MSP: Multi-Stage Prompting for Making Pre-trained Language Models Better Translators

Anonymous ACL submission

Abstract

Prompting has recently been shown as a promising approach for applying pre-trained language models to perform downstream tasks. We present Multi-Stage Prompting, a simple and automatic approach for leveraging pre-trained language models to translation tasks. To better mitigate the discrepancy between pre-training and translation, MSP divides the translation process via pre-trained language models into three separate stages: the encoding stage, the re-encoding stage, and the decoding stage. During each stage, we independently apply different continuous prompts for allowing pre-trained language models better shift to translation tasks. We conduct extensive experiments on three translation tasks. Experiments show that our method can significantly improve the translation performance of pre-trained language models.

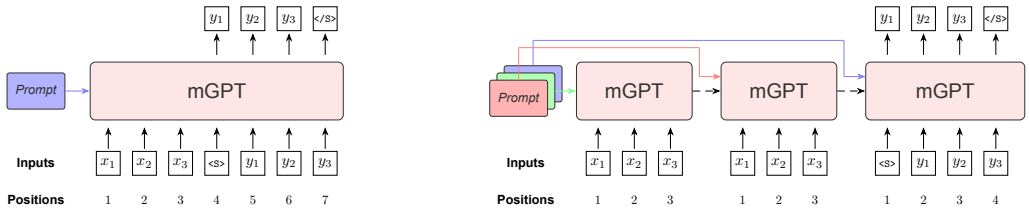
1 Introduction

Prompting (Brown et al., 2020; Lester et al., 2021), which refers to the approach of generating task-specific outputs from language models (LMs) by conditioning on extra information (known as *prompts*), has emerged as a new way of using LMs to perform natural language processing (NLP) tasks (Gao et al., 2020; Liu et al., 2021). While being efficient in parameters (Lester et al., 2021), prompting can enable mixed-task inference, which is not possible for other related approaches like finetuning or adapter-based tuning (Li and Liang, 2021; Lester et al., 2021). Prompting also opens the possibility of using a single pre-trained LM to perform all NLP tasks (Liu et al., 2021).

Machine translation (MT), which involves transformations between two languages, is considered one of the most challenging tasks in NLP (Koehn and Knowles, 2017). While neural machine translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017) is the current *de facto* approach for machine translation, using pre-trained LMs as translators via prompting is appealing in several aspects. For example, for the method described in this paper, supporting a new translation direction with a pre-trained LM occupies disk spaces below 20M, which is much smaller than training a separate neural machine translation model, where the model size is typically larger than 60M per language pair for the Transformer architecture. [1] Furthermore, the pre-trained LM also retains the ability to perform other downstream tasks, which is an important characteristic that has not been validated available on neural machine translation models.

However, it is challenging to leverage pre-trained LMs to translation tasks via prompting. First, finding an appropriate prompt for a translation task is not trivial and requires specific designs (Brown et al., 2020; Gao et al., 2020; Li and Liang, 2021; Lester et al., 2021). Second, the prompting method with a single prompt may be sub-optimal for steering pre-trained LMs to translation tasks, as there is a clear discrepancy between the objectives of translation and pre-training. Translation imposes strict *semantic equivalence* and *language space* constraint, in which a source sentence must translate to a semantically equivalent sentence in the target language space. As the objective of pre-training is usually to reconstruct parts of the input sentence (Radford et al., 2018; Devlin et al., 2019), the generation of a pre-trained LM conditioned on a source sentence will likely be in the source language space with non-equivalent semantics. Therefore, using a single prompt to guide the LM for mitigating both the semantic and language gap is likely to be sub-optimal. Third, prevalent generative LMs such as GPTs use a decoder-only architecture (Radford et al., 2018), which is unidirectional and may be sub-optimal for encoding source sentences (Devlin et al., 2019). While re-

[1] Assume using the *transformer-base* setting with a vocabulary size of 32K.



(a) Basic (single-stage) prompting for MT.

(b) Multi-stage prompting.

Figure 1: Overview of using prompts for steering a multilingual GPT (mGPT) model to machine translation tasks. Note that we reset the position ids during each stage in multi-stage prompting for ease of implementation. All stages use the same mGPT model.

cent works in prompting like prefix-tuning (Li and Liang, 2021) or prompt-tuning (Lester et al., 2021) alleviate the first challenge by introducing differentiable continuous prompts, the last two challenges remain to be addressed.

In this paper, we present Multi-Stage Prompting (MSP) for addressing the challenges of steering pre-trained language models to translation tasks. MSP encapsulates the idea of breaking translation tasks into simpler consecutive stages, allowing the pre-trained LM to learn "smoother transitions" to translation tasks by providing different prompts at different stages. For GPT-style pre-trained LMs, we design a three-stage prompting scheme for modeling the translation process, which consists of an *encoding stage*, a *re-encoding stage*, and a *decoding stage*. Specifically, the pre-trained LM focuses on learning source representations at the encoding stage and learns refined bidirectional representations by re-encoding source sentences at the re-encoding stage. Therefore, the LM can produce better translations with refined source representations at the decoding stage. Following prefix-tuning (Li and Liang, 2021) and prompt tuning (Lester et al., 2021), we use independent trainable continuous prompts at different stages, which are learned through back-propagation. The difference between basic (single-stage) prompting and multi-stage prompting is illustrated in Figure 1.

We demonstrate the effectiveness of our method with a multilingual GPT (mGPT) model on Romanian-English, English-German, and English-Chinese translation tasks. Experiments verify that compared with prompt tuning or prefix-tuning, MSP can significantly improve the translation performance of pre-trained LMs. Our method improves the translation performance of pre-trained language models via prompt tuning and prefix-tuning by 18.6 and 4.1 BLEU points on average over the three translation tasks, respectively, suggesting that MSP is a more effective prompting method for translation tasks.

2 Background

2.1 Prompting

Prompting is an approach of using an LM to perform downstream tasks by adding extra information for the LM to condition during its generation (Lester et al., 2021). This extra information, also known as a *prompt*, plays an important role in prompting methods and is often prepended to LM's input for better control of its generation. Depending on the form of prompts, prompting methods can be divided into two categories: using textual prompts or using continuous prompts.

Textual prompts are typically composed of natural language tokens. As a representative approach of textual prompts, Brown et al. (2020) use manually designed prompts to steer GPT-3's generation. A typical prompt used in GPT-3 consist of a task description and a few task-specific examples. Gao et al. (2020) and Shin et al. (2020) propose different automatic methods to generate textual prompts. Textual prompts are typically understandable by humans. However, Shin et al. (2020) indicate that automatically generated textual prompts may lack interpretability.

Continuous prompts, which consist of a sequence of continuous vectors, have gained increasing popularity recently. For example, in (Li and Liang, 2021), the continuous prompts consist of a sequence of key-value pairs (also called prefixes). Lester et al. (2021) propose a simplified version of continuous prompts, which consists of virtual tokens that are only added to the embedding layer. Compared with textual prompts, using continuous prompts is generally more powerful but less interpretable (Lester et al., 2021).

2.2 mGPT

In this paper, we use GPT (Radford et al., 2018, 2019; Brown et al., 2020) as the backbone LM for machine translation tasks. GPTs are a series of causal language models based on the Transformer architecture (Vaswani et al., 2017). To be more suitable for translation tasks that involve multiple languages, we introduce a *multilingual GPT* (mGPT) model instead of using a standard GPT-2 model. The main difference between mGPT and GPT-2 is the training data. mGPT is trained on the mC4 dataset (Xue et al., 2020), which is a multilingual dataset covering over 101 languages. For further details about mGPT, please refer to Appendix A.1.

Let $\mathbf{z} = [z_1, \ldots, z_n]$ be a sequence of tokens, mGPT uses an autoregressive Transformer network to model the conditional probability $P(z_t|\mathbf{z}_{<t})$, where $t \in [1, n]$ and $\mathbf{z}_{<t} = [z_1, \ldots, z_{t-1}]$. We use $f_{\text{LM}}(\boldsymbol{z}, \boldsymbol{H}; \boldsymbol{\theta})$ to denote the Transformer network, where $\boldsymbol{z}$ is a word embedding, $\boldsymbol{H}$ is a sequence of past activations, and $\boldsymbol{\theta}$ denotes the parameters of the Transformer network.

Initially, the inputs to the Transformer network are z_1 and $\boldsymbol{H}_0$, where $\boldsymbol{H}_0$ is an empty sequence. The Transformer network produces two outputs: the final output $\boldsymbol{g}_1 \in \mathbb{R}^d$ and the activation $\boldsymbol{h}_1 \in \mathbb{R}^{2N \times d}$, [2] where d denotes the hidden size of the Transformer network and N is the number of layers of the Transformer network.

For subsequent inputs z_t and $\boldsymbol{H}_{t-1}$, where $\boldsymbol{H}_{t-1} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{t-1}]$, the computation is formally described as

$$\boldsymbol{g}_t, \boldsymbol{h}_t = f_{\text{LM}}(\boldsymbol{e}_{z_t}, \boldsymbol{H}_{t-1}), \tag{1}$$

where $\boldsymbol{e}_{z_t}$ denotes the word embedding of z_t. To make the notation simpler, we use the following equation to denote the repeated application of f_{LM} over a sequence $\mathbf{z}^{i:j} = [z_i, \ldots, z_j]$ given past activations $\boldsymbol{A}$:

$$\boldsymbol{G}^{i:j}, \boldsymbol{H}^{i:j} = f_{\text{LM}}(\boldsymbol{Z}^{i:j}, \boldsymbol{A}), \tag{2}$$

where $\boldsymbol{Z}^{i:j} = [\boldsymbol{e}_{z_i}, \ldots, \boldsymbol{e}_{z_j}]$, $\boldsymbol{G}^{i:j} = [\boldsymbol{g}_i, \ldots, \boldsymbol{g}_j]$, and $\boldsymbol{H}^{i:j} = [\boldsymbol{h}_i, \ldots, \boldsymbol{h}_j]$.

Finally, the conditional probability $P(z_t|\mathbf{z}_{<t})$ is modeled as follows:

$$P(z_t|\mathbf{z}_{<t}) = \frac{\exp\left(\boldsymbol{e}_{z_t}^{\mathsf{T}} \cdot \boldsymbol{g}_t\right)}{\sum_{i=1}^{|V|} \exp\left(\boldsymbol{e}_{z_i}^{\mathsf{T}} \cdot \boldsymbol{g}_t\right)}, \tag{3}$$

where $|V|$ is the vocabulary size, and "$\cdot$" denotes matrix production.

[2] $\boldsymbol{h}$ is a concatenation of a set of key-value pairs $\{\langle \boldsymbol{k}^{(i)}, \boldsymbol{v}^{(i)} \rangle | i = 1 \ldots N\}$ in the Transformer network.

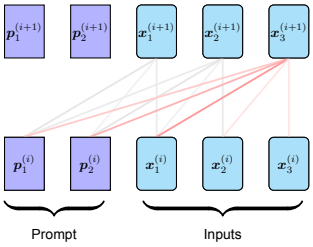


Figure 2: A deep continuous prompt is prepended to the inputs in all attention layers, which affects the computation of all attention layers. We do not distinguish keys and values for simplicity.

3 Multi-Stage Prompting

We propose multi-stage prompting (MSP), a simple and lightweight method for steering pre-trained LMs to translation tasks. We first describe the concept of deep continuous prompts in Section 3.1. Then we detail the stages and training objective in Section 3.2 and Section 3.3, respectively. Finally, we describe the reparameterization of deep continuous prompts in Section 3.4.

3.1 Deep Continuous Prompts

We adopt "continuous prompts" (Li and Liang, 2021; Lester et al., 2021) instead of using textual prompts in our method. Using continuous prompts allows learning through differentiable methods like back-propagation (Lester et al., 2021). To be specific, we use deep continuous prompts which are in the same form as in (Li and Liang, 2021). Formally, a prompt $\boldsymbol{P}$ is a sequence of L continuous vectors $[\boldsymbol{p}_1, \ldots, \boldsymbol{p}_L]$. Each vector $\boldsymbol{p}_i \in \mathbb{R}^{2N \times d}$ $(1 \leq i \leq L)$ is a concatenation of key-value pairs in all N Transformer layers, which directly affect the computation of every attention layer. We give an illustration of conditioning on a deep continuous prompt in Figure 2.

3.2 Stages

To effectively mitigate the semantic and language gap between the pre-training and translation, we propose multi-stage prompting which divides the procedure of using pre-trained LMs as translators into three separate stages: the encoding, the re-encoding, and the decoding stages. Given different prompts at different stages, the pre-trained LM is expected to behave differently during each stage and is more capable of generating translations.

Given a source sentence $\mathbf{x} = [x_1, \ldots, x_S]$ and a

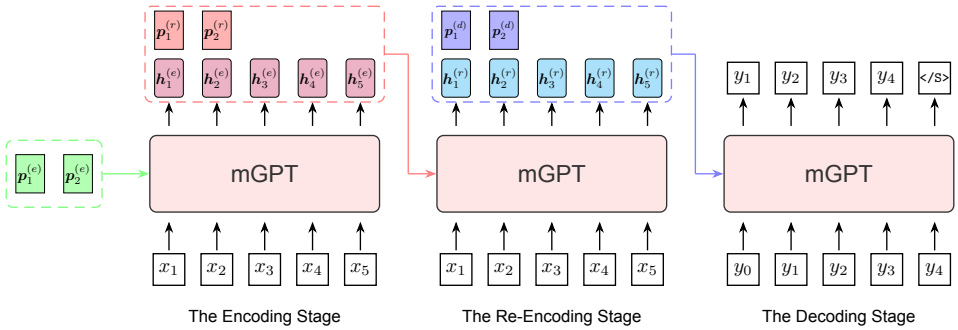


Figure 3: Detailed computations involved in the multi-stage prompting for machine translation tasks. We use rectangles to denote prompt vectors and rounded rectangles to denote activations.

target sentence $\mathbf{y} = [y_1, \ldots, y_T]$, the details of the three stages are described as follows:

The Encoding Stage. At the encoding stage, the pre-trained LM encodes the source sentence $\mathbf{x}$ into a sequence of activations $\boldsymbol{H}_e^{1:S}$ by using an encoding stage prompt $\boldsymbol{P}_e$. This procedure is the same as basic prompting. Formally, it can be described as follows:

$$\boldsymbol{G}_e^{1:S}, \boldsymbol{H}_e^{1:S} = f_{\mathrm{LM}}(\boldsymbol{X}^{1:S}, \boldsymbol{P}_e). \quad (4)$$

The Re-encoding Stage. At the re-encoding stage, the pre-trained LM produces fine-grained representations of the source sentence by re-encoding $\mathbf{x}$ given past activations $\boldsymbol{H}_e^{1:S}$ and a re-encoding stage prompt $\boldsymbol{P}_r$, which allows each representation to condition on all words in $\mathbf{x}$. This procedure can be described as

$$\boldsymbol{G}_r^{1:S}, \boldsymbol{H}_r^{1:S} = f_{\mathrm{LM}}(\boldsymbol{X}^{1:S}, [\![\boldsymbol{P}_r; \boldsymbol{H}_e^{1:S}]\!]), \quad (5)$$

where $[\![\boldsymbol{P}_r; \boldsymbol{H}_e^{1:S}]\!]$ denotes the concatenation of two sequences $\boldsymbol{P}_r$ and $\boldsymbol{H}_e^{1:S}$.

The Decoding Stage. Finally, we obtain the hidden vectors $\boldsymbol{G}_d^{1:T}$ for predicting the probability of the target sentence $\mathbf{y}$ at the decoding stage, given the refined source representations $\boldsymbol{H}_r^{1:S}$ and a decoding stage prompt $\boldsymbol{P}_d$:

$$\boldsymbol{G}_d^{1:T}, \boldsymbol{H}_d^{1:T} = f_{\mathrm{LM}}(\boldsymbol{Y}^{1:T}, [\![\boldsymbol{P}_d; \boldsymbol{H}_r^{1:S}]\!]). \quad (6)$$

Figure 3 gives a detailed illustration of MSP. By dividing the translation process into multiple stages and applying different prompts, we expect the pre-trained LM model can generate better translations.

3.3 Training Objective

We use the cross-entropy loss for learning prompts. Given $\boldsymbol{G}_d^{1:T} = [\boldsymbol{g}_1^{(d)}, \ldots, \boldsymbol{g}_T^{(d)}]$ in Eq. (6), the training objective is formally described as follows:

$$\begin{aligned}\mathcal{L} &= \frac{1}{T}\sum_{t=1}^{T} P(y_t|\mathbf{y}_{<t}, \mathbf{x}) \\ &= \frac{1}{T}\sum_{t=1}^{T} \frac{\exp\left(\boldsymbol{e}_{z_t}^{\mathsf{T}} \cdot \boldsymbol{g}_t^{(d)}\right)}{\sum_{i=1}^{|V|} \exp\left(\boldsymbol{e}_{z_i}^{\mathsf{T}} \cdot \boldsymbol{g}_t^{(d)}\right)}.\end{aligned} \quad (7)$$

Note that the parameters $\boldsymbol{\theta}$ of the pre-trained LM are fixed during training.

3.4 Reparameterization

Li and Liang (2021) suggest that using a neural network to reparameterize continuous prompts is more robust to different choices of hyperparameters. In contrast to their approach which uses an MLP network to reparameterize continuous prompts, we introduce a much simpler *scaled reparameterization* method, in which a continuous prompt is reparameterized as a product of a learnable scalar and an embedding. More precisely, the reparameterization of the three prompts are as follows:

$$\boldsymbol{P}_e = \max(\alpha_e, 1.0) \times \boldsymbol{\phi}_e, \quad (8)$$
$$\boldsymbol{P}_r = \max(\alpha_r, 1.0) \times \boldsymbol{\phi}_r, \quad (9)$$
$$\boldsymbol{P}_d = \max(\alpha_d, 1.0) \times \boldsymbol{\phi}_d. \quad (10)$$

As a result, the set of trainable parameters $\boldsymbol{\phi}$ in our method is $\boldsymbol{\phi} = \{\alpha_e, \alpha_r, \alpha_d, \boldsymbol{\phi}_e, \boldsymbol{\phi}_r, \boldsymbol{\phi}_d\}$, which contains much less tunable parameters than an MLP network. α_e, α_r, and α_d are initialized to 1.0 at the beginning of training.

Scaled reparameterization enables directly adjusting the value of prompts by a tunable scaling factor, leading to a much faster convergence without loss of performance. Further analysis is presented in Section 4.6.

Method	#Params.	Ro-En	En-De	En-Zh	Average
Prompt Tuning	131K	17.7	5.9	4.5	9.4
Prefix-Tuning	26M	32.5	17.5	21.9	23.9
MSP (*Ours*)	19M	**34.7**	**21.2**	**28.1**	**28.0**

Table 1: BLEU score on three different translation tasks for different prompting methods. All prompting methods use the same pre-trained language model "mGPT". "#Params." denotes the number of tunable parameters during training.

LM	Architecture	#M-Params.	Method	BLEU
mT5-XXL (Zhang et al., 2021)	Encoder-Decoder	13B	Finetuning	24.0
CPM-2 (Zhang et al., 2021)	Encoder-Decoder	11B	Prompt Tuning	24.1
CPM-2 (Zhang et al., 2021)	Encoder-Decoder	11B	Finetuning	26.2
Ernie 3.0 (Sun et al., 2021a)	Encoder-Decoder	10B	Finetuning	26.8
mGPT (*Ours*)	Decoder	560M	MSP	**28.1**

Table 2: Comparisons with previous studies on the WMT20 En-Zh translation task. "#M-Params." indicates the number of parameters of pre-trained LMs.

4 Experiments

4.1 Setup

Datasets We conduct experiments on Romanian-English (Ro-En), English-German (En-De), and English-Chinese (En-Zh) translation tasks to verify our proposed method. For the Ro-En translation task, we used the WMT16 Romanian-English dataset, which consists of 0.6M bilingual sentence pairs and 2M back-translated sentence pairs.[3] We used *newsdev2016* as the development set and *newstest2016* as the test set. For the En-De translation task, we used the WMT14 English-German dataset, which consists of 4.5M sentence pairs. The development set is *newstest2013* and the test set is *newstest2014*. For the En-Zh translation task, we used the WMT20 English-Chinese dataset as the training corpus, which consists of 28M sentence pairs. The development set is *newstest2019* and the test set is *newstest2020*. The details of preprocessing and postprocessing are given in Appendix A.2.

Metric. We used case-sensitive BLEU (Papineni et al., 2002) as the evaluation metric. The BLEU score is calculated using the SACREBLEU toolkit (Post, 2018).[4]

Baselines. We used the mGPT model as the backbone LM in all our experiments, which contains 560M parameters. We compare our method with the following prompting methods: [5]

- Prompt tuning (Lester et al., 2021). A prompting method that only prepends virtual tokens to the embedding layer of pre-trained LMs.
- Prefix-tuning (Li and Liang, 2021). A prompting method that uses deep continuous prompts, which prepend virtual tokens to all key-value pairs in attention layers of pre-trained LMs. We use an MLP network to reparameterize a continuous prompt during training as suggested in (Li and Liang, 2021).

Implementations. All our models are trained on a machine with 8 RTX 3090Ti GPUs. For all prompting methods, we set the prompt length to 128. For the training, we use the Glorot uniform initilalizer (Glorot and Bengio, 2010) to initialize tunable parameters unless otherwise noted. We use Adam (Kingma and Ba, 2014) ($\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 1\times 10^{-9}$) as the optimizer with a batch size of roughly 32K tokens. We use the same learning rate schedule as described in (Vaswani et al., 2017). We set the maximum learning rate to 0.02

[3] http://data.statmt.org/rsennrich/wmt16_backtranslations/ro-en

[4] Signature: nrefs:1|case:mixed|eff:no|tok:{13a,zh}|smooth:exp|version:2.0.0

[5] In our preliminary experiments, we also experimented with the few-shot approach as described in (Brown et al., 2020). However, we found mGPT often failed to generate meaningful translations.

for prompt tuning and MSP, and 7e-4 for prefix-tuning. We train prompts for a total of 80K steps for prompt tuning and prefix-tuning, and 40K steps for MSP. The number of warmup steps is set to 4K. For the inference, we use the beam search algorithm to obtain translation from the mGPT model, and the beam size is set to 4. The length penalty is determined by the results evaluated on the development set. We set the length penalty to 1.0 for the En-Zh translation task and 0.0 for other translation tasks. We implement our models on top of the open-source NMT toolkit THUMT (Tan et al., 2020) and the Transformers library (Wolf et al., 2020).

4.2 Main Results

Table 1 shows the results for the Ro-En, En-De, and En-Zh translation tasks.

As the most parameter-efficient among the three prompting methods, prompt tuning introduces only 131K parameters during training for each translation task. However, it only achieves 9.4 BLEU points on average over the three translation tasks. Lester et al. (2021) indicate that language model capacity is a key ingredient for prompt tuning to succeed. As mGPT is a pre-trained LM with only 560M parameters, the results coincide with the conclusion of Lester et al. (2021).

Prefix-tuning, which uses deep continuous prompts, achieves an average of 23.9 BLEU points over the three translation tasks. The results indicate that using deep continuous prompts is beneficial for steering mGPT to translation tasks. However, introducing deep continuous prompts inevitably requires more free parameters. The MLP network used in prefix-tuning introduces about 26M parameters for each translation task during training in our experiments.

Finally, MSP achieves 28.0 BLEU points on average over the three translation directions and outperforms prompt tuning and prefix-tuning by 18.6 and 4.1 BLEU points, respectively. MSP introduces 19M parameters for each translation task during training, which is more than prompt tuning but less than prefix-tuning. MSP explicitly divides the translation process using mGPT into three separate stages, which are not present in prompt tuning and prefix-tuning. The results suggest that MSP is more effective in instructing pre-trained LMs to perform translation than prompt tuning and prefix-tuning. Besides comparisons with prompting methods, we also provide comparisons between mGPT with MSP and Transformer-based NMT model in Appendix A.3.

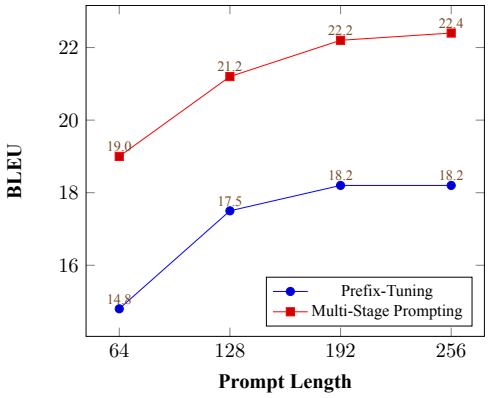


Figure 4: Comparison between MSP and prefix-tuning on the WMT14 En-De translation task with different prompt lengths.

4.3 Comparison with Other LMs

Table 2 gives the results of mT5-XXL (Zhang et al., 2021), CPM-2 (Zhang et al., 2021), Ernie 3.0 (Sun et al., 2021a), and mGPT on the WMT20 En-Zh translation task. Except for mGPT, other LMs are based on the encoder-decoder architecture. Despite using a much smaller pre-trained LM with about 5% parameters of mT5-XXL, CPM-2, and Ernie 3.0, MSP achieves the best performance over prompt tuning and full fine-tuning. Therefore, we show that MSP is an efficient and effective approach to steering pre-trained LMs to translation tasks.

4.4 Effect of Prompt Length

Figure 4 shows the effect of prompt length for prefix-tuning and MSP. We omit the comparison to prompt tuning because of its inferior performance. We found that using longer prompts generally leads to better performance for both prefix-tuning and MSP, but with diminishing returns. This finding is consistent with previous studies (Li and Liang, 2021; Lester et al., 2021). Furthermore, MSP consistently outperforms prefix-tuning when using the same prompt length. Even MSP with a prompt length of 64 performs better than prefix-tuning with a prompt length of 256 (19.0 vs. 18.2). The results further confirm that MSP is a better prompting method than prefix-tuning for steering pre-trained LMs to translation tasks.

Method	#Params.	Training	Inference	BLEU
Single-stage	6.3M	14h	0.10 s/sent.	17.9
Two-stage (encoding/decoding)	12.6M	14h	0.10 s/sent.	20.2
+ Re-encoding (*default*)	19.0M	21h	0.11 s/sent.	21.2
+ 2nd Re-encoding	25.1M	29h	0.11 s/sent.	21.8
+ Prompt sharing	6.3M	21h	0.11 s/sent.	19.8

Table 3: Comparison of using different stage settings on the WMT14 En-De translation task. "#Params." denotes the number of trainable parameters. "Training" denotes the total training time. "Inference" denotes the inference speed measured on *newstest2014* using 8 GPUs. "s/sent." denotes seconds per sentence.

4.5 Effect of Stages

Table 3 shows the comparison of using different stage settings on the WMT14 En-De translation task. Using single-stage prompting achieves 17.9 BLEU points. By comparison, explicitly separating encoding and decoding stages improve the translation performance over single-stage prompting by 2.3 BLEU points, which indicates the importance of differentiating stages. Adding a re-encoding stage further improves the translation performance by 1.0 BLEU point, suggesting that the re-encoding stage is effective. Adding a second re-encoding stage further improves the translation performance by 0.6 BLEU points. Although adding stages introduces more trainable parameters, it should be noted that sharing a single prompt for the encoding/re-encoding/decoding stages also improves over the single-stage prompting by 1.9 BLEU points. The results suggest that most improvements are attributed to the explicit separation of stages rather than increased parameters. Adding more stages generally slows the training speed. However, we do not observe notable inference speed drop as re-encoding stages are computed one time in parallel during inference.

4.6 Effect of Reparameterization

Figure 5 shows the comparison between MSP using scaled reparameterization and without using reparameterization. Using scaled reparameterization converges faster than without using reparameterization. These two methods achieve nearly the same translation performance when the training is converged. As a result, using scaled reparameterization can make the convergence much faster and reduce the total training time.

4.7 Analysis

Knowledge. As continuous prompts are learned using bilingual sentence pairs, an interesting question arises: Is the translation knowledge stored in the continuous prompts or the pre-trained LM? To answer this question, we discard the prompts and feed the mGPT model the concatenation of a source and the corresponding target sentence as input, and calculate the cosine similarities between the source and target hidden activations on each mGPT layer. We found that although the prompts are not given, the nearest pairs of tokens between the source and target language frequently turn out to coincide with bilingual alignments. This finding reveals to some

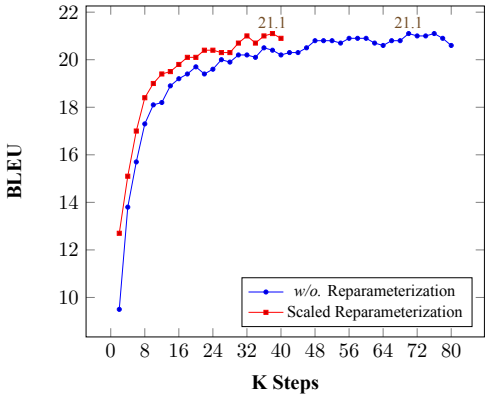


Figure 5: Comparison between using scaled reparameterization and without using reparameterization on the WMT14 translation task. The BLEU score is evaluated on *newstest2013*.

Prompt	Distribution
w/o prompt	en (16%), ru (10%)
Prefix-tuning	zh (80%), ja (12%)
MSP (encoding stage)	en (51%), la (14%)
MSP (re-encoding stage)	en (24%), la (17%)
MSP (decoding stage)	zh (91%), ja (9%)

Table 4: Language distribution of the free generations using mGPT by conditioning on different prompts learned by different prompting methods on the WMT20 En-Zh dataset.

extent that the translation knowledge mainly resides in the pre-trained LM instead of the learned continuous prompts, while the prompts play a role in guiding the model to perform translation during generation. Examples are given in the Appendix.

Bottleneck. We study the bottleneck of the current prompting method. We train a separate Transformer encoder and an adapter network that directly maps a source sentence into a deep continuous prompt, leaving the mGPT model only serving as a decoder. This model introduces 378M tunable parameters and achieves 25.9 BLEU points on the WMT14 En-De translation task. Compared with 21.2 BLEU points by MSP, the result shows that there is still room to advance the translation performance of pre-trained LM by improving the prompting method, such as using *dynamic prompts* (Liu et al., 2021) for each input sentence. However, as translation knowledge may coming from the pre-trained LM, the translation performance may also bottlenecked by the capability of the backbone LM.

Interpretability. We did not find our learned continuous prompts to be interpretable, which agrees with the findings of Shin et al. (2020) and Lester et al. (2021). However, we do observe prompts of different stages changing the behavior of mGPT significantly. Specifically, we sample 100 examples generated from mGPT by providing prompts of different stages learned on the English-Chinese translation task and identify the language ids of generated texts using the `langid` toolkit. The top-2 identified language distributions of each generation are shown in Table 4. Without providing continuous prompts, mGPT generates a random sentence from a random language. By given continuous prompts learned by prefix-tuning, the mGPT mostly generates texts related to Chinese. For MSP, it is noticeable that there is a transition from English to Chinese. mGPT generates English-related text given the encoding stage prompt. The distribution of languages becomes smoother when providing the re-encoding stage prompt. Finally, mGPT generates Chinese texts dominantly given the decoding stage prompt. The results coincide with our intuition that MSP helps the pre-trained LM to learn "smoother transitions" to the translation task.

5 Related Work

Prompting. Brown et al. (2020) propose to use a task description and a few examples to adapt the GPT-3 model to downstream tasks, which is referred to as in-context learning. Their prompts are manually designed. Gao et al. (2020) present LM-BFF for automatic prompts generation. They use T5 model (Raffel et al., 2019) to generate templates for prompting pre-trained LMs. Li and Liang (2021) propose prefix-tuning, which uses continuous vectors as prompts. These prompts are trained using task-specific data and optimized through back-propagation. Lester et al. (2021) propose prompt tuning, which is similar to prefix-tuning but with fewer trainable parameters. Our method is also based on prompting. We use continuous prompts for steering PLMs to translation tasks. Unlike Li and Liang (2021) and Lester et al. (2021) who present general frameworks, our method is focused on improving the translation performance of pre-trained LMs.

Using Pre-trained LMs as Translators. Stickland et al. (2021) investigate using BART and mBART models for machine translation tasks, their approach relies on adapter networks and finetuning parts of pre-trained LMs. Guo et al. (2020) build a non-autoregressive NMT model by using a source BERT model as the encoder and a target BERT as the decoder with adapter layers. Sun et al. (2021b) propose grafting a source BERT model and a target GPT model for translation tasks. All these methods are adapter-based, which injects additional tunable modules into the pre-trained LMs. As a result, the pre-trained LMs lose the ability to perform mixed-task inference. Our approach is based on prompting, which only uses prompts for steering the pre-trained LMs to translation tasks. Zhang et al. (2021) investigate using prompt tuning for steering CPM-2 model to the WMT20 English-Chinese translation task. Furthermore, their approach applied to encoder-decoder architecture pre-trained LMs while ours applied to decoder-only pre-trained LMs.

6 Conclusion

We have presented multi-stage prompting, a method for making pre-trained models better translators. Experiments show that with multi-stage prompting, pre-trained language models can generate better translations, showing the potential of using pre-trained language models for translation tasks. In future work, we plan to extend our methods to pre-trained language models with the encoder-decoder architecture.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*.

Graeme Blackwood, Miguel Ballesteros, and Todd Ward. 2018. Multilingual neural machine translation with task-specific attention. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3112–3122.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Junliang Guo, Zhirui Zhang, Linli Xu, Hao-Ran Wei, Boxing Chen, and Enhong Chen. 2020. Incorporating BERT into Parallel Sequence Decoding with Adapters. In *Advances in Neural Information Processing Systems*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Matt Post. 2018. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *Technical report, OpenAI*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2021. Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3440–3453.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021a. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Zewei Sun, Mingxuan Wang, and Lei Li. 2021b. Multilingual translation via grafting pre-trained language models. *arXiv preprint arXiv:2109.05256*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*.

Zhixing Tan, Jiacheng Zhang, Xuancheng Huang, Gang Chen, Shuo Wang, Maosong Sun, Huanbo Luan, and Yang Liu. 2020. THUMT: An Open-Source Toolkit for Neural Machine Translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (AMTA 2020)*, pages 116–122.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Zhengyan Zhang, Yuxian Gu, Xu Han, Shengqi Chen, Chaojun Xiao, Zhenbo Sun, Yuan Yao, Fanchao Qi, Jian Guan, Pei Ke, et al. 2021. CPM-2: Large-scale Cost-effective Pre-trained Language Models. *arXiv preprint arXiv:2106.10715*.

A Appendix

A.1 Details of Multilingual GPT

We used a multilingual GPT (mGPT) (Radford et al., 2019) model as the pre-trained language model in all our experiments. The mGPT model is trained using the Megatron-LM toolkit (Shoeybi et al., 2019) [6] on the mC4 dataset (Xue et al., 2020), [7] which contains massive web crawled data covering 101 languages. The model consists of 24 transformer layers, and the hidden size d of the model is set to 1,024. We used the same tokenization and vocabulary as the mT5 model (Xue et al., 2020). The vocabulary size is 250,100. The total number of parameters of the mGPT model is about 560M.

A.2 Preprocessing and Postprocessing

We do not apply any additional preprocessing during pre-training. Preprocessing like tokenization is done automatically with the *sentencepiece* program. For learning prompts, we do not apply additional preprocessing on translation tasks except Romanian-English translation task, where we use a script [8] to remove diacritics in the Romanian side. Because the mT5 tokenizer automatically uses Unicode NFKC normalization, which results in non-standard punctuation for Chinese (e.g. “， ”→ “,”). Therefore, we use a rule-based method to replace non-standard punctuation back to standard counterparts for Chinese.

A.3 Comparison with Transformer

We compare our method with the state-of-the-art Transformer NMT model (Vaswani et al., 2017) [9] on the following two datasets:

- TedTalks dataset (Blackwood et al., 2018), which is an English-centric multilingual corpus including 59 languages with sentence pairs ranging from 3K to 200K per direction. We only report results for 5 languages pairs. The Transformer model is trained on all available parallel sentences, serving as a strong NMT baseline. For mGPT with MSP, we individually train prompts on each language pair following the same procedure described in this paper.

- WMT14 English-German (En-De) dataset, which is a widely used dataset for machine translation.

The results of “X→En” and “En→X” translation tasks are shown in Table 5 and Table 6, respectively. Although mGPT with MSP is independently trained on each language pair, the model still outperforms the strong multilingual NMT baseline by 2.8 and 1.8 BLEU points on “X-En” and “En-X” directions, respectively. This result demonstrates that using pre-trained LMs as translators with an appropriate prompting method has the potential to outperform a strong traditional Transformer NMT model.

Table 7 shows the comparison between and our mGPT model with MSP on the WMT14 En-De translation task. While there is still a noticeable performance gap between Transformer and mGPT with MSP, using mGPT as a translator with MSP is much parameter-efficient than training a separate NMT model. Supporting En-De translation with mGPT only introduces 19M parameters with MSP method. In comparison, the model size of the Transformer model for En-De translation is 450M. While mGPT model can perform other downstream tasks by providing different prompts, such abilities have not been validated on the Transformer NMT model. Besides being efficient in disk spaces, learning prompts for the En-De translation task are also faster than training a separate NMT model. It takes 21 hours to train prompts for MSP, whereas 72 hours for training a Transformer model.

[6] https://github.com/NVIDIA/Megatron-LM

[7] https://huggingface.co/datasets/mc4

[8] https://github.com/rsennrich/wmt16-scripts/blob/master/preprocess/normalise-romanian.py

[9] We used the *transformer-big* setting. Tokenizations and vocabularies are the same with mGPT for fair comparisons.

Model	#Params.	Bg	Es	It	Ru	Zh	Avg.
Transformer	437M	35.2	38.0	34.2	22.6	17.6	29.5
mGPT (MSP)	19M	**38.9**	**42.1**	**37.8**	**24.4**	**18.3**	**32.3**

Table 5: Results on the TedTalks "X→En" translation directions.

Model	#Params.	Bg	Es	It	Ru	Zh	Avg.
Transformer	437M	29.2	34.0	29.2	16.7	21.2	26.1
mGPT (MSP)	19M	**34.1**	**38.4**	**32.8**	**19.2**	14.9	**27.9**

Table 6: Results on the TedTalks "En→X" translation directions.

Model	#Params.	BLEU
Transformer (big)	450M	27.9
mGPT (MSP)	19M	21.2

Table 7: Resutls on the WMT14 En-De dataset. "#Params." denotes the number of tunable parameters during training.

English	"They say there were boys around, that was not the case at all," he said.
Chinese	他表示：“他们说周围有好几个男孩子，但事实并非如此。”
Alignments	他/he 表示/said :"/" 他们/They 说/say 周围/around 有/were 好/boys 几个/were 男孩/boys 子/boys ,/, 但/that 事实/case 并非/not 如此/all 。"/.
English	Saudi Arabia To Offer Tourist Visas For First Time, Abolish Abaya Rule
Chinese	沙特阿拉伯首次提供旅游签证，废除阿巴亚长袍规定
Alignments	沙/Saudi 特/Arabia 阿拉/Arabia 伯/Arabia 首次/Offer 提供/Offer 旅游/Tourist 签证/Visa ,/, 废/olish 除/olish 阿/Saudi 巴/baya 亚/baya 长/Rule 袍/Visa 规定/Rule

Table 8: Alignments induced by computing cosine similarities between target hidden keys and source hidden keys of the 15th Transformer layer of mGPT. We use "/" to separate Chinese and English tokens.