Dynamic Attention Aggregation with BERT for Neural Machine Translation

JiaRui Zhang[1,2], HongZheng Li[†], ShuMin Shi[†], HeYan Huang[*†], Yue Hu[1,2] and XiangPeng Wei[1,2]

[1]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[2]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China*
[†]*School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China*
{zhangjiarui, huyue, weixiangpeng}@iie.ac.cn, {lihongzheng, bjssm, hhy63}@bit.edu.cn

***Abstract*—The recently proposed BERT has demonstrated great power in various natural language processing tasks. However, the model does not perform effectively on cross-lingual tasks, especially on machine translation. In this work, we propose three methods to introduce pre-trained BERT into neural machine translation without fine-tuning. Our approach consists of a) a linear-attention aggregation that leverages a parameter matrix to capture the key knowledge of BERT, b) a self-attention aggregation which aims to learn what is vital for input and output, and c) a switch-gate aggregation to dynamically control the balance of the information flowing from the pre-trained BERT or the NMT model. We conduct experiments on several translation benchmarks and substantially improve over 2 BELU points on the IWSLT'14 English - German task with switch-gate aggregation method compared to a strong baseline, while our proposed model also performs remarkably on the other tasks.**

I. Introduction

Pre-training Language Models (LM) like ELMO [1], BERT [2], GPT-2 [3], [4], cross-lingual language model (XLM) [5] and MASS [6] have significantly boost the performances of several natural language processing (NLP) tasks by transferring prior knowledge learned from large amount of unlabeled data to downstream tasks [1], such as classification, question answering, and sequence labeling. Among these different pre-training mechanisms, BERT [2], which employs Transformer encoder architecture [7] and trains a bi-directional LM, has become one of the most successful techniques. Since then, there are large numbers of variants of BERT, like XLNet [8], RoBERTa [9], which achieve state-of-the-art benchmarks [10] for many NLP tasks.

In recent years, Neural Machine Translation (NMT) has achieved remarkable performance on large-scale parallel corpora [11], [12], [13]. Considering the complexity of existing NMT models, it is difficult to improve the effectiveness directly. Naturally, the idea of using pre-trained models to improve the performance of machine translation has arisen and is brought up by some researchers [5], [6], [14]. However, due to the limitation of computing resources, training a pre-trained model from scratch is difficult [15]. Thus, we consider using the existing pre-trained BERT model to improve the effectiveness of machine translation.

* Corresponding author

However, incorporating BERT into NMT is quite challenging and using BERT directly doesn't always achieve satisfactory results. Compared with other tasks where direct BERT fine-tuning works well, NMT has two distinct features, the availability of large amounts of training corpora and the high capacity of the baseline NMT models (i.e. Transformer) [16], which need lots of updating steps to adapt to the high-capacity model well on large-scale data [17]. However, updating overmuch causes BERT to forget universal knowledge from pre-training, which is named the *catastrophic forgetting problem* [18].

The existing practical accomplishments are integrating BERT as an additional representation into each layer of the encoder or decoder [15], or using BERT as a teacher model to guide the training on Transformer encoder side [17]. However, in BERT-fused NMT [15], the fusion method of pre-trained BERT and Transformer encoder or decoder layer representation is relatively simple (just averaged). Moreover, the settings of the drop-net trick during the training and inference are not consistent, and the authors have not explained this. In CTNMT [17], there is no improvement on the decoder side.

In this work, we propose three new fusion methods of pre-trained BERT and Transformer encoder or decoder layer representation, which follow the previous work (BERT-fused NMT; CTNMT). Firstly, at each layer of the encoder and decoder, we introduce a parameter matrix to weigh the representations of BERT and vanilla NMT. The vanilla NMT model represents in-domain information of the input, while the BERT model represents out-of-domain information of the input. Intuitively, the importance of the two parts can be automatically measured by a parameter matrix. secondly, we use a self-attention mechanism to incorporate BERT into vanilla NMT, which can learn the word dependencies within a sentence and capture the internal structure of the sentence. Thirdly, We utilize a switch gate mechanism, inspired by the significant results on gated recurrent units in RNN [19]. Different from CTNMT , we add switch gates at each layer of the encoder and decoder. As each layer of Transformer will learn different features of the input (for example, shallow layers learn syntactic information and deep ones learn semantic information) [20], [21], integrating BERT with each level of the Transformer can better introduce the knowledge of BERT to the input adequately. Similarly,

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

each layer of BERT also represents different syntactic and semantic information [22], [23]. In this study, we compare the fusion results of different BERT and Transformer layers.

We conduct experiments on WMT and IWSLT tasks. For IWSLT tasks, our experiments on English → German, German → English and English → French datasets show gains of up to 2.13, 1.22 and 1.16 BLEU scores respectively. For WMT tasks, with Transformer-base architecture, we increase 2.14 BELU scores compared to a strong baseline on En → De dataset, and even surpass some models whose parameters are far more than our model. Experimental results demonstrate that our model can significantly outperform the strong standard baseline (i.e. Transformer).

The main contributions of our works can be summarized as:

- We propose three new attention aggregation methods to dynamically incorporate BERT into Neural Machine Translation with a Transformer-based framework.
- We empirically show that fusion different layers of BERT and vanilla NMT model have an important impact on the translation effect.
- Our approaches achieve significant improvements over the strong baseline Transformer model on three common translation datasets.

The rest parts of this paper are as follows. Related work is introduced in Section 2. The Approach of our model is reported in Section 3. Then we show our experiment details in Section 4. Finally, in Section 5, we conclude our work and further discuss some future plans.

II. RELATED WORK

With the latest advances and performance improvements in neural networks, pre-training technology develops rapidly. After Mikolov et al. [24] and Pennington et al. [25] proposing word2vec and Glove, we could apply distributional representations to individual words, which greatly improved the capability of NLP models. ELMO [1] proposed to merge the different network layers to obtain effective word representations, based on the bi-directional LSTM model [26], which was widely applied to question answering, textual entailment, etc. Inspired by ELMO, Radford et al. [3] employed a self-attention network based language model (GPT) to replace the BI-LSTM structure, which further improved the performance of the pre-training model. Then, BERT [2] appeared, which was one of the widely adopted pre-training approaches for model initialization. The structure of BERT is evolved from the encoder of Transformer [7]. Owing to the great achievements of BERT on various NLP tasks, variants of BERT have been proposed. Lample et al. [5] adapted the objectives of BERT to incorporate cross-lingual supervision from parallel data to learn cross-lingual language models (XLM). Masked Sequence to Sequence Pre-training (MASS) [6] adopted the encoder-decoder framework to reconstruct a sentence fragment from the remaining part of the sentence that had not been masked [27].

Several approaches had been proposed recently to incorporate BERT into Neural Machine Translation. Intuitively, the conventional method was using the BERT model to initialize the NMT model and then fine-tuning it on downstream tasks, but it was even worse than the standard Transformer without using BERT [15]. Another common approach of utilizing BERT was employing it as inputs to the NMT model [1]. Although it was better than the fine-tuning ones, it did not make the most use of BERT in NMT, and also had not improved significantly.

Based on the above two traditional methods, [15] proposed a new algorithm, BERT-fused NMT, in which they fed BERT into all Transformer encoder and decoder layers by introducing attention mechanism rather than just using BERT as input embeddings only. Almost at the same time, [17] put forward CTNMT, a model that contained three techniques to integrate the knowledge of pre-trained BERT and vanilla NMT, namely asymptotic distillation, dynamic switch for knowledge fusion, and rate-scheduled updating. Besides, [45] used an APT framework for acquiring knowledge from pre-trained model to NMT, which included two modules: a) a dynamic fusion mechanism; b) a knowledge distillation paradigm.

Our approaches follow the previous BERT attention mechanism in BERT-fused NMT and introduce the switch gate mechanism into the model like CTNMT. Simultaneously, we also propose two other patterns to improve the model.

III. APPROACH

In this section, we first introduce the background of NMT, then present our methods in detail.

A. Background

NMT aims to maximize the conditional probability $p\left(y_1, \cdots, y_{T'} | x_1, \cdots, x_T\right)$, given a source and target sentence pair $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_1, x_2, \cdots, x_T)$ and $\mathbf{y} = (y_1, y_2, \cdots, y_{T'})$, T and $T^{'}$ are the length of $\mathbf{x}$ and $\mathbf{y}$, respectively.

At training time, based on encoder-decoder architecture, NMT first transforms the input sentence with tokens $x_1, x_2, \cdots, x_T$ into a sequence of hidden states $\boldsymbol{H} = \{\boldsymbol{h}_t\}_{t=1}^{T}$, and then uses the hidden states to predict the conditional distribution of each target token $p\left(y_j | \boldsymbol{H}, y_{<j}\right)$, given the previous ground truth target tokens [28]. The training loss on corpus $\boldsymbol{C}$ is defined as:

$$\boldsymbol{L}\left(\theta_{mt}\right) = \frac{1}{|\boldsymbol{C}|} \sum_{(\mathbf{x},\mathbf{y}) \in C} -\log \boldsymbol{P}\left(\boldsymbol{y} | \boldsymbol{x}; \theta_{mt}\right) \tag{1}$$

where θ_{mt} is a set of model parameters. At inference, the target sentence is generated by left-to-right decoding.

Different neural architectures have been proposed with the goal of improving effectiveness on NMT, in which Transformer [7] has shown strong results and surpassed prior state-of-the-art architectures based on recurrent networks [12], [29]. Thus, in this work, we choose Transformer as our base model.

Attention mechanism [11] is widely applied to the field of NMT, which focuses on important points and ignore other unimportant factors. The attention-based models allow a single token (e.g., a word or subword) in a sequence to be represented

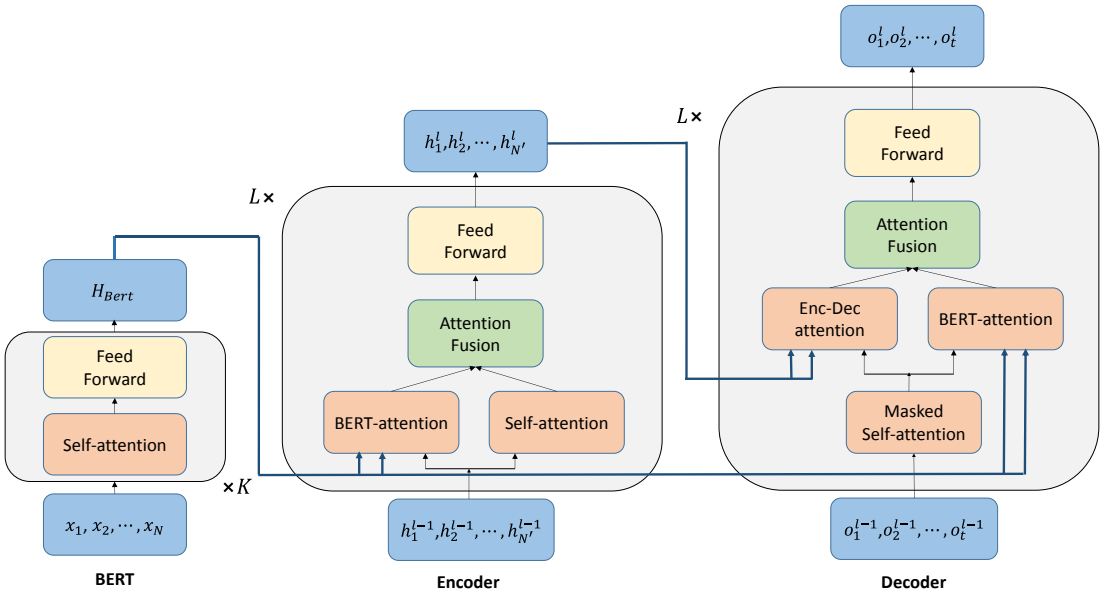

Fig. 1. The main architecture of our model

as a combination of all tokens in the sequence [29]. Let $att(q, K, V)$ denote the attention model,in which q, K and V represent query, key and value respectively. q is a d-dimensional vector, and in this paper, K and V are the same matrices. Each $k_i \in K$ is also d-dimension vectors, $i \in [|K|]$. The attention model can be calculated by:

$$\alpha_i^T = \frac{1}{Z} \cdot \exp\left(\frac{(W_q q)^T (W_k k_i)}{\sqrt{d_k}}\right), \tag{2}$$

$$Z = \sum_{i=1}^{|K|} \exp\left(\frac{(W_q q)^T (W_k k_i)}{\sqrt{d_k}}\right) \tag{3}$$

$$\mathrm{att}(q, K, V) = \sum_{i=1}^{|V|} \alpha_i^T W_v v_i, \tag{4}$$

where W_q, W_K and W_V are parameters of the NMT model, and d_k is the dimension of each attention head.

B. Algorithm

In this section, we present three methods to incorporate BERT into NMT, and an illustration of our model is shown in Fig. 1. The whole training process can be summarized as follows step by step:

- Feeding the input x into BERT, Transformer encoder and decoder. Especially, we tokenize the input with BERT tokenizer and standard tokenizer respectively.
- Sending the output of BERT (usually the last layer) into the attention model named BERT-attention, then fuse the BERT-attention into self-attention or encoder-decoder-attention [7] in each layer of encoder or decoder with three new methods.
- Training with the pre-trained Transformer model and frozen BERT model based on [7]. Continue the decoding process until meeting the end token.

The key difficulty of this training algorithm is to design the fusion methods of BERT-attention and self-attention or encoder-decoder attention. In this paper, we propose three different strategies: *linear aggregation*, *self-attention aggregation* and *switch-gate aggregation*.

1) ***Linear aggregation****:* Mikolov et al. [30] and Wu et al. [31] indicate the representation space of similar languages can be transmitted through the linear mapping. In our scenario, BERT-attention represents the universal information and Transformer represents the special downstream task information, which is in the same language. Intuitively, we can train a parameter matrix to balance the two parts.

Assuming $\boldsymbol{x}$ and $\boldsymbol{y}$ are the all source and target sentences from corpus C. For any sentence pair (x, y), where $x \in \boldsymbol{x}$ and $y \in \boldsymbol{y}$, let n_x and n_y denote the number of the units (e.g. wordpiece) in x and y. And we define H_{Bert} denoting the output of BERT. The $h_{Bert,i} \in H_{Bert}$ represents the i-th token of the input x through BERT encoding.

On Transformer encoder, let H_E^l be the hidden representation of the l-th layer. Specially, H_E^0 refers to the word embedding layer. Denote the i-th cell in H_E^l as e_i^l for any $i \in [n_x]$.

$$\begin{aligned} e_i^l = W_i^l \big(&\mathrm{att}_S\left(e_i^{l-1}, H_E^{l-1}, H_E^{l-1}\right) \\ &+ \mathrm{att}_B\left(e_i^{l-1}, H_{Bert}, H_{Bert}\right)\big), \forall i \in [n_x] \end{aligned} \tag{5}$$

Where $W_i^l \in \mathbb{R}^{2d*d}$ (d means the dimension of word embedding) is the matrix to weigh the representation of BERT and vanilla NMT, and att_S and att_B represent attention models (4) with distinct parameters. Then each e_i^l is processed

by the non-linear activation unit, the norm layer [32] and the feed-forward neural network the same as standard Transformer [7]. The encoder eventually outputs H_E^L from the last layer.

During the Transformer decoder preceding time step t, let $O_{<t}^l = \left(o_1^l, \cdots, o_{t-1}^l\right)$ denotes the hidden state of l-th layer. Similar to vanilla NMT [7], o_1^0 is a special token that indicating the start of a sentence, and o_t^0 represents the embedding of predicted word at time-step $t-1$. In the l-th layer, o_t^l can be calculated as:

$$\begin{aligned} \tilde{o}_t^l &= \mathrm{att}_{MS}\left(o_t^{l-1}, O_{<t+1}^{l-1}, O_{<t+1}^{l-1}\right) \\ o_t^l &= \hat{W}_i^l\left(\mathrm{att}_E\left(\tilde{o}_t^l, H_E^L, H_E^L\right)\right. \\ &\quad \left. + \mathrm{att}_B\left(\tilde{o}_t^l, H_{BERT}, H_{BERT}\right)\right) \end{aligned} \tag{6}$$

Where att_{MS}, att_E and att_B are masked self-attention model, encoder-decoder attention model and BERT attention model respectively. After (6), o_t^l is handled by the non-linear activation unit, the norm layer and the feed-forward network, in which we can eventually obtain o_t^L. Then, put o_t^L into a linear transformation and softmax layer to get the t-th predicted word $\hat{y}_t$. When meeting $< EOS >$ tag, the process of decoding finishes.

*2) **self-attention aggregation**:* In Transformer-based architectures, a self-attention mechanism is applied to draw global dependencies between inputs and outputs rather than the use of recurrence in neural networks [7]. We follow this idea and use it to find out which words the model should pay more attention to.

On Transformer encoder, different from the above method *linear aggregation*, *self-attention aggregation* mechanism first concatenates the self-attention calculated by the previously hidden representation and BERT-output attention unit as a new embedding D_E^l at l-th layer, then we send D_E^l to a new self-attention layer. Specially, we denote the i-th cell in D_E^l as d_i^l for any $i \in [n_x]$.

$$\tilde{d}_i^l = \left[\mathrm{a}_S\left(d_i^{l-1}, D_E^{l-1}, D_E^{l-1}\right); \mathrm{a}_B\left(d_i^{l-1}, H_{Bert}, H_{Bert}\right)\right] \tag{7}$$

$$d_i^l = \mathrm{a}_S\left(\tilde{d}_i^l, \tilde{D}_E^l, \tilde{D}_E^l\right) \tag{8}$$

Where a_S and a_B are shorthands of att_S and att_B respectively, $[;]$ is the concatenation operation, which combines information from BERT and vanilla NMT model. Then each d_i^l is processed by a norm layer and a feed-forward network as a descending dimension layer. Finally, the encoder outputs D_E^L from the last layer.

On Transformer decoder, which is similar to the method on encoder side, we let $M_{<t}^l = \left(m_1^l, \cdots, m_{t-1}^l\right)$ represents the hidden state of l-th layer, during the Transformer decoder preceding time step t. In the l-th layer, m_t^l can be defined as :

$$\hat{m}_t^l = \mathrm{a}_{MS}\left(m_t^{l-1}, M_{<t+1}^{l-1}, M_{<t+1}^{l-1}\right) \tag{9}$$

$$\tilde{m}_t^l = \left[\mathrm{a}_E\left(\hat{m}_t^l, D_E^L, D_E^L\right); \mathrm{a}_B\left(\hat{m}_t^l, H_{Bert}, H_{Bert}\right)\right] \tag{10}$$

$$m_t^l = \mathrm{a}_{MS}\left(\tilde{m}_t^l, \tilde{M}_{<t+1}^l, \tilde{M}_{<t+1}^l\right) \tag{11}$$

Then we add a linear descending dimension layer applying to m_t^l. The following steps are the same as the vanilla NMT model.

*3) **switch-gate aggregation**:* The gated recurrent cells have the ability to learn which data in a sequence should be kept in or left out [26], [33], inspired by which we propose to apply the similar idea of gates to the dynamic regulation of the balance of the information flowing from the pre-trained BERT or the vanilla NMT model. By doing that, they can pass relevant information down the long chain of sequences to make predictions [34].

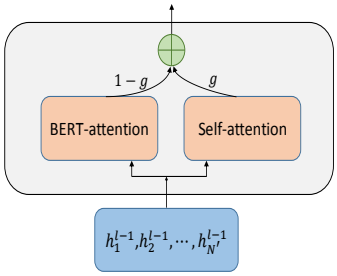

Fig. 2. The switch-gate aggregation

This mechanism has the same computation pattern on both Transformer encoder and decoder. Thus we adopt the encoder side as an example showed in Fig. 2. Formally, a context switch gate contains a sigmoid activation layer and an element-wise multiplication operation which can be defined as:

$$ATT_S^l = \|_{i=0}^{n_x-1}\, \mathrm{att}_S\left(e_i^{l-1}, H_E^{l-1}, H_E^{l-1}\right) \tag{12}$$

$$ATT_B^l = \|_{i=0}^{n_x-1}\, \mathrm{att}_B\left(e_i^{l-1}, H_{Bert}, H_{Bert}\right) \tag{13}$$

$$g_l = \sigma\left(W_l \cdot ATT_S^l + U_l \cdot ATT_B^l + b_l\right) \tag{14}$$

ATT_S^l and ATT_B^l are concat layers that concatenate the self-attention and BERT output attention information respectively at l-th layer. Where $\sigma(\cdot)$ is the logistic function, H_E^l represents the hidden state of the vanilla NMT at l-th layer, H_{Bert} is the output of the pre-trained BERT model, and b_l is the bias. Then we integrate the NMT model and pre-trained BERT as:

$$H_E^l = g \odot ATT_S^l + (1-g) \odot ATT_B^l \tag{15}$$

Where $\odot$ refers to element-wise multiplication. If g is set to 0, which means completely ignoring the information on Transformer encoder, the network simply acts as the fine-tuning approach, and if g is set to 1, the network degrades to the vanilla NMT model. On the NMT decoder side, the calculation method is similar to the above.

TABLE I
BLEU SCORES ON THREE IWSLT DATASETS

System	Architecture	En-Fr	En-De	De-En
Baseline				
Vaswani et al. [7]	Transformer base	40.86	28.49	34.93
Liu et al. [35]	Transformer base + RAdam	41.01	28.52	34.98
Lample et al. [5]	XLM + fine-tuning	40.71	29.27	33.15
Zhu et al. [15]	BERT-fused NMT	41.59	30.28	35.90
Our NMT systems				
Ours	Self-attention aggregation	42.03*	30.37	35.29
Ours	Linear aggregation	41.80	30.29	35.41
Ours	Switch-gate aggregation	**42.12**	**30.62**	**36.15**

"$*$" indicates that we only fuse the last layer of Transformer and BERT (See subsection "About NMT layers" for more details)

IV. EXPERIMENTS

We conduct our experiments on English $\leftrightarrow$ German and English $\rightarrow$ French tasks to verify our approach, the training details are as follows.

A. Datasets

For the rich-resource scenario, we work on WMT'14 En $\rightarrow$ De task [1], whose corpus size is 4.5M. $newstest2013$ is used as the validation set and $newstest2014$ as the test set, which is the same as [7].

For the low-resource scenario, the English $\leftrightarrow$ German training set is from IWSLT'14 [36] that compromises 160k sentence pairs. We lowercase all words, split 7k sentence pairs from the training dataset for validation and concatenate $dev2010$, $dev2011$, $tst2010$, $tst2011$, $tst2012$ as the test set [37], [38]. For English $\rightarrow$ French task, we choose IWSLT'17 as our training set, which contains 236k sentence pairs. we use $dev2010$ as our validation and the fusion of $tst2010$, $tst2011$, $tst2012$, $tst2013$, $tst2014$ and $tst2015$ as our test set.

In all translation tasks, we tokenize all data with MOSE tokenizer [39], and apply byte pair encoding (BPE) [40] to encode words through sub-word units. We build shared vocabularies of 32K, 10K and 16k sub-words for WMT'14 English $\rightarrow$ German, IWSLT'14 English $\leftrightarrow$ German and IWSLT'17 English $\rightarrow$ French respectively. We measure the translation quality with BLEU scores [41].

B. Training Details

To make this work easier to reproduce, this paper gives detailed strategies for training the proposed model. All experiments run on *fairseq-py*[2] [42].

For WMT'14 En $\rightarrow$ De task, the model configuration is *transformer_wmt_en_de*, in which the embedding size is 512, attention heads are 8 and feed-forward layer dimension is 2048. The hyper-parameters setting resembles [7].

For other low-resource tasks, the model configuration is *transformer_iwslt_de_en*, representing a six-layer model with embedding size 512 and feed-forward neural network dimension 1024. We employ $adam$ [43] as the optimization of the network with β_1 = 0.9, β_2 = 0.98 and $weight - decay$ = 0.0001. The $learning - rate$ is 0.0005 with inverse_sqrt scheduler, where $warmup{-}init{-}lr$ is 10^{-7}. Label smoothing ε_{ls} = 0.1 is used as regularization [44]. We set the batch size to 4000 per batch and limit sentence length to 100 BPE tokens. Note that the above hyperparameter settings are the same as those used in the baseline models.

We choose BERT-base-uncased[3] model for English $\rightarrow$ German and English $\rightarrow$ French tasks and Bert-base-german-cased[4] model for German $\rightarrow$ English. The BERT model is frozen during training. Besides, on the *self-attention aggregation* mechanism, we set attention fusion heads to 4 and 8 for IWSLT and WMT tasks respectively.

We first train an NMT model until convergence and initialize our model with the obtained model, and the rest parameters are initialized with xavier uniform. For English $\rightarrow$ German task, we choose the second-to-last hidden states of BERT to help the training of the vanilla NMT model, while the last layer of BERT is adopted in other tasks.

C. Main Results on IWSLT Datasets

The BLEU scores of IWSLT translation tasks are reported in TABLE I. We compare our system with the other four systems including a) Transformer base [7], b) RAadm which is a theoretically sound variant of Adam [35], c) XLM which is a cross-lingual pre-trained language model, and d) BERT-fused NMT [15], in which BERT-fused NMT is the strongest baseline that has significant advantages over other three baselines. On the above tasks, we reproduce the experiments presented in the four baseline paper. However, We do not achieve the results in [15].

In our model, we evaluate the effectiveness of the proposed three strategies respectively. Obviously, switch-gate performs best over the other two patterns, and improves the BLEU scores of the three tasks by 1.26, 2.13, 1.22 points respectively, demonstrating the effectiveness of our method. Linear aggregation algorithm also has a great improvement on En $\rightarrow$ Fr translation task by 0.94 BLEU. However, the self-attention mechanism does not achieve our expected results on IWSLT'14 De $\rightarrow$ En task, and even performs further worse

[1]http://www.statmt.org/wmt14/translation-task.html

[2]https://github.com/pytorch/fairseq

[3]https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-uncased-pytorch_model.bin

[4]https://s3.amazonaws.com/models.huggingface.co/bert/bert-base-german-cased-pytorch_model.bin

TABLE II
BLEU SCORES ON WMT'14 EN → DE DATASET

System	Architecture	BLEU
Baseline		
Vaswani et al. [7]	Transformer base	27.31
Yang et al. [17]	+ Dynamic switch	29.40*
Weng et al. [45]	+ Dynamic fusion	28.77
Weng et al. [45]	+ knowledge distillation	29.23
Vaswani et al. [7]	Transformer big	28.46
Lample et al. [5]	+ Fine-tuning	27.70
Lample et al. [5]	+ Frozen feature	28.70
Our NMT systems		
Ours	Transformer base	27.37
Ours	+ Self-attention aggregation	28.87
Ours	+ Linear aggregation	29.11
Ours	+ Switch-gate aggregation	**29.51**

"$*$": beam size is 8 and the dimensions of hidden states are 768

than BERT-fused NMT. We analyze the following two reasons for this result. Firstly, we consider that some information is lost when BERT attention unit and vanilla NMT self-attention unit are merged. In the future work, we will verify this assumption. Secondly, the self-attention module is updated through the backpropagation of the loss between the label and the predicted value. No other supervision information is introduced. Therefore, the supervision of the model is limited and it is easy to overfit the label.

D. Main Results on WMT'14 En → De Dataset

The results on WMT'14 En → De dataset are presented in TABLE II. We compare our model with several strong baselines including Transformer and some pre-trained methods in NMT.

Compared with Yang et al (CTNMT) [17], we also apply the gate mechanism to the decoder side. Experiments show that although their dimensions of all the hidden states and beam size values are larger, our *switch-gate aggregation* method still performs better. Weng et al. [45] propose an APT framework to acquire knowledge from the pre-trained model to NMT, which also has a dynamic fusion mechanism to fuse task-specific features adapted from general knowledge into NMT network. The biggest difference between our fusion methods is the different granularity, in which Weng et al. calculate each token of each layer in the Transformer and the pre-trained model. However, it is not that the finer the calculation granularity, the better the translation effect. According to experiments, except for the *linear aggregation* method, our BLEU scores are higher than them.

Transformer big model with fine-tuning even falls behind the baseline, and with a frozen feature approach during fine-tuning, it achieves a few gains over the Transfomer big model. The experimental results are consistent with our intuition that fine-tuning will cause *catastrophic forgetting problem* [18] and also verify the fine-tuning method does not fit NMT. All our methods with Transformer base architecture are better than the Transformer big model whose parameter size is far larger than ours.

TABLE III
RESULTS OF ABLATION STUDY (%).

Settings	BLEU
Transformer base	**40.86**
Switch-gate aggregation	**42.12**
- BERT-encoder attention	41.57
- feed BERT to all layers	41.45
- BERT-decoder attention	41.95
- feed BERT to all layers	41.82
- without frozen BERT	41.53
- aggregation-dropout	41.93
- pre-trained NMT model	41.07
Linear aggregation	**41.80**
- BERT-encoder attention	41.27
- BERT-decoder attention	41.64
- without frozen BERT	41.42
- aggregation-dropout	41.47
- activation-function	41.59
Self-attention aggregation	**42.03**
- BERT-encoder attention	41.44
- BERT-decoder attention	41.63
- aggregation-dropout	41.86

E. Ablation study

We conduct the following studies and the corresponding results can be found in TABLE III. The experiments are mainly conducted on IWSLT'14 En → Fr translation task.

1) For switch-gate strategy: We first verify whether each component is essential: we remove the BERT-encoder attention in (12-15) (i.e., att_B). The BLEU score drops from 42.12 to 41.57, which shows attention models are important for better performance. Besides, we feed BERT into the last layer of the decoder instead of all layers, and it makes BLEU scores drop by 0.12 points. It suggests that incorporating BERT into all layers can more fully fuse the two parts of information. The operation of removing BERT-decoder attention is similar to the above, which leads to a notable decrease in BLEU. Obviously, adding BERT into the encoder is more effective than the decoder. Then we jointly train the vanilla NMT model with BERT model. Although it can also boost the baseline from 40.86 to 41.53, it is not as good as freezing the BERT model. Aggregation-dropout is removed afterward, which is a dropout net after (15), and BLEU score drops by 0.19 points slightly. Finally, we train the NMT model from a randomly initialized model instead of using a pre-trained *fairseq* Transformer model, BLEU score affects a significant decline by 1.05 points comparing to 42.12, which nearly catches up the baseline.

2) For the other two strategies: We conduct similar experiments on *linear aggregation* and *self-attention aggregation*. We also find BERT attention models are important for better performance, in which BERT-encoder attention is more effective. Especially on the *linear aggregation* strategy where we remove the non-linear activation unit, which is the next step of (5) and (6). The BLEU score is 41.59, not as good as our proposed method. On the *self-attention aggregation* strategy, we just apply BERT to the last layer of the vanilla NMT, and there is a significant decline in the results of removing different

TABLE IV
RESULTS ON IWSLT WITH DIFFERENT LAYERS OF VANILLA NMT

Model	Layers	BLEU
Transformer	N/A	40.86
Embedding	N/A	41.38
Switch-gate aggregation	1st $\sim$ 5th	41.54
	Last layer	41.93
	All layers	**42.12**
Linear aggregation	1st $\sim$ 5th	41.59
	Last layer	41.71
	All layers	**41.80**
Self-attention aggregation	1st $\sim$ 5th	41.55
	Last layer	**42.03**
	All layers	41.64

modules, which is similar to *linear aggregation* strategy.

The above experiments and analysis can demonstrate how our approaches work and point out the effectiveness of each component in our architecture.

F. About NMT layers

We compare the effectiveness of applying our methods to different layers of vanilla NMT model on IWSLT'17 En $\rightarrow$ Fr dataset. The results are shown in TABLE IV. *Embedding method* means using pre-trained BERT as inputs to the NMT model, leaving the underlying linguistic information ignored, inspired by [1]. $1st \sim 5th$ represents integrating BERT into the first 5 layers of the Transformer model, while *Last layer* only fuses BERT with the last layer of the Transformer, and *All layers* are incorporating BERT into all levels of Transfomer.

On the embedding layer, the pre-trained embedding performs better than the embedding from the vanilla NMT, which can get a considerable improvement. Moreover, all of our methods are better than using pre-trained models as inputs to the NMT model.

On the *Switch-gate aggregation* and *Linear aggregation* mechanism, *All layers* obtain the best performance, while *Last layer* performs best on *Self-attention aggregation*. We think that the first two methods *Switch-gate aggregation* and *Linear aggregation* are essentially weighted fusion of the BERT and transformer representations, and adding more layers is equivalent to adding more additional information to each layer of the NMT model according to its requirements, thus the experimental result of *All layers* displays best. In contrast, *Self-attention aggregation* is merging the BERT and transformer representations into a new vector, and then automatically learn the information required for the machine translation task. We think that the shallow layers of Transformer focus on transforming the input representation to the high layer, and not directly connected to downstream tasks, so the external contextual knowledge does not help them much. Besides, High layers of the Transformer can obtain more gain from BERT compared with low layers.

G. About BERT layers

BERT captures the structure properties of language and composes a hierarchy of linguistic signals ranging from surface to semantic features [23].

TABLE V
RESULTS ON IWSLT WITH DIFFERENT LAYERS OF BERT

Model	En $\rightarrow$ De	En $\rightarrow$ Fr
Last Hidden Layer	30.51	**42.12**
Second-to-Last Hidden Layer	**30.62**	41.64
Third-to-Last Hidden Layer	30.41	41.32
Fourth-to-Last Hidden Layer	30.40	41.28

In [17], they find that the second-to-last layer of BERT works significantly better than the last hidden state. However, they only experimented on WMT14 English $\rightarrow$ German corpus, which represents the rich-resource scenario containing 4.5M parallel data.

We experiment on a low-resource scenario with *switch-gate aggregation* algorithm, in which we choose IWSLT'14 English $\rightarrow$ German, and IWSLT'17 English $\rightarrow$ French datasets. The results are shown in TABLE V.

On IWSLT'14 En $\rightarrow$ De translation task, we find that the second-to-last layer of BERT works best comparing with other hidden states, while on IWSLT'17 En $\rightarrow$ Fr translation task, the last layer performs significantly better than the second-to-last layer. Intuitively, the last layer of BERT focuses more on downstream tasks and better represents information at the semantic level. So on English $\rightarrow$ German, the last layer is biased to the LM tasks. However, on English $\rightarrow$ French translation task, where some words in English and French are close or even the same, the last layer of the BERT model works best with relatively similar semantic space.

V. CONCLUSION AND FUTURE WORK

In this work, we propose three simple but effective approaches for neural machine translation to incorporate BERT into Transformer, i.e., *self-attention aggregation*, *linear aggregation* and *switch-gate aggregation*, in which the BERT model is fully utilized by the encoder and decoder. Among the above methods, *switch-gate aggregation* performs best on several translation tasks, which leverages the gate mechanism to dynamically control the balance of the information flowing from the pre-trained BERT or the Transformer model. While the empirical results are strong, *switch-gate aggregation* surpasses Transformer-base by 2.13 BLEU score on IWSLT 14 English $\rightarrow$ German translation task. On other tasks, our approaches also achieve remarkable performance.

For future work, we consider starting from the semantic level to more refinedly extract the information related to the machine translation task in the BERT model. Then we want to compress the model and speed up inference time.

VI. ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (No.61732005, No.61671064) and the National Key RD Program of China under Grant (No.2018YFC0832104).

REFERENCES

[1] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

[4] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, 2019.

[5] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.

[6] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mass: Masked sequence to sequence pre-training for language generation," *arXiv preprint arXiv:1905.02450*, 2019.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[8] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *arXiv preprint arXiv:1906.08237*, 2019.

[9] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[10] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *arXiv preprint arXiv:1804.07461*, 2018.

[11] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," *Advances in NIPS*, 2014.

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

[13] B. Ji, Z. Zhang, X. Duan, M. Zhang, B. Chen, and W. Luo, "Cross-lingual pre-training based transfer for zero-shot neural machine translation," *arXiv preprint arXiv:1912.01214*, 2019.

[14] S. Edunov, A. Baevski, and M. Auli, "Pre-trained language model representations for language generation," *arXiv preprint arXiv:1903.09722*, 2019.

[15] J. Zhu, Y. Xia, L. Wu, D. He, T. Qin, W. Zhou, H. Li, and T. Liu, "Incorporating {bert} into neural machine translation," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=Hyl7ygStwB

[16] S. Clinchant, K. W. Jung, and V. Nikoulina, "On the use of bert for neural machine translation," *arXiv preprint arXiv:1909.12744*, 2019.

[17] J. Yang, M. Wang, H. Zhou, C. Zhao, Y. Yu, W. Zhang, and L. Li, "Towards making the most of bert in neural machine translation," *arXiv preprint arXiv:1908.05672*, 2019.

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

[19] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[20] Z.-Y. Dou, Z. Tu, X. Wang, S. Shi, and T. Zhang, "Exploiting deep representations for neural machine translation," *arXiv preprint arXiv:1810.10181*, 2018.

[21] Q. Wang, F. Li, T. Xiao, Y. Li, Y. Li, and J. Zhu, "Multi-layer representation fusion for neural machine translation," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 3015–3026.

[22] I. Tenney, D. Das, and E. Pavlick, "Bert rediscovers the classical nlp pipeline," *arXiv preprint arXiv:1905.05950*, 2019.

[23] G. Jawahar, B. Sagot, D. Seddah, S. Unicomb, G. Iñiguez, M. Karsai, Y. Léo, M. Karsai, C. Sarraute, É. Fleury *et al.*, "What does bert learn about the structure of language?" in *57th Annual Meeting of the Association for Computational Linguistics (ACL), Florence, Italy*, 2019.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[25] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "Spanbert: Improving pre-training by representing and predicting spans," *arXiv preprint arXiv:1907.10529*, 2019.

[28] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[29] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[30] T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *arXiv preprint arXiv:1309.4168*, 2013.

[31] P. Wu, S. Huang, R. Weng, Z. Zheng, J. Zhang, X. Yan, and J. Chen, "Learning representation mapping for relation detection in knowledge base question answering," *arXiv preprint arXiv:1907.07328*, 2019.

[32] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[33] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in neural information processing systems*, 2015, pp. 2377–2385.

[34] M. Nguyen, "Illustrated guide to lstm's and gru's: A step by step explanation," *Online] Towards Data Science*, 2018.

[35] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019.

[36] M. Cettolo, C. Girardi, and M. Federico, "Wit3: Web inventory of transcribed and translated talks," in *Conference of European Association for Machine Translation*, 2012, pp. 261–268.

[37] S. Edunov, M. Ott, M. Auli, D. Grangier, and M. Ranzato, "Classical structured prediction losses for sequence to sequence learning," *arXiv preprint arXiv:1711.04956*, 2017.

[38] M. Cettolo, J. Niehues, S. Stüker, L. Bentivogli, and M. Federico, "Report on the 11th iwslt evaluation campaign, iwslt 2014," in *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, 2014, p. 57.

[39] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens *et al.*, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, 2007, pp. 177–180.

[40] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[41] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[42] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," *arXiv preprint arXiv:1904.01038*, 2019.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," *arXiv preprint arXiv:1701.06548*, 2017.

[45] R. Weng, H. Yu, S. Huang, S. Cheng, and W. Luo, "Acquiring knowledge from pre-trained model to neural machine translation," *arXiv preprint arXiv:1912.01774*, 2019.