Deep Fusing Pre-trained Models into Neural Machine Translation

Rongxiang Weng[1,2], **Heng Yu**[2], **Weihua Luo**[2], **Min Zhang**[1]

[1]School of Computer Science and Technology, Soochow University, Suzhou, China
[2]Machine Intelligence Technology Lab, Alibaba Group, Hangzhou, China
wengrongxiang@gmail.com, yuheng.yh@alibaba-inc.com, weihua.luowh@alibaba-inc.com,minzhang@suda.edu.cn

Abstract

Pre-training and fine-tuning have become the de facto paradigm in many natural language processing (NLP) tasks. However, compared to other NLP tasks, neural machine translation (NMT) aims to generate target language sentences through the contextual representation from the source language counterparts. This characteristic means the optimization objective of NMT is far from that of the universal pre-trained models (PTMs), leading to the standard procedure of pre-training and fine-tuning does not work well in NMT. In this paper, we propose a novel framework to deep fuse the pre-trained representation into NMT, fully exploring the potential of PTMs in NMT. Specifically, we directly replace the randomly initialized Transformer encoder with a pre-trained encoder and propose a layer-wise coordination structure to coordinate PTM and NMT decoder learning. Then, we introduce a partitioned multi-task learning method to fine-tune the pre-trained parameter, reducing the gap between PTM and NMT by progressively learning the task-specific representation. Experimental results show that our approach achieves considerable improvements on WMT14 En2De, WMT14 En2Fr, and WMT16 Ro2En translation benchmarks and outperforms previous work in both autoregressive and non-autoregressive NMT models.

Introduction

Pre-trained models (PTMs), such as GPT (Radford et al. 2018, 2019), BERT (Devlin et al. 2019), XLM (Lample and Conneau 2019), have obtained large success in many natural language processing (NLP) tasks, from understanding (NLU) (Devlin et al. 2019; Lample and Conneau 2019) to generation (NLG) (Song et al. 2019; Lewis et al. 2019). Generally, training a PTM requires a large amount of unlabeled data and well-designed self-supervised training objectives (*e.g.*, masked language model (MLM) (Devlin et al. 2019) or auto-regressive language model (Radford et al. 2018)). Various downstream tasks then directly utilize the pre-trained knowledge by fine-tuning the parameters of PTMs with the task-specific training data (Qiu et al. 2020).

However, neural machine translation (NMT) does not get promising results in this transformation, especially for the rich-resource language pairs (Yang et al. 2019; Zhu et al. 2020; Weng et al. 2020a; Liu et al. 2020; Chen et al. 2021). Unlike most NLP tasks, as a bilingual generation task (Bahdanau, Cho, and Bengio 2014; Cho et al. 2014; Vaswani et al. 2017), the goal of NMT is to generate target language sentences through the contextual information from the source language counterparts. Both of pre-trained encoders (Radford et al. 2018; Devlin et al. 2019; Lample and Conneau 2019) and sequence to sequence models (Song et al. 2019; Lewis et al. 2019) cannot fit this goal, *i.e.*, these models only learn representations in the same language. This phenomenon leads to that simply fine-tuning the parameters of PTMs suffering from the *catastrophic forgetting* problem (Goodfellow et al. 2013; Yang et al. 2019). Thus, how to explore the potential of PTMs in NMT is still an issue worthy of study.

Recently, some studies have noticed the aforementioned problem and proposed several effective fusion-based methods. Zhu et al. (2020) proposed a BERT-fused NMT to fuse representations of BERT into NMT. Yang et al. (2019) proposed asymptotic distillation and dynamic switch methods for further utilizing pre-trained knowledge. Meanwhile, Weng et al. (2020a) proposed a dynamic fusion mechanism to choose the pre-trained representations dynamically. Their primary motivation is to extract pre-trained representations from PTMs and feed them into the NMT model. The pre-trained representation acts as a supplement to the NMT (*i.e.*, *shallow fusion*), which is not sufficient to make use of PTMs and renders the scale of NMT becomes very large. In addition, another shortcoming is that they completely abandon the fine-tuning and treat PTMs as a static knowledge base. However, a proper fine-tuning strategy is helpful to reduce the gap between PTMs and NMT, prompting the pre-trained representation to be more suitable for translation (Gururangan et al. 2020).

In this paper, towards making more effective use of PTMs in NMT, we propose a *deep fusion* framework, taking the previous fusion-based approach a step further. On the one hand, we propose a *layer-wise coordination structure* to coordinate the learning of the PTM and the NMT decoder, rather than regarding the PTM as a plug-in module of NMT. In our structure, the PTM replaces the randomly initialized Transformer encoder, and the decoder generates target sentences just relying on the contextual representation of the PTM. Different from the traditional coordination structure (He et al. 2018), each decoder layer only captures the representation from the corresponding encoder layer. We adopt a *coordina-*

tor to connect the PTM and the NMT decoder, which serves as a bridge to *map universal representation to task-specific representation*. The coordinator aggregates the multiple layer representations of PTM, integrating all layer representations into the NMT decoder (Dou et al. 2018a; Wang et al. 2018).

On the other hand, we propose a *partitioned multi-task training strategy* with two *bilingual intermediate tasks* to eliminate the intrinsic gap between PTMs and NMT. Firstly, given the source language sentence, we introduce a *target sentence prediction* (TSP) task to ask for the PTM to identify the corresponding target sentence and a *target words prediction* (TWP) task to predict words in this target sentence. Then, each task takes on different responsibilities though optimizing different parts of the model.[1] To be more specific, the translation task only trains the NMT decoder to learn to translate, while two intermediate tasks with the MLM objective will optimize the encoder part to avoid the *forgetting* problem.

To demonstrate the effectiveness of the proposed approach, we experiment on widely used machine translation tasks in both autoregressive (AT) (Vaswani et al. 2017) and non-autoregressive (NAT) (Ghazvininejad et al. 2019) NMT models. Our approach with the AT model achieves 31.43, 43.91, and 39.62 BLEU scores on the WMT14 En$\rightarrow$De, WMT14 En$\rightarrow$Fr and WMT16 Ro$\rightarrow$En, respectively. When combined with the NAT model, our approach gets 2.14 gains on the WMT14 En$\rightarrow$De task. We summarize our contributions as follows:

- We study the problem of using PTMs in NMT models, namely PTMs only learn representations in the same language, while NMT models need the representations across languages.
- We propose a novel deep fusion framework, including a layer-wise coordination structure and a partitioned multi-task learning paradigm, to fully explore the potential of PTMs in NMT with limited model size.
- Experimental results on three machine translation benchmarks show that our approach achieves considerable improvements over strong baselines and related work in both autoregressive and non-autoregressive NMT models.

The Proposed Approach

In this section, we will briefly introduce the Transformer based NMT (Vaswani et al. 2017) and MLM (Devlin et al. 2019), and give some necessary notations. Then, we will introduce the proposed approach based on them in detail.

Background and Notations

Neural machine translation. Given a parallel sentence pair $\{\mathbf{x}, \mathbf{y}\}$, where $\mathbf{x}$ and $\mathbf{y}$ are the source and target sequences, respectively. In the NMT, $\mathbf{y}$ is generated by $\textsc{Dec}(\textsc{Enc}(\mathbf{x};\theta_\text{E});\theta_\text{D})$, where $\textsc{Enc}(\cdot)$ and $\textsc{Dec}(\cdot)$ are the encoder and the decoder networks, respectively. The θ_E and θ_D are the parameters of the encoder and decoder. Both of them are composed of multiple self-attention layers (Vaswani et al. 2017), the number of layer is denoted N.

The encoder encodes $\mathbf{x}$ to the contextual representation $\mathbf{R}_N \in \mathbb{R}^{I\times d_\text{enc}}$, where I is the length of $\mathbf{x}$ and d_enc is the hidden size of the encoder. Formally, the nth layer's representation $\mathbf{R}_n$ is calculated by:

$$\mathbf{R}_n = \textsc{Enc}_n(\mathbf{R}_{n-1}) = \text{LN}(\widetilde{\mathbf{R}}_n + \textsc{Ffn}(\widetilde{\mathbf{R}}_n)), \quad (1)$$

$$\widetilde{\mathbf{R}}_n = \text{LN}(\textsc{Att}(\mathbf{R}_{n-1}, \mathbf{R}_{n-1}, \mathbf{R}_{n-1}) + \mathbf{R}_{n-1}), \quad (2)$$

where $\text{LN}(\cdot)$, $\textsc{Ffn}(\cdot)$ and $\textsc{Att}(\cdot)$ are layer normalization (Ba, Kiros, and Hinton 2016), feed forward network and self-attention network (Vaswani et al. 2017).

Similar to the encoder, the target representation $\mathbf{S}_N \in \mathbb{R}^{J\times d_\text{dec}}$ is computed by

$$\mathbf{S}_n = \textsc{Dec}_n(\mathbf{R}_N, \mathbf{S}_{n-1}) = \text{LN}(\mathbf{H}_n + \textsc{Ffn}(\mathbf{H}_n)), \quad (3)$$

$$\mathbf{H}_n = \text{LN}(\textsc{Att}(\widetilde{\mathbf{S}}_n, \mathbf{R}_N, \mathbf{R}_N) + \mathbf{S}_{n-1}), \quad (4)$$

$$\widetilde{\mathbf{S}}_n = \text{LN}(\textsc{Att}(\mathbf{S}_{n-1}, \mathbf{S}_{n-1}, \mathbf{S}_{n-1}) + \mathbf{S}_{n-1}), \quad (5)$$

where $\mathbf{S}_{n-1}$ is from the previous layer and d_dec is the decoder hidden size which is the same as d_enc, J is the length of the output sentence.

The training objective of NMT is to minimize the negative log-likelihood, denoted by:

$$\mathcal{L}_\text{T} = -\log P(\mathbf{y}|\mathbf{x};\theta_E,\theta_D), \quad (6)$$

$$P(\mathbf{y}|\mathbf{x}) = \text{softmax}(\text{FFN}(\mathbf{S}_N)). \quad (7)$$

Masked lanaguage model. Devlin et al. (2019) proposed to use the *masked language model task* (MLM) to pre-train a Transformer encoder.

Specifically, given the input sentence $\mathbf{x}$, a sequence $\mathbf{x}^R$ is constructed by randomly masking some tokens of $\mathbf{x}$. The MLM task is to recover the masked tokens: $\mathbf{x} = \textsc{Enc}(\mathbf{x}^R;\theta_\text{P})$, where θ_P is the parameters of the encoder. The training objective of MLM is

$$\mathcal{L}_\text{MLM} = -\log P(\mathbf{x}|\mathbf{x}^R;\theta_\text{P}), \quad (8)$$

$$P(\mathbf{x}|\mathbf{x}^R;\theta_\text{P}) = \text{softmax}(\text{FFN}(\mathbf{R}_N)). \quad (9)$$

Layer-wise Coordination Structure

Generally, the input of each Transformer decoder layer is a static contextual representation from the encoder and the output of the previous decoder layer (Eq.3-Eq.5). However, previous work (Wei et al. 2020; Dou et al. 2018b; Wang et al. 2018) shown that a dynamic contextual representation from the encoder's multi-layer representations can help to generate target sentences better. Compared to the vanilla encoder of NMT models, PTMs trained by large-scale unlabeled data have richer contextual information. A suitable approach to utilizing multi-layer representations is essential for effectively exploiting pre-trained knowledge.

We adopt a layer-wise coordination structure to feed each decoder layer with the contextual representations from the PTM. Different from He et al. (2018), which coordinates the encoder and the decoder layer-by-layer strictly, we introduce an inner structure (named *coordinator*) to *soft* connect the PTM and the decoder with a *multi-to-multi* manner.

[1]Our approach is a variant of the behavioral fine-tuning in https://ruder.io/recent-advances-lm-fine-tuning.

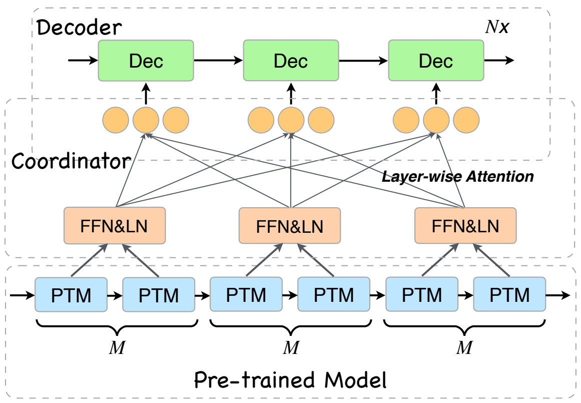

Figure 1: The overview of the proposed layer-wise coordination structure.

Specifically, we define the PTM as $\text{PTM}(\cdot;\theta_\text{P})$, θ_P is the parameters of the PTM, and divide it into N parts, each part has M Transformer layers. N is equal to the number of decoder layer. In other words, the total number of layers of the PTM is $N \cdot M$, and the overall layers of our model is $N \cdot (M+1)$. The proposed coordinator includes two modules, an aggregation layer and a layer-wise attention mechanism. We firstly adopt the aggregation layer to densely aggregate representations in the same part:

$$\mathbf{R}_n^A = \text{LN}(\text{FFN}(\mathbf{R}_{n,1}^P + \cdots + \mathbf{R}_{n,M}^P)), \tag{10}$$

where $\mathbf{R}_{n,M}^P \in \mathbb{R}^{I \times d_\text{ptm}}$ is the hidden states from the PTM. Then, the contextual representation $\mathbf{C}_n \in \mathbb{R}^{I \times d_\text{dec}}$ is computed by the *layer-wise attention* mechanism:

$$\mathbf{C}_n = \text{FFN}(\sum_{k=1}^{N} \alpha_k \mathbf{R}_k^A), \tag{11}$$

where the $\text{FFN}(\cdot)$ is a mapping function (*i.e.*, $d_\text{ptm} \rightarrow d_\text{dec}$) to integrate the PTM representations into the decoder. The attention weight α_k is calculated by

$$\alpha_k = \frac{\exp(e_k)}{\sum_{t=1}^{N} \exp(e_t)}, \tag{12}$$

$$e_k = \text{FFN}((\frac{1}{I}\sum_{i=1}^{I} \mathbf{r}_{n,i}^A) \cdot (\frac{1}{I}\sum_{i=1}^{I} \mathbf{r}_{k,i}^A)). \tag{13}$$

where $\mathbf{r}_{n,i}^A$ and $\mathbf{r}_{k,i}^A$ are the ith vector from $\mathbf{R}_n^A$ and $\mathbf{R}_k^A$, respectively. Finally, the decoder representation is:

$$\mathbf{S}_n = \text{DEC}_n(\mathbf{C}_n, \mathbf{S}_{n-1}), \tag{14}$$

In this process, the PTM and the coordinator can be considered as the encoder. The role of the aggregation layer is to merge and regroup the similar pre-trained representations. And the layer-wise attention determines which representation is more important for the current decoder layer. The overview is shown in the Figure 1.

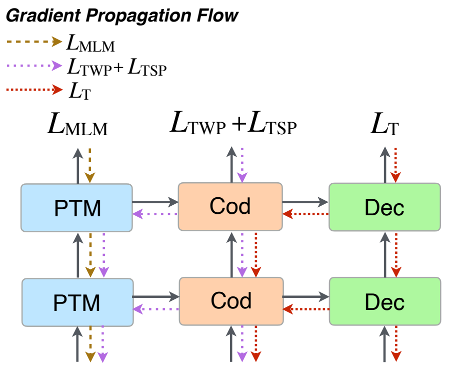
Gradient Propagation Flow
L_MLM
L_TWP+ L_TSP
L_T
L_MLM
L_TWP +L_TSP
L_T
PTM
Cod
Dec

Figure 2: The overview of the partitioned multi-task learning framework. The Cod means the coordinator. The dash lines are the gradient propagation flow of different tasks.

Partitioned Multi-task Training

We introduce a partitioned multi-task training to reduce the gap between PTMs and NMT. Besides the NMT training objective, we design two simple training objectives as intermediate tasks to learning bilingual representation. Furthermore, the masked language model (MLM) (Devlin et al. 2019) task is also employed to avoid the *forgetting* problem.

Target words prediction. The first training objective is the *target words prediction* (TWP), which is to assess *whether the word is in the target sentence* according to the source sentence. Specifically, we introduce a sub-task to predict the target sentence's bag-of-words, forcing the contextual representation to capture lexical-level knowledge of the target sentence. Following Zhao, Zhao, and Eskenazi (2017); Weng et al. (2017), we assume each target word is independent of each other. Formally, given an output sentence $\mathbf{y}$, the loss of TWP task is defined as:

$$\begin{aligned}\mathcal{L}_\text{TWP} &= -\log P(\mathbf{y}_{bow}|\bar{\mathbf{r}}_N^A;\theta_\text{C},\theta_\text{P}) \\ &= -\log \prod_{j=1}^{J} \frac{P(y_j)}{\sum_k^{V^t} P(y_k)},\end{aligned} \tag{15}$$

$$\bar{\mathbf{r}}_N^A = \frac{1}{I}\sum_{i=1}^{I} \mathbf{r}_{N,i}^A, \tag{16}$$

where θ_C is the parameters of the proposed coordinator, V^t is the target vocabulary and $\mathbf{y}_{bow}$ is a unordered set containing all words in the given target sentence $\mathbf{y}$.

Target sentence prediction. Inspired by Wei et al. (2021) and Chen et al. (2020), we introduce a contrastive learning based training objective, named *target sentence prediction* (TSP), to predict the target sentence.

Given the representation $\bar{\mathbf{r}}_N^A$ (Eq.16) from $\mathbf{x}$, we denote the sentence representation $g(\mathbf{x})$ is computed by $\text{FFN}(\bar{\mathbf{r}}_N^A)$, where $\text{FFN}(\cdot)$ is a linear projection layer (Chen et al. 2020). Then, the training objective here is defined as:

$$\mathcal{L}_\text{TSP} = -\log \frac{\exp(s(g(\mathbf{x}), g(\mathbf{y})))}{\sum_{b=1}^{B} \exp(s(g(\mathbf{x}), g(\hat{\mathbf{y}}_b)))}, \tag{17}$$

where $s(\cdot)$ is the cosine similarity, $\hat{\mathbf{y}}_b$ is the negative sample in the contrastive learning framework (Khosla et al. 2020). In the NLP, the key factor of employing contrastive learning is how to get reasonable negative samples (Fang and Xie 2020; Lee, Lee, and Hwang 2020). Here, we use two methods to collect negative samples. The first one is considering other target sentences in the same batch as negative samples. The next one is replacing some words in the $\mathbf{y}$ to build negative samples. We choose 50% words in the $\mathbf{y}$, and swap, replace and delete them randomly. The synthetic sentences have similar words to the target sentence but do not have correct semantic. Each method will produce $\frac{1}{2}B$ sentences, where B is the batch size. Finally, we can get a set of negative examples $\{\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, ..., \hat{\mathbf{y}}_B\}$.

Training. Each task in our multi-tasking learning framework optimizes different parts of the model. As shown in the Figure 2, the optimization parts of the translation task ($\mathcal{L}_{\text{T}}$) includes the decoder (θ_{D}) and the coordinator (θ_{C}). While the proposed TWP and TSP tasks optimize the coordinator and the PTM (θ_{P}). The purpose of this is to enable the generated contextual representation to obtain bilingual information. Then, the MLM only optimizes the PTM to avoid forgetting the pre-trained knowledge. Finally, the training function can be summarized as

$$\mathcal{L} = \mathcal{L}_{\text{T}} + \mathcal{L}_{\text{TWP}} + \mathcal{L}_{\text{TSP}} + \mathcal{L}_{\text{MLM}}. \tag{18}$$

Experiment

Data-sets

We evaluate our approach on three WMT translation tasks[2], including WMT 14 English to German (En→De), WMT14 English to French (En→Fr) and WMT16 Romanian to English (Ro→En). Here, the En→De is the most widely used benchmark in machine translation. The En→Fr has the most training data of any public dataset. And the Ro→En could be treated as a low-resource language pair.

Following previous work (Vaswani et al. 2017; Caswell, Chelba, and Grangier 2019), on the En→De task, the training set has about 4.5M sentence pairs. We use `newstest2013` as validation set which has 3000 sentence pairs, and `newstest2014` as test set which has 3003 sentence pairs. On the En→Fr task, our training set has about 36M sentence pairs. We use `newstest2013` as validation set which has 3000 sentence pairs, and `newstest2014` as test set which has 3003 sentence pairs. On the Ro→En task, our training set has about 0.6M sentence pairs. We use `newstest2015` as validation set which has 2000 sentence pairs, and `newstest2016` as test set which has 2000 sentence pairs.

Implementation Detail

We adopt cased multilingual BERT[3] (mBERT) as the default PTM in our experiments for fairly comparing to previous work. The depth of the mBERT is 12 and the hidden size is 768. The hyperparameter M is set 2 and the d_{dec} is set as 768. So, the layer number of the decoder are 6. We use the vocabulary from the mBERT as the source vocabulary and the target vocabulary is limited to 30k. We implement the proposed approach with the in-house implementation derivated from *tensor2tensor*.

We use label smoothing with the value 0.1 and dropout with the rate of 0.1. Adam (Kingma and Ba 2014) is used to update parameters, and the learning rate is set as 0.0001. The batch size is set as 64 and the max sentence length is limited to 80. All experiments are conducted on 8 V100 GPUs, and we accumulate the gradient 4 iterations in En→De and En→Fr tasks.

After the training stage, we use beam search as the decoding algorithm, and the beam size is set as 4. We measure the translation quality with the BLEU score (Papineni et al. 2002) computed by `multi-bleu.perl` script provided by Moses[4], and report the tokenized case-sensitive score.

Main Results

Translation quality. We firstly evaluate the proposed approach on WMT14 En→De and En→Fr tasks. All results are summarized in the Table 1. Line 1 and 2 are the Transformer-Base and Transformer-Big baselines from Vaswani et al. (2017). We also implement them with the same setting in our experiment, which are shown in the line 7 and 8. Ours outperform the results from Vaswani et al. (2017).

Compared to the above setting, our model has a deeper encoder. So, we implement two Transformer baselines with 12 and 20 encoder layers, which are denoted as Transformer-12 (line 9) and Transformer-20 (line 10). Here, we set the hidden size of them as 768 and use the *pre-norm* (Wang et al. 2019) in the Transformer-20 for achieving better performance. Moreover, we also report some advanced work about NMT with a deep encoder (line 3-6) (Bapna et al. 2018; Wang et al. 2019; Wu et al. 2019; Li et al. 2020).

Our model gets 31.59 and 44.21 BLEU scores on the En→De and En→Fr tasks, respectively. Compared to the Transformer-Base/Big, our model (line 11) achieves 3.81/2.46 BLEU score improvements. Then, compared to the Transformer-20, which is the strongest baseline in Table 1, our model achieves 1.77 and 1.28 improvements. Compared to previous work, our model gets the best performance in both of the two tasks. Moreover, our model only add a limited number of parameters. Experimental results on the above two translation tasks show that our approach can effectively exploit PTMs to improve translation quality.

Compared to previous work. We also make a comparison to some related work in our experiment. The overall results are reported in Table 2. The first one is that using mBERT to initialize the encoder's parameters of the Transformer-12. Then, in the NMT training process, the encoder is fine-tuned (line 2) and frozen (line 3), respectively. Following Weng et al. (2020a), we adopt knowledge distillation to learn the knowledge from mBERT to NMT (line 4). In addition, we use two sequence to sequence pre-trained models, *i.e.*, mBART (Liu

[2]http://www.statmt.org/wmt17/translation-task.html

[3]https://github.com/google-research/bert

[4]https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

#	Model	#Param	En→De	En→Fr
1	Transformer-Base (Vaswani et al. 2017)	65M	27.3	38.1
2	Transformer-Big (Vaswani et al. 2017)	213M	28.4	41.0
3	Transparent (Bapna et al. 2018)	137M	28.04	–
4	Deep NMT (Wang et al. 2019)	137M	29.3	–
5	Depth Growing NMT (Wu et al. 2019)	268M	29.92	43.27
6	SD-Transformer(Li et al. 2020)	437M	30.46	43.29
7	Transformer-Base	62M	27.78	38.96
8	Transformer-Big	207M	29.13	41.70
9	Transformer-12 ($d_{model} = 768$)	123M	29.32	42.39
10	Transformer-20 ($d_{model} = 768$, *pre-norm*)	159M	29.82	42.93
11	*Ours*	171M	**31.59**†	**44.21**†

Table 1: The translation quality of our model (*Ours*). The column #Param is the number of parameters. † indicate significantly better than the baseline ($p < 0.01$), which is computed by *compare-mt* (Neubig et al. 2019).

#	Model	#Param	En→De	En→Fr
1	Transformer-12	123M	29.32	42.39
2	Transformer-12 + Use mBERT to initialize the encoder	164M	29.19	41.46
3	Transformer-12 + Use frozen mBERT as the encoder	164M	28.17	40.03
4	Transformer-12 + Knowledge Distillation	123M	29.97	42.88
5	Use mBART (Liu et al. 2020) to initialize NMT	610M	28.82	41.77
6	Use MASS (Song et al. 2019) to initialize NMT	123M	29.41	42.03
7	CTNMT (Yang et al. 2019)	–	30.1	42.3
8	BERT-fused NMT (Zhu et al. 2020)	∼500M	30.75	43.78
9	*Ours*	171M	**31.59**	**44.21**

Table 2: The comparison of our approach (*Ours*) and previous approaches. #Param is the number of parameters.

et al. 2020) and MASS (Song et al. 2019), to initialize and fine-tune the parameters of NMT (line 5 and line 6).[5] Finally, two approaches about incorporating PTMs into NMT, *i.e.*, CTNMT (Yang et al. 2019) and BERT-fused NMT (Zhu et al. 2020), are reported in line 7 and 8.

The fine-tuning method cannot get substantial improvements. The knowledge distillation method only can get marginal improvements in both En→De and En→Fr tasks. CTNMT and BERT-fused NMT can get considerable improvements. While the proposed model largely outperforms all of them. Typically, compared to the BERT-fused NMT, which needs to using BERT-Large on-the-fly, our approach can get better results with fewer computational resources.

Employed on non-autoregressive NMT. Our work only changes the encoder of NMT, which is not affected by decoding mode. Thus, whether our approach can work on non-autoregressive translation (NAT) models (Gu et al. 2017; Ghazvininejad et al. 2019) is worthy to investigate. Compared to the autoregressive NMT, NAT relies more on the quality of the contextual representation.

Here, we employ the proposed approach based on the Mask-Predict (Ghazvininejad et al. 2019) framework[6] and evaluate in the En→De task. The results are shown in Table 3, our work largely outperforms the Mask-Predict baseline (+2.14). Then, compared to the BERT-Fused NAT (Zhu et al. 2020) and AB-Net (Guo et al. 2020), our work get 1.44 and 0.48 BLEU score gains, respectively. This experiment shows that our approach can get better contextual representation for NMT no matter which decoding modes.

Model	#Speedup	En→De
(Vaswani et al. 2017)	1.0x	27.3
(Gu et al. 2017)	2.36x	19.17
(Ghazvininejad et al. 2019)	1.7x	27.03
(Zhu et al. 2020)	0.95x	27.73
(Guo et al. 2020)	1.26x	28.69
Transformer-20	1.0x	29.82
Ours	1.57x	29.17

Table 3: The results of our method on the non-autoreggressive NMT (NAT). #Speedup is the decoding efficiency.

[5]We convert the parameters of the public mBART and MASS to our code base. The results in our experiment are consistent with the Liu et al. (2020).

[6]The detail of the setting is shown in Ghazvininejad et al. (2019).

Model	En→De	Ro→En
(Zhu et al. 2020)	–	39.10
(Caswell, Chelba, and Grangier 2019)	31.40	–
Transformer-Base	27.78	31.93
Transformer-12	29.32	33.79
+ Back-translation	31.63	37.51
Ours	31.59	36.71
+ Back-translation	32.56	39.48

Table 4: The comparison of our model and back-translation.

Combined with back-translation. An interesting question is whether pre-trained models and back-translation can work together. Here, to verify that, we make an experiment on the En→De and Ro→En tasks. Specifically, for the En→De task, we use 24M synthetic data from Caswell, Chelba, and Grangier (2019). For the Ro→En task, we use 2M back-translated data from Sennrich, Haddow, and Birch (2016).

The results are shown in Table 4, our method with back-translation can achieve 0.93 and 1.97 improvements compared to Transformer-12 with the same synthetic data in two tasks. Moreover, our model with back-translation gets 0.97 and 2.77 gains compared to only using parallel data. The results demonstrate that our method could work with back-translation to achieve better performance.

A hypothesis of our model can work together with back-translation is that the PTM enhances the encoding ability of NMT, while the back-translation improves the ability of the translation model. We will carry out further research and analysis in the future work.

Initializing the parameters of the decoder with PTMs. Generally, the target side unlabeled data can be used to pre-train a language model, which can serve as the decoder of NMT. Thus, we use all 216M monolingual data from WMT (Edunov et al. 2018) to pre-train a GPT model (Radford et al. 2018). Then, the pre-trained GPT is used to initialize the parameters of the decoder. As a comparison, we also use the mBERT to initialize the decoder in our framework.

The results are shown in Table 5, no matter what PTM is used, the fine-tuning method does not work. In addition, when freezing the parameters of the PTM, the performance drops dramatically, which shows that the current methods of using pre-trained decoders are not appropriate for NMT.[7]

Analysis

The effectiveness of training objectives. To verify the effectiveness of each objective in the partitioned multi-task learning framework, we make an ablation study here.

The results are shown in Table 6, both of the *target words prediction* ($\mathcal{L}_{\text{TWP}}$) and *target sentence prediction* ($\mathcal{L}_{\text{TSP}}$) affect the final performance. When ablating one of the two tasks, BLEU score drops about 0.4. BLEU drops 0.65 when ablating $\mathcal{L}_{\text{TWP}}$ and $\mathcal{L}_{\text{TSP}}$ together. When removing the $\mathcal{L}_{\text{MLM}}$, the BLEU score drops 0.53. Furthermore, when only using MLM task, our model gets comparable results to BERT-fused NMT, while the model size of BERT-fused NMT is about three times than ours (500M vs. 170M).

Then, we design four settings to evaluate our model: 1). without pre-training, 2). adopting the monolingual BERT on our model, 3). training the PTM by all tasks, in which the $\mathcal{L}_{\text{T}}$ is used to optimize the θ_{P}, and 4). freezing the PTM in the training process. The results are shown in the last four rows in Table 6, all of them reduce translation quality to varying degrees. Further, compared to training by NMT objective, freezing the PTM gets better performance. This results suggest that forgetting is a more serious problem than the representation mismatch.

The effectiveness of the hyperparameter M. In our approach, the hyperparameter M determines the grain of representation aggregation and the depth of the decoder. So, we make an experiment to compare the effects of different M. Typically, the PTM we used has 12 layers, so the decoder depth N is equal to $12/M$.

The results are shown in Table 7, our default setting ($M = 2$) gets the best performance and the decoding speed is similar to Transformer-20. When the M is 1, in which the PTM connects the decoder layer-by-layer, BLEU drops 0.45 compared to the default setting and the speed is only 0.41x compared to the Transformer-base. When the M is 4, BLEU drops 0.9 and the speed approaches the Transformer-base. This experiment shows that a suitable granularity of aggregation is useful for exploiting PTMs. Furthermore, when the M is 6, our model gets the similar performance of Transformer-Big, while the decoding speed is faster than the Transformer-Base. We think it can be used in some real-time scenarios.

Model	En→De	Δ
Ours	31.59	–
Initialize decoder by BERT	30.73	-0.86
Initialize decoder by GPT	31.22	-0.37
Use frozen BERT as decoder	25.48	-6.11
Use frozen GPT as decoder	27.69	-3.90

Table 5: The results of our model with pre-trained decoders.

Model	En→De	Δ
Ours	31.59	–
$-\mathcal{L}_{\text{TWP}}$	31.21	-0.38
$-\mathcal{L}_{\text{TSP}}$	31.17	-0.42
$-\mathcal{L}_{\text{TWP}}-\mathcal{L}_{\text{TSP}}$	30.94	-0.65
Without pre-training	30.17	-1.42
Adopt monolingual BERT	30.99	-0.60
Optimize θ_{P} by all tasks	30.62	-0.97
Freeze θ_{P} for all tasks	30.98	-0.61

Table 6: The ablation study of the additional training objectives in the proposed model.

[7]The prompt learning (Lester, Al-Rfou, and Constant 2021; Li and Liang 2021) is a promising direction to use pre-trained decoders in NMT. We will study it in the future.

Model	M/N	#Speedup	BLEU
Transformer-Base	–	1.0x	27.78
Transformer-Big	–	0.26x	29.13
Transformer-20	–	0.58x	29.82
Ours	1/12	0.41x	31.14
	2/6	0.52x	31.59
	4/3	0.93x	30.69
	6/2	1.21x	29.31

Table 7: The results of our model with different M. N is the depth of the decoder. #Speedup is the decoding efficiency.

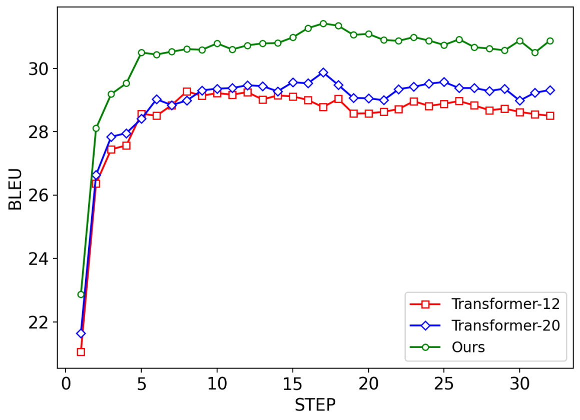

Figure 3: The BLEU curve of Transformer-12, Transformer-20 and our work on the test set of the WMT14 En→De task.

The convergence speed. We sampled checkpoints every 5000 training steps to evaluate the convergence speed. The results are shown in Figure 3. For simplicity, the STEP in the figure is the number of actual training steps divided by 5,000. The Transformer-12 converges faster than other models (between Step 5 and 10), because of its shallow layers. Our model and Transformer-20 converge almost simultaneously (between Step 15 and 20). Compared to Transformer-20, our method will not affect the convergence speed. Furthermore, our model can achieve a better result very early (Step 5), which is the result of the pre-trained encoder.

Related Work

Data-augmentation for NMT. Data-augmentation is widely used in NMT. Sennrich, Haddow, and Birch (2016) proposed back-translation to generate synthetic parallel data from target language unlabeled data. Zhang et al. (2018) proposed to jointly train the source-to-target and target-to-source NMT models with the pseudo parallel data. Although back-translation is useful, it is time-consuming and heavily relies on the quality of the synthetic data. Recently, how to use pre-trained models (PTMs) in NMT attracts many researchers' attention. Zhu et al. (2020) proposed a BERT-fused NMT to fuse pre-trained representations into NMT. Yang et al. (2019) proposed asymptotic distillation and dynamic switch methods for further utilizing PTMs. Weng et al. (2020a) proposed to use knowledge distillation to transfer knowledge from PTMs to NMT, which get considerable improvements. However the issue of how to exploiting PTMs into NMT is worthy of study. Unlike them, our approach has the advantage of fine-tuning and fusion to incorporate PTMs into NMT effectively.

Pre-training and fine-tuning. The paradigm of using a large amount of unlabeled data to pre-train a model and fine-tune it with labeled data has achieved great success. Peters et al. (2018) firstly proposed a Bi-LSTM based PTM (ELMo) to learn contextual information. Radford et al. (2018) proposed to use a self-attention network (GPT) as backbone. Then, Devlin et al. (2019) proposed to use a masked language model objective to capture bi-directional encoder representation (BERT). Following them, several varieties are proposed, Song et al. (2019) and Liu et al. (2020) extended to a sequence to sequence structure for text generation, Lample and Conneau (2019) and Conneau et al. (2020) extend to a cross-lingual manner for cross-lingual tasks.

Another key factor is the fine-tuning strategy, which largely affects the performance of PTMs (Gururangan et al. 2020). Fang et al. (2020) proposed several fine-tuning tricks according to different tasks. Gururangan et al. (2020) pointed out that using in-domain data continuously trains the PTM before fine-tuning can improve the performance of downstream tasks. Inspired by them, our work adopts multi-task learning to fine-tune the PTM in the NTM training process, reducing the gap between the bilingual and monolingual tasks.

Multi-task learning in NMT. Multi-task learning has been widely used in NMT. Dong et al. (2015) proposed to share an encoder between different translation tasks to exploit multi lingual knowledge. Luong et al. (2015) proposed to jointly learn the translation task for different languages, the parsing task and the image captioning task, with a shared encoder or decoder. Zhang and Zong (2016) proposed to use multi-task learning for incorporating source side monolingual data in NMT. Weng et al. (2017) and Ma et al. (2018) proposed to introduce a bag-of-words training objective to model the future information in NMT. Wang, Zhai, and Hassan (2020) proposed to use multi-task learning in multilingual NMT. Weng et al. (2020b) exploited multi-task learning framework to improve the faithfulness of NMT. Different from these attempts, our work focus on tuning the PTM to map universal representation to task-specific representation.

Conclusion

In this paper, we propose to deep fuse pre-trained models into NMT effectively and efficiently. In terms of the model structure, we propose a layer-wise coordination structure to exploit the pre-trained representation into NMT. We introduce a partitioned multi-task learning framework for learning the task-specific representation, reducing the gap from the different training objectives. Our extensive experiments demonstrate that the proposed approach achieves considerable gains on three WMT translation tasks and outperforms previous work with much smaller model sizes.

Acknowledgements

We would like to thank the reviewers for their insightful comments. This work is supported by the National Natural Science Foundation of China (Grant No. 62036004) and National Key R&D Program of China (2018YFB1403202).

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv:1607.06450*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473*.

Bapna, A.; Chen, M. X.; Firat, O.; Cao, Y.; and Wu, Y. 2018. Training deeper neural machine translation models with transparent attention. In *EMNLP*, 3028–3033.

Caswell, I.; Chelba, C.; and Grangier, D. 2019. Tagged back-translation. *arXiv:1906.06442*.

Chen, G.; Ma, S.; Chen, Y.; Dong, L.; Zhang, D.; Pan, J.; Wang, W.; and Wei, F. 2021. Zero-Shot Cross-Lingual Transfer of Neural Machine Translation with Multilingual Pretrained Encoders. In *EMNLP*, 15–26.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *ICML*, 1597–1607.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*, 1724–1734.

Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzmán, F.; Grave, E.; Ott, M.; Zettlemoyer, L.; and Stoyanov, V. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *ACL*, 8440–8451.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Dong, D.; Wu, H.; He, W.; Yu, D.; and Wang, H. 2015. Multi-Task Learning for Multiple Language Translation. In *ACL*, 1723–1732.

Dou, Z.-Y.; Tu, Z.; Wang, X.; Shi, S.; and Zhang, T. 2018a. Exploiting Deep Representations for Neural Machine Translation. In *EMNLP*, 4253–4262.

Dou, Z.-Y.; Zhou, H.; Huang, S.-J.; Dai, X.; and Chen, J. 2018b. Dynamic Oracle for Neural Machine Translation in Decoding Phase. In *LREC*.

Edunov, S.; Ott, M.; Auli, M.; and Grangier, D. 2018. Understanding Back-Translation at Scale. In *EMNLP*, 489–500.

Fang, H.; and Xie, P. 2020. CERT: Contrastive Self-supervised Learning for Language Understanding. *arXiv:2005.12766*.

Fang, Y.; Wang, S.; Gan, Z.; Sun, S.; and Liu, J. 2020. FILTER: An Enhanced Fusion Method for Cross-lingual Language Understanding. *arXiv:2009.05166*.

Ghazvininejad, M.; Levy, O.; Liu, Y.; and Zettlemoyer, L. 2019. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. In *EMNLP-IJCNLP*, 6112–6121.

Goodfellow, I. J.; Mirza, M.; Xiao, D.; Courville, A.; and Bengio, Y. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv:1312.6211*.

Gu, J.; Bradbury, J.; Xiong, C.; Li, V. O.; and Socher, R. 2017. Non-autoregressive neural machine translation. *arXiv:1711.02281*.

Guo, J.; Zhang, Z.; Xu, L.; Wei, H.-R.; Chen, B.; and Chen, E. 2020. Incorporating BERT into Parallel Sequence Decoding with Adapters. In *NeurIPS*, 10843–10854.

Gururangan, S.; Marasović, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; and Smith, N. A. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *ACL*, 8342–8360.

He, T.; Tan, X.; Xia, Y.; He, D.; Qin, T.; Chen, Z.; and Liu, T.-Y. 2018. Layer-wise coordination between encoder and decoder for neural machine translation. In *NIPS*, volume 31.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2020. Supervised contrastive learning. *arXiv:2004.11362*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Lample, G.; and Conneau, A. 2019. Cross-lingual Language Model Pretraining. *arXiv:1901.07291*.

Lee, S.; Lee, D. B.; and Hwang, S. J. 2020. Contrastive learning with adversarial perturbations for conditional text generation. *arXiv:2012.07280*.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. *arXiv:2104.08691*.

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

Li, B.; Wang, Z.; Liu, H.; Jiang, Y.; Du, Q.; Xiao, T.; Wang, H.; and Zhu, J. 2020. Shallow-to-Deep Training for Neural Machine Translation. In *EMNLP*, 995–1005.

Li, X. L.; and Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv:2101.00190*.

Liu, Y.; Gu, J.; Goyal, N.; Li, X.; Edunov, S.; Ghazvininejad, M.; Lewis, M.; and Zettlemoyer, L. 2020. Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Luong, M.-T.; Le, Q. V.; Sutskever, I.; Vinyals, O.; and Kaiser, L. 2015. Multi-task sequence to sequence learning. *arXiv:1511.06114*.

Ma, S.; Sun, X.; Wang, Y.; and Lin, J. 2018. Bag-of-Words as Target for Neural Machine Translation. In *ACL*, 332–338.

Neubig, G.; Dou, Z.-Y.; Hu, J.; Michel, P.; Pruthi, D.; and Wang, X. 2019. compare-mt: A Tool for Holistic Comparison of Language Generation Systems. In *NAACL (Demonstrations)*, 35–41.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*, 311–318.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep Contextualized Word Representations. In *NAACL-HLT*, 2227–2237.

Qiu, X.; Sun, T.; Xu, Y.; Shao, Y.; Dai, N.; and Huang, X. 2020. Pre-trained Models for Natural Language Processing: A Survey. *SCIENCE CHINA Technological Sciences*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *CoRR*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. In *OpenAI blog*, volume 1, 9.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *ACL*, 86–96.

Song, K.; Tan, X.; Qin, T.; Lu, J.; and Liu, T.-Y. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *ICML*, 5926–5936.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, volume 30.

Wang, Q.; Li, B.; Xiao, T.; Zhu, J.; Li, C.; Wong, D. F.; and Chao, L. S. 2019. Learning deep transformer models for machine translation. In *ACL*, 1810–1822.

Wang, Q.; Li, F.; Xiao, T.; Li, Y.; Li, Y.; and Zhu, J. 2018. Multi-layer Representation Fusion for Neural Machine Translation. In *COLING*, 3015–3026.

Wang, Y.; Zhai, C.; and Hassan, H. 2020. Multi-task Learning for Multilingual Neural Machine Translation. In *EMNLP*, 1022–1034.

Wei, X.; Weng, R.; Hu, Y.; Xing, L.; Yu, H.; and Luo, W. 2021. On Learning Universal Representations Across Languages. In *ICLR*.

Wei, X.; Yu, H.; Hu, Y.; Zhang, Y.; Weng, R.; and Luo, W. 2020. Multiscale Collaborative Deep Models for Neural Machine Translation. *ACL*, 414–426.

Weng, R.; Huang, S.; Zheng, Z.; Dai, X.; and Chen, J. 2017. Neural Machine Translation with Word Predictions. In *EMNLP*, 136–145.

Weng, R.; Yu, H.; Huang, S.; Cheng, S.; and Luo, W. 2020a. Acquiring Knowledge from Pre-trained Model to Neural Machine Translation. In *AAAI*, 9266–9273.

Weng, R.; Yu, H.; Wei, X.; and Luo, W. 2020b. Towards enhancing faithfulness for neural machine translation. In *EMNLP*, 2675–2684.

Wu, L.; Wang, Y.; Xia, Y.; Tian, F.; Gao, F.; Qin, T.; Lai, J.; and Liu, T.-Y. 2019. Depth growing for neural machine translation. In *ACL*, 5558–5563.

Yang, J.; Wang, M.; Zhou, H.; Zhao, C.; Yu, Y.; Zhang, W.; and Li, L. 2019. Towards Making the Most of BERT in Neural Machine Translation. In *EMNLP*, 9378–9385.

Zhang, J.; and Zong, C. 2016. Exploiting Source-side Monolingual Data in Neural Machine Translation. In *EMNLP*, 1535–1545.

Zhang, Z.; Liu, S.; Li, M.; Zhou, M.; and Chen, E. 2018. Joint training for neural machine translation models with monolingual data. In *AAAI*, volume 32.

Zhao, T.; Zhao, R.; and Eskenazi, M. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*, 654–664.

Zhu, J.; Xia, Y.; Wu, L.; He, D.; Qin, T.; Zhou, W.; Li, H.; and Liu, T.-Y. 2020. Incorporating BERT into Neural Machine Translation. In *arXiv:2002.06823*.