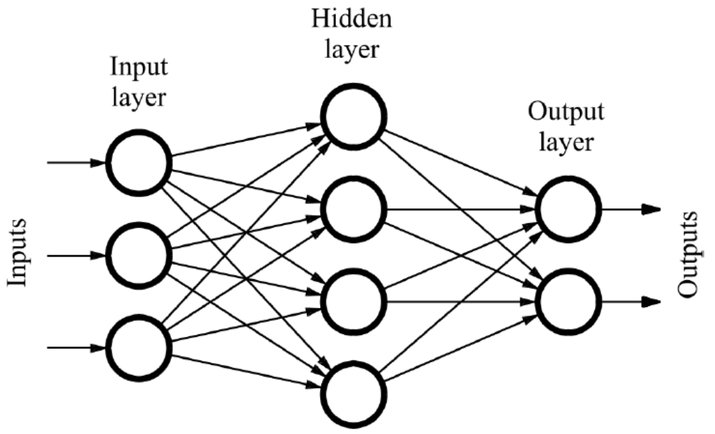


Figure 2.1: Sample of a feedforward neural net with one hidden layer from Quiza and Davim, 2011. [55]

bias, and go through a non-linear activation function to produce the output value. The values pass iteratively to the next layer until the output layer node generates the final output. All weights and biases form the trainable parameters of the entire neural network.

After the neural net generates an output, the difference between the predicted output and the true values/labels, known as loss, can be computed using different loss functions such as Cross Entropy or Mean Squared Error. The smaller the loss is, the better the neural net predicts. All parameters (weights and biases) are initialized randomly and adjusted to reduce the loss during the training so as to improve the accuracy of predictions. It does so by backpropagating the loss and updating the weights based on their calculated gradients using some optimization functions such as Stochastic Gradient Descent (SGD), Adam, etc. Let θ be the parameter of the neural network, $\mathcal{L}$ be the loss function and η be the learning rate, then the update can be expressed as:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}(\theta)$$

The number of layers in the neural network defines its depth. It is widely considered that a large depth is necessary to build a powerful model which can have more ability to learn different representations. However, it brings up another famous question called the vanishing/exploding gradient problem. Briefly speaking, with deeper layers, the product of the derivative (gradient) shrinks/explodes during the backpropagation and lower layers fail to receive a reasonable amount of gradients to make adjustments. As a result, lower layers do not learn/totally explode during the training.

2.2 Recurrent Neural Networks

Recurrent neural networks (RNNs) are another type of artificial neural networks where connections between nodes can create a cycle so that the previous output can be used as the next input and affect the subsequent computation. For example, if we encounter a text generation task where the prediction of the next token depends on the context created by the previous words, we cannot use a feedforward neural net which assumes every input is independent of each other, but instead, we can use an RNN which has a "memory" component to store the states and incorporate the dependencies.

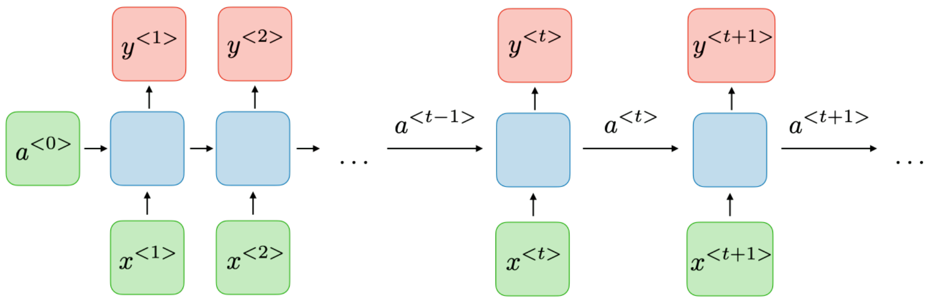


Figure 2.2: An unrolled recurrent neural network (RNN).[1]

We provide a graph (Figure 2.2) in which we unroll the RNN and analyze its behavior. In the beginning, an RNN is provided with initial hidden state a^0 and input x^1, it predicts a^1 and y^1 and stores a^1 as its latest "memory". In the next step, when given x^1, it uses the stored a^1 jointly to generate a^2 and y^2 and update a^2 as the latest "memory". In keeping with this, at every time step t, we can express the calculation as follows:

$$a^t = g(W_{aa}a^{t-1} + W_{ax}x^t + b_a)$$

$$y^t = f(W_{ya}a^t + b_y)$$

where $W_{aa}, W_{ax}, b_a, W_{ya}, b_y$ are learnable weights and biases that are shared temporally and g, f are activation functions.

As before, all parameters are updated using backpropagation during training. Though an RNN is capable of processing inputs of any length, it has difficulty remembering long-range dependencies. [7] Later, Long Short Term Memory (LSTM) [24] and Gated Recurrent Unit (GRU) [12] solve the problem by adding the gating mechanism to retain the past information and the vanishing/exploding problem. However, RNNs are still time-consuming to

[1] https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks

train and infer due to their sequential structure. Later, we will introduce Transformer 2.4 which is designed to process sequential data in parallel.

2.3 Sequence-to-Sequence Models

Sutskever et al. [50] propose Sequence-to-sequence (Seq2seq) models for solving tasks when input sequences and output sequences have different lengths. It has been widely applied in different language modeling tasks such as Machine Translation [50], Text Summarization [51], Dialogue Generation[67, 63], etc. Based on the previous RNN section, we notice that an RNN learns how to take the current hidden state and combine it with an input vector to produce the subsequent hidden state and the output vector. A Seq2Seq model is essentially composed of two multilayered RNNs: an encoder and a decoder. The encoder processes the source sentence and builds an internal vector of fixed dimensionality to encapsulate its understanding of the source sentence. The decoder takes the encoded representation and previous words in the target sentence to generate the next word step by step. By default, the decoder's internal vector is initialized by the encoder's last internal vector. Normally, the target sentence starts with "begin of sentence" token ($\langle BOS \rangle$) to shift the expected output by one time steps and is followed by "end of sentence" token ($\langle EOS \rangle$) to tell the decoder where to stop the generation. We provide a Seq2seq model in Figure 2.3.

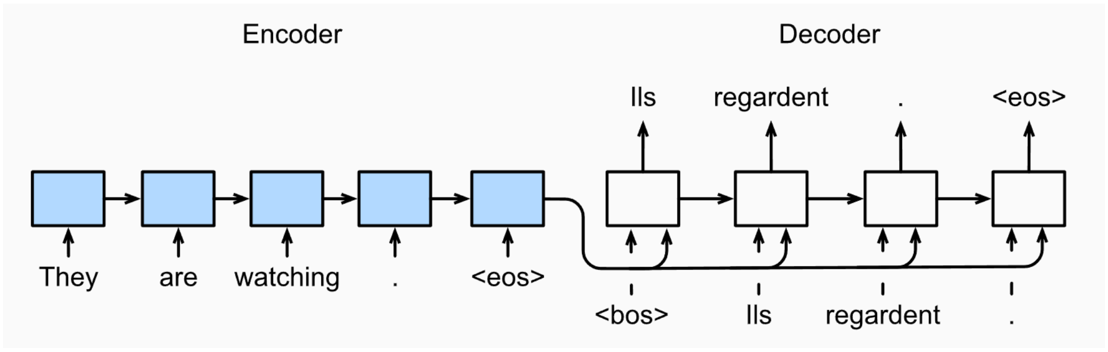


Figure 2.3: A Sequence-to-sequence model. [2]

To train a Seq2seq model, one forces the decoder to generate gold sequences by penalizing it for assigning low probabilities to gold sequences. Losses are summed on all

[2]https://d2l.ai/chapter_recurrent-modern/seq2seq.html

probabilities generated at every token in the target sentence and we use the generated loss to adjust the parameters (backpropagation mentioned earlier).

With the last hidden state of the encoder being the only encoded representation, Seq2seq model is incapable of capturing long source sentences. Later, Bahdanau et al. [3] and Luong et al. [47] integrate the attention mechanism into Sequence-to-sequence models, which allows the decoder to flexibly select a set of source words to attend to when predicting the target word. The attention mechanism is one of the most powerful techniques in machine learning that allows a model to automatically pay more attention to certain words and phrases in the past and it has demonstrated its effectiveness through qualitative analysis of Machine Translation tasks.

2.4 Transformers

Though recurrent neural networks can provide better performance in sequence modeling with the attention mechanism, the recurrent nature restricts them to processing every sentence word by word and it becomes prohibitively time-consuming to train RNNs on massive longer sequence datasets. In order to remedy this problem, Vaswani et al. [66] propose Transformer which is entirely built from self-attention blocks to compute representations and enables parallel computation that significantly reduces the training time.

Transformer follows the encoder-decoder structure where the encoder maps the input sentence to a sequence of fixed-dimension vectors that the decoder can utilize to generate an output sentence. In Figure 2.4, the left half depicts the encoder structure, which contains N stacked layers. In the paper, N is set to 6. Each layer consists of two sub-layers named multi-head self-attention and feedforward neural network. Multi-head self-attention is the key for the encoder to encapsulate the understanding of all processed tokens. We will explain the attention block in more detail in the next paragraph. Every sublayer is wrapped with a residual connection and a layer normalization that has proved effective at solving the vanishing/exploding gradient problem and stabilizing the hidden state dynamics. The decoder on the right-hand side is also composed of N stacked layers, which include three sub-layers, masked multi-head self-attention, multi-head cross-attention, and a feedforward neural network. According to the paper, masked multi-head self-attention is the trick that prevents the decoder from peeking at future tokens that it needs to generate and allows it to only pay attention to previous words. Multi-head cross attention enables the decoder to attend to the representation computed by the encoder, which supports the information flow in the encoder-decoder structure.

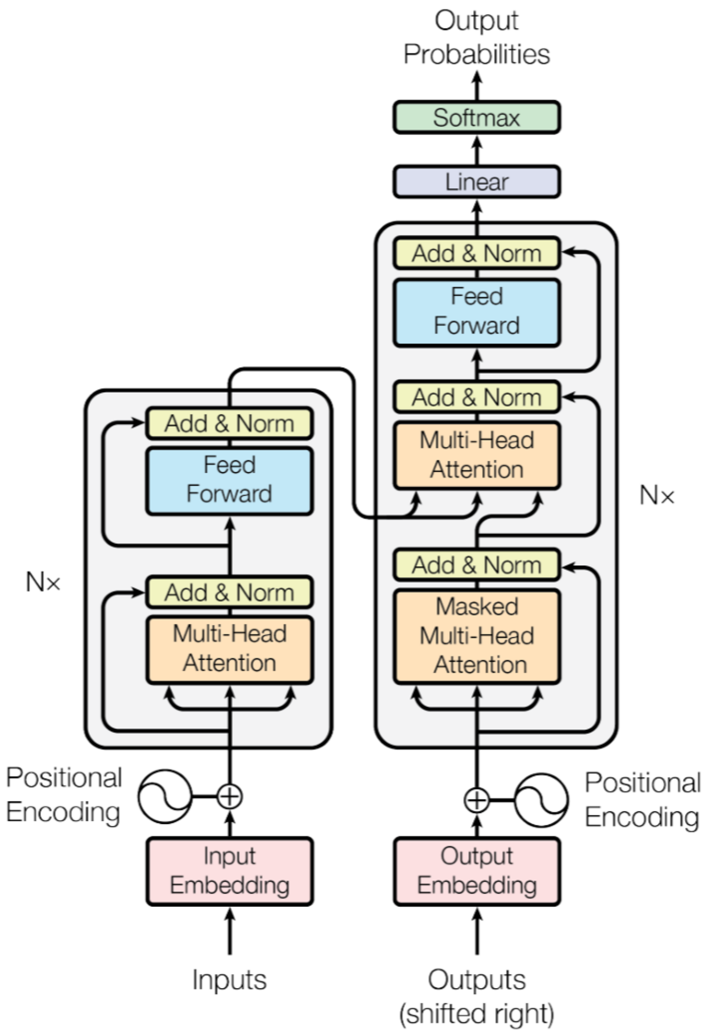


Figure 2.4: Transformer architecture from Vaswani et al. [66]

The most essential part of the transformer is “Attention”, or “Scaled Dot-Product Attention”. Basically, the stacked input X of sequence length l is firstly multiplied with weight matrices W^Q, W^K, W^V to produce the stacked matrices Query Q, Key K and Value V. Then the Query is multiplied with the Key to obtain the attention scores for values in V at different sequence positions. After scaling down the attention scores product, the output is multiplied with Value V to produce the final value. We represent the matrix of outputs as:

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

d_k is the dimension of queries and keys and $\frac{1}{\sqrt{d_k}}$ is the scaling factor. Instead of using scaled-dot-product attention directly, authors suggest adopting “Multi-Head Attention”, where X is projected to queries, keys and values h times. In this way, the model can benefit from richer representation subspaces between pieces of the input sequence. These h attention computations are performed in much smaller hidden dimensions and will be computed in parallel so as to reduce computational costs. Following that, all attention computations will be combined together back to the model’s hidden dimensions. The multi-head process can be represented as follows:

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \\ \text{where head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned}$$

where W_i^Q, W_i^K, W_i^V and W^O are projection matrices to be learned.

In order to make the attention-based encoder-decoder work better, Transformer also has word embedding layers before the encoder and the decoder, as shown in Figure ref-Transformer. The word embedding helps to convert input tokens to continuous vectors of the model’s hidden dimensionality for later computation. Besides, in order to convert the decoder output vector to the conditional probability of the next token over the vocabulary, they apply a linear layer that shares the same weights as the aforementioned embedding layer. As the transformer removes the recurrent structure, which might result in the loss of information about the relative position of the input tokens, the transformer further includes a sinusoid-wave-based positional encoding into the input embeddings to preserve the important distance information.

With the attention mechanism, transformers manage to solve long-range text generation much more powerfully and efficiently compared to previous recurrent models, and they have become the de facto structure in the field of Natural Language Processing. Later, three main types of transformer-based models have been developed: (1) decoder-based models, (2) encoder-based models and (3) encoder-decoder-based models.

Among decoder-based models, the GPT series (GPT, GPT-2, GPT3) [56, 57, 8] are most commonly used. They are composed of a stack of transformer decoder blocks and trained to predict the next token with previously provided tokens. We build our experiments on GPT-2 and we will discuss them in detail in the next section.

Encoder-based models are composed of a stack of transformer encoder blocks. Typically, they are trained with the "Masked Language Model" (MLM) objective, which involves reading a masking-corrupted sentence and predicting the masked word to complete the sentence based on the provided context. Due to transformer encoder blocks' inherent bidirectionality, these models are able to comprehend the context coming from both left and right collectively and develop a deeper understanding of the sentence compared to unidirectional decoder-based transformers. As an example, Google AI Language researchers propose Bidirectional Encoder Representations from Transformers (BERT) [11] and Robustly Optimized BERT (RoBERTa) [46] which have shown breakthroughs in a variety of NLP tasks such as natural language inference (MNLI task), sentiment analysis (SST2 task) and question answering (SQuAD task). While BERT can be used for text generation tasks [68], their performance is not particularly impressive because they are not trained in an autoregressive style from the beginning, and researchers normally seek decoder-based models like GPTs for natural language generation (NLG).

Meanwhile, models like Bidirectional and Auto-Regressive Transformers (BART) [34] and Text-to-text Transfer Transformer (T5) [58] adopt the standard encoder-decoder structure to generalize both autoencoding and autoregressive abilities. In BART, the encoder is trained to recover corrupted sentences, while its decoder is trained to generate text sequentially. T5 frames and unifies text-based problems into a text-to-text format where inputs are sentences with gaps and outputs are phrases that fit into these gaps to complete sentences. They achieve state-of-the-art results across multiple NLP tasks including document summarization, machine translation and text classification.

2.4.1 Generative Pretrained Transformers (GPTs)

The series of Generative Pretraining models [56, 57, 8] are all transformer-decoder-based models which follow the language modeling approach. More specifically, language models produce the conditional probability of a given sequence of words occurring in a sentence and their goal is to maximize the joint likelihood of all target words. Due to their ability to generate words sequentially, they can also be referred as autoregressive models. Despite increases in model size and dataset volume, the evolvement from GPT, GPT-2 to GPT-3 illustrates the strength and capability of *Pretraining*.

Before GPT, “Fully supervised learning” on every single task plays a pivotal role in numerous natural language processing tasks, and continuing to develop with novel architectures and algorithms. [3, 66] In order to achieve strong performance on a specific task, models are normally trained on a large amount of data samples with curated labels. Consequently, this leads to two major limitations. First, the model’s performance is highly dependent on the amount and quality of the human engineered dataset. It takes a lot of effort to create a satisfactory clean dataset, which typically requires thousands of thousands samples. Second, the model’s performance degrades dramatically when transferring from the training dataset to other datasets. i.e., the resulting model is task-specific, thereby limiting its applicability.

With the advent of GPT, academia has accelerated the development of a more human-like “Pretrain then fine-tune” paradigm, where language models can be first generatively learned on noisy, unlabeled, large-scale natural language tasks, developing an initial parameter representation from a variety of tasks, and then undergo minor adaptation to a specific domain with a supervised objective discriminatively, resulting in significant gains on different downstream scarce tasks. [22]

Later GPT-2 changes the format to provide instructions (the task description, input and output) in natural language words in pretraining and fine-tuning, and therefore unifies the supervised objective during fine-tuning to be the same as the unsupervised objective trained during the pretraining stage. Instead of designing different sequences and objectives elaborately during fine-tuning, GPT-2 is provided with natural language sequences, and is expected to do inferences based on natural words and provide the answers for different downstream tasks more easily. GPT-2 provides models up to 48 layers (1.5 billion parameters) which is 10 times larger than GPT and pretrained on 40GB text data from over 8 million documents. Though GPT-2 is competitive with supervised baselines in several language modeling datasets, it is still struggling with some tasks such as text summarization. In addition, the paper discloses an interesting phenomenon that GPT-2 with more parameters and trained on a larger dataset can surpass its smaller counterpart on many tasks. Attempting to build even stronger models, Open AI built GPT-3 which pushes the limit to 175 billion parameters, which is 100 times more parameters than GPT-2. Owing to its large capacity, it is capable of writing human-like articles and performing tasks it has never been explicitly trained to. The technique that the model develops its skills during the pretraining and being able to perform downstream tasks without gradient updates during fine-tuning is called “in-context learning” or ”prompting”, which we will explore in more detail in section 3.2.1.

2.4.2 Dialogue Generation Models

Dialogue generation models or conversational agents are designed to generate fluent and coherent responses based on given previous utterances. The advent of self-attention architecture along with the "Pretrain then fine-tune" paradigm has shed lights on dialogue generation, where a large number of recent open-domain dialogue systems have been built on large-scale transformer structures and extensive pretrained datasets. These works include Meena [1], DialoGPT [76], PLATO [4], PLATO-2 [5], BlenderBot [59], LaMDA [65]. Among these, Meena, DialoGPT, LaMDA are built on transformer-decoder-based language model like GPT-2 while PLATO, PLATO-2, BlenderBot utilizes a standard Seq2seq transformer structure to generate the next utterance. In this thesis, we concentrate on DialoGPT, which is a large, publicly available and flexibly tunable conversational agent. DialoGPT extends GPT-2 to strengthen the capability of dialogue generation by further training on 147M Redddit posts from 2005 through 2017. More importantly, DialoGPT provides different scales of transformer-decoder models from 12 to 48 layers for researchers to experiment with and has shown its flexibility and superiority when being fine-tuned on different downstream datasets. [40, 16]

In recent studies, scaling up model and dataset sizes has been shown to be beneficial for existing systems [76, 59, 65], and considerable effort has gone into expanding model size to improve generation capabilities. For instance, LaMDA scales the model to 137B parameters and requires a pretraining dataset containing 1.56T words. There emerges an increasingly urgent need to address the efficiency problem caused by these huge models: Does there exist any efficient method to steer these models flexibly in various downstream tasks? In this thesis, we will focus on adapting large-scale pretrained language models to different dialogue tasks in a minimally effortful manner.

Chapter 3

Related work

In the chapter, we describe related work in two domains where we are going to propose our approach. The first one is controlled dialogue generation, where we mainly focus on transformer-based fine-tuning methods. These methods can be classified into conditional fine-tuning and lightweight fine-tuning. The second type consists of Activation updating, Weighted decoding and Additive network based on where the pretrained model introduces the adjustment: pretrained model hidden states, output conditional distribution, or an additional small module. Another section depicts the development of prompting and prompt-tuning and shows the progression from task-level prompts, to attribute-level prompts and instance-level (instance-specific) prompts. By doing so, we pave the way for introducing instance-specific prompts in controlled dialogue generation in the next chapter.

3.1 Controlled Dialogue Generation

The advent of transformers [66] and large-scale training [56, 11] has advanced the development of dialogue generation, which aims to generate $P(X)$ unconditionally when given input sentence X. Despite their high quality and fluency[76, 59, 4, 65], these large conversational models generate responses based on conversation history without considering extra factors such as preferred topic, emotion, dialogue intent/action, etc. Meanwhile, controlled dialogue generation introduces control attribute A so that the model is expected to generate $P(X|A)$ within the restriction given by A. Normally, the control attribute can be the desired label attribute, such as a specific topic, language formality, or a series of sentences describing the user's persona or background knowledge. Compared to other controlled text generation tasks, controlled dialogue generation usually requires a higher level of consistency, engagingness, and informativeness [25].

The topic of controlled generation in large language models has received increased attention. There have been several successful PLMs-based approaches to generating controlled text using pretrained language models in recent years, and we classify them into four categories: (1) *Conditional fine-tuning* (2) *Activation updating* (3) *Weighted decoding* and (4) *Additive network*.

Conditional fine-tuning demonstrates that control code can be applied and learned during the training stage. [30] proposes Conditional Transformer Language (CTRL) which trains a conditional transformer from scratch and attempts to generate consistent content based on various control code such as govern styles, domains or even defined URLs. The method prepends control codes a to starting words $x = (x_1, \dots, x_n)$ and continues generating the following words by modeling the chain rule of probability as $P(x|a) = \prod_{i=n+1} P(x_i|a, x_1, \dots, x_{i-1})$. As a result, it releases a 48-layer, 1.63 billion-parameter conditional transformer which can be easily governed by control information. Nevertheless, this type of method requires updating all parameters in the model, which not only makes it difficult to scale up to larger models but also to integrate new control codes.

Activation updating performs conditional generation during the decoding stage rather than the training stage to avoid retraining large-scale pretrained language models for every single attribute. [10] introduces Plug-and-Play Language Models (PPLM) which is quite popular in controlled text generation. The intuition behind this method is to train a small attribute model on top of the frozen pretrained language model, utilize the output of the attribute model to iteratively perturb the language model's activation states using gradient descent, and finally generate tokens towards the expected control attribute.

We denote the expected control attribute as a and the sequence of input words generated so far as $x = (x_1, \dots, x_n)$. Under the recurrent generation view, we denote the language model's key-value pairs for every transformer layer as H_t at time-step t. Hence, H_0 represents the states of the pretrained language model before any generation. At every decoding step t, H_t will be shifted towards the sum of two gradients: (1) higher log-likelihood of the given attribute a under the conditional attribute model $p(a|x)$ and (2) higher log-likelihood of x_t under the unconditional language model $p(x)$. The first likelihood is computed by the aforementioned additional attribute model and the second likelihood is computed by Kullback-Leibler (KL) Divergence and Post-norm Geometric Mean Fusion generated between the modified language model (H_t) and the unmodified language model (H_0). In this way, the modified model is able to stay close to the pretrained version and retain its fluency while generating more attribute-related words.

For time-step t, let ΔH_t denote the update to H_t and ΔH_t has an initial value of 0, ΔH_t can be written as follows:

$$\Delta H_t \leftarrow \Delta H_t + \alpha \frac{\nabla_{\Delta H_t}(\log p(a|H_t + \Delta H_t) + \lambda \log p(H_t + \Delta H_t))}{\|\nabla_{\Delta H_t}(\log p(a|H_t + \Delta H_t) + \lambda \log p(H_t + \Delta H_t))\|^{\gamma}}$$

where α is the step size, γ is the scaling coefficient for the normalization term for each layer of the language model and λ is the scalar for the unconditional likelihood. After repeating a few steps (the paper uses 3 to 10 steps), the language model's state is updated as

$$H_{t+1} = H_t + \Delta H_t$$

There are following works [18, 48] that add different attribute models and additional strategies on top of PPLM and show satisfactory results. Though PPLM is flexible and light in comparison to full-model training, it has several drawbacks:

1. Since it involves a few steps of gradient updates at each decoding step, the generation is pretty time-consuming (see Table 4.1 and Table 4.2) and especially not suitable for interactive dialogue generation.

2. The method is restricted to label control or Bag of Words (BoW) control. In the paper, the control attribute model can be either a label classifier or a predefined BoW. In the BoW scenario, authors use a set of related keywords that specify the desired topic (e.g. "biology", "theory", etc. under the topic "science") and use the sum of the likelihoods of all keywords in the set to produce the desired gradient update. As a result, it becomes difficult to implement the method in sentence control scenarios such as using the user's persona, where the control attribute is several sentences rather than a label or a topic with curated keywords. Therefore, we use this method as one of the baselines in the label control scenario in later experiments, but not in the sentence control scenario.

Weighted decoding [17, 71, 31] is another alternative to prevent full-model fine-tuning which uses discriminators to steer the pretrained language model. Instead of modifying the base model's state representation, weighted decoding methods typically employ the attribute model to reweight the base model's output distribution. [17] presents a modified scoring function that perturbs the final output distribution and increases the likelihood of a bag of desired candidate tokens. Likewise, FUDGE [71] trains a binary discriminator to rescore the output conditional distribution via Bayes Rule. Specifically, given a sequence of words $x = (x_1, \dots, x_n)$ and the control attribute a, FUDGE avoids modeling $P(x_i|x_1, \dots, x_{i-1}, a)$ directly as *conditional fine-tuning*, and it utilizes the Bayesian factorization to transform the distribution as

$$P(x_i|x_1, \dots, x_{i-1}, a) \propto P(a|x_1, \dots, x_i)P(x_i|x_1, \dots, x_{i-1})$$

and it becomes sufficient to train the first term with an independent binary classifier. These two methods have shown promising results on text generation but they rely on either binary label or bag of words, which is not scalable to our multi-labels and document control tasks.

Later, GEDI [31] also proposes to transform the conditional probability via Bayes Rule, but it further normalizes the distribution over binary-class distributions with several heuristics: It utilizes control code a and a corresponding undesired or anti-control code $\bar{a}$ when training $P(a|x_1, \dots, x_i)$, which is not applicable to our tasks where there is no anti-control code in user's persona sentence description.

In addition, since weighted decoding methods are not involved in the pretrained model's computation and only modify the final distributions generated by the pretrained model, they are restricted by low-variance and high-biased pretrained models and can easily return common words instead of related targeting words when attribute weights are set low [14] or they may lose the fluency if attribute weights are set high[62].

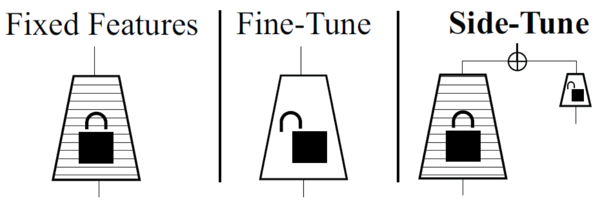


Figure 3.1: Side-tuning model structure from Zhang et al., 2020.

Recently, [73] extends the idea of *Weighted decoding* to *Additive network* with a method called Side-tuning (Figure 3.1), which adds a side conditional network to understand the control code and freeze the base model during the training. If the base model is denoted as B and the small side model is denoted as S, given input x, the curated output becomes $B(x) \oplus S(x)$. $\oplus$ can be defined as α blending where $\alpha \in [0, 1]$ is a learnable parameter to control the feature constraints, and the output becomes $\alpha B(x) + (1 - \alpha)S(x)$.

SideControl [14] implements this idea in controlled dialogue generation, where it utilizes the base model's pretrained language understanding ability to digest the conversation context, trains a side network to encode control attributes and performs a point-wise α

blending to generate the final output. Using the same annotation as Side-tuning, SideControl first generates the hidden states from the base model B on input x as $h_b = B(x)$. This is the same output as directly using a pretrained language model. Then SideControl uses the side network S, which they set as a bidirectional LSTM or a feedforward neural net, to encode the control attribute a into the base representation as $h_s = S(h_b, a)$. Finally, the final output h is combined using α blending as $h = \alpha h_b + (1 - \alpha) h_s$. Additionally, in order to train the side net to encode control codes more accurately and explicitly, they introduce the control attribute loss $L_{control}$ and the final loss is a combination of the cross entropy loss from the base model and control attribute loss, denoting as $L = L_{base} + \lambda L_{control}$ (λ is task-specific hyperparameter).

SideControl shows better controllability and text quality compared to other parameter-efficient methods in *Activation updating* and *Weighted decoding* categories. More importantly, they do not have restrictions over the control attribute and can serve as a competitive baseline model in our later experiments. However, the combination of the control attribute occurs after the base model's generation and its performance depends on the base model's ability to understand the conversation context. It is challenging to adapt the pretrained model to unfamiliar downstream tasks where the pretrained model may fail to generate reasonable base representations in the first place. Furthermore, SideControl overlooks the powerful attention mechanism within the pretrained model and it might be more expressive if the control code is injected directly into the pretrained model to participate in the attention computation.

To summarize, controlled dialogue generation has evolved into different categories to efficiently integrate control code with dialogue generation. These methods generally incorporate control attributes on top of the base model. However, the idea of inserting control attributes before the base model or interleaving the base model with control attributes is rarely explored, and we propose a new *Additive network* method that can interact with the pretrained model during the generation.

3.2 Prompt-tuning

3.2.1 Prompting

As mentioned earlier, "Pretrain then fine-tune" has been the dominant paradigm when deploying generative pretrained language models (PLMs) to downstream tasks since the advent of GPT [56] and BERT [11]. It has achieved remarkable performance in different

NLP tasks including question answering, commonsense reading, and so forth. However, this requires storing a full copy of parameter states for every downstream task during the fine-tuning stage, which is memory-consuming and expensive to serve when working with large-scale models like GPT-3 [8], which requires 175B parameters for each task.

The emergence of GPT-3, on the other hand, revealed a possibility to induce the pre-trained language model's learning ability without any tuning. For a given task, inputs are transformed into cloze-style phrases, along with none or some number of similar examples demonstrating the task, and the model is required to complete the sentence using a reasonable word over the masked token. The given task description and examples are known as discrete prompt and the method is called *prompting* or *in-context learning.* For example, when GPT-3 is given some instructions such as "Please unscramble the letters into a word, and the word: taefed = ", it can naturally finish the sentence by outputting "defeat". It can even imitate particular writers' work by only conditioning on the given title, the author's name and a few starting words. It turns out that GPT-3 can adapt its pretrained learning ability quickly towards specific tasks with mere task descriptions and possible guiding examples.

Later work [60, 61] proposes a promising application of GPT-3 called pattern-exploiting training (PET) and iterative PET (iPET) which incorporate minor gradient-based finetuning into GPT-3 and achieve much better performance compared to strong semi-supervised methods. They define a "pattern" that wraps every input into a predefined template with mask tokens and a "verbalizer" which maps meaningless labels to words in the model's vocabulary. For instance, in a movie review sentiment analysis task, we define the "pattern" to become "⟨sentence⟩, <u>the movie is</u> ⟨mask⟩" and "verbalizer" to be V where $V(y_0) =$ good and $V(y_1) =$ bad. If an input sentence is "It is totally worth the price", it will be transformed as "It is totally worth the price, <u>the movie is</u> ⟨mask⟩" and then be forwarded to the model. Since GPT-3 is pretrained in a generative manner, it may assign a higher possibility to positive labels such as "good" compared to "bad". The probabilities for two classes are then obtained using the softmax function applied to extracted vocab tokens as follows: $p(y_0) = \dfrac{\exp p('good')}{\exp p('bad') + \exp p('good')}$. Cross-entropy loss is computed between p and the desired output to update the parameters of the language model. In this way, they found that pretrained GPT-3 can quickly learn with mere 32 training examples and be used to annotate a large set of unlabeled data, which can be provided as training examples for other much smaller models' fine-tuning.

Prompting has demonstrated shockingly satisfactory performance on various downstream tasks after only one pretraining session. The "freezing" PLM with flexible prompts technique is appealing because one copy of the model can be shared across different tasks, resulting in a much smaller carbon footprint. However, the model's performance is largely restricted by the maximum conditional input length, the model size, and manual guesswork for prompts. Specifically, different prompts can generate significant instability in model performance[78, 60, 61, 28]. [78] has shown that prediction accuracy varies from 90% to 50% across different prompt formats on a sentiment analysis task SST-2. More importantly, creating effective prompts demands domain knowledge of datasets and a thorough understanding of language models.

Previous works have focused on automatically searching for better discrete prompts to better solve the task at hand (prompt engineering) [28, 64, 15, 6]. For example, AutoPrompt [64] generates a set of trigger tokens through a gradient-based strategy iteratively and concatenates them with inputs conforming to predefined templates to probe a masked language model's ability on several downstream application tasks. [15] introduces encoder-decoder-based pretrained model T5 to prune brute-force search on suitable label words and decode template tokens automatically. Though automated prompt search shows effectiveness in optimization and superior performance compared to manual prompt, *prompting* is still sub-optimal relative to *fine-tuning* [33, 45, 43]. In addition, some automatically generated prompts might seem mysterious and lack interpretability [64, 15]; for instance, "Hi" is chosen to represent the "entailment" class in SNLI, a natural language inference dataset. [15].

3.2.2 Prompt-tuning

Recently, there has been an increased interest in *continuous prompts / prompt-tuning*, which bridges the gap between prompting and fine-tuning, while remaining efficient during training [33, 38, 45, 44]. Continuous prompts remove the trouble of prompt engineering and extend the prompt selection to the entire space of embeddings, including vector embeddings that do not correspond to any human-interpretable natural language tokens. It commonly introduces another small module to encode and generate prompts so that the template or prompt tokens can be easily tuned on different tasks without being parameterized by the pretrained language models. As a result, soft prompts are found to be more expressive than discrete prompts [33, 43]. In prompt tuning, the prompt encoding module is commonly referred to as a "prompt encoder".

[33] introduces **Soft Prompt-tuning** (Figure 3.2) where the backbone model is frozen during the training and an additional embedding layer is updated for prompt encoding. Suppose the main frozen model is parameterized as $\mathbf{M}_\theta$ and the prompt encoder is parameterized as $\mathbf{E}_\alpha$, when given a data sample $D = (X, Y)$ where X represents the source

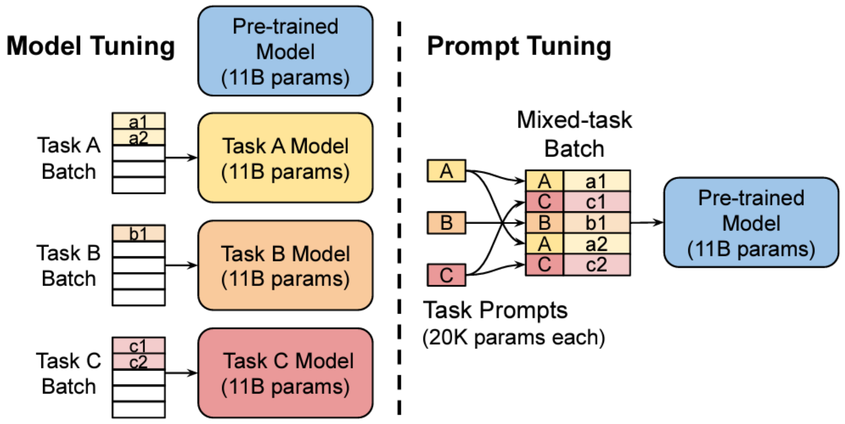

Model Tuning
Pre-trained Model (11B params)
Task A Batch
a1
a2
Task A Model (11B params)
Task B Batch
b1
Task B Model (11B params)
Task C Batch
c1
c2
Task C Model (11B params)
Prompt Tuning
Mixed-task Batch
A
B
C
A a1
C c1
B b1
A a2
C c2
Pre-trained Model (11B params)
Task Prompts (20K params each)

Figure 3.2: Soft Prompt Tuning model structure from Lester et al., 2021.

tokens and Y represents the target tokens, prompting (or discrete prompts) would add a sequence of task description words P to X and optimizes

$$Pr_{\mathbf{M}_\theta}(Y|[P;X])$$

Comparatively, **Soft Prompt-tuning** generates P' from E_α and maximizes

$$Pr_{\boldsymbol{M}_\theta,\boldsymbol{E}_\alpha}(Y|[P';X])$$

They show that the approach on T5-Large can outperform few-shot prompt design with GPT-3 175B (over 220 times larger) and even match the performance to full model-tuning when the underlying model scales up to T5-11B [58] with only 0.01% task-specific parameters introduced.

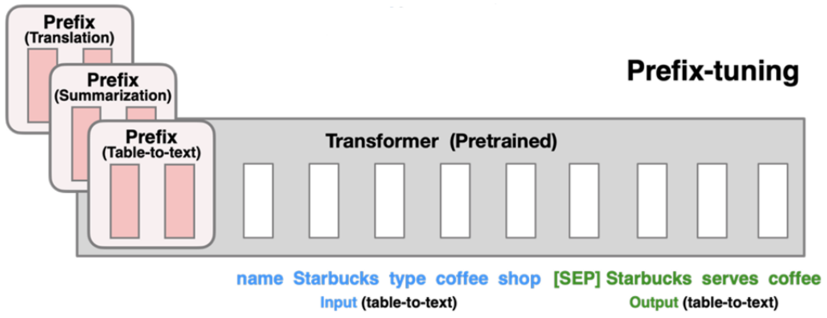


Figure 3.3: Prefix Tuning model structure from Li and Liang, 2021.

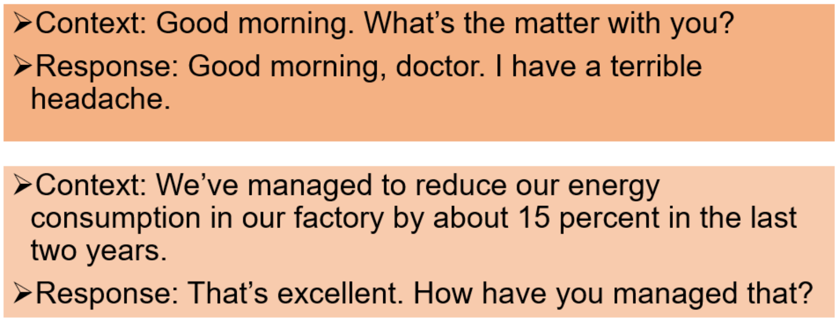
Concurrently, **Prefix-tuning** [38, 44] (Figure 3.3) proposes a more effective technique that adds soft tokens in the form of key-value pairs at every attention block of the transformer. Similar to the above annotation, the method changes the prompt encoder from an embedding layer $\boldsymbol{E}_\alpha$ to a multi-layer perceptron $\boldsymbol{m}_\alpha$ and the generated prompt becomes $P'' = [P'_1, P'_2, \dots, P'_l]$ where l represents the pretrained model's layer count. Prefix-tuning aims to maximize the following likelihood objective

$$Pr_{\boldsymbol{M}_\theta, \boldsymbol{m}_\alpha}(Y|[P''; X]) = Pr_{\boldsymbol{M}_\theta, \boldsymbol{m}_\alpha}(Y|[([P'_1, P'_2, \dots, P'_l) \diamond (h_1, h_2, \dots, h_l)])$$

where the term h_i is the input source hidden states at every transformer layer and $\diamond$ is layer-wise concatenation between prompt and intermediate source hidden states. Experimentally, Prefix-tuning achieves a comparable performance with fine-tuning with only 0.1% parameters in both table-to-text generation and summarization tasks. Prefix-tuning even outperforms fine-tuning in low data regimes and has better extrapolation to unseen topics by preserving pretrained language model parameters.

Later papers refer to the prompt added at the input embedding layer as *shallow prompt-tuning (or shallow prompt)* and the prompt prepended at every layer of the transformer as *deep prompt-tuning (or deep prompt).*

However, both deep prompts and shallow prompts assume a *static prompt / task-level prompt* for all samples within a task, neglecting the fact that samples might vary greatly and each scenario might require a different prompt [29]. Identifying an appropriate prompt dynamically for different scenarios is difficult, especially in the field of open-domain conversation generation where we may encounter diverse conversational contexts and responses within a task (Figure 3.4).



➢Context: Good morning. What's the matter with you?

➢Response: Good morning, doctor. I have a terrible headache.

➢Context: We've managed to reduce our energy consumption in our factory by about 15 percent in the last two years.

➢Response: That's excellent. How have you managed that?

Figure 3.4: Diverse dialogues within a task.

There are recent papers exploring possible finer-grained prompts in both *attribute-level* and *instance-level*.

Control-prefixes [9] utilize an attribute-level prompt along with a task-level prompt to generate output values. In every task, the method will train N attribute-level prompt candidates to cope with different features (e.g. “sport”, “technology” in the news domain context). Control-prefixes outperform Prefix-tuning in Text Simplification, Data-to-Text task, etc. However, Control-prefixes can handle only a limited number of labels and are not scalable to tasks with thousands of categories. Instance-Dependent Prompt Generation (IDPG) [70] proposes a finer-grained instance-level deep prompt by first generating the representation h_x of input x using the frozen pretrained language model and then generating deep prompt p where $p = f(h_x)$ and f is a light-weight prompt encoder. But it requires two passes through the frozen language model for every sample which causes a longer training and inference time. Additionally, the method is only shown capable of performing classification tasks rather than more complex generation tasks. Later Instance-aware prompt learning (IPL) [29] adds instance-specific shallow prompts by including a look-up module to re-weight prompt tokens before passing the updated embedding-only prompt into the transformer. But IPL updates all model parameters, which loses the efficiency benefits of prompting. Recent works such as Contrastive prefixes [53] and Tailor [72] both propose *attribute-based prompts* to include either single-attribute or multi-attribute prompts into controlled text generation tasks, which reveal the powerful potential of controllability of continuous prompts.

Although there has been a flourishing emergence of finer-grained prompt-tuning methods, few of them have investigated prompt-tuning in dialogue generation. Recently, [20] presents DialogPrompt which performs instance-specific prompting for dialogue generation by conditioning the prompt on the entire dialogue history. It shows that dynamic instance-specific prompts are more suitable for dialogue generation compared to generic task prompts. However, their prompting module consists of GPT-2, which is a full-fledged language model, and the approach is as costly as storing an entire fine-tuned base model.

Overall, despite the rise of attribute prompts and instance-specific prompts, tricks still appear, such as fully updating large-scale models or confining the method to classification tasks. In addition, the potential for parameter-efficient prompt-tuning has not been exploited yet in open-domain dialogue generation, including controlled dialogue generation.

Chapter 4

Methodology

4.1 Controlled DialogPrompt

In this work we present Controlled DialogPrompt (Controlled DP) for controlled dialogue generation inspired by static prompts from Soft Prompt-tuning [33] and Prefix-tuning [38]. Different from static prompts where prompts are fixed for a task, Controlled DialogPrompt is expected to provide control attribute information such as the dialogue intention or the user's persona within the prompt and steer the pretrained model efficiently.

Consider a controlled generation task where a sample dialogue $D = [u_1, u_2, \dots, u_N]$ of N tokens consists of the conversation context $C = [u_1, u_2, \dots, u_n]$ $(n < N)$ and the corresponding response $R = [u_{n+1}, u_{n+2}, \dots, u_N]$. The control attribute $A = [a_1, \dots, a_M]$ can be either a label with one token or a set of sentences with multiple tokens (detailed attribute settings are provided in Section 4.2.1).

We begin with the pretrained autoregressive language model $\mathcal{M}_\theta$ which is parameterized by pretrained weights θ. Here we use the pretrained DialoGPT [76].

Normally, fine-tuning (Figure 4.1 top) performs gradient updates based on the following objective:

$$\max_\theta P_\theta(R|A, C) = \sum_{L=n}^{N-1} \log P_\theta(u_{L+1}|a_1, \dots, a_M, u_1, \dots, u_L)$$

where P is the conditional distribution. L is the length of prior input tokens and the concatenated length is $M + L$.

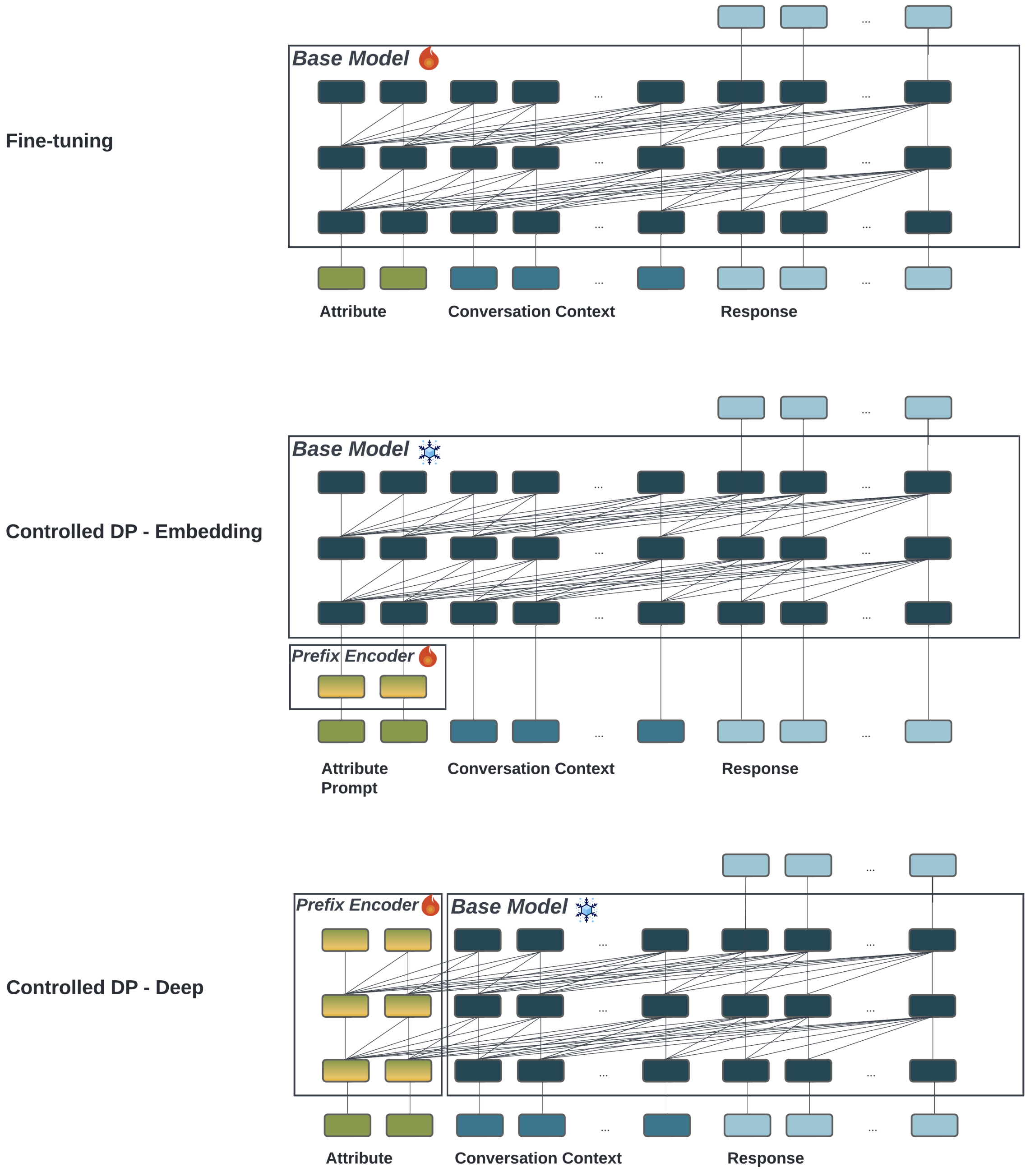


Figure 4.1: Diagrams illustrating Controlled DialogPrompt for response generation.

Concretly, the autoregressive backbone model first applies an input embedding layer $\boldsymbol{e_\theta}$ to convert the concatenation of attribute tokens and input tokens to a sequence of vectors of model dimension d_{model} as follows:

$$[\boldsymbol{e_\theta}(a_1), \ldots, \boldsymbol{e_\theta}(a_M), \boldsymbol{e_\theta}(u_1), \boldsymbol{e_\theta}(u_2), \ldots, \boldsymbol{e_\theta}(u_L)]$$

where $\boldsymbol{e_\theta}(a), \boldsymbol{e_\theta}(u) \in \mathbb{R}^d$.

The sequence of input embeddings then forwards through a stack of multi-headed self-attention layers followed by position-wise feedforward layers to produce an output distribution over target tokens. At every attention layer, the model calculates key, value and query as

$$K = UW_K,\ Q = UW_Q,\ V = UW_V$$

where $U \in \mathbb{R}^{(M+L)\times d_{model}}$ is input activation at every layer, $W_K, W_Q, W_V \in \mathbb{R}^{d_{model}\times h\times d_h}$, h is the number of parallel attention heads and $d_h = d_{model}/h$. The backbone model produces attention output as

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

with scaling factor $\dfrac{1}{\sqrt{d_k}}$.

And lastly, the model projects the final output from the last transformer block to the vocabulary-size logit vectors and derives the conditional probability for the next token.

In fine-tuning, control attribute A is encoded together with context C by the model's parameters θ. Though powerful, it requires a full-scale model update and is especially costly to serve when the model size scales up. Besides, recent studies have demonstrated that the model is prone to overfitting the training downstream task and its extrapolation ability deteriorates after tuning and shifting towards the task [38, 33]. In this sense, We attempt to freeze the pretrained model to preserve its pretrained learning ability as well as maintain the parameter-efficiency, and encode the control attribute separately from the conversation context so that the model can understand and modify the attribute information independently with greater flexibility. We would like to encode the control attribute with prompt encoders and in terms of the injected prompt depth, we classify these methods into *Controlled DialogPrompt-Embedding* and *Controlled DialogPrompt-Deep*. Briefly speaking, *Controlled DialogPrompt-Embedding* inserts the controlled prompt at the embedding layer, whereas *Controlled DialogPrompt-Deep* inserts the controlled prompt at every layer of the backbone.

Controlled DP-Embedding (Figure 4.1 middle) trains another embedding layer to encode the control attribute A. We denote this embedding layer as $\boldsymbol{e}_{\phi}$. The updated objective becomes

$$\max_{\phi} P_{\theta,\phi}(R|A,C) = \sum_{L=n}^{N-1} \log P_{\theta,\phi}(u_{L+1}|a_1,\ldots,a_M,u_1,\ldots,u_L)$$

and the input embedding changes to

$$[\boldsymbol{e}_{\boldsymbol{\phi}}(a_1),\ldots,\boldsymbol{e}_{\boldsymbol{\phi}}(a_M),\boldsymbol{e}_{\boldsymbol{\theta}}(u_1),\boldsymbol{e}_{\boldsymbol{\theta}}(u_2),\ldots,\boldsymbol{e}_{\boldsymbol{\theta}}(u_L)]$$

After concatenating the prompt embedding and the word embedding, the upper activation layers are computed the same by the pretrained model as fine-tuning.

Controlled DP-Deep (Figure 4.1 bottom), on the other hand, suggests enhancing the influence of the controlled prompt by generating key-value pairs prepended to each attention layer of the frozen pretrained dialogue system rather than only at the embedding. Mathematically, the objective is the same as for *Controlled DP-Embedding*:

$$\max_{\phi} P_{\theta,\phi}(R|A,C) = \sum_{L=n}^{N-1} \log P_{\theta,\phi}(u_{L+1}|a_1,\ldots,a_M,u_1,\ldots,u_L)$$

but the input embedding only needs to compute context tokens as:

$$[\boldsymbol{e}_{\boldsymbol{\theta}}(u_1),\boldsymbol{e}_{\boldsymbol{\theta}}(u_2),\ldots,\boldsymbol{e}_{\boldsymbol{\theta}}(u_L)]$$

The key difference is that for every attention block, the prompt encoder generates an additional set of key-value pairs beforehand, denoted as K_a, $V_a \in \mathbb{R}^{K\times h\times d_h}$ and the pretrained model will integrate and process both prompt key-value pairs and context key-value pairs at the same time in the forward pass. The attention computing now becomes:

$$\begin{gathered}
\text{Attention}(Q',K',V') = \text{softmax}(\frac{Q'K'^T}{\sqrt{d_k}})V' \\
Q' = U'W_Q \\
K' = \textbf{concat}(\boldsymbol{K_a}, U'W_K) \\
V' = \textbf{concat}(\boldsymbol{V_a}, U'W_V) \\
U' \in \mathbb{R}^{\boldsymbol{L}\times d_{model}}, W_K, W_Q, W_V \in \mathbb{R}^{d_{model}\times h\times d_h} \\
Q' \in \mathbb{R}^{\boldsymbol{L}\times h\times d_h}, K', V' \in \mathbb{R}^{\boldsymbol{(M+L)}\times h\times d_h}
\end{gathered}$$

Finally, the pretrained model calculates the conditional probability for the next token by projecting the final output of the last transformer block onto the logit vectors.

Controlled DP-Deep facilitates the controlled generation in the following ways:

(i) Controlled DP freezes the pretrained models and trains a versatile prompt so that it can retain the functionality of the pretrained model and guide the model to generate restricted text through training and selecting appropriate prompts and hence fulfill the objective of controllability.

(ii) Controlled information is given as additional key-value pairs K_a, V_a, so it won't take place in the context window size of the input to the pretrained transformer. It becomes possible to extend the controlled attribute to extremely long sentences, such as conversation background knowledge documents because we no longer have a length restriction imposed by the pretrained model. Since prompted key-value pairs are precomputed, we can also compute document representations offline without a heavy computational load per sample.

(iii) Compared to *Controlled DP-Embedding*, *Controlled DP-Deep* injects attributes as the form of key-value pairs at every layer, which directly participates in pretrained model's attention computation and interpolates original attention output and is therefore much more expressive than the embedding-only prompt.

4.2 Experimental Design

4.2.1 Datasets

DialogAct Label Control

Dailydialog [39](Figure 4.2) is a widely used daily conversation dataset that provides a dialogue act for every sentence. Dialogue acts indicate the communication function of each utterance and there are 4 types of dialogue acts: Inform, Questions, Directives, and Commissive: (1) ”Inform” indicates the speaker should provide information within the sentence; (2) ”Questions” means the speaker should raise a question to seek for some information; (3) ”Directives” indicates the speaker should come up with a request, instruct, suggest and (4) ”Commissive” means the speaker should capture the user's commitments to perform certain actions such as accepting/rejecting requests or suggestions.

We follow the standard split of the original Dailydialog dataset, limit the conversation context to a maximum of four sentences, and remove any sentence that has more than 25 words to maintain computation efficiency. As a result, we obtain 61,669 training samples, 5769 validation samples, and 5453 testing samples.

During the response quality evaluation, We additionally use the Dailydialog multi-reference dataset from [21] in metrics computation, which provides 5 references to each response in order to mitigate the one-to-many possible response problem.

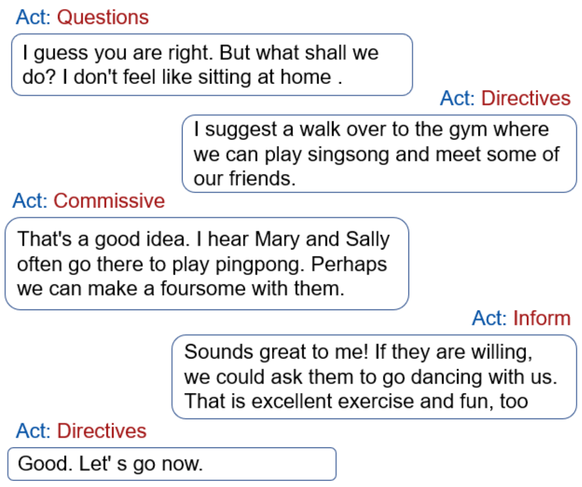

Act: Questions
I guess you are right. But what shall we do? I don't feel like sitting at home .
Act: Directives
I suggest a walk over to the gym where we can play singsong and meet some of our friends.
Act: Commissive
That's a good idea. I hear Mary and Sally often go there to play pingpong. Perhaps we can make a foursome with them.
Act: Inform
Sounds great to me! If they are willing, we could ask them to go dancing with us. That is excellent exercise and fun, too
Act: Directives
Good. Let' s go now.

Figure 4.2: Example dialogue from DailyDialog dataset.

User's Persona Document Control

FoCus[26](Figure 4.3) is a new persona-grounded dataset. Unlike DailyDialog, FoCus aims to build a dialogue agent that provides informative answers based on the user's persona about the geographical landmark; therefore, it is more content-rich and challenging. The selected knowledge candidate sentence is prepended to the conversation and regarded as part of the input. For every utterance, the dataset provides 5 user's persona sentences to condition on, and we include all of them without manual selection, so we can simulate real-life settings more accurately. In other words, models are expected to condition on related personas.

Since the grounded answer of the test set has not been released, we shuffle and split the original training set to construct our training samples and validation samples (70% training and 30% validation) and the original validation set as our testing samples. We further restrict conversation context to at most three sentences because the bot's utterances are much longer than human's utterances. In total, we have 8738 conversations for training, 3746 conversations for validation, and 1000 conversations for testing (49,198 samples for training, 21,134 samples for validation, and 5,639 samples for testing).

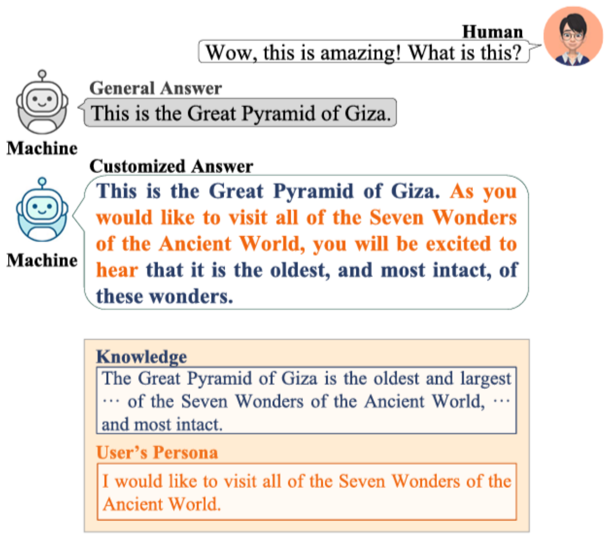

Human
Wow, this is amazing! What is this?
General Answer
This is the Great Pyramid of Giza.
Machine
Customized Answer
This is the Great Pyramid of Giza. As you would like to visit all of the Seven Wonders of the Ancient World, you will be excited to hear that it is the oldest, and most intact, of these wonders.
Machine
Knowledge
The Great Pyramid of Giza is the oldest and largest ⋯ of the Seven Wonders of the Ancient World, ⋯ and most intact.
User's Persona
I would like to visit all of the Seven Wonders of the Ancient World.

Figure 4.3: Example dialogue from Focus dataset.

4.2.2 Baseline Model Settings

To demonstrate the better performance of Controlled DialogPrompt, we compare our model with other competitive controlled dialogue generation methods and prompt-tuning techniques.

During our experiments, we have set both DialoGPT-Medium and DialoGPT-large as the frozen backbone model and trained all models on two Nvidia V100 32G GPUs. Models are trained for 10 epochs with a training batch size of 16 and a learning rate of 1e-4 except for fine-tuning, which is set to 5e-5 in the FoCus dataset and 1e-5 in the Dailydialog dataset. When setting the learning rate for each experiment, we search from the set $\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$ and choose the one rendering lowest evaluation loss.

- **Pretrained DialoGPT** [76]: DialoGPT-Medium and DialoGPT-large has shown its superiority for a wide range of open-domain dialogue generation tasks by pretraining on a massive corpus. The goal is to demonstrate that the model has learned an understanding of general language modeling during the pretraining phase.

- **Fine-tuning**: Fine-tuning, though memory-consuming, is the most straightforward and prevalent adaptation technique to downstream tasks. Fine-tuning has been considered as the benchmark for all light-weight fine-tuning methods including prompt-tuning.

- **PPLM** [14]: Plug-and-Play Language Models(PPLM) is a competitive controlled dialogue generation baseline model which updates the language model's activation states at every decoding step. It is only included in the label control comparison. In order to ensure the model can generate a full response, we set the maximum generation length to 20, which is twice the average response length. Besides, we set the number of gradient updates to 10, step size $\alpha = 0.2$, kl scale $\lambda = 0.01$, scaling coefficient $\gamma = 0.95$.

- **SideControl** [14]: This is a competitive controlled dialogue generation baseline model which adds a side network on top of the base model and leverages the output distribution with control signals. We follow the training setting in this paper for label control (Dailydialog dataset) and we search and select the hyperparameter $\lambda = 1e-5$ in document control (Focus dataset) and training batch size 16 with others kept the same.

- **Soft Prompt-tuning (static shallow prompt)** [33]: The method applies a static task prompt to the embedding of every input. We experiment with different lengths $\{10, 50\}$ of the static shallow prompt and use the better length 50 based on the lower evaluation loss.

- **Prefix-tuning (static deep prompt)** [38]: Prefix prompts are added to every layer during computation. We experiment with different lengths of $\{10, 50\}$ and we report the better prompt result with length 10 which shows lower evaluation loss.

- **Controlled DP - Embedding (instance-specific shallow prompt)**: The shallow version of our method with controlled prompts is added only in the embedding layer. It is used to demonstrate the expressiveness of the deep Controlled DialogPrompt.

  In the label control scenario (Dailydialog dataset), the embedding layer size is set to [label_size $\times$ model_hidden_size] to encode several labels, which is close to Soft Prompt-tuning and is much smaller compared to Controlled DP-Deep models. In the document control scenario (Focus dataset), the embedding layer size is set to [vocab_size × model_hidden_size] in order to encode words, resulting in a significant increase in parameter size because DialoGPT's vocabulary size is 50257.

- **Controlled DP - Deep: Controlled DP - MLP / 2-layer Transf-Dec (instance-specific deep prompt)**: We explore different prompt encoder structures, among which MLP prompt encoder shares the frozen pretrained transformer embedding layer to reduce tunable parameters.

  In Controlled DP - MLP, we follow the setting and the reparameterization trick in Prefix-tuning where we design two fully connected layers with tanh activation applied to each token of the control attribute, but the hidden size is changed to 512. In Controlled DP - 2-layer Transf-Dec, we train a two-layer transformer decoder with an embedding size of 256. The embedding size of each architecture was chosen to yield roughly the same number of parameters. This number of parameters is about 5%-6% of the number of parameters of the language model.

During our experiments, we have set both DialoGPT-Medium and DialoGPT-large as the frozen backbone model and trained all models on two Nvidia V100 32G GPUs. Models are trained for 10 epochs with a training batch size of 16 and a learning rate of 1e-4 except for fine-tuning, which is set to 5e-5 in the FoCus dataset and 1e-5 in the Dailydialog dataset. When setting the learning rate for prompt-tuning methods, we search from the set $\{1e-5, 5e-5, 1e-4, 5e-4, 1e-3\}$ and choose the one rendering the lowest evaluation loss. Models that achieve the lowest validation losses are saved during the training. We perform optimization with the AdamW optimizer with maximum gradient clipping set to 1. For decoding, we choose top-k sampling provided in Huggingface where k=10 and temperature T=0.9. The random seed is set to 42 in all of our experiments.

4.2.3 Evaluation Methods

Automated metrics

We are going to evaluate both the controllability and the response quality of different controlled methods.

Controllability We mainly follow [14] to evaluate whether models can customize responses based on specified control attributes for controllability.

In label control, we fine-tune an independent BERT classifier [11] which can take a sentence and predict its dialogue intention. We train the classifier on the same training set and achieve 83.23% accuracy on the test set. During the training, we use Adam optimizer with a linear warmup of 100 steps for the learning rate $1e-5$, and we set evaluation_step to 500 and save the model with the lowest evaluation loss.

In document control, we also compute the cosine similarity between the GloVe embedding of the generated responses and grounded persona documents. GloVe is an unsupervised learning algorithm trained to derive word embeddings by aggregating global word co-occurrence matrices from a given corpus. Here we obtain the sentence embedding by averaging all the word embeddings within a sentence. The GloVe embedding is pretrained on Wikipedia 2014 + Gigaword 5 with 6B tokens and 400K vocabulary size. We use vectors with dimensionality 100. Since the FoCus dataset contains human-annotated labels for persona sentences, only those that are actually used are evaluated.

Response Quality

How can we evaluate the performance of a conversational agent? A simple way to evaluate the response is to show the generated sentence side-by-side with the grounded responses and ask humans to rate its quality. But human evaluation is expensive and time consumed that cannot be reused. We present several widely used automatic evaluation metrics to provide more comprehensive results to evaluate different conversational agents' abilities. We utilize different variants of n-gram-based metrics such as BLEU (B-2, B-4) [52], NIST (N-2, N-4) [13], ROUGE-L [41], METEOR [2] to evaluate fluency and adequacy and distinct n-gram distribution metrics such as Dist (D-1, D-2) [35] and Entropy (E-4) [75] to measure the diversity of the response.

BLEU BiLingual Evaluation Understudy(BLEU) [52] is a popular automatic metric for evaluating machine-translated text and has been widely used in dialogue generation. BLEU scores measure the degree of overlapping between machine-generated text named hypothesis and a set of references, ranging from 0 to 1. The metric firstly involves computing the precision of n-grams for the defined n: all candidate n-gram counts are collected and clipped by their corresponding maximum count in all the references, summed and divided by the total number of all unclipped candidate n-gram counts. If we have M candidate n-grams in total and denote the precision score as p_n for n-gram, then

$$M = \sum_{\text{n-gram} \in \text{Candidate}} Count(\text{n-gram})$$

$$p_n = \frac{1}{M} \sum_{\text{n-gram} \in \text{Candidate}} Count_{clip}(\text{n-gram})$$

Let c be the length of the candidate and r be the closest reference sentence length, then the metric requires the calculation of the brevity penalty BP to discourage excessively short candidates:

$$BP = \begin{cases} 1, & \text{if } c > r. \\ e^{(1-r/c)}, & \text{otherwise.} \end{cases}$$

Then, BLEU is calculated as the multiplication of the brevity penalty with the geometric average of the precision scores.

$$\text{BLEU} = BP \ \exp(\sum_{n=1}^{N} w_n \log p_n)$$

In our experiments, we report the more commonly used BLEU-2 and BLEU-4 [42], i.e. $N = 2$ or 4 and use uniform weights $w_n = 1/N$.

NIST NIST [13] is a variant of BLEU which assigns heavier credits for rarer words and values more on informative n-grams. The information weight for each n-gram is computed as:

$$\text{Info}(w_1 \dots w_n) = \log_2(\frac{\text{the \# of occurrences of } w1 \dots w_{n-1}}{\text{the \# of occurrences of } w1 \dots w_n})$$

In addition, NIST changes the geometric mean of co-occurrences to the arithmetic average to prevent counterproductive variance in low co-occurrences for larger N. It also switches to a smaller brevity penalty for smaller variations in candidate lengths.

ROUGE-L Recall Oriented Understudy for Gisting Evaluation (ROUGE) [41] is a set of metrics specifically designed for evaluating automatic summarization, and can also used in dialogue generation evaluation. BLEU focuses on precision, that is how many n-grams in the candidate appear in the reference, whereas ROUGE focuses on recall, that is how many n-grams in the reference are displayed in the candidate sentence. If we use a similar format as BLEU that we have M reference n-grams in total and denote the recall score as ROUGE-N for n-gram, then

$$M = \sum_{\text{n-gram} \in \text{Reference}} Count(\text{n-gram})$$

$$\text{ROUGE-N} = \frac{1}{M} \sum_{\text{n-gram} \in \text{Reference}} Count_{match}(\text{n-gram})$$

where $Count_{match}(\text{n-gram})$ is the maximum number of n-grams co-occurring in a candidate sentence.

We use an advanced version called ROUGE-L which is based on the longest common subsequence(LCS). Basically, the LCS is the longest sequence of words shared between the candidate and reference, and words do not have to be consecutive, but they must be arranged in the correct order. It has the flexibility to automatically include the longest in-sequence common n-grams and it is inexpensive to compute. In addition to BLEU metrics, it serves as a complementary tool. However, this method does not manage synonyms and only measures syntactical matches.

METEOR METEOR [2] calculates both precision and recall and includes a weighted F-score to capture the matched words' order nicely. If more than one reference translation is possible, the metric provides the best score.

The algorithm first finds a set of mappings between unigrams named alignment between the candidate and the reference by calculating either through the exact match, stem match, or synonymy match. After the final alignment is formalized, suppose there is m unigrams mapping, m_c, m_r is the length of candidate and response, then precision P and recall R is calculated as:

$$P = \frac{m}{m_c}, R = \frac{m}{m_r}$$

Then $Fmean$ is computed by combining P and R via a harmonic-mean which renders:

$$Fmean = \frac{10PR}{R + 9P}$$

Since previous steps focus on unigram mapping and the metric seeks longer-term matching, it adds a penalty p to encourage longer matches. This firstly requires all adjacent unigrams to be grouped together to form the fewest possible chunks. Intuitively, the longer consecutive adjacent unigrams (n-grams) exist, the fewer chucks will be formed. Suppose c is the number of chunks it forms and u is the number of unigrams that are previously mapped, the final score s is calculated as:

$$p = 0.5(\frac{c}{u})^3$$

$$s = (1 - p)Fmean$$

Dist Diversity(Dist) [35] is a straightforward method to report the degree of diversity by simply calculating the number of distinct unigrams and bigrams in the generated responses. Thus it is a reference-free metric and is divided by the total number of generated tokens. It is recently used in the neural conversation model evaluation [75, 76, 10] as a measurement to avoid repetitive and dull responses.

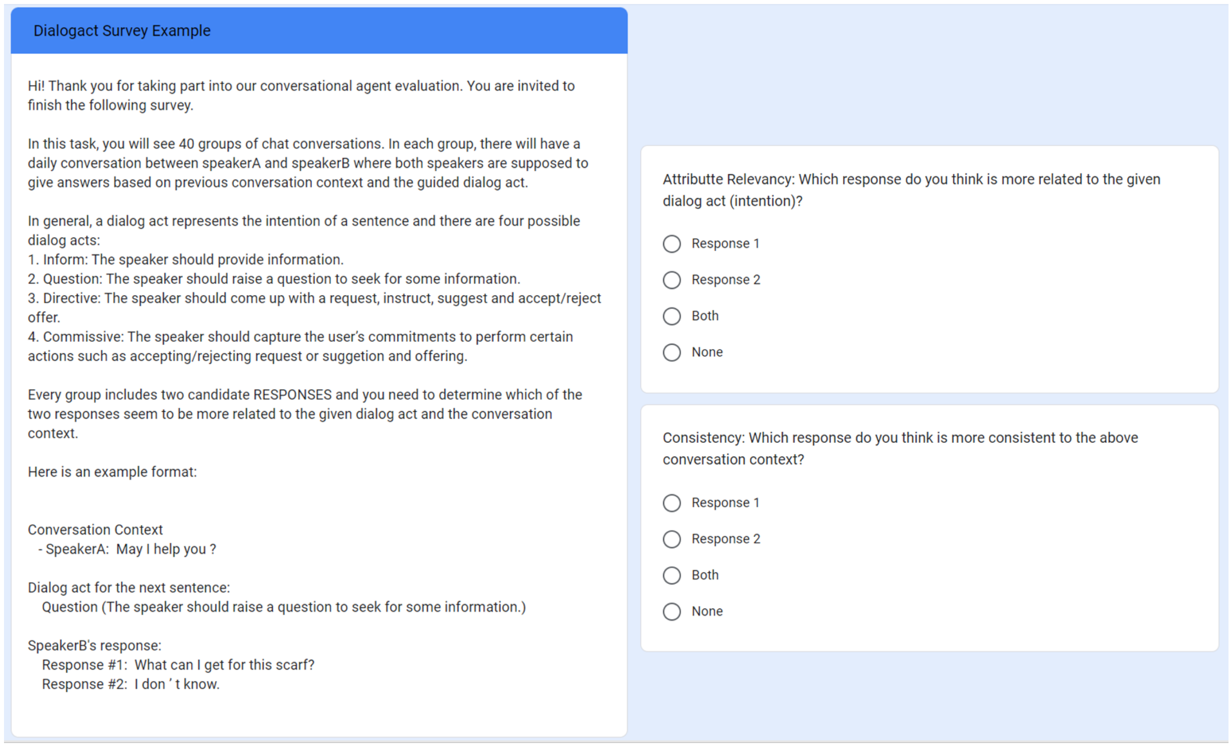
Entropy Entropy-N(E-N) [75] points out that Dist metrics overlook the n-grams frequency differences. If n-gram A and n-gram B occur m times each, it is preferred than n-gram A occurring once but n-gram B occurring $2m - 1$ times. In that case, Entropy-N takes the n-gram distribution into account and prefers to have a more even distribution of different n-grams. Denote $f(m)$ as the frequency of the n-gram m and E-N is calculated as follow:

$$M = \sum_{m \in \text{generated tokens}} f(m)$$

$$\text{E-N} = -\frac{1}{M} \sum_{m \in \text{generated N words}} f(m) \log \frac{f(m)}{M}$$

Avg Len It reports the average length of generated response sentences in the corpus.

Decoding Cost We follow [14] to report the average decoding second per token, which can be used to compare generation efficiency across different models. We pick 30 dialogue contexts as the testing set and calculate each model's total decoding time which is divided by the total number of tokens produced. The metric is set to clearly reflect the high computational cost of PPLM method.



Dialogact Survey Example

Hi! Thank you for taking part into our conversational agent evaluation. You are invited to finish the following survey.

In this task, you will see 40 groups of chat conversations. In each group, there will have a daily conversation between speakerA and speakerB where both speakers are supposed to give answers based on previous conversation context and the guided dialog act.

In general, a dialog act represents the intention of a sentence and there are four possible dialog acts:
1. Inform: The speaker should provide information.
2. Question: The speaker should raise a question to seek for some information.
3. Directive: The speaker should come up with a request, instruct, suggest and accept/reject offer.
4. Commissive: The speaker should capture the user's commitments to perform certain actions such as accepting/rejecting request or suggetion and offering.

Every group includes two candidate RESPONSES and you need to determine which of the two responses seem to be more related to the given dialog act and the conversation context.

Here is an example format:

Conversation Context
- SpeakerA: May I help you ?

Dialog act for the next sentence:
Question (The speaker should raise a question to seek for some information.)

SpeakerB's response:
Response #1: What can I get for this scarf?
Response #2: I don ' t know.

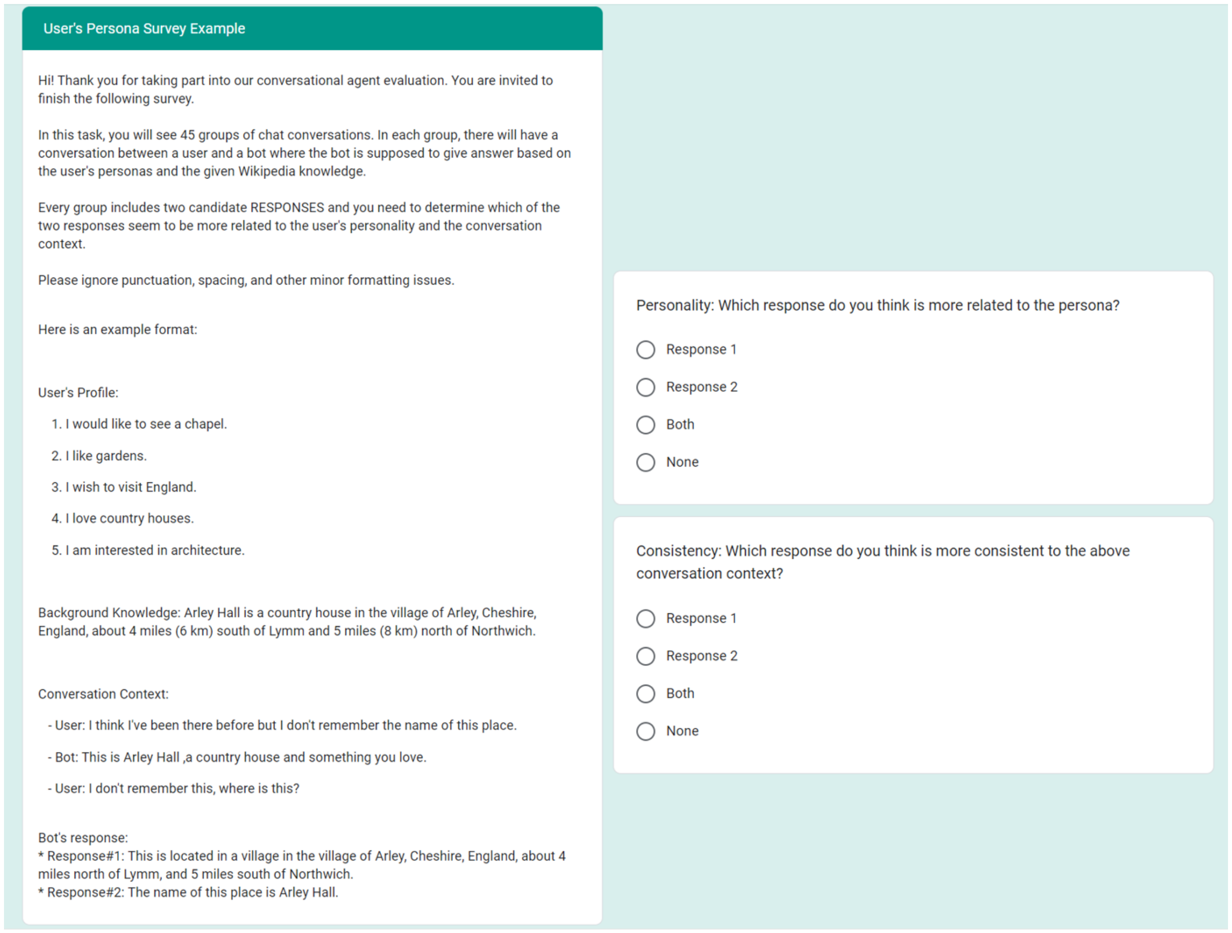
Attributte Relevancy: Which response do you think is more related to the given dialog act (intention)?

- ○ Response 1
- ○ Response 2
- ○ Both
- ○ None

Consistency: Which response do you think is more consistent to the above conversation context?

- ○ Response 1
- ○ Response 2
- ○ Both
- ○ None

Figure 4.4: Human evaluation survey snippet of DialogAct.


User's Persona Survey Example

Hi! Thank you for taking part into our conversational agent evaluation. You are invited to finish the following survey.

In this task, you will see 45 groups of chat conversations. In each group, there will have a conversation between a user and a bot where the bot is supposed to give answer based on the user's personas and the given Wikipedia knowledge.

Every group includes two candidate RESPONSES and you need to determine which of the two responses seem to be more related to the user's personality and the conversation context.

Please ignore punctuation, spacing, and other minor formatting issues.

Here is an example format:

User's Profile:

1. I would like to see a chapel.
2. I like gardens.
3. I wish to visit England.
4. I love country houses.
5. I am interested in architecture.

Background Knowledge: Arley Hall is a country house in the village of Arley, Cheshire, England, about 4 miles (6 km) south of Lymm and 5 miles (8 km) north of Northwich.

Conversation Context:

- User: I think I've been there before but I don't remember the name of this place.
- Bot: This is Arley Hall ,a country house and something you love.
- User: I don't remember this, where is this?

Bot's response:
* Response#1: This is located in a village in the village of Arley, Cheshire, England, about 4 miles north of Lymm, and 5 miles south of Northwich.
* Response#2: The name of this place is Arley Hall.

Personality: Which response do you think is more related to the persona?

○ Response 1
○ Response 2
○ Both
○ None

Consistency: Which response do you think is more consistent to the above conversation context?

○ Response 1
○ Response 2
○ Both
○ None

Figure 4.5: Human evaluation survey snippet of User's Persona.

Human Evaluation

In order to demonstrate the expressiveness of (1) instance-specific prompts and (2) deep prompts, we carry out the human evaluation on Controlled DialogPrompt with every other competitive prompt-tuning methods, covering static shallow prompt, static deep prompt, and instance-specific shallow prompt. We pick the base model to be DialoGPT-Large.

Human evaluation is set to measure consistency between dialogue context and response and attribute controllability. Similar to ACUTE-Eval in [37, 59], we adopt single-turn pairwise evaluations to prevent annotator bias in numerical score evaluation. In each comparison group, there are two questions designed separately to assess the response's dialogact/persona controllability as well as consistency to the previous conversation context. We sample 15 conversations from each comparison group and there are 5 conversations overlapped across different groups. Annotators are industrial NLP researchers and NLP graduate students. We collected 900 annotations in total. Surveys are conducted using Google Forms, and snippets from each scenario are provided separately in Figure 4.4 and Figure 4.5.

4.3 Results and Analysis

Table 4.1 summarizes the automatic evaluation results on the DialogAct label control task with the base model as DialoGPT-Medium. Table 4.2 summarizes the automatic evaluation results on the DialogAct label control task with the base model as DialoGPT-Large.

Controlled DialogPrompt - Deep outperforms all other parameter-efficient controlled methods on controllability and achieves accuracy pretty close to fine-tuning. It confirms the effectiveness of injecting the controlling attribute into every layer of the base model, which can interfere with the language modeling directly. Though Controlled DialogPrompt - Deep achieves satisfactory results on language fluency, SideControl achieves higher scores on NIST and Entropy. NIST is weighted-BLEU with higher weights on rarer words and Entropy-4 counts the distribution of distinct 4-grams, and we notice SideControl tends to generate more diverse phrases than other models which makes it outperform on these metrics. PPLM achieves surprisingly better performance on Diversity, but it incurs a much higher decoding cost, which is 60 times slower than Controlled DP with DialoGPT-Medium and 100 times slower than Controlled DP with DialoGPT-Large. As the base model grows, the cost of updating all its parameters increases. This impedes PPLM from being implemented with more complex models such as GPT-3.

Compared to Prefix-tuning (static task prompts), Controlled DialogPrompt - Deep achieves better performance consistently, especially on deep prompt levels. Since the controlled attribute is injected independently through the prompts, it does not affect the understanding and generation ability of the pretrained transformer. Both Controlled DP deep methods show higher controllability and response quality than Controlled DP embedding, in line with [38, 44, 54] indicating the expressiveness of deep prompts. Also, Controlled DP deep methods show performance close to fine-tuning and even outperform on some metrics such as NIST when training less than 5% of total parameters.

Human evaluation result table 4.3 also shows that Controlled DP deep has a significantly higher winning rate than other prompting techniques on both control attribute relevancy and conversation consistency. We provide sample generated dialogues in 4.7 and 4.8. In these examples, Controlled DP - Deep is highly compatible with provided dialogue intention (DialogAct) while generating a fluent and well-coordinated response.

Table 4.4 summarizes the automatic evaluation results on the user's persona document control task with the base model as DialoGPT-Medium. Table 4.5 summarizes the automatic evaluation results on the user's persona document control task with the base model as DialoGPT-Large.

Unlike simple daily conversations in the label control scenario, dialogue generation based on the user's persona and knowledge-rich FoCus dataset is much more challenging and requires careful consideration of the control attribute. In this example, SideControl significantly degrades its performance when only manipulating the control attribute on top of the base model. It has difficulty adhering to its goal of generating relevant responses. It extracts some keywords from the knowledge and the conversation context and creates hallucinated facts on the topic, which can be observed in the provided example 4.9. In contrast, methods, such as Prefix-tuning and Controlled DialogPrompt, that provide controlled information before or interleaving the base model form a better understanding of the user's persona and maintain a complete generation of dialogue responses.

Particularly, Controlled DialogPrompt displays advantages over other prompting methods in terms of response quality, which shows a promising sign that controlled DialogPrompt can be adapted to more challenging document control scenarios. Although controlled DialogPrompt methods perform slightly lower than Prefix-tuning on the similarity scores with given user's persona and Entropy-4 values, we find it to be highly consistent with the previous conversation history in human evaluation (Figure 4.6). Similar results are observed with FoCus [26] where models with high generation abilities do not always ensure high grounding abilities. In addition, the difference between static/instance-specific deep prompts and static/instance-specific shallow prompts emphasizes the direct impact of deep prompts in complex tasks. The advantage of deep prompts is consistent in different sizes of base models (DialoGPT-Medium and DialoGPT-Large). Fine-tuning performs the best but with approximately 20 times more tunable parameters. We provide sample generated dialogues in 4.9 and 4.10. In these examples, Controlled DP - Deep can flexibly quote provided user's persona based on given dialogue contexts and generate more personalized responses.

Wikipedia Knowledge Document Control

From previous experiments, we observe Controlled DialogPrompt's superiority in controlling user's persona. Controlled DialogPrompt can integrate the provided personas flexibly and generate related responses conditioned on previous conversation context. For example, in Example 4.10, Controlled DialogPrompt - Deep takes in *"it is among the greatest examples of Georgian architecture"* and *"I am interested in architecture"* and generates *"The Royal Crescent is one of the Georgian architectural masterpieces. You should really visit this place since you love architecture!"*. The generation is derived from provided information but with a reasonable minor adjustment, which enriches the corpus.

However, we later evaluate Controlled DialogPrompt with knowledge being injected as the prompt and unexpectedly find that the performance greatly degrades. Table 4.11 shows the automatic evaluation results and Table 4.12 4.12 gives a few examples to better illustrate our points. Compared with previous persona based Controlled DialogPrompt (denoted as "Controlled DialogPrompt (on Persona)"), Controlled DialogPrompt (on Knowledge) does not have knowledge concatenated with conversation context to form the input. The grounded knowledge sentence is provided in the prompt and the model is expected to generate the response when knowledge is encoded by the prompt encoder and conversation context is fed as input to the base model. Moreover, we provide Fine-tuning baselines with and without knowledge to visualize lower and upper bounds for knowledge usage.

From the table, Controlled DP (Knowledge) performs much worse than Controlled DP (Persona) and just performs slightly better than Fine-tuning without knowledge. Combined with the provided examples 4.12, we notice the model cannot copy the information precisely from the provided knowledge sentence. To illustrate, it modifies *"Grade I listed building"* to *"Grade VIII listed building"* and replaces *"from 1924 to 1939"* with *"back in 1926"*. Presumably, we expect the model to condition on / relate to the user's persona but not to copy the knowledge sentence accurately. Controlled prompt might be suitable for conditioning but not information repetition. Therefore, when implementing knowledge prompts that rely heavily on knowledge during response generation such as Focus, we find that the model has difficulty replicating the knowledge exactly as specified. In our future

Method	ϕ%	**Controllability Accuracy**	**BLEU ↑**		**NIST ↑**		**ROUGE-L ↑**
			B-2	B-4	N-2	N-4	
Pretrained	0%	46.60%	14.92%	3.05%	**1.54**	**1.57**	27.58%
Fine-tuning	100%	**79.90%**	**20.78%**	**5.81%**	1.18	1.21	**34.23%**
PPLM	0.001%	69.74%	15.54%	3.10%	1.06	1.08	26.86%
SideControl	1.8%	53.42%	18.03%	4.52%	**1.27**	**1.29**	31.84%
Soft Prompt-tuning	0.014%	67.23%	19.13%	4.83%	0.99	1.01	32.39%
Prefix-tuning	3.6%	74.93%	19.19%	4.70%	1.07	1.09	32.87%
Controlled DialogPrompt (Embedding)	0.001%	68.42%	18.60%	4.57%	1.08	1.10	32.17%
Controlled DialogPrompt (MLP)	3.6%	79.11%	19.58%	4.91%	1.15	1.18	**33.33%**
Controlled DialogPrompt (2-layer Transf-Dec)	4.2%	**79.35%**	**19.76%**	**5.12%**	1.18	1.21	33.18%

Method	ϕ%	**METEOR ↑**	**Dist ↑**		**Entropy ↑**	**Avg Len**	**Decoding Cost**
			D-1	D-2	E-4		sec/token
Pretrained	0%	11.32%	5.51%	32.52%	**10.46**	11.72	0.023
Fine-tuning	100%	**13.18%**	5.97%	**34.35%**	10.28	10.19	0.024
PPLM	0.001%	10.92%	**6.13%**	29.23%	9.58	10.49	1.792
SideControl	1.8%	12.32%	4.67%	29.62%	**10.28**	10.68	0.031
Soft Prompt-tuning	0.014%	12.39%	5.13%	30.84%	10.20	9.92	0.027
Prefix-tuning	3.6%	12.55%	5.36%	31.64%	10.25	10.10	0.029
Controlled DialogPrompt (Embedding)	0.001%	12.37%	5.05%	30.44%	10.23	10.15	0.029
Controlled DialogPrompt (MLP)	3.6%	12.79%	5.47%	**31.98%**	10.27	10.25	0.028
Controlled DialogPrompt (2-layer Transf-Dec)	4.2%	**12.89%**	5.14%	30.55%	10.23	10.29	0.026

Table 4.1: **Automated metrics** on **Dailydialog** multi-reference evaluation. The backbone model is **DialoGPT-Medium**. ϕ% denotes the % of tunable parameters to the frozen-LM parameters required at training time. Red number is the best value in every metric on all methods. Blue number is the best value in every metric among parameter-efficient fine-tuning methods.

work, we intend to solve this problem so that the large-scale model can perform efficient controlled generation with complicated features.

Method	$\phi\%$	**Controllability Accuracy**	**BLEU ↑** B-2	B-4	**NIST ↑** N-2	N-4	**ROUGE-L ↑**
Pretrained	0%	58.30%	10.31%	1.73%	0.18	0.18	19.43%
Fine-tuning	100%	**80.25%**	**21.03%**	**5.70%**	0.96	0.98	**34.38%**
PPLM	0.001%	56.63%	15.19%	3.17%	1.45	1.47	27.40%
SideControl	1.3%	52.92%	17.05%	3.78%	**1.51**	**1.54**	31.31%
Soft Prompt-tuning	0.008%	70.51%	18.15%	4.08%	0.56	0.57	31.58%
Prefix-tuning	3.1%	75.02%	19.94%	5.12%	0.91	0.93	33.29%
Controlled DialogPrompt (Embedding)	0.001%	69.06%	**20.11%**	4.91%	0.71	0.73	32.80%
Controlled DialogPrompt (MLP)	3.1%	78.36%	19.92%	**5.43%**	0.98	1.01	33.12%
Controlled DialogPrompt (2-layer Transf-Dec)	3.3%	**78.58%**	19.86%	5.26%	1.01	1.04	**33.35%**

Method	$\phi\%$	**METEOR ↑**	**Dist ↑** D-1	D-2	**Entropy ↑** E-4	**Avg Len**	**Decoding Cost** sec/token
Pretrained	0%	7.30%	**7.61%**	**40.00%**	10.03	7.54	0.038
Fine-tuning	100%	**13.05%**	6.02%	34.51%	10.21	9.68	0.034
PPLM	0.001%	11.38%	5.47%	27.56%	9.80	11.46	4.613
SideControl	1.3%	12.33%	4.12%	27.00%	**10.33**	11.36	0.042
Soft Prompt-tuning	0.008%	11.46%	5.33%	30.82%	10.02	8.88	0.037
Prefix-tuning	3.1%	12.54%	5.59%	32.46%	10.17	9.64	0.035
Controlled DialogPrompt (Embedding)	0.001%	12.19%	5.18%	30.07%	10.03	9.18	0.033
Controlled DialogPrompt (MLP)	3.1%	12.61%	5.71%	32.42%	10.20	9.83	0.035
Controlled DialogPrompt (2-layer Transf-Dec)	3.3%	**12.64%**	**5.82%**	**33.16%**	10.23	9.92	0.036

Table 4.2: **Automated metrics** on **Dailydialog** multi-reference evaluation. The backbone model is **DialoGPT-Large**. $\phi\%$ denotes the % of tunable parameters to the frozen-LM parameters required at training time. Red number is the best value in every metric on all methods. Blue number is the best value in every metric among parameter-efficient fine-tuning methods.

Attribute Relevancy: Which response do you think is more related to the given dialog act?				
Method A		Neutral	Method B	
Controlled DP (Deep)	30.7%	49.3%	20.0%	Soft Prompt-tuning
Controlled DP (Deep)	25.3%	58.7%	16.0%	Prefix-tuning
Controlled DP (Deep)	34.7%	56.0%	9.3%	Controlled DP (Shallow)
Consistency: Which response do you think is more consistent to the above conversation context?				
Method A		Neutral	Method B	
Controlled DP (Deep)	32.0%	48.0%	20.0%	Soft Prompt-tuning
Controlled DP (Deep)	37.3%	46.7%	16.0%	Prefix-tuning
Controlled DP (Deep)	38.7%	36.0%	25.3%	Controlled DP (Shallow)

Table 4.3: **Human evaluation** concerning prompting methods on **Dailydialog** dataset. "Controlled DP (Deep)" represents Controlled DialogPrompt with 2-layer transformer decoder as the prompt module. "Controlled DP (Shallow)" represents Controlled DialogPrompt on the embedding layer. "Neutral" means both methods are equal and there is no preference.

Method	ϕ%	Controllability Similarity	BLEU ↑ B-2	BLEU ↑ B-4	NIST ↑ N-2	NIST ↑ N-4	ROUGE-L ↑
Pretrained	0%	52.11%	1.77%	0.49%	0.02	0.02	6.39%
Fine-tuning	100%	**70.61%**	**35.13%**	**24.52%**	**5.55**	**6.01**	**26.45%**
SideControl	7.4%	58.86%	6.98%	2.28%	0.65	0.67	11.24%
Soft Prompt-tuning	0.014%	63.04%	21.30%	12.50%	3.36	3.56	17.64%
Prefix-tuning	7.3%	66.30%	28.32%	18.02%	4.55	4.86	21.55%
Controlled DialogPrompt (Embedding)	14.5%	63.46%	18.14%	9.08%	2.82	2.97	17.07%
Controlled DialogPrompt (MLP)	7.3%	66.12%	30.10%	19.64%	4.79	5.12	22.59%
Controlled DialogPrompt (2-layer Transf-Dec)	7.7%	**66.59%**	**33.71%**	**23.16%**	**5.27**	**5.68**	**24.64%**

Method	ϕ%	METEOR ↑	Dist ↑ D-1	Dist ↑ D-2	Entropy ↑ E-4	Avg Len
Pretrained	0%	3.68%	6.98%	32.86%	10.15	9.71
Fine-tuning	100%	**23.33%**	**8.66%**	**41.39%**	**11.31**	23.50
SideControl	7.4%	6.48%	4.17%	23.00%	10.62	16.23
Soft Prompt-tuning	0.014%	14.95%	7.42%	35.21%	11.13	22.16
Prefix-tuning	7.3%	19.19%	7.79%	37.76%	11.27	24.06
Controlled DialogPrompt (Embedding)	14.5%	12.91%	5.71%	29.62%	11.05	22.63
Controlled DialogPrompt (MLP)	7.3%	20.20%	7.79%	37.61%	11.26	24.42
Controlled DialogPrompt (2-layer Transf-Dec)	7.7%	**22.08%**	**7.92%**	**38.37%**	**11.31**	24.86

Table 4.4: **Automated metrics** on **Focus** validation dataset. The Backbone model is **DialoGPT-Medium**. ϕ% denotes the % of tunable parameters to the frozen-LM parameters required at training time. Red number is the best value in every metric on all methods. Blue number is the best value in every metric among parameter-efficient fine-tuning methods.

Method	ϕ%	Controllability Similarity	BLEU ↑ B-2	BLEU ↑ B-4	NIST ↑ N-2	NIST ↑ N-4	ROUGE-L ↑
Pretrained	0%	51.40%	1.63%	0.42%	0.02	0.02	6.62%
Fine-tuning	100%	**75.21%**	**37.38%**	**25.77%**	**5.80**	**6.30**	**27.71%**
SideControl	5.3%	57.79%	7.31%	2.44%	0.61	0.63	12.33%
Soft Prompt-tuning	0.008%	62.69%	18.01%	9.50%	2.72	2.87	16.53%
Prefix-tuning	6.2%	**66.89%**	27.18%	16.73%	4.35	4.63	21.38%
Controlled DialogPrompt (Embedding)	8.3%	61.16%	13.01%	5.12%	1.89	1.96	14.84%
Controlled DialogPrompt (MLP)	6.2%	64.96%	26.82%	17.09%	4.25	4.54	21.40%
Controlled DialogPrompt (2-layer Transf-Dec)	5.0%	66.34%	**31.85%**	**21.67%**	**5.00**	**5.40**	**24.20%**

Method	ϕ%	METEOR ↑	Dist ↑ D-1	Dist ↑ D-2	Entropy ↑ E-4	Avg Len
Pretrained	0%	3.67%	7.62%	34.44%	10.15	9.28
Fine-tuning	100%	**24.43%**	**7.93%**	**38.20%**	**11.28**	24.76
SideControl	5.3%	6.78%	5.27%	26.81%	10.63	15.72
Soft Prompt-tuning	0.008%	13.29%	6.77%	32.19%	10.96	20.86
Prefix-tuning	6.2%	18.56%	7.60%	36.88%	**11.25**	23.77
Controlled DialogPrompt (Embedding)	8.3%	10.28%	5.21%	26.45%	10.82	20.15
Controlled DialogPrompt (MLP)	6.2%	18.47%	**7.85%**	37.58%	11.18	22.57
Controlled DialogPrompt (2-layer Transf-Dec)	5.0%	**21.16%**	**7.85%**	**37.86%**	11.24	23.55

Table 4.5: **Automated metrics** on **Focus** validation dataset. The Backbone model is **DialoGPT-Large**. ϕ% denotes the % of tunable parameters to the frozen-LM parameters required at training time. Red number is the best value in every metric on all methods. Blue number is the best value in every metric among parameter-efficient fine-tuning methods.

Personality: Which response do you think is more related to the persona?				
Method A		Neutral	Method B	
Controlled DP (Deep)	41.3%	53.4%	5.3%	Soft Prompt-tuning
Controlled DP (Deep)	22.7%	50.6%	26.7%	Prefix-tuning
Controlled DP (Deep)	29.3%	49.4%	21.3%	Controlled DP (Shallow)
Consistency: Which response do you think is more consistent to the above conversation context?				
Method A		Neutral	Method B	
Controlled DP (Deep)	44.0%	42.7%	13.3%	Soft Prompt-tuning
Controlled DP (Deep)	28.0%	64.0%	8.0%	Prefix-tuning
Controlled DP (Deep)	41.3%	49.4%	9.3%	Controlled DP (Shallow)

Table 4.6: **Human evaluation** concerning prompting methods on Focus dataset. "Controlled DP (Deep)" represents Controlled DialogPrompt with 2-layer transformer decoder as the prompt module. "Controlled DP (Shallow)" represents Controlled DialogPrompt on the embedding layer. "Neutral" means both methods are equal and there is no preference.

DialogAct: Question The speaker should raise a question to seek for some information.

Conversation Context:
- I heard that one of their specialties is the samosa.
- Well, let's get a plate of those then.
- Sounds good. What are you going to have for your main course?
- I think I might have a dahl.

Different model responses with the base model being DialoGPT-Medium

Pretrained: I can't tell if you're being sarcastic or not.
Fine-tuning: OK, I'll be right back... What are you going to have for dessert?
PPLM: I like the idea! Dahl sounds delicious, thanks!
SideControl: You're going to love that. it's got a high energy and strong taste.
Soft Prompt-tuning: I'll have a try of that. Is it a good one?
Prefix-tuning: Have you got anything for dessert?
Controlled DP (Shallow): That's great. How much is that?
Controlled DP (Deep): That sounds good, what kind of dahl is it?

Different model responses with the base model being DialoGPT-Large

Pretrained: : do you have a special dahl recipe? I want one to share.
Fine-tuning: That sounds good. How are you planning to spend your evening?
PPLM: I like dahl
SideControl: Sounds good.
Soft Prompt-tuning: Oh, that's too bad, I like the kind that have the whole country in its mouth.
Prefix-tuning: How about a dahl?
Controlled DP (Shallow): What do you think of the samosa?
Controlled DP (Deep): You know, that sounds quite good. I heard they are pretty good. Do you remember what kind of dahl you have?

Table 4.7: **Sample utterances** from different models when provided with conversation context, grounded knowledge and dialogue intent.

DialogAct: Directives The speaker should come up with a request, instruct, suggest.

Conversation Context:
- I want a pair of sports shoes, Adidas.
- What size do you want?
- I wear size 40.

Different model responses with the base model being DialoGPT-Medium

Pretrained: Try out the adidas x adi sneakers
Fine-tuning: Well, you needn't buy this pair for me. I am going to buy a different pair for you.
PPLM: I'll buy them off you
SideControl: OK. Can I see it?
Soft Prompt-tuning: How about size 43 for me?
Prefix-tuning: Then I'll get you one for free
Controlled DP (Shallow): Size 41, sir.
Controlled DP (Deep): How about these? They are very lightweight, and comfortable.

Different model responses with the base model being DialoGPT-Large

Pretrained: , get a pair.
Fine-tuning: Then, go get some Red Wing.
PPLM: I'd buy you a pair of Vans, Adidas, for 20, please.
SideControl: How about this one?
Soft Prompt-tuning: Adidas?
Prefix-tuning: Let me see. How about this pair?
Controlled DP (Shallow): Would you like to buy a pair of leather shoes?
Controlled DP (Deep): We'll get you a pair of size 40. They have all kinds of styles.

Table 4.8: **Sample utterances** from different models when provided with conversation context, grounded knowledge and dialogue intent.

User's Persona	**Grounded Knowledge**
I am going to visit Bavaria. I like military museums. I like old history. I like historical sites. I have a desire to be a soldier on peacekeeping missions.	The Bayerisches Armeemuseum is the Military History Museum of Bavaria. It was founded in 1879 in Munich and is located in Ingolstadt since 1972.
Conversation Context: **User:** I know this place, but I don't remember the name of this place.	
Different model responses with the base model being DialoGPT-Medium **Pretrained:** Haha :D **Fine-tuning:** This is the Bayerisches Armeemuseum, a Military History Museum of Bavaria. As you said you will be visiting Bavaria, I have been looking for places that might interest you. **SideControl:** The name of this place is the museum, the museum is a place you like, the museum contains the world of the British, the place is the museum of the American, the museum is a place you might like. **Soft Prompt-tuning:** This is the Bayerisches Armeemuseum in Munich. You may recognize the place since this is a Military history museum and it is in Munich. **Prefix-tuning:** This is Bayerisches Armeemuseum located in Munich, Germany. As a military history buff, you are going to like this place! **Controlled DP (Shallow):** Bayerisches Armeemuseum is the Military History Museum of Bavaria, which you may have heard of since you have a desire to be the military museums. I will tell you that the museum itself is located in Munich. **Controlled DP (Deep):** This is the Bayerisches Armeemuseum, located in Munich, Germany. If you are interested in military history, put it on the list!	
Different model responses with the base model being DialoGPT-Large **Pretrained:** I think he said it was in Bavaria. **Fine-tuning:** It is the Bayerisches Armeemuseum in Bavaria. Since you told me that you are going to visit Bavaria and that you like military museums, I have been looking for places that may interest you. **SideControl:** It is a military base, a military history-listed house, and a former military hospital. Since you told me you like old history, I have this information that is interesting. **Soft Prompt-tuning:** This is the Bayerisches Armeem Museum of Bavaria and it was founded in 1879 in Munich after being founded in 1879 in Munich. **Prefix-tuning:** The name of this place is Bayerisches Armeemuseum, a military history museum, located in Bavaria, since 2002. **Controlled DP (Shallow):** This is the Armeem museum in Munich, Bavarian military museum. Since its founded in 1879, the Bayerisches Armeem Museum place is named Bayerisches Armeem Museum. **Controlled DP (Deep):** This is Bayerisches Armeemuseum, the Military History Museum of Bavaria. Since you are studying military history, you should visit here!	

Table 4.9: **Sample utterances** from different models when provided with conversation context, grounded knowledge and user's persona.

User's Persona	**Grounded Knowledge**
I am from England. I live in Great Britain. I would like to visit Bath. I am interested in architecture. I love the Georgian style.	Designed by the architect John Wood, the Younger and built between 1767 and 1774, it is among the greatest examples of Georgian architecture to be found in the United Kingdom and is a Grade I listed building.

Conversation Context:
User: Could you tell me where in England it is located?
Bot: Specifically, the Royal Crescent is located in Bath, a place that you want to visit someday.
User: What is the Royal Crescent known for?

Different model responses with the base model being DialoGPT-Medium

Pretrained: It's the only building to exist on the Isle of Wight, the only building in the area.
Fine-tuning: As you're interested in architecture, you'll be happy to hear that this is one of the most famous Georgian architecture to be found in the UK.
SideControl: It was used by the Georgian Empire to use the place as a trading post, for which it was the second most populous building in the world.
Soft Prompt-tuning: As an example, it is one of the greatest examples of Georgian architecture in the United Kingdom.
Prefix-tuning: As you love building, I am sure you'd love to know that the Royal Crescent is known for Georgian architecture.
Controlled DP (Shallow): The Royal Crescent is, in Bath is known as the Royal Crescent.
Controlled DP (Deep): Since you like Georgian architecture, you might be interested to hear this building is known for being one of the greatest examples of the country's Georgian architecture.

Different model responses with the base model being DialoGPT-Large

Pretrained: It is in Bath
Fine-tuning: As an architecture and design enthusiast, you'll enjoy visiting the Grand Crescent which is Grade I listed.
SideControl: It is a place with a high level of architectural style, which you might enjoy, since you love architecture.
Soft Prompt-tuning: Being the seat of the monarch of a royal family, you will be pleased to know that the royal family also built it.
Prefix-tuning: As you're interested in architecture, the Royal Crescent is known as one of the most amazing things in the country to be found in England. You could check the architecture website of the Royal Crescent to see if there's anything you love.
Controlled DP (Shallow): The Royal Crescent is is a kind of Crescent you would find in a British city.
Controlled DP (Deep): The Royal Crescent is one of the Georgian architectural masterpieces. You should really visit this place since you love architecture!

Table 4.10: **Sample utterances** from different models when provided with conversation context, grounded knowledge and user's persona.

Method	ϕ%	**BLEU ↑**		**NIST ↑**		**ROUGE-L ↑**
		B-2	B-4	N-2	N-4	
Pretrained	0%	1.63%	0.42%	0.02	0.02	6.62%
Fine-tuning (wt knowledge)	100%	37.38%	25.77%	5.80	6.30	27.71%
Fine-tuning (wo knowledge)	100%	13.32%	4.99%	2.02	2.09	16.73%
Controlled DialogPrompt (on Persona)	5.0%	31.85%	21.67%	5.00	5.40	24.20%
Controlled DialogPrompt (on Knowledge)	5.0%	14.20%	4.84%	2.27	2.33	16.69%

Method	ϕ%	**METEOR ↑**	**Dist ↑**		**Entropy ↑**	**Avg Len**
			D-1	D-2	E-4	
Pretrained	0%	3.67%	7.62%	34.44%	10.15	9.28
Fine-tuning (wt knowledge)	100%	24.43%	7.93%	38.20%	11.28	24.76
Fine-tuning (wo knowledge)	100%	10.10%	5.35%	28.87%	10.91	21.95
Controlled DialogPrompt (on Persona)	5.0%	21.16%	7.85%	37.86%	11.24	23.55
Controlled DialogPrompt (on Knowledge)	5.0%	10.89%	5.99%	31.34%	11.12	23.44

Table 4.11: **Automated metrics** on **Focus** validation dataset comparing prompts on persona or knowledge. The Backbone model is **DialoGPT-Large**. ϕ% denotes the % of tunable parameters to the frozen-LM parameters required at training time. This illustrates the disadvantage of Controlled DialogPrompt in terms of knowledge control.

User's Persona	**Grounded Knowledge**
I am from England. I live in Great Britain. I would like to visit Bath. I am interested in architecture. I love the Georgian style.	Designed by the architect John Wood, the Younger and built between 1767 and 1774, it is among the greatest examples of Georgian architecture to be found in the United Kingdom and is a Grade I listed building.
Conversation Context: **User:** Could you tell me where in England it is located? **Bot:** Specifically, the Royal Crescent is located in Bath, a place that you want to visit someday. **User:** What is the Royal Crescent known for? **Controlled DialogPrompt (on Persona):** The Royal Crescent is one of the Georgian architectural masterpieces. You should really visit this place since you love architecture! **Controlled DialogPrompt (on Knowledge):** Being in Britain, you are familiar with the Royal Crescent. It is a large building and has been designated as a Grade VIII listed building in the country's Heritage Sites.	

User's Persona	**Grounded Knowledge**
I would like to visit the Nazareth House again. I love Benevolent institutions. I am interested in History. I have curiosity about the Description of this place. I would like to know when it was Built.	It was built from 1924 to 1939.
Conversation Context: **User:** Can you describe this house to me? **Bot:** You have curiosity about the description of Nazareth House and I will tell you. Nazareth House is prominently located on an elevation along Wynnum North Road. The complex consists of a number of buildings including the original building, the Convent and Chapel and two more recent additions, St Joseph's Hostel and the nursing home known as Larmeniere. **User:** Does this house look old to me, when it was built? **Controlled DialogPrompt (on Persona):** Yes, you have curiosity about the building, I will tell you it was built from 1924 to 1939. **Controlled DialogPrompt (on Knowledge):** No, actually, It started as a Church back in 1926, it has been renovated recently.	

Table 4.12: **Sample utterances** from Controlled DialogPrompt models with different prompt information.

Chapter 5

Conclusion and Future Work

In this work, we presented a novel prompt-tuning technique dependent on the provided attribute for controlled dialogue generation. The control attribute can either be a dialogue intent label or user's persona description. This instance-specific prompt includes the attribute encoding step and language modeling step. In the attribute encoding step, we use either a multilayer perceptron or a two-layer transformer decoder to represent the attribute text into a sequence of continuous vectors. In the language modeling step, we provide the encoded representation in two different ways, either in the embedding layer and concatenated with input word embeddings (shallow prompt) or in every layer of the base model and provided as extra key-value pairs (deep prompt). In both ways, we analyze the performance and discover they can outperform previous static prompt-tuning baselines.

We present a comprehensive description of prior studies about controlled dialogue generation and prompt-tuning that served as the inspiration for the proposed approach. We carry out experiments for two scenarios: label control with Dailydialog [39] and document control with Focus [26]. We show that deep controlled prompt outperforms superiorly compared to other controlled methods in terms of its controllability and consistency. The advantage is more significant in document control where the dataset centers on landmark discussions and provides content-rich user's persona description. When other controlled methods are not directly involved in the base model's generation, they fail to steer the pretrained model smoothly as they do in the label control scenario.

In addition, we perform an ablation study that compares deep prompts to shallow prompts in both static prompts and instance-specific controlled prompts. According to the pair-wise human evaluation results, deep prompts yield better results than shallow prompts since deep prompts can be attended to in every attention block instead of the embedding layer only for shallow prompts.

In the last part of the Results section, we point out Controlled DialogPrompt's inability to accurately transcribe knowledge from document. This might be related to the fact that the existing model lacks a corresponding structure to ensure entity consistency, such as the address or typical dates. It is more common to copy these entities from the knowledge documents rather than generating them from scratch. CopyNet [19] might be able to resolve this issue. It uses a trainable pointer to increase the attention distribution over related tokens in the knowledge document. It has shown improved performance in task-oriented dialogue systems recently [69]. Following that, we may obtain different prompts conditioned on different documents and we will be able to explore the integration of multiple Controlled DialogPrompts, which has never been studied before to the best of our knowledge.

In addition, due to limited computing power, we have not applied our parameter-efficient prompt-tuning technique on larger models such as OPT [74] which has a version that contains up to 175B parameters. It would be very interesting to leverage such a huge language model's pretrained ability and test its performances on different dialogue datasets. More importantly, it can push the limit of model size to a point where fine-tuning becomes intractable due to the limited GPU memory. Meanwhile, prompt-tuning modules can be trained easily and stored for various downstream tasks without too much overhead.

References

[1] Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.

[2] Abhaya Agarwal and Alon Lavie. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. *Proceedings of WMT-08*, 2007.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[4] Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. Plato: Pre-trained dialogue generation model with discrete latent variable. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 85–96, 2020.

[5] Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhen Guo, Zhibin Liu, and Xinchao Xu. Plato-2: Towards building an open-domain chatbot via curriculum learning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2513–2525, 2021.

[6] Eyal Ben-David, Nadav Oved, and Roi Reichart. Pada: A prompt-based autoregressive approach for adaptation to unseen domains. *arXiv preprint arXiv:2102.12206*, 2021.

[7] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

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

[9] Jordan Clive, Kris Cao, and Marek Rei. Control prefixes for text generation. *arXiv preprint arXiv:2110.08329*, 2021.

[10] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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)*, pages 4171–4186, 2019.

[12] Rahul Dey and Fathi M Salem. Gate-variants of gated recurrent unit (gru) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, pages 1597–1600. IEEE, 2017.

[13] George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, 2002.

[14] Wanyu Du and Yangfeng Ji. Sidecontrol: Controlled open-domain dialogue generation via additive side networks. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2175–2194, 2021.

[15] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, 2021.

[16] Xiang Gao, Yizhe Zhang, Michel Galley, Chris Brockett, and William B Dolan. Dialogue response ranking training with large-scale human feedback data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 386–395, 2020.

[17] Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, 2017.

[18] Tushar Goswamy, Ishika Singh, Ahsan Barkati, and Ashutosh Modi. Adapting a language model for controlled affective text generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2787–2801, 2020.

[19] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK 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)*, pages 1631–1640, 2016.

[20] Xiaodong Gu, Kang Min Yoo, and Sang-Woo Lee. Response generation with context-aware prompt learning. *arXiv preprint arXiv:2111.02643*, 2021.

[21] Prakhar Gupta, Shikib Mehri, Tiancheng Zhao, Amy Pavel, Maxine Eskenazi, and Jeffrey P Bigham. Investigating evaluation of open-domain dialogue systems with human generated multiple references. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 379–391, 2019.

[22] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pages 2712–2721. PMLR, 2019.

[23] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, 2020.

[24] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[25] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32, 2020.

[26] Yoonna Jang, Jungwoo Lim, Yuna Hur, Dongsuk Oh, Suhyune Son, Yeonsoo Lee, Donghoon Shin, Seungryong Kim, and Heuiseok Lim. Call for customized conversation: Customized conversation grounding persona and knowledge. *arXiv preprint arXiv:2112.08619*, 2021.

[27] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. *arXiv preprint arXiv:2203.12119*, 2022.

[28] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[29] Feihu Jin, Jinliang Lu, Jiajun Zhang, and Chengqing Zong. Instance-aware prompt learning for language understanding and generation. *arXiv preprint arXiv:2201.07126*, 2022.

[30] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[31] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. Gedi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, 2021.

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

[33] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, 2021.

[34] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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*, pages 7871–7880, 2020.

[35] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, 2016.

[36] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. A survey of pretrained language models based text generation. *arXiv preprint arXiv:2201.05273*, 2022.

[37] Margaret Li, Jason Weston, and Stephen Roller. Acute-eval: Improved dialogue evaluation with optimized questions and multi-turn comparisons. *arXiv preprint arXiv:1909.03087*, 2019.

[38] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for 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 1: Long Papers)*, pages 4582–4597, 2021.

[39] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset.

[40] Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie 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)*, pages 128–138, 2021.

[41] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[42] Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Mike Noseworthy, Laurent Charlin, and Joelle 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*, pages 2122–2132, 2016.

[43] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

[44] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[45] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[46] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[47] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[48] Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. Plug-and-play conversational models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2422–2433, 2020.

[49] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, 2019.

[50] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[51] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, 2016.

[52] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[53] Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, 2022.

[54] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, 2021.

[55] Ramon Quiza and J. Davim. *Computational Methods and Optimization*, pages 177–208. 01 2011.

[56] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training.

[57] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

[58] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

[59] Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, et al. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, 2021.

[60] Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, 2021.

[61] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, 2021.

[62] Abigail See, Stephen Roller, Douwe Kiela, and Jason 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)*, pages 1702–1723, 2019.

[63] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586, 2015.

[64] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, 2020.

[65] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[66] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[67] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[68] Alex Wang, Kyunghyun Cho, and CIFAR Azrieli Global Scholar. Bert has a mouth, and it must speak: Bert as a markov random field language model. *NAACL HLT 2019*, page 30, 2019.

[69] Weizhi Wang, Zhirui Zhang, Junliang Guo, Yinpei Dai, Boxing Chen, and Weihua Luo. Task-oriented dialogue system as natural language generation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2698–2703, 2022.

[70] Zhuofeng Wu, Sinong Wang, Jiatao Gu, Rui Hou, Yuxiao Dong, VG Vydiswaran, and Hao Ma. Idpg: An instance-dependent prompt generation method. *arXiv preprint arXiv:2204.04497*, 2022.

[71] Kevin Yang and Dan 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*, pages 3511–3535, 2021.

[72] Kexin Yang, Dayiheng Liu, Wenqiang Lei, Baosong Yang, Mingfeng Xue, Boxing Chen, and Jun Xie. Tailor: A prompt-based approach to attribute-based controlled text generation. *arXiv preprint arXiv:2204.13362*, 2022.

[73] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III*, pages 698–714, 2020.

[74] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[75] Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. Generating informative and diverse conversational responses via adversarial information maximization. *Advances in Neural Information Processing Systems*, 31, 2018.

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

[77] Zhengyan Zhang, Xu Han, Hao Zhou, Pei Ke, Yuxian Gu, Deming Ye, Yujia Qin, Yusheng Su, Haozhe Ji, Jian Guan, et al. Cpm: A large-scale generative chinese pre-trained language model. *AI Open*, 2:93–99, 2021.

[78] Tony Z Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*, 2021.