Multilingual Translation via Grafting Pre-trained Language Models

Zewei Sun[1], Mingxuan Wang[1], Lei Li[2*]
[1] ByteDance AI Lab
{sunzewei.v,wangmingxuan.89}@bytedance.com,
[2] University of California, Santa Barbara
lilei@cs.ucsb.edu

Abstract

Can pre-trained BERT for one language and GPT for another be glued together to translate texts? Self-supervised training using only monolingual data has led to the success of pre-trained (masked) language models in many NLP tasks. However, directly connecting BERT as an encoder and GPT as a decoder can be challenging in machine translation, for GPT-like models lack a cross-attention component that is needed in seq2seq decoders. In this paper, we propose **Graformer** to graft separately pre-trained (masked) language models for machine translation. With monolingual data for pre-training and parallel data for grafting training, we maximally take advantage of the usage of both types of data. Experiments on 60 directions show that our method achieves average improvements of 5.8 BLEU in x2en and 2.9 BLEU in en2x directions comparing with the multilingual Transformer of the same size[1].

1 Introduction

In recent years, pre-trained (masked) language models have achieved significant progress in all kinds of NLP tasks (Devlin et al., 2019; Radford et al., 2019). Among them, neural machine translation (NMT) is also explored by several attempts (Yang et al., 2020a; Zhu et al., 2020b; Rothe et al., 2020). The pre-training and fine-tuning style becomes an important alternative to take advantage of monolingual data (Yang et al., 2020c,b; Liu et al., 2020; Pan et al., 2021).

An intuitive question comes as: Can we bridge BERT-like pre-trained encoders and GPT-like decoders to form a high-quality translation model? Since they only need monolingual data, we can reduce the reliance on the large parallel corpus.
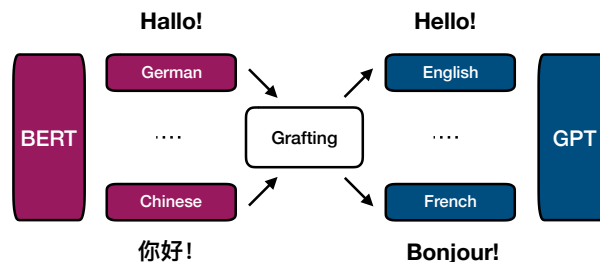

Figure 1: Grafting pre-trained (masked) language models like BERT and GPT for machine translation.

Moreover, if the combination of models is universal, it can be applied to translation for multiple languages, as is shown in Figure 1.

However, though many works successfully gain improvements by loading encoder/decoder parameters from BERT-like pre-trained encoders (Zhu et al., 2020b; Guo et al., 2020), they do not achieve satisfactory results with loading decoder parameters from GPT-like pre-trained decoders (Yang et al., 2020a; Rothe et al., 2020). Theoretically, the well-trained decoder model like GPT should bring better generation ability to the translation model. We suggest the outcome may be attributed to the architecture mismatch.

Pre-trained (masked) language models predict the current word solely based on the internal context while the translation decoder has to capture the source context. Specifically, the decoder in NMT has a "cross-attention" sub-layer that plays a transduction role (Bahdanau et al., 2015), while pre-trained models have none, as is shown in Figure 2. This mismatch between the generation models and conditional generation models makes it a challenge for the usage of pre-trained models as translation decoders.

Therefore, some previous works manually insert cross-attention sub-layer or adapters (Rothe et al., 2020; Ma et al., 2020; Guo et al., 2020). However, the extra implantation may influence the ability of the pre-trained model. Other works try to avoid

---

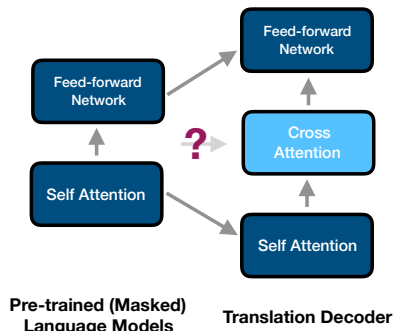[1]Our code will be public in https://github.com/sunzewei2715/Graformer

Figure 2: Taking the popular architecture Transformer (Vaswani et al., 2017) as an example, the translation model has a "cross-attention" sub-layer, while pre-trained (masked) language models have none.

this problem by directly pre-training a seq2seq model and conduct fine-tuning (Tang et al., 2020; Yang et al., 2020b; Luo et al., 2020). However, the pre-training objective is usually a variant of auto-encoding (Song et al., 2019; Liu et al., 2020), which is different from the downstream translation objective and may not achieve adequate improvements (Lin et al., 2020).

In this paper, we mainly focus on exploring the best way to simultaneously take advantage of the pre-trained representation model and generation model (e.g., BERT+GPT) without limiting their strengths. The primary target is to link the generation model to the source side and maintain the invariability of the architecture in the meantime. Therefore, we propose **Graformer**, with pre-trained models grafted by a connective sub-module. The structure of the pre-trained parts remains unchanged, and we train the grafting part to learn to translate. For universality and generalization, we also extend the model to multilingual NMT, achieving mBERT+mGPT.

Generally, the translation process can be divided into three parts: representation, transduction, and generation, respectively achieved by the encoder, cross-attention, and decoder. In multilingual NMT, the transduction can only be trained with multiple parallel data. But the rest two can be pre-trained with multiple monolingual data, which is tens or hundreds of the size of parallel one. To maximize the efficacy of each part, we firstly pre-train a multilingual BERT and multilingual GPT. Then they are grafted to implement translation. With the architecture consistency, we can reserve the language knowledge of the pre-trained models and obtain a strong translation model flexibly at the same time.

Experiments on 30 language directions show that our method improves the results of multilingual NMT by 2.9 and 5.8 BLEU on average. It also achieves gains of 9.2 to 13.4 BLEU scores on zero-shot translation settings. In addition, it verifies that such translation capability can be well transferred to other languages without fine-tuning on the target parallel corpus.

2 Related Work

This paper is related to a chain of studies of multilingual translation and pre-trained models.

2.1 Multilingual Neural Machine Translation

With the development of NMT, multilingual neural machine translation (MNMT) also attracts a great amount of attention. Dong et al. (2015); Firat et al. (2016a,b) take early attempts and confirm its feasibility. The most well-known work is from Johnson et al. (2017), who conduct a series of interesting experiments. And the usage of the language token style is widely accepted. Also, many subsequent works continuously explore new approaches in MNMT, such as parameter sharing (Blackwood et al., 2018; Wang et al., 2019b; Tan et al., 2019a), parameter generation (Platanios et al., 2018), knowledge distillation (Tan et al., 2019b), learning better representation (Wang et al., 2019a), massive training (Aharoni et al., 2019; Arivazhagan et al., 2019), interlingua (Zhu et al., 2020a), and adpater (Zhu et al., 2021). These works mainly utilize parallel data.

There are also some works taking advantage of monolingual corpus. Zhang et al. (2020); Wang et al. (2020) use back-translation (BT) to improve MNMT. However, for MNMT, BT is tremendously costly, reaching $O(n)$, or even $O(n^2)$. Siddhant et al. (2020); Wang et al. (2020) adopt multi-task learning (MTL), combining with other tasks such as masked language model (MLM) (Devlin et al., 2019), denoising auto-encoding (DAE) (Vincent et al., 2008), or masked sequence-to-sequence generation (MASS) (Song et al., 2019). However, the optimization target is different from translation, which may interfere with the training and limit the usage of extremely large-scale monolingual data.

2.2 Pre-trained Models

In recent years, pre-train models have become very popular in both research and industry communities.

With downstream fine-tuning, plenty of significant results are achieved in NLP field (Qiu et al., 2020).

Devlin et al. (2019); Liu et al. (2019); Conneau and Lample (2019); Conneau et al. (2020) take masked language model (MLM) as the training target. The input tokens are randomly masked, and the model learns the representation by maximizing their likelihood. Radford et al. (2018, 2019); Brown et al. (2020) use language model (LM) as their learning goal. With historical contexts, the model acquires language knowledge by learning to predict the next word. Raffel et al. (2020); Xue et al. (2020); Lewis et al. (2020); Liu et al. (2020); Lin et al. (2020) choose direct sequence-to-sequence (seq2seq) for training. The pre-train tasks can be machine translation, question answering, classification, etc.

2.3 Pre-trained Models for NMT

Since pre-trained models can significantly boost relevant tasks, several recent studies try to combine them with NMT. They can be roughly divided into two groups, depending on whether the models are pre-trained uniformly or separately.

2.3.1 United Style

The first category is pre-training directly on seq2seq tasks and providing downstream MT with consistent architectures. Tang et al. (2020) tune translation models from a pre-trained seq2seq model, mBART (Liu et al., 2020), and obtain significant improvements. Yang et al. (2020c) pre-train a seq2seq model with some input tokens replaced by another language from lexicon induction. Luo et al. (2020) pre-train the encoder and decoder in a single model that shares parameters. Then the parameters are partially extracted for tuning, depending on the tasks (NLU or NLG).

However, the pre-training objective of these works is usually a variant of auto-encoding (Song et al., 2019; Liu et al., 2020), which is different from the downstream translation objective and may not achieve adequate improvements (Lin et al., 2020).

2.3.2 Fused Style

The second category is pre-training the encoder or decoder independently and fusing them with the translation model in the fine-tuning stage. Yang et al. (2020a); Zhu et al. (2020b); Guo et al. (2020); Ma et al. (2020) fuse BERT/RoBERTa into NMT with extra encoders or adapters. Yang et al. (2020b)
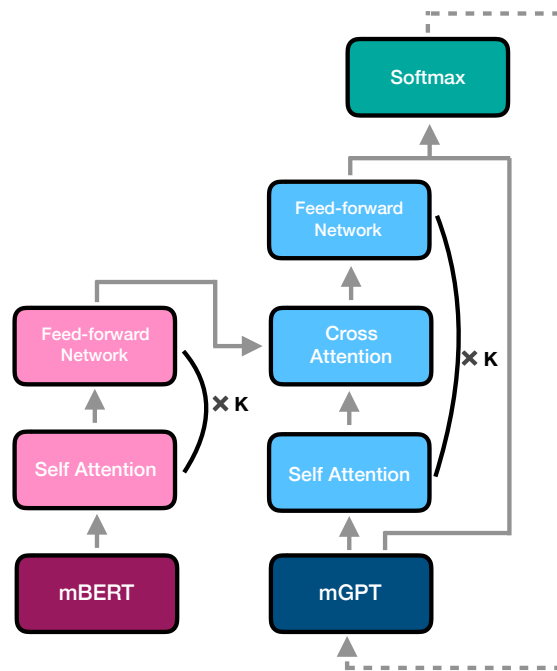

Figure 3: The model architecture of Graformer. The pre-trained multilingual encoder (mBERT) and decoder (mGPT) are grafted to achieve multilingual translation. The dashed line means feeding in the last token.

propose alternating language modeling as the target of the pre-trained encoder. Rothe et al. (2020) explore the usage of GPT but still manually insert extra cross-attention. Weng et al. (2020) use dynamic fusion mechanism and knowledge distillation to integrate the representation of the pre-trained models into NMT models.

These works either do not touch the decoder side or modify the architecture and conduct fine-tuning to fuse BERT/GPT into the decoder model. As mentioned in Section 1, the modification of the model architecture may influence the model ability and harm the performance.

3 Approach

To maintain the original model structure of pre-trained models, we propose Graformer, as is in Figure 3. For the encoder side, we stack another K-layers encoder ($K = 6$, in this paper) on pre-trained mBERT to help it adapt to the translation training. For the decoder side, we do similarly, except we append cross-attention layers to extract conditional context from the source. Unlike previous works, we maintain the integrality of mBERT and mGPT and do not change their architectures.

Finally, we employ a residual connection (He

et al., 2016) that we combine the hidden state outputted by mGPT and the grafting decoder. The summed context is then fed into the softmax layer. This integration is for utilizing the generation ability of the pre-trained decoder to help to generate a better language model.

As mentioned in Section 1, we try to take advantage of both multiple parallel data and multiple monolingual data so as to maximize the efficacy of representation, transduction, and generation, respectively. Therefore, our training methods can be separated into two stages: *1)* pre-train on the multiple monolingual data and obtain independent encoder (representation) and decoder (generation); *2)* fine-tune on the multilingual parallel data to graft two models (transduction).

3.1 Pre-train Multilingual BERT (Encoder for Representation)

Inspired by Devlin et al. (2019); Liu et al. (2019); Conneau and Lample (2019); Conneau et al. (2020), we use masked language model (MLM) as the training goal with the masked probability of 15%. Specifically, we adopt Transformer (Vaswani et al., 2017) encoder with N layers ($N = 6$, in this paper). To make cross-lingual token representation more universal, we add no language token as previous works do. The training goal is as follows:

$$\mathcal{L}_{MLM} = - \sum_{\hat{x} \in m(\mathbf{x})} \log p(\hat{x}|\mathbf{x}_{\backslash m(\mathbf{x})}) \quad (1)$$

$m(\mathbf{x})$ and $\backslash m(\mathbf{x})$ denote the masked words and rest words from $\mathbf{x}$

3.2 Pre-train Multilingual GPT (Decoder for Generation)

Inspired by Radford et al. (2018, 2019); Brown et al. (2020), we use auto-regressive language model (LM) as the training goal. Specifically, we adopt Transformer (Vaswani et al., 2017) decoder with N layers ($N = 6$, in this paper). To specify the generation language, we set a unique language token (e.g., `<2en>`) as the first input for the language model. The training goal is as follows:

$$\mathcal{L}_{LM} = - \sum_{t=1}^{T} \log p(x_t|\mathbf{x}_{<t}) \quad (2)$$

T denotes the length of sequence. $\mathbf{x}_{<t} = $ `<2lang>`$, x_1, x_2, ..., x_{t-1}$.

3.3 Fine-tune Multilingual Translation (Grafting for Transduction)

After obtaining the pre-trained encoder and decoder, we tune the model to link the representation model and generation model. The training goal is as follows:

$$\mathcal{L}_{MT} = \text{softmax}(W_{o_1} h_N + W_{o_2} h_{N+K}) \quad (3)$$

h_N denotes the hidden state of the last layer in mGPT. h_{N+K} denotes the hidden state of the last layer in the grafting decoder. W_{o_1} and W_{o_2} denote the corresponding output matrix. The former one shares the same parameters with the target-side embedding.

In the tuning stage, we freeze the pre-trained decoder parameters (including W_{o_1}) and tune the grafting parameters as well as the pre-trained encoder. Our ablation study shows that this setting yields the best performance, as is in the experiment section.

4 Experiments

In this paper, we perform many-to-many style multilingual translation (Johnson et al., 2017). The detailed illustrations of the datasets and implementation are as follows.

4.1 Datasets and Preprocess

- **Pre-training:** We use News-Crawl corpus [2] plus WMT datasets. We conduct deduplication and label the data by language. In the end, we collect 1.4 billion sentences in 45 languages, which is only one-fifth of that of mBART (Liu et al., 2020). The detailed list of languages and corresponding scales is in Appendix A.
- **Multilingual Translation:** We use TED datasets, the most widely used MNMT datasets, following Qi et al. (2018); Aharoni et al. (2019). We extract 30 languages [3] from & to English, with the size of 3.18M sentence pairs in raw data and 10.1M sentence pairs in sampled bidirectional data. The detailed list of language pairs and scales is in Appendix A. We download the data from the open source [4]

---

[2] http://data.statmt.org/news-crawl
[3] We use the corpus of "zh_cn" instead of "zh".
[4] https://github.com/neulab/word-embeddings-for-nmt

Model	bg	bn	bs	cs	de	el	es	et	fa	fi
Transformer	32.0	12.5	30.3	23.7	28.7	30.7	34.7	17.5	20.4	17.1
mBART	-	-	-	26.4	32.8	-	38.1	20.9	-	19.9
Graformer	**38.5**	**18.1**	**36.5**	**29.4**	**35.5**	**37.4**	**40.7**	**24.0**	**26.9**	**23.0**

Model	fr	hi	hr	hu	it	ja	kk	lt	mk	mr
Transformer	33.1	18.7	30.4	19.8	31.3	10.1	7.6	20.1	29.8	9.4
mBART	36.5	22.9	-	-	34.7	12.0	8.9	23.6	-	-
Graformer	**39.2**	**25.1**	**36.7**	**26.1**	**37.2**	**13.7**	**10.5**	**27.2**	**35.7**	**13.0**

Model	nl	pl	pt	ro	ru	sr	ta	tr	uk	zh
Transformer	28.9	19.7	34.8	28.6	20.8	29.0	5.8	18.7	23.4	15.6
mBART	32.9	-	-	32.2	22.6	-	-	22.6	-	18.1
Graformer	**35.2**	**25.1**	**41.5**	**35.1**	**25.1**	**35.6**	**10.2**	**25.5**	**28.9**	**19.9**

Table 1: The results of x→en directions, with average improvements of 5.8 against baseline (22.8→28.6)

Model	bg	bn	bs	cs	de	el	es	et	fa	fi
Transformer	28.8	11.3	23.4	16.6	23.7	25.9	33.0	14.0	12.5	12.1
mBART	-	-	-	17.7	25.8	-	35.2	14.1	-	13.2
Graformer	**33.0**	**14.1**	**26.3**	**20.2**	**27.8**	**29.8**	**37.5**	**16.1**	**14.2**	**14.4**

Model	fr	hi	hr	hu	it	ja	kk	lt	mk	mr
Transformer	33.5	15.3	23.2	14.7	28.9	11.1	3.4	12.8	22.2	9.3
mBART	35.8	16.5	-	-	30.6	12.6	3.0	14.2	-	-
Graformer	**37.8**	**18.1**	**26.8**	**17.2**	**32.5**	**12.8**	**3.8**	**15.9**	**25.7**	**10.6**

Model	nl	pl	pt	ro	ru	sr	ta	tr	uk	zh
Transformer	25.9	12.8	32.0	24.7	16.1	18.7	13.6	11.6	17.3	21.2
mBART	28.9	-	-	27.1	16.9	-	-	**13.4**	-	22.2
Graformer	**29.0**	**15.8**	**36.6**	**29.1**	**19.0**	**21.4**	**14.7**	13.3	**19.5**	**23.0**

Table 2: The results of en→x directions, with average improvements of 2.9 against baseline (19.0→21.9)

and conduct detokenization with Moses Detokenizer (Koehn et al., 2007) [5].

- **Zero-shot and Bilingual Translation:** We use WMT 2014 German-English (4.5M sentence pairs) and French-English (36M sentence pairs) datasets.
- **Sample:** Upsampling is an important way to improve the performance of low-resource pairs (Arivazhagan et al., 2019). Therefore, sentences are sampled according to a multinomial distribution with probabilities $\{q_i\}$, where $q_i \propto p_i^\alpha$, p_i is the proportion of $language_i$. For monolingual pre-training, we follow (Conneau and Lample, 2019; Liu et al., 2020) and set $\alpha = 0.7$. For parallel fine-tuning, we follow (Arivazhagan et al., 2019) and and set $\alpha = 0.2$ ($T = 5$).
- **Tokenization:** Like previous works, we use

sentencepiece (Kudo and Richardson, 2018) and learn a joint vocabulary of 64000 tokens.

4.2 Implementation Details

- **Architecture:** We use Transformer (Vaswani et al., 2017) as our basic structure with pre-norm style (Xiong et al., 2020), and GELU (Hendrycks and Gimpel, 2016) as activation function. Specifically, we adopt 1024 dimensions for the hidden state, 4096 dimensions for the middle FFN layer, and 16 heads for multi-head attention. Learnable position embedding is also employed. For baseline models, we use 12 layers. For pre-trained ones, we use Transformer encoder and decoder (without cross-attention) with 6 layers, respectively. For the grafting part, we add another 6 layers.
- **Training:** We train the models with a batch size of 320,000 tokens on 16 Tesla V100 GPUs. For pre-training, we go through the

total data for five times. Parameters are optimized by using Adam optimizer (Kingma and Ba, 2015), with $\beta_1 = 0.9$, $\beta_2 = 0.98$, with $warmup\_steps = 4000$. Without extra statement, we use dropout $= 0.3$ (Srivastava et al., 2014). Label smoothing (Szegedy et al., 2016) of value $= 0.1$ is also adopted. Besides, we use fp16 mixed precision training (Micikevicius et al., 2018) with Horovod library with RDMA inter-GPU communication (Sergeev and Del Balso, 2018).

- **Evaluation:** We uniformly conduct beam search with size $= 5$ and length penalty $\alpha = 0.6$. For hi, ja, and zh, we use Sacre-BLEU (Post, 2018). Otherwise, we use tokenized BLEU (Papineni et al., 2002) with the open-source script [6].

4.3 Main Results

As is shown in Table 1 and 2, our methods obtain significant improvements across all language pairs. For x→en and en→x pairs, advances of nearly 6 BLEU and 3 BLEU are achieved. We also compare the results with loading from mBART, a well-known multilingual pre-trained sequence-to-sequence model (Liu et al., 2020) [7]. Due to the language difference, we only tune the model on a part of languages. With both 12-layers depth and 1024-dimensions width, our method outperforms mBART on almost all pairs, proving the superiority of Graformer comparing with pre-training in United Style mentioned in Section 2. It is worth noticing that we only use the one-fifth amount of the data of mBART.

4.4 Ablation Study

To verify the contribution of each part of our model, we do a series of ablation studies. As is shown in Table 3 and 4, we can draw at least four empirical conclusions.

Encoder needs tuning, decoder needs not. In Table 3, comparing Row 1 with Row 2, and Row 5 with Row 8, we can see that the tuning of the encoder is essential. It can bring further improvements. However, freezing pre-trained decoder parameters is a better choice. Comparing Row 3 with Row 4, and Row 6 with Row 8, we can see that

tuning may lead to a drop for decoder. It seems that the pre-trained decoder model learns much more knowledge, and its original language model can better guide the generation.

Decoder matters more. In Table 3, comparing Row 1,2,3,4, we can see that the pre-trained decoder yields more progress than the pre-trained encoder. This shows that involving only pre-trained encoders like BERT into MT is limited. The performance can be further enhanced with the introduction of pre-trained decoders.

Residual connection contributes. In Table 3, comparing Row 7 with Row 8, we can see that the residual connection from the pre-trained decoder can further boost the results. The well-trained language model effectively helps the translation model. It also shows the importance of incorporating the knowledge-rich generation model.

Row	Encoder	Decoder	x→en	en→x
0	-	-	22.8	19.0
1	Freeze	-	23.2	19.2
2	Fine-tune	-	27.0	20.2
3	-	Freeze	27.8	21.0
4	-	Fine-tune	25.2	19.9
5	Freeze	Freeze	25.8	20.4
6	Fine-tune	Fine-tune	27.0	19.4
7	Fine-tune	Freeze*	28.1	20.9
8	Fine-tune	Freeze	**28.6**	**21.9**

Table 3: Each number is the average BLEU of 30 language directions. "-" means not loading from pre-trained models. "*" means the residual connection is abandoned.

Encoder	Decoder	x→en	en→x
6+6	6+6	28.6	21.9
6+6	6+5	28.7	21.7
6+6	6+4	28.2	21.6
6+6	6+3	28.3	21.6
6+6	6+2	28.2	21.1
6+6	6+1	27.9	18.0
6+5	6+6	28.6	21.3
6+4	6+6	28.5	21.5
6+3	6+6	28.4	21.7
6+2	6+6	28.4	21.0
6+1	6+6	28.0	20.8
6	6+6	28.0	20.7

Table 4: Each number is the average BLEU of 30 language directions. "x+y" means the combination of x-layers pre-trained (masked) language models and y-layers grafting models.

---

[6] https://github.com/pytorch/fairseq/blob/master/examples/m2m_100/tok.sh

[7] https://dl.fbaipublicfiles.com/fairseq/models/mbart/mbart.cc25.v2.tar.gz

Layer number has slight effects. In Table 4, as the number of layers decreases, the performance drops slightly for both the encoder and decoder. But the extent of the decline is limited. Even no extra encoder layer or one-layer extra decoder can maintain a relatively high performance.

4.5 Well-trained Language Model Helps

Except for BLEU, we also study how the pre-trained generation model influence the translation model. We speculate that the pre-trained decoder helps to translate through combining the well-trained language model. Therefore, we collect and compare the perplexity of the models on the validation sets.

As is in Table 5, we can see that our method significantly lowers the perplexity comparing to the baseline model. The pre-trained decoder brings in better representation and language knowledge. Also, the residual connection from the original pre-trained decoder can further improve the results, illustrating the enlightening role the well-trained language model plays.

Model	x→en	en→x
Transformer	8.64	8.76
Graformer *	5.60	6.58
Graformer	**5.27**	**6.21**

Table 5: The perplexity of models. Each number is the average result of 30 language directions. "*" means the residual connection is abandoned.

4.6 Better than Fused Styles

Besides *United Style* (mBART), we also compare our method with *Fused Style*. Specifically, we choose two typical works, as are in Figure 4: *1)* loading parameters directly and ignoring cross-attention (denoted as "Direct") (Rothe et al., 2020; Ma et al., 2020); *2)* insert extra cross-attention layers into each decoder sub-layer and freeze pre-trained models (denoted as "Adapter") (Guo et al., 2020). We re-implement the models with the same depth and width as Graformer.

The crucial difference is that we leave the pre-trained decoder module unchanged and complete. Other works inject extra layers internally, such as cross-attention or adapters. Specifically, they go like $layer_1 \rightarrow adapter_1 \rightarrow layer_2 \rightarrow adapter_2 \rightarrow ... \rightarrow layer_N \rightarrow adapter_N$. The well-trained bond between $layer_i$ and $layer_{i+1}$ is
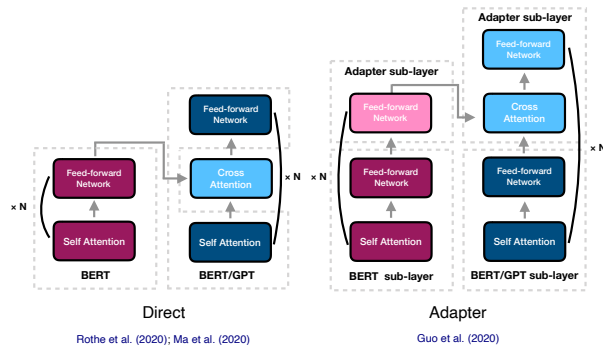

Figure 4: The model architecture of "Direct" (left) (Rothe et al., 2020; Ma et al., 2020) and "Adapter" (right) (Guo et al., 2020).

Model	BLEU↑		Perplexity↓	
	x→en	en→x	x→en	en→x
Direct	27.1	20.5	6.61	8.06
Adapter	27.4	19.8	5.78	6.71
Graformer	**28.6**	**21.9**	**5.27**	**6.21**

Table 6: Each number is the average BLEU/Perplexity of 30 language directions. Our model outperform related methods in fused style.

broken, which can not activate the full potential of the pre-trained decoder.

Differently, we maintain the original structure and even feed its output into the final layer. These strategies are all for the sake of fully taking advantage of the pre-trained generation model. As is in Table 6, our approach outperforms other two methods (The detailed results are in Appendix B).

4.7 Graformer Maintains Good Performance in Few-Shot Translation

We also conduct few-shot experiments. We randomly select 30%, 10%, 3%, 1% of the data and reproduce the experiments. As is in Figure 5,6, as the scale of datasets decreases, the performance of baseline drops dramatically and fails to generate comprehensible sentences (BLEU < 5). However, our method keeps relatively higher results even with only 1% data. And with the less data provided, the gap between Graformer and baseline is much larger (5.8→12.1, 2.9→7.1). Again, it proves that the usage of multiple monolingual data can benefit MNMT greatly since its scale is tens or hundreds of times of the parallel one.
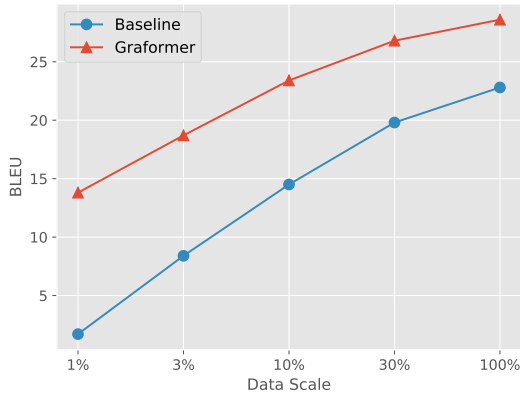
Figure 5: The results of x→en directions. As the data scale decrease from 100% to 1%, the gap is getting larger (5.8→12.1).

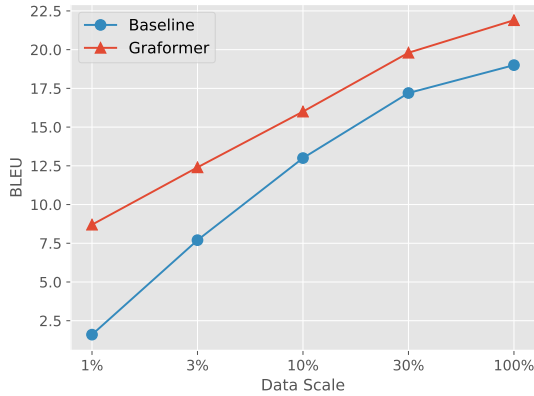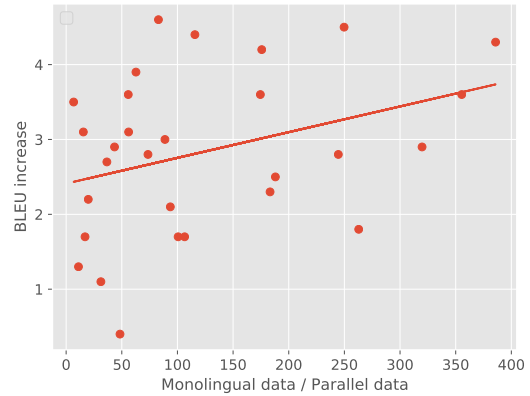Figure 7: Each point represents a language. The x-axis means the quotient of the monolingual data scale divided by the parallel data scale. The y-axis means the BLEU improvements of en→x directions.

Train	Model	Testing	
		de→en	fr→en
de→en	Transformer	31.9	6.7
	Graformer	**33.4**	15.2
	Graformer$_{fe}$	33.0	**20.1**
fr→en	Transformer	5.1	35.1
	Graformer	10.8	**36.0**
	Graformer$_{fe}$	**16.8**	35.5

Table 7: Zero-shot experiments on WMT Datasets. "fe" means freezing the pre-trained encoder. Notice that even the model does not see parallel sentences for a testing language, our method can achieve 11.7 and 13.4 BLEU improvement.

Train	Model	Testing	
		de→en	fr→en
de→en	Transformer	33.6	1.7
	Graformer	**36.9**	3.4
	Graformer$_{fe}$	35.4	**11.9**
fr→en	Transformer	1.5	37.3
	Graformer	4.5	**40.7**
	Graformer$_{fe}$	**10.7**	39.8

Table 8: Zero-shot experiments on TED Datasets. "fe''" means freezing the pre-trained encoder. Notice that even the model does not see parallel sentences for a testing language, our method can achieve 10.2 and 9.2 BLEU improvement.

Figure 6: The results of en→x directions. As the data scale decrease from 100% to 1%, the gap is getting larger (2.9→7.1).

4.8 The More Monolingual data, The Better

To further analyze the effectiveness of monolingual data, we look into the relationship between the BLEU advance and the data scale. As is in Figure 7, as the quotient of the monolingual data scale divided by the parallel data scale increases, the BLEU improvements gradually go up. It shows the extra benefit provided by the monolingual data, especially in the large-scale scene. Since the parallel data is rare, Graformer can be an essential approach to enhance low-resource language pairs.

4.9 Graformer Boosts Zero-Shot Translation

To verify whether the multilingual pre-trained model learns cross-lingual knowledge, we also conduct a crossed experiment of zero-shot translation. Firstly, we use our approach to train models only on German-English corpus and then conduct inference on French-English test sets. Converse ones are done similarly. We perform experiments on both TED and WMT datasets, with the encoder frozen (Graformer$_{fe}$) and tuned (Graformer).

As is in Table 7 and 8, we can draw similar conclusions. On the one hand, the performance of the original direction is improved, as expected. On

the other hand, the inference results in the other direction are also significantly boosted at the same time. It is worth noting that our models are trained with none of the test directions but obtain BLEU score high than 10.

More specifically, if the encoder is frozen, the results of the main direction can be slightly lowered, but the results of the zero-shot one will be significantly improved. It illustrates that the untuned pre-trained model contains much more cross-lingual knowledge and can be better transferred to untrained pairs.

4.10 Graformer Works in Bilingual Translation

To verify the effect of our methods, we also conduct experiments on bilingual translation. We use WMT14 English-German and English-French Datasets. In this series of settings, the datasets and vocabulary of both pre-training and tuning are limited in the bilingual corpus. For en-fr training, we adopt $dropout = 0.1$, following Vaswani et al. (2017).

The results, along with several strong related pre-training works, are listed in Table 9. Those related works all take advantage of pre-trained models and significantly improve the translation. Our method boosts the performance of bilingual translation and is at the top level. It proves the universal effectiveness of Graformer.

Model	en→de	en→fr
Transformer	28.9	41.8
Yang et al. (2020a)	30.1	42.3
Weng et al. (2020)	29.2	-
Yang et al. (2020b)	29.2	-
Zhu et al. (2020b)	30.8	**43.8**
Rothe et al. (2020)	30.6	-
Guo et al. (2020)	30.6	43.6
Graformer	**31.0**	43.6

Table 9: Bilingual translation results of English-German and English-French of WMT14. Comparing objects are strong results reported by recent works. Graformer boosts the performance and is at the top level.

5 Conclusion

In this paper, we propose Graformer, grafting multilingual BERT and multilingual GPT for multilingual neural machine translation. By pre-training the representation part (encoder) and generation part (decoder) of the model, we leverage the monolingual data to boost the translation task. And different from other previous fusing methods, we maintain the original architectures. With this approach, we can fully take advantage of the pre-trained models, including their well-trained capacity for representation and generation. Experimental results show that our method can significantly improve the performance and outperform similar related works. A series of empirical analyses of perplexity, few-shot translation, and zero-shot translation also shows its universality.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *NAACL-HLT*.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Graeme Blackwood, Miguel Ballesteros, and Todd Ward. 2018. Multilingual neural machine translation with task-specific attention. In *ICLR*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *NeurIPS*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *ACL-IJCNLP*.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016a. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *NAACL-HLT*.

Orhan Firat, Baskaran Sankaran, Yaser Al-Onaizan, Fatos T Yarman Vural, and Kyunghyun Cho. 2016b. Zero-resource translation with multi-lingual neural machine translation. In *EMNLP*.

Junliang Guo, Zhirui Zhang, Linli Xu, Hao-Ran Wei, Boxing Chen, and Enhong Chen. 2020. Incorporating bert into parallel sequence decoding with adapters. In *NeurIPS*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Melvin Johnson, Mike Schuster, Quoc Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In *ACL*.

Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pretraining multilingual neural machine translation by leveraging alignment information. In *EMNLP*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *TACL*.

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

Fuli Luo, Wei Wang, Jiahao Liu, Yijia Liu, Bin Bi, Songfang Huang, Fei Huang, and Luo Si. 2020. Veco: Variable encoder-decoder pre-training for cross-lingual understanding and generation. *arXiv preprint arXiv:2010.16046*.

Shuming Ma, Jian Yang, Haoyang Huang, Zewen Chi, Li Dong, Dongdong Zhang, Hany Hassan Awadalla, Alexandre Muzio, Akiko Eriguchi, Saksham Singhal, et al. 2020. Xlm-t: Scaling up multilingual machine translation with pretrained cross-lingual transformer encoders. *arXiv preprint arXiv:2012.15547*.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. 2018. Mixed precision training. In *ICLR*.

Xiao Pan, Mingxuan Wang, Liwei Wu, and Lei Li. 2021. Contrastive learning for many-to-many multilingual neural machine translation. In *ACL*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Emmanouil Antonios Platanios, Mrinmaya Sachan, Graham Neubig, and Tom Mitchell. 2018. Contextual parameter generation for universal neural machine translation. In *EMNLP*.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *NAACL-HLT*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *TACL*.

Alexander Sergeev and Mike Del Balso. 2018. Horovod: fast and easy distributed deep learning in tensorflow. *arXiv preprint arXiv:1802.05799*.

Aditya Siddhant, Ankur Bapna, Yuan Cao, Orhan Firat, Mia Xu Chen, Sneha Kudugunta, Naveen Arivazhagan, and Yonghui Wu. 2020. Leveraging monolingual data with self-supervision for multilingual neural machine translation. In *ACL*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2019. Mass: Masked sequence to sequence pre-training for language generation. In *ICML*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *CVPR*.

Xu Tan, Jiale Chen, Di He, Yingce Xia, QIN Tao, and Tie-Yan Liu. 2019a. Multilingual neural machine translation with language clustering. In *EMNLP-IJCNLP*.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019b. Multilingual neural machine translation with knowledge distillation. In *ICLR*.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *ICML*.

Xinyi Wang, Hieu Pham, Philip Arthur, and Graham Neubig. 2019a. Multilingual neural machine translation with soft decoupled encoding. In *ICLR*.

Yining Wang, Long Zhou, Jiajun Zhang, Feifei Zhai, Jingfang Xu, and Chengqing Zong. 2019b. A compact and language-sensitive multilingual translation method. In *ACL*.

Yiren Wang, ChengXiang Zhai, and Hany Hassan. 2020. Multi-task learning for multilingual neural machine translation. In *EMNLP*.

Rongxiang Weng, Heng Yu, Shujian Huang, Shanbo Cheng, and Weihua Luo. 2020. Acquiring knowledge from pre-trained model to neural machine translation. In *AAAI*.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *ICML*.

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*.

Jiacheng Yang, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Weinan Zhang, Yong Yu, and Lei Li. 2020a. Towards making the most of bert in neural machine translation. In *AAAI*.

Jian Yang, Shuming Ma, Dongdong Zhang, ShuangZhi Wu, Zhoujun Li, and Ming Zhou. 2020b. Alternating language modeling for cross-lingual pre-training. In *AAAI*.

Zhen Yang, Bojie Hu, Ambyera Han, Shen Huang, and Qi Ju. 2020c. Csp: Code-switching pre-training for neural machine translation. In *EMNLP*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *ACL*.

Changfeng Zhu, Heng Yu, Shanbo Cheng, and Weihua Luo. 2020a. Language-aware interlingua for multilingual neural machine translation. In *ACL*.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020b. Incorporating bert into neural machine translation. In *ICLR*.

Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. 2021. Serial or parallel? plug-able adapter for multilingual machine translation. *arXiv preprint arXiv:2104.08154*.

A Languages and Scales

The languages of datasets are listed in Table 10 and Table 11, for pre-training and translation training, respectively. We use significantly less data than mBART(Liu et al., 2020). According to its paper (and some naive summation), they use 208 billion tokens in 1.4T in total. We only use 42 billion tokens in 0.18T.

B Results of Fused Style Methods

The results of other Fused Style methods are in Table 12 and 13.

Language	Scale	Language	Scale
am	119643	ky	279440
bg	38305118	lt	4992036
bn	3916068	lv	13059185
bs	1955342	mk	209389
cs	90149511	ml	182467
de	329456604	mr	325364
el	8159512	nl	1205639
en	326422361	or	444212
es	65422557	pa	218067
et	7023190	pl	14480947
fa	1304611	ps	948310
fi	23127824	pt	9260529
fr	121133895	ro	21285406
gu	535156	ro*	20509504
hi	32491838	ru	94788355
hr	6718607	so	168710
hu	40181635	sr	3798788
it	39682711	sw	455488
iu	781877	ta	1251716
ja	19579066	te	882347
kk	1956205	tr	17494020
km	4410059	uk	1486906
kn	502499	zh	25401930
		all	1.40B

Table 10: Languages used for pre-training and their scales (in sentences). "ro*" means processed Romanian.

Language	Scale	Language	Scale
bg	174444	ja	204090
bn	4649	kk	3317
bs	5664	lt	41919
cs	103093	mk	25335
de	167888	mr	9840
el	134327	nl	183767
es	196026	pl	176169
et	10738	pt	51785
fa	150965	ro	180484
fi	24222	ru	208458
fr	192304	sr	136898
hi	18798	ta	6224
hr	122091	tr	182470
hu	147219	uk	108495
it	204503	zh	5534
		all	3.18M

Table 11: Language pairs (from & to English) used for translation training and their scales (in sentences).

Model	bg	bn	bs	cs	de	el	es	et	fa	fi
Direct	36.8	18.0	35.2	28.5	33.9	35.3	39.1	22.5	24.8	21.2
Adapter	38.0	18.1	36.8	29.2	34.3	36.2	40.1	23.3	23.7	21.9
Graformer	**38.5**	**18.1**	**36.5**	**29.4**	**35.5**	**37.4**	**40.7**	**24.0**	**26.9**	**23.0**

Model	fr	hi	hr	hu	it	ja	kk	lt	mk	mr
Direct	38.0	23.6	35.3	24.4	36.0	12.3	10.1	25.4	33.8	12.1
Adapter	38.7	24.1	36.2	24.9	36.5	11.7	10.1	25.9	34.7	11.2
Graformer	**39.2**	**25.1**	**36.7**	**26.1**	**37.2**	**13.7**	**10.5**	**27.2**	**35.7**	**13.0**

Model	nl	pl	pt	ro	ru	sr	ta	tr	uk	zh
Direct	33.2	23.6	40.1	33.6	23.9	33.9	8.7	23.3	27.8	18.5
Adapter	33.8	24.0	41.1	34.2	24.3	34.9	7.1	22.9	27.6	17.9
Graformer	**35.2**	**25.1**	**41.5**	**35.1**	**25.1**	**35.6**	**10.2**	**25.5**	**28.9**	**19.9**

Table 12: The results of x→en directions for "Direct" (Rothe et al., 2020; Ma et al., 2020) and "Adapter" (Guo et al., 2020).

Model	bg	bn	bs	cs	de	el	es	et	fa	fi
Direct	30.7	12.2	24.5	18.2	25.1	27.8	35.3	14.9	13.3	13.1
Adapater	31.0	10.5	24.3	18.5	25.4	26.9	35.3	15.3	9.6	13.4
Graformer	**33.0**	**14.1**	**26.3**	**20.2**	**27.8**	**29.8**	**37.5**	**16.1**	**14.2**	**14.4**

Model	fr	hi	hr	hu	it	ja	kk	lt	mk	mr
Direct	35.4	16.7	25.2	15.7	30.7	12.2	4.0	14.5	24.4	10.5
Adapater	35.8	15.3	24.8	15.7	30.2	9.2	3.8	14.5	24.5	9.0
Graformer	**37.8**	**18.1**	**26.8**	**17.2**	**32.5**	**12.8**	**3.8**	**15.9**	**25.7**	**10.6**

Model	nl	pl	pt	ro	ru	sr	ta	tr	uk	zh
Direct	28.1	14.4	34.3	26.9	17.5	20.2	15.6	12.3	18.7	21.8
Adapater	26.9	13.9	34.2	26.7	17.1	19.7	11.6	11.7	18.1	20.5
Graformer	**29.0**	**15.8**	**36.6**	**29.1**	**19.0**	**21.4**	**14.7**	13.3	**19.5**	**23.0**

Table 13: The results of en→x directions for "Direct" (Rothe et al., 2020; Ma et al., 2020) and "Adapter" (Guo et al., 2020).