Towards Better Word Alignment in Transformer

Kai Song, Xiaoqing Zhou, Heng Yu, Zhongqiang Huang, Yue Zhang, Weihua Luo, Xiangyu Duan, and Min Zhang

***Abstract*— While neural models based on the Transformer architecture achieve the State-of-the-Art translation performance, it is well known that the learned target-to-source attentions do not correlate well with word alignment. There is an increasing interest in inducing accurate word alignment in Transformer, due to its important role in practical applications such as dictionary-guided translation and interactive translation. In this article, we extend and improve the recent work on unsupervised learning of word alignment in Transformer on two dimensions: a) parameter initialization from a pre-trained cross-lingual language model to leverage large amounts of monolingual data for learning robust contextualized word representations, and b) regularization of the training objective to directly model characteristics of word alignments which results in favorable word alignments receiving more concentrated probabilities. Experiments on benchmark data sets of three language pairs show that the proposed methods can significantly reduce alignment error rate (AER) by at least 3.7 to 7.7 points on each language pair over two recent works on improving the Transformer's word alignment. Moreover, our methods can achieve better alignment results than GIZA++ on certain test sets.**

***Index Terms*—Neural network, neural machine translation, Transformer, word alignment, language model pre-training, alignment concentration.**

I. Introduction

DIFFERENT from traditional statistical machine translation methods (SMT) [1]–[3], which make explicit use of word alignment as part of the model training process, neural machine translation (NMT) [4]–[7] works by taking an end-to-end approach, using a neural network to encode a given source sentence and incrementally predicting its target translation by calculating a distribution over a target vocabulary at each step, where no word alignment is required during model training or decoding.

Manuscript received September 27, 2019; revised February 7, 2020 and April 14, 2020; accepted May 11, 2020. Date of publication May 28, 2020; date of current version June 22, 2020. This work was supported in part by the National Natural Science Foundation of China under Grants 61525205 and 61673289 and in part by the Alibaba Group through Alibaba Innovative Research Program. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Eric Fosler-Lussier. *(Corresponding author: Kai Song.)*

Kai Song is with the Soochow University, Suzhou 215000, China, and also with the Alibaba DAMO Academy, Hangzhou 310051, China (e-mail: songkai.neu@gmail.com).

Xiaoqing Zhou, Xiangyu Duan, and Min Zhang are with the Soochow University, Suzhou 215000, China (e-mail: xqzhou@stu.suda.edu.cn; xiangyuduan@suda.edu.cn; minzhang@suda.edu.cn).

Heng Yu, Zhongqiang Huang, and Weihua Luo are with the Alibaba DAMO Academy, Hangzhou 310051, China (e-mail: yuheng.yh@alibaba-inc.com; z.huang@alibaba-inc.com; weihua.luowh@alibaba-inc.com).

Yue Zhang is with the Westlake University, Hangzhou 310000, China (e-mail: yue.zhang@wias.org.cn).

Digital Object Identifier 10.1109/TASLP.2020.2998278

Intuitively, word alignment as a source of bilingual knowledge can be useful because of its practical significance. There is an increasing interest in inducing accurate word alignment in neural translation models. In addition, word alignment is helpful to improve the translation of terminology and low-frequency words. In dictionary-suggested translation [8]–[10], translation quality can benefit from leveraging external bilingual lexicons for terminology constraints [11]–[14]. When translating low-frequency words by methods based on placeholder tags [15], [16], word alignment is the key information for the replacement operations during both training and decoding procedures. Also, word alignment provides useful linguistic information on translation output. For example, producing reasonable word alignment can help to analyse translation errors [17], such as over-translation and under-translation [18]. Also, highlight the source aligned word of the current translated word can be helpful in interactive translation with the human in the loop [19]–[21].

NMT models have shown large advantages on a myriad of data sets. Among them, Transformer [7] based NMT attributes its superior performance to the multi-layer and multi-head self-attention architecture. In this paper we focus on inducing high quality word alignment during decoding in Transformer. Although its use of multi-layer and multi-head attention leads to the state-of-the-art performance in machine translation and other tasks, multiple studies have found that Transformer's target-to-source attention does not correlate well with word alignment [17], [22]. Several approaches have been proposed to make Transformer model produce better word alignment.

The first method learns word alignment from external alignment signals by supervising original attention head [23] or additional parameters [10], [17]. The second method learns word alignment from bilingual data only, without using alignment supervision. The word alignment is interpreted from the standard Transformer's attention head [22] or from dedicated neural models [24]. Notably, Zenkel *et al.* (2019) [24] introduced a separate alignment layer to the Transformer architecture to learn word alignment directly from bilingual text without extra supervision, resulting in alignment that is significantly more accurate than that obtained from target-to-source attentions in the decoder. Most of these works rely on taking intermediate representations produced by Transformer NMT, the learned representations are restricted to available bilingual training corpora, which are limited in practice for many language pairs. Moreover, its use of a simple word-generation objective is not sufficient to capture other desirable alignment characteristics, such as alignment concentration and source fertility [25], [26], which are utilized in traditional alignment models in SMT.

We build upon the work of Zenkel *et al.* (2019) [24] and propose to address the aforementioned limitations on two dimensions. First, we make use of monolingual data, which are more abundantly available than bilingual data, to better capture the similarity of words within and across languages. Inspired by recent work on pre-training contextualized word representations from monolingual data for downstream language understanding tasks [27]–[29], we initialize and fine-tune the parameters of the Transformer NMT and the alignment layer from a cross-lingual language model, which is pre-trained on monolingual text of source and target languages. Similarity of words within and across languages can be better captured in the pre-trained model [30], which is intuitively helpful to the word alignment problem. Second, we introduce soft constraints on attention weights produced by the alignment layer to encourage the characteristics of proper word alignment. In particular, we propose to regularize the training objective of the alignment layer so that the distribution of alignment probabilities can be more concentrated on the most relevant source words. As a result, the attention probabilities are more consistent with word alignments.

Experiments on benchmark data sets of three language pairs show that the proposed methods are effective. Alignment error rate (AER) is significantly reduced by at least 3.7 to 7.7 points on each language pair compared with two recent works on improving the Transformer's word alignment [22], [24]. Moreover, our methods achieve better alignment results than GIZA++ [31] on certain test sets.

II. Background

A. Transformer

Transformer-based NMT [7] uses a self-attention network for both encoder and decoder. The encoder is composed of J stacked neural layers. For time step i in layer j, the hidden state $\mathbf{h}_i^j$ is calculated by employing self-attention over the hidden states in layer $j-1$, which are $\{\mathbf{h}_1^{j-1}, \mathbf{h}_2^{j-1}, \ldots, \mathbf{h}_m^{j-1}\}$, where m is the number of source-side words. In particular, $\mathbf{h}_i^j$ is calculated as follows: First, a self-attention sub-layer is employed to encode the context. Then *attention weights* are computed as scaled dot products between the current query $\mathbf{h}_i^{j-1}$ and all the keys $\{\mathbf{h}_1^{j-1}, \mathbf{h}_2^{j-1}, \ldots, \mathbf{h}_m^{j-1}\}$, normalized with a softmax function. The context vector is then represented as *weighted sum* of the values projected from the hidden states in the previous layer, which are $\{\mathbf{h}_1^{j-1}, \mathbf{h}_2^{j-1}, \ldots, \mathbf{h}_m^{j-1}\}$. The hidden states in the previous layer and the context vector are then connected by residual connection, followed by a layer normalization function [32] to produce a candidate hidden state $\mathbf{h}'^j_i$, where $\mathbf{h}'$ represents a *candidate* hidden state. Finally, another sub-layer, a feed-forward network (FFN), is connected with $\mathbf{h}'^j_i$ through a residual connection, followed by a layer normalization function to obtain the hidden state $\mathbf{h}_i^j$.

The decoder is also composed of stacked layers, e.g., J. For time step t, layer j not only consists of a self-attention sub-layer and a FFN layer, but also a target-to-source attention sub-layer between them: First, a self-attention sub-layer is calculated by employing self-attention mechanism over the hidden states in the previous target layer, which are $\{\mathbf{s}_1^{j-1}, \mathbf{s}_2^{j-1}, \ldots, \mathbf{s}_{t-1}^{j-1}\}$, resulting in candidate hidden state $\mathbf{s}'^j_t$. Then, a target-to-source sub-layer is inserted after the first self-attention sub-layer. In particular, $\mathbf{s}'^j_t$ is taken as query (Q), and the keys (K) and values (V) are projected from the source hidden states in the last layer of the encoder. The attention weights $\{\alpha_{t,1}^j, \alpha_{t,2}^j, \ldots, \alpha_{t,m}^j\}$ are used to gain source context c_t^j, which is a weighted sum of source-side hidden states. Another candidate state $\mathbf{s}''^j_t$ is calculated by employing self-attention mechanism over the source context c_t^j and the candidate hidden state $\mathbf{s}'^j_t$, which is produced by the first sub-layer. Finally, a last feed-forward sub-layer is connected with $\mathbf{s}''^j_t$ through a residual connection, followed by a layer normalization function to obtain the hidden state $\mathbf{s}_t^j$.

The hidden state $\mathbf{s}_t^j$, which is produced by decoder's last layer, is then followed by a linear layer, which is a simple, fully connected neural network that projects the $\mathbf{s}_t^j$ into a vector which has the same size with target-side vocabulary. A softmax layer based on the decoder's last layer $\mathbf{s}_t^J$ is used to produce a probability distribution over the target-side vocabulary:

$$p(y_t|y_1, \ldots, y_{t-1}, \mathbf{x}) = \mathrm{softmax}(\mathbf{s}_t^J \cdot \mathbf{W}), \tag{1}$$

where $\mathbf{x}$ denotes the source sentence, $\{y_1, y_2, \ldots, y_t\}$ denote the target words, and $\mathbf{W}$ is the parameter of the linear operation, which is usually called "output embedding" and is usually tied to the target-side word embedding ("input embedding") in a number of Transformer implementations [33].

III. Baselines

A. Alignment Extraction in Transformer

A common way to extract target-to-source word alignment from a Transformer model is to choose the source word that has the maximum accumulated attention weight from the current target word [8], [34], [35], i.e.,

$$\gamma(t) = \underset{i \in \{1,\ldots,m\}}{\mathrm{argmax}} \sum_{j=1}^{J} \alpha_{t,i}^j, \tag{2}$$

where i is a candidate position of the aligned source word. For decoding step t in layer j, $\alpha_{t,i}^j$ is an average of all the target-to-source attention weights, which are produced by all the attention heads:

$$\alpha_{t,i}^j = \frac{1}{N} \sum_{n=1}^{N} \alpha_{t,i,n}^j, \tag{3}$$

where N denotes the number of attention heads and $\alpha_{t,i,n}^j$ denotes the target-to-source attention weight toward ith source-side position produced by nth attention head in layer j at decoding step t.

As also noted in Koehn *et al.* (2017), Li *et al.* (2019) and Ding *et al.* (2019)'s work [17], [22], [36], word alignment derived from a vanilla Transformer by the above method has a high error rate, as illustrated in Fig. 1.

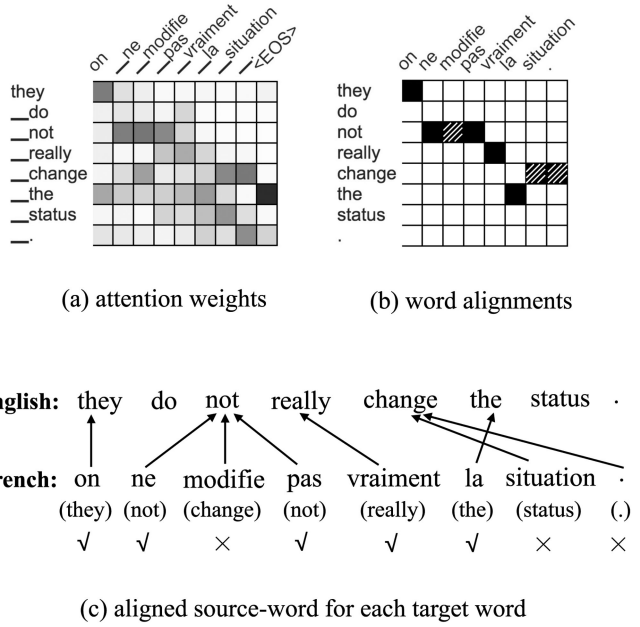


Fig. 1. (a) accumulated attention weight matrix from French sub-word tokens to English sub-word tokens, with darker gray-level for higher weights; (b) word alignments extracted from the accumulated attention weight matrix, with thatched grids indicating erroneous alignment; (c) extracted word alignments compared to reference. The English word in the bracket under a French word is its translation. "√" denotes a correct alignment link, "×" denotes an incorrect alignment link.

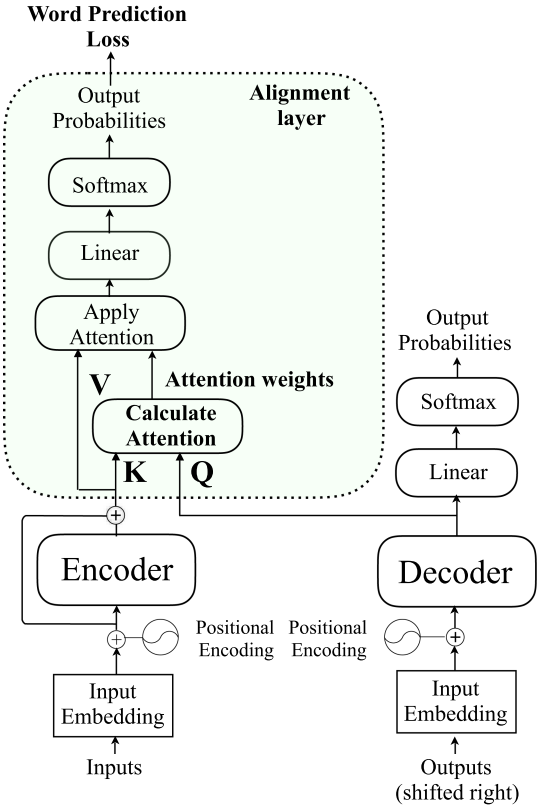


Fig. 2. Transformer with dedicated alignment layer.

B. Transformer With Dedicated Alignment Layer

Zenkel *et al.* (2019) [24] proposed to use a dedicated alignment layer for learning more accurate word alignment in Transformer. Their model architecture contains a standard Transformer for NMT and a separate alignment layer with inputs connected to contextualized word representations in the Transformer model. As shown in Fig. 2, the alignment layer also uses the target-to-source attention mechanism similar to that in the decoder. The hidden states in the final layer of the decoder are used as query vectors Q, while the key and value vectors K and V are taken from the average of the source input word embeddings and the hidden states on the final layer of the encoder:

$$V = \left\{ \frac{1}{2}(\mathbf{h}_1^J + \mathbf{x}_1), \ldots, \frac{1}{2}(\mathbf{h}_m^J + \mathbf{x}_m) \right\}, \tag{4}$$

and $K = V$. The output of the attention layer is then connected to a fully connected linear layer and a softmax layer to compute the probability of the target word, similar to that of the decoder. The linear layer functions as the output embeddings for the alignment layer.

There is a notable difference between the target-to-source attention in the alignment layer and the target-to-source attention in a standard Transformer model in the way they are trained. The standard Transformer model is trained on the parallel training data with the typical maximum likelihood translation objective. After that, the parameters of the Transformer NMT model are fixed and the alignment layer is trained on the same parallel training data with its own maximum likelihood translation objective:

$$\mathcal{L}_t^{word} = -\log(p(y_t|\mathbf{s}_t^J, \mathbf{x})) \tag{5}$$

where $\mathbf{s}_t^J$ is the hidden state on the final layer of the decoder for the t-th word in the target side. The inputs of the alignment layer, i.e., the Q, K, V vectors, are computed from the standard Transformer which is kept fixed. As such, the alignment layer learns the projection matrix for the Q, K, V vectors to capture the correlation between source and target words. During inference, the aligned source word of the current predicted target word y_t is obtained by choosing the source position with the maximum attention weight calculated in the alignment layer. The multi-head attention mechanism used in the alignment layer only has one single attention head.

Zenkel *et al.* (2019) [24] also provide a method that directly optimizes attention weights for each parallel sentence in the test set during the decoding process. For each test sentence, the attention weights are reset to the original attention weights as inferred from the trained model and then updated towards the current sentence: First, a forward pass is executed on the entire network to obtain the attention weights in the alignment layer for each word of the whole target sentence. Then the attention weights are used and optimized to maximize the likelihood of the target word using stochastic gradient descent (SGD).

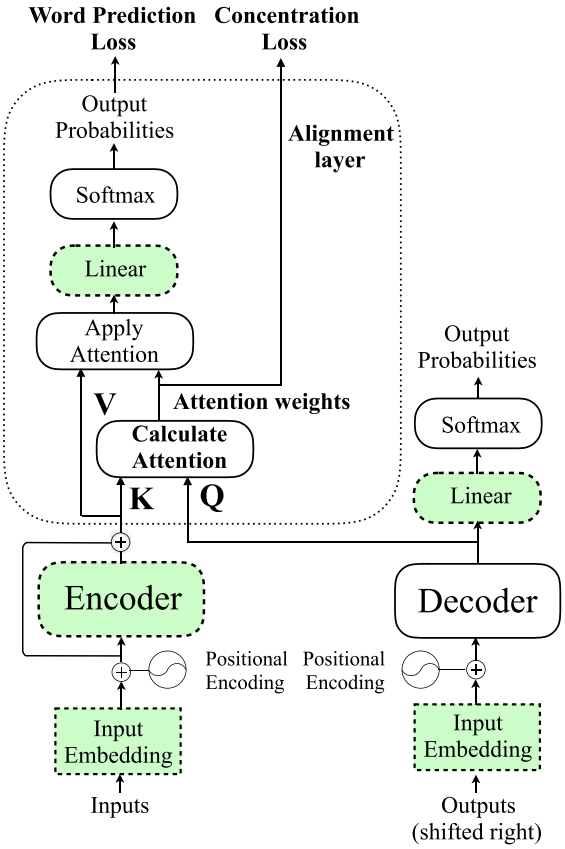


Fig. 3. Our model architecture. The parameters of the colored components are initialized by the pre-trained cross-lingual masked language model.

This method optimizes for accuracy, but is hardly applied in most of the real-world scenarios, because relying on back-propagation or performing multiple steps of SGD to find appropriate attention weights for each sentence is time-consuming. However, on some of the test sets, the optimization method can obtain alignment accuracy even as good as GIZA++, which also includes test set for updating model parameters. In this paper, we also affirm the effectiveness of our methods when using this optimization method during inference, which is discussed in Section V-D. Since the method is similar to the notion of dynamic evaluation proposed in Krause *et al.* [37], [38]'s work, we use *dynamic evaluation* to refer this optimization method in the rest of this paper.

IV. Method

We extend Zenkel *et al.* (2019) [24]'s method in two dimensions: language model pre-training to utilize large amounts of monolingual data and regularization of the alignment layer to encourage concentration of alignment probabilities. As shown in Fig. 3, we also use Transformer with a dedicated alignment layer described in Section III-B as the model architecture, but choose to initialize selected components of the model by a pre-trained cross-lingual masked language model. In addition, we introduce a concentration loss to regularize the original word prediction loss and encourage word alignment distributions with less ambiguity.

A. Pre-Training on Monolingual Data

Lample *et al.* (2019) [29] proposed a Transformer-based cross-lingual masked language model,[1] henceforth "XLM" in this paper, for learning contextualized representations from unpaired monolingual data of different languages. We choose XLM as the pre-trained language model for two reasons. First, it is based on the same standard Transformer architecture with multi-layer multi-head self-attentions that is also used in the translation part of our model. Second, XLM enables learning of contextualized word representations from large amounts of unpaired monolingual data of different languages that are able to capture word similarity across languages [30], which intuitively could be helpful for learning word alignment.

Following Lample *et al.* (2019) [29], BPE [39] codes are learned from the combination $\mathbb{C}$ of the source ($\mathbb{C}_s$) and target ($\mathbb{C}_t$) monolingual corpora with random sampling to ensure equal balance between the source and the target. A shared vocabulary is created for both the source and target languages and is used in the XLM model, the NMT model, and the alignment layer.

The training samples are generated from $\mathbb{C}$. Following Devlin *et al.* (2019) [27], given a sentence $\mathbf{c} \in \mathbb{C}$, a modified version $\hat{\mathbf{c}}$ is obtained by sampling 15% of the tokens from the input sentence and replacing them with a [MASK] token 80% of the time, with a random token 10% of the time, and kept unchanged 10% of the time. The model is trained to predict the masked tokens:

$$\mathcal{L}(\theta; \mathbb{C}) = \frac{1}{|\mathbb{C}|} \sum_{\mathbf{c} \in \mathbb{C}} \sum_{m \in \mathbb{M}} \log P(\mathbf{c}_m | \hat{\mathbf{c}}; \theta), \tag{6}$$

where $\mathbb{M}$ is the set of the masked positions in $\mathbf{c}$, c_m is the m-th token in $\mathbf{c}$ and θ is the parameters of the XLM model, which contains the parameters of Transformer's encoder, language embedding, positional embeddings and cross-lingual word embedding.

The entire encoder of the Transformer model is initialized from the pre-trained XLM model. The input and output embeddings of the decoder are also initialized from the corresponding parameters of the XLM model, with the remaining parameters of the decoder initialized randomly. In addition, we also initialize the output embeddings of the alignment layer, i.e., the linear projection matrix before softmax, with the pre-trained XLM output embeddings. The effect of initializing the linear projection matrix with the pre-trained XLM model is analyzed in the experiments section. Inspired by Press *et al.* (2016) [33] and Sennrich *et al.* (2017) [40], we tie the output embeddings in the alignment layer with the output embeddings in the original decoder of Transformer. With a combination of pre-trained and randomly initialized parameters, the NMT model is fine-tuned on the parallel training data using the standard maximum likelihood translation objective. Although the improvement of translation quality is not the focus of this paper, as to be shown in

[1] In their paper, this method is denoted as "XLM (MLM)".

the experiments section, fine-tuning from the pre-trained XLM model results in improved translation quality.

The alignment layer is trained after the NMT training. For a given sentence, the parameters of the Transformer NMT are fixed and used to obtain both contextualized word representations of the source sentence and the query vector of the alignment layer. The use of pre-training method allows the model to learn more robust representations from large amounts of monolingual data than from the limited parallel data. As discussed in Section III-B, for each proposed target word, its aligned source word can be immediately obtained by choosing the source position that has the maximum attention weight in the alignment layer. This is in contrast to the traditional word alignment methods, such as GIZA++ and FastAlign, in which the entire target sentence needs to be generated before performing word alignment as a downstream task.

B. Alignment Concentration

In the target-to-source attention layer of the Transformer model, the source positions with higher attention weights provide more influence on the generation of the target word. Because of the context-dependent nature of sentence translation, this typically results in a smooth distribution of attention weights over the source words [41], [42], as illustrated in Fig. 1(a). However, this can lead to poor word alignment [17], [22] as its objective is to find the most relevant source words responsible for generating the target word, since not all of the source words are helpful. A characteristic of good alignment is that the alignment distribution is *concentrated* on just one or a few source words [25], [26].

To this end, we introduce a concentration loss based on entropy to constrain the probability distribution of attention weights. The entropy of attention distribution represents the uncertainty in choosing the most relevant source word for the current target word. The concentration loss is defined as follows and is added as a part of the training objective function of the alignment layer.

$$\mathcal{L}_t^{concentration} = -\sum_{i=1}^{m}(\beta_{t,i}\log(\beta_{t,i})), \tag{7}$$

where $\beta_{t,i}$ is the attention weight of source position i in decoding step t, which is calculated by the attention mechanism in the alignment layer. By minimizing the concentration loss, we reduce the uncertainty of word alignment and encourage the alignment distribution to be more concentrated. During the training of the alignment layer, the objective function $\mathcal{L}$ consists of two components:

$$\mathcal{L} = \sum_{i=1}^{n}(\mathcal{L}_t^{word} + \lambda\mathcal{L}_t^{concentration}), \tag{8}$$

where $\mathcal{L}_t^{word}$ denotes the original word prediction loss in the alignment layer (Section III-B) and λ is the hyper-parameter that balances the preference between the normal word prediction loss and the alignment concentration loss.

V. Experiments

We conduct experiments based on fairseq [43], an open source implementation[2] of Transformer [7]. All methods are evaluated on the benchmark data sets of three language pairs: English-Romanian (En-Ro), English-German (En-De) and English-French (En-Fr). Alignment error rate (AER) [44] and BLEU [45] are used for measuring word alignment accuracy and translation quality, respectively.

A. Data

Bilingual Data: All the training data is obtained from and pre-processed with the same open-source tool[3] released in Zenkel *et al.* (2019)'s work [24]. Specifically, the training data for En-Ro, En-De and En-Fr has 0.4, 1.1 and 1.9 million parallel sentences, respectively. The training data for De-En is taken from Europarl v8 corpus. The training data for En-Ro and En-Fr are taken from word alignment shared task of HLT-NAACL 2003 workshop. For FastAlign and GIZA++, the train set and test set are merged and used for training, and the word alignment on the test portion is taken out for evaluation. We randomly set aside the 2,000 sentences of the training data for each language as the development set.

Monolingual Data: All the monolingual data is taken from the NewsCommonCrawl corpus released for WMT tasks. Specially, the monolingual data for English, German and French comes from WMT2014 to WMT2018 tasks, with 44.5, 86.8, and 66.6 million sentences for each language, respectively. The monolingual data for Romania is taken from WMT2015 with 2.7 million sentences. The tokens numbers for English, German, French and Romania are 1.02 billion, 1.54 billion, 1.60 billion and 64.9 million, respectively.

Test Data: We evaluate word alignment accuracy on publicly available hand-aligned benchmark test sets for En-Ro, En-Fr,[4] and En-De,[5] with 248, 447 and 508 sentences respectively for each language pair. Each language pair contains two directions of word alignment tasks, we use En→De as the development set to choose the value of the hyper-parameter λ when the alignment concentration loss is used. The test sets for evaluating translation quality are from the WMT news translation task, which are "newstest2016," "newstest2014" and "newstest2014" for En-Ro, En-De and En-Fr respectively.

B. Experimental Settings

We use BPE [39] in all experiment settings. For each language pair, the BPE codes are learned from the combination of the source and target sentences of the parallel training data, and a shared vocabulary is used for both the source and target. Both the merge operation and the vocabulary size are set to 20 K for each language pair.

We use the following configuration for all experiment settings. Both the encoder and decoder use 6 layers of attentions with 8

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

[3][Online]. Available: https://github.com/lilt/alignment-scripts

[4][Online]. Available: https://www-i6.informatik.rwth-aachen.de/goldAlignment/

[5][Online]. Available: http://web.eecs.umich.edu/~mihalcea/wpt/index.html

attention heads each. The size of embeddings and hidden states is set to 512. The feed-forward layer has 2,048 cells and ReLU [46] is used as the activation function. Adam [47] is used for training, with warmup steps set to 4,000 and learning rate set to 0.0005. We use label smoothing [48] with a confidence score of 0.9 and the drop-out [49] probabilities are set to 0.3.

In order to use pre-trained XLM models for parameter initialization, we use the same number of hidden states, attention layers, and attention heads in XLM models as in the Transformer NMT models. The XLM models are trained using the toolkit released by Lample *et al.* (2019) [29] with default settings, except for the aforementioned configuration.

C. System Configurations

We compare our method with both the Transformer-based word alignment methods and the traditional methods. For fair comparison with traditional methods, following Zenkel *et al.* (2019) [24] and Ding *et al.* (2019) [22], we conduct our experiments under two kinds of settings: without using dynamic evaluation and using dynamic evaluation.

Without using the dynamic evaluation method described in Section III-B, which means the test sets are not used for updating model parameters during the evaluation process. Under this condition, we compare three Transformer-based word alignment methods. One is the most common but naive way, the other two are from Zenkel *et al.* (2019) [24] and Ding *et al.* (2019) [22], which are two recent works on improving Transformer's word alignment. Zenkel *et al.* (2019) [24] achieves the best alignment results among recent works on improving Transformer's word alignment. In this condition, we also compare with FastAlign-online where the word alignments are obtained by using the inference mode in the FastAlign toolkit [50].

Under the condition of *using* the dynamic evaluation method described in Section III-B, which means the test sets are leveraged for updating model parameters during the evaluation process, we compare our methods with Zenkel *et al.* (2019) [24], GIZA++ [31] and FastAlign-offline. Under this condition, both Zenkel *et al.* (2019) [24] and our methods optimize attention weights towards each test sentence during inference.

Baseline 1: Extract Alignment from Vanilla Transformer: The most common way of extracting word alignment from vanilla Transformer, which is described in Section III-A, is used as a naive baseline.

Baseline 2: Extract Alignment from Dedicated Alignment Layer: Zenkel *et al.* (2019) [24] obtain the state-of-the-art alignment accuracy on benchmark data sets. We re-implement their work as our second baseline. In addition, they also proposed a method (we use "dynamic evaluation" to denote it in this paper) that optimizes the attention weights of the alignment layer towards each test sentence during inference, which is described in Section III-B. We also re-implement this method in our experiments to compare with GIZA++ and FastAlign-offline.

Since the reference word alignment is based on words, not BPE tokens. To map sub-word alignment to word alignment, we consider that a source word is aligned to a target word if any of their sub-word units are aligned. This method is used in Zenkel *et al.* (2019) [24]'s work, we use it in our work for a fair comparison. We will study other alternative methods that can map sub-word alignment to word alignment in our future work.

Traditional Methods: We also compare our methods with traditional methods, which are GIZA++,[6] FastAlign-offline and FastAlign-online.[7]

GIZA++, in its off-the-shelf form, cannot produce word alignment on a standalone test set. Although it is possible to align unseen test data using Giza++, but the documentation is hard to find and it is unpleasant to use.[8] A common way of obtaining the word alignment of unseen test data is to include the test sets in the training data first, then the alignments are produced during training after which the test portion is taken out for evaluation.

FastAlign provides both offline and online modes: For FastAlign-offline, the alignment results are obtained in the same way as with GIZA++. For FastAlign-online, the parameters are trained only on the train set. The alignment results of test sets are inferred by using the trained model parameters. Different from GIZA++ and FastAlign-offline, test set is not used for training.

Our System: As described in Section IV, we extend the method of Zenkel *et al.* (2019) [24] with two improvements: leveraging monolingual data through a pre-trained XLM model and modeling alignment concentration in the alignment layer. For each language pair, one XLM model is used for both translation directions. For example, the same XLM model is used for En→Ro and Ro→En. The value of the hyper-parameter λ in Equation 8 is empirically set to 15 according to the experiment on development set. In addition, when comparing with GIZA++, for fair comparison, we make use of the dynamic evaluation method introduced in section III-B, which optimizes attention weights for each test sentence during decoding and is used in Zenkel *et al.* (2019) [24].

D. Results

Not using dynamic evaluation: Under the condition of not using the dynamic evaluation method described in Section III-B, meaning that the model parameters will not be updated according to each test sentence during the evaluation process, Table I shows the comparison of our methods with different systems. Compared with the naive baseline ("*Baseline1*"), our methods reduce the averaged AER scores by 8.1, 24.4 and 22.1 points on En-Ro, En-De and En-Fr, respectively. Compared with Zenkel *et al.* (2019) [24] ("*Baseline2*"), the reductions of averaged AER scores are 3.7, 4.1 and 7.7 points. Our re-implementation of Zenkel *et al.* (2019) [24] is verified to produce comparable alignment accuracy as reported in the original paper. In addition, our methods outperform Ding *et al.* (2019) [22] on the same data sets by an average of 7.5, 11.3 and 8.4 AER points. Our methods can outperform FastAlign-online on En-Ro and

[6][Online]. Available: https://github.com/moses-smt/giza-pp

[7][Online]. Available: https://github.com/clab/fast_align

[8][Online]. Available: https://github.com/moses-smt/mgiza/blob/master/mgizapp/scripts/force-align-moses.sh

TABLE I
AER [%] SCORES OF DIFFERENT METHODS UNDER THE CONDITION OF NOT USING DYNAMIC EVALUATION, THE LOWER THE BETTER. AER OF DING *et al.* (2019) [22] AND ZENKEL *et al.* (2019) [24] ARE QUOTED FROM THEIR PAPER. OTHER SYSTEMS ARE DESCRIBED IN SECTION V-C. "AVG" IS THE AVERAGED AER SCORES OF BOTH LANGUAGE DIRECTIONS FOR EACH LANGUAGE PAIR. "BIDIR" IS THE AER SCORE OF THE SYMMETRIZED ALIGNMENT RESULTS, WHICH ARE OBTAINED BY COMBINING BOTH UNI-DIRECTIONAL ALIGNMENT RESULTS OF EACH LANGUAGE PAIR USING THE GROW-DIAG-FINAL HEURISTIC POSTPROCESSING STEPS DESCRIBED IN OCH AND NEY (2003) [26]. "EN→RO" DENOTES THE WORD ALIGNMENTS OBTAINED IN THE ENGLISH TO ROMANIA TRANSLATION TASK. † DENOTES THE DEVELOPMENT SET WHICH IS USED FOR CHOOSING λ EMPIRICALLY

Systems	En→Ro	Ro→En	Avg	Bidir	En→De†	De→En	Avg	Bidir	En→Fr	Fr→En	Avg	Bidir
FastAlign-online	41.1	37.1	39.1	35.5	34.4	30.8	32.6	30.6	**18.9**	**16.8**	**17.9**	16.3
Ding et al. (2019) [22]	41.4	41.2	41.3	32.7	43.0	36.4	39.7	29.0	29.7	25.9	27.8	15.3
Zenkel et al. (2019) [24]	38.7	37.7	38.2	32.8	34.7	31.5	33.1	27.1	26.0	26.1	26.1	15.2
Vanilla Transformer (*Baseline1*)	44.5	39.3	41.9	33.6	60.7	44.8	52.8	47.7	41.2	41.8	41.5	27.6
+Alignment Layer (*Baseline2*)	38.6	36.4	37.5	31.7	33.4	31.5	32.5	25.6	30.0	24.1	27.1	14.5
+Pre-training	33.2	35.4	34.3	30.7	30.1	28.4	29.3	24.2	21.6	20.0	20.8	12.4
+Concentration	**32.6**	**34.9**	**33.8**	**30.0**	**28.8**	**28.0**	**28.4**	**24.0**	19.6	19.1	19.4	**12.0**

TABLE II
AER [%] SCORES OF DIFFERENT METHODS UNDER THE CONDITION OF *USING* DYNAMIC EVALUATION. GIZA++ AND FASTALIGN-OFFLINE RESULTS ARE QUOTED FROM ZENKEL *et al.* [24]. † DENOTES THE DEVELOPMENT SET WHICH IS USED FOR CHOOSING λ EMPIRICALLY

Systems	En→Ro	Ro→En	Avg	Bidir	En→De†	De→En	Avg	Bidir	En→Fr	Fr→En	Avg	Bidir
FastAlign-offline	35.5	33.8	34.7	32.1	32.0	28.4	30.2	27.0	16.4	15.9	16.2	10.5
GIZA++	32.2	**28.7**	30.5	27.9	**23.1**	**21.0**	**22.1**	21.4	**8.0**	**9.8**	**8.9**	**5.9**
Zenkel et al. (2019) [24]	34.8	32.3	33.6	27.6	30.4	26.6	28.5	21.2	23.8	20.5	22.2	10.0
Alignment Layer (*Baseline2*)	35.9	33.2	34.6	28.2	29.5	31.8	30.7	22.6	27.3	20.5	23.9	11.0
+Pre-training	29.9	30.8	30.4	26.7	25.0	24.2	24.6	20.1	17.7	22.4	20.1	11.2
+Concentration	**29.6**	30.7	**30.2**	**26.6**	24.8	24.0	24.4	**19.7**	17.6	22.0	19.8	10.5

En-De, the reductions of averaged AER scores are 5.3 and 4.2 points respectively. Surprisingly, even without using test sets for training, on two of the three language pairs, our methods can outperform "FastAlign-offline," which includes the test set during training (See Table II).

The comparison between "*Baseline2*" and "Pre-training" shows that the pre-trained XLM models produce an average reduction of AER on three language pairs of 3.2, 3.2 and 6.3 points, respectively. By adding the training regularization, the averaged AER is further reduced by 0.5, 0.9 and 1.4 points on En-Ro, En-De and En-Fr, respectively.

In addition to the significant improvements on the averaged AER scores of each language pair, we observe that the AER score on the symmetrized alignments (See the "Bidir" column in Table I), which are obtained by combining both uni-directional alignment results of each language pair using the grow-diag-final heuristic rule described in Och and Ney (2003) [26], also improves, although to a less degree. However, in real-time applications, only uni-directional alignments are feasible and our results indicate that alignments in one direction are adequate.

Using dynamic evaluation: Table II shows the comparison of AER scores under the condition of using the dynamic evaluation method described in Section III-B, meaning that the model parameters will be updated according to each test sentence during the evaluation process. Both "*Baseline2*" and our methods optimize attention weights toward each test sentence during inference under this setting.

Compared with our re-implementation ("*Baseline2*"), our methods have better results (both averaged AER scores and AER scores of symmetrized word alignments) on all of the three language pairs. Compared with the results reported in Zenkel *et al.* (2019) [24], our methods have better symmetrized alignment results on En-Ro and En-De,[9] where the reductions are 1.0 and 1.5, respectively. Moreover, our methods have better averaged AER scores on all of the three language pairs, where the reductions are 3.4, 4.1 and 2.4 points, respectively.

By adding the pre-trained XLM models (see "+ Pre-training") and the training regularization (see "+ Concentration") to "*Baseline2*" progressively, the AER scores reduce constantly, showing that our methods can stably improve alignment quality, which is consistent with the results in Table I.

Our methods achieve better averaged AER scores than GIZA++ on En-Ro, while on the other two language pairs, GIZA++ is better. Our methods outperform FastAlign-offline on two of the three language pairs, which are En-Ro and En-De. Regarding the AER scores of symmetrized alignments, our methods outperform GIZA++ on En-Ro and En-De, and obtain better or equal results compared with FastAlign-offline on all the three language pairs.

For En-Fr in Table I and En-De in Table II, the best uni-directional results are from traditional methods, however the best symmetrized results are from the Transformer-based methods. Zenkel *et al.* (2019) also found the neural approaches profit more from symmetrizing both directions compared to the statistical approaches, which is similar to our findings. According to their speculation, the neural alignment models always use all the source words but only the generated target words, which might be a contributing factor to the strong reduction in error rates by combining two uni-directional results. This may also have some

[9]Our re-implementation of Zenkel *et al.* (2019) [24] under the setting of *not using* dynamic evaluation is verified to produce comparable or better results compared with the results reported in their paper (See Table I). However, under the setting of *using* dynamic evaluation, our results are not comparable to their results due to different training hyper-parameters used in their experiments and in ours, such as learning rate, warmup steps, etc.

TABLE III
THE ABLATION STUDY, "THIS WORK" REFERS TO THE SAME SYSTEM SHOWN IN THE LAST LINE OF TABLE I. "W/O INITIALIZATION" REFERS THAT THE LINEAR PROJECTION MATRIX IN THE ALIGNMENT LAYER IS NOT INITIALIZED WITH THE OUTPUT EMBEDDING MATRIX IN THE PRE-TRAINED XLM MODEL

Systems	En→Ro	Ro→En	Avg	Bidir	En→De	De→En	Avg	Bidir	En→Fr	Fr→En	Avg	Bidir
This work	32.6	34.9	33.8	30.0	28.8	28.0	28.4	24.0	19.6	19.1	19.4	12.0
w/o initialization	35.4	34.6	35.0	31.0	33.2	31.0	32.1	25.3	20.0	19.9	20.0	12.4

TABLE IV
PRECISION, RECALL AND F-MEASURE SCORES [%] OF THE TWO SYSTEMS USED IN TABLE III

Systems	En→Ro	Ro→En	En→De†	De→En	En→Fr	Fr→En
(This work)						
AER	32.6	34.9	28.8	28.0	19.6	19.1
Precision	68.5	66.8	69.3	70.1	76.2	76.9
Recall	66.5	63.4	73.2	74.9	86.1	87.2
F-measure	67.5	65.1	72.1	72.4	80.8	81.7
(w/o initialization)						
AER	35.4	34.6	33.2	32.1	20.0	19.9
Precision	65.6	67.0	65.0	65.7	76.4	78.1
Recall	63.6	63.8	68.8	70.3	87.1	83.5
F-measure	64.6	65.4	66.8	67.9	81.4	80.7

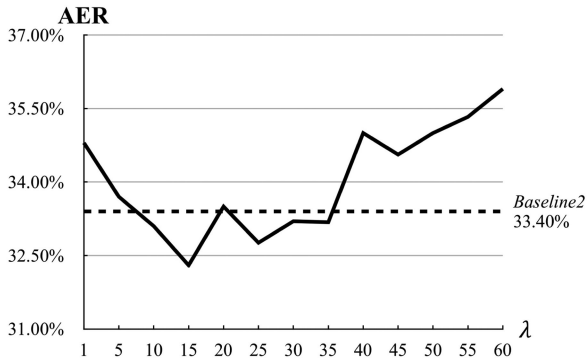


Fig. 4. AER on En→De task (development set) when introducing alignment concentration as the training regularization into "*Baseline2*". λ is a hyper-parameter, which is described in Equation 8. Horizontal axis denotes different values of λ and vertical axis denotes AER scores.

correlations with the method we used to obtain word alignment from sub-word alignment. We will study the effect of different mapping methods in our future work.

Effect of initializing the linear projection matrix in the alignment layer: As described in Section IV-A, the output embeddings of the alignment layer are tied to the output embeddings of the original decoder, both of them are initialized with the output embeddings of the pre-trained XLM model. We did an ablation study to particularly analyze the effect of initializing the linear projection matrix of the alignment layer, which is shown in Table III. Without initializing the output embeddings of the alignment layer with the pre-trained model, the AER scores increase with varying degrees in different language pairs, which means worse alignment qualities.

Precision, recall and F-measure: Although AER is a good measure, but looking at the precision and recall of the alignments is also important, which are stated in Table IV. By comparing different measures of the two systems used in Table III, we can find that AER scores are generally consistent with precision or F-measure scores. Lower AER scores appear together with higher precision or F-measure scores, indicating better word alignment qualities. The only exception is En→Fr, however, the gap of AER score between the two systems are small, as well as the gap of precision score, we can consider the results are within a reasonable fluctuating range.

Effect of taking different values for λ: As described in Section IV-B, we use a hyper-parameter λ to balance the preference of normal word prediction loss and alignment concentration loss. En→De is used as development set to choose λ which will be used for all language directions. Fig. 4 gives a comparison of AER scores when λ takes different values. In particular, we introduce the alignment concentration loss into "*Baseline2*," taking different values for λ during the training of the alignment layer, as a result, the trained models achieve different alignment accuracy. The minimum AER score is achieved when λ is set to 15 and this setting is used for all language directions.

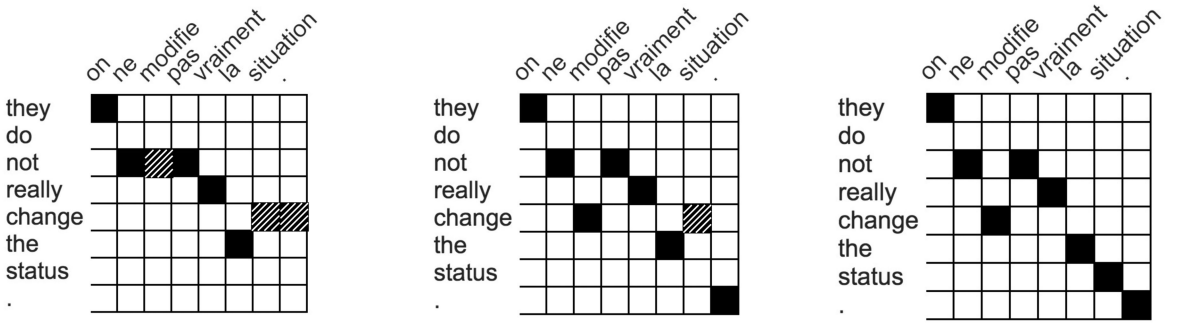
Decoding Speed and Translation Quality: As described in Section 2, both Zenkel *et al.* (2019) [24] and our methods use an alignment layer to derive word alignment as part of the decoding process. The amount of computation in the alignment layer during decoding is much smaller than training. During the training process of the alignment layer, both the attention weights and the word prediction loss need to be calculated. During decoding, only the attention weights need to be calculated, which is used to obtain the aligned source word of current target-side word. The word prediction loss in the alignment layer does *not* need to be calculated during decoding, which takes up most of the computation in the alignment layer during training.

TABLE V
DECODING SPEED (TOKENS/SEC.) TESTED ON TRANSLATION TEST SETS OF THREE LANGUAGE PAIRS. TRAINING TIME IS NOT INCLUDED IN THIS TABLE

Systems	newtest2016		newtest2014		newtest2014	
	EnRo	RoEn	EnDe	DeEn	EnFr	FrEn
Baseline1	91.64	85.72	92.36	91.68	98.15	88.28
This work	81.58	77.09	85.05	76.98	85.36	79.16

Table V shows a comparison of decoding speed, which is tested on the test sets of WMT translation tasks on three language pairs. The difference between "*Baseline1*" and our method comes from the forward computation to get the attention weights in the alignment layer. On Tesla M40 GPU card, with batch size set to 1 and beam size set to 4, the averaged decoding speed of our algorithm is 80.87 tokens/sec, as opposed to 91.31 tokens/sec in the vanilla Transformer. The relative reduction of decoding speed is 12.9%.

As described in Section IV-A, the alignment layer is trained after the NMT training is finished. Since the parameters of the Transformer are fixed during the training of the alignment layer, the translation quality is exactly the same before and after the training of the alignment layer. Table VI shows the comparison of translation quality. Although this work focuses on improving



(a) Baseline 1: Vanilla Transformer (b) Baseline 2: Alignment layer (c) Using pre-trained XLM model

Fig. 5. Word alignments derived from different En→Fr systems. Thatched grids denote alignment errors.

TABLE VI
COMPARISON OF BLEU SCORES ON WMT TEST SETS

Systems	newstest2016		newstest2014		newstest2014	
	EnRo	RoEn	EnDe	DeEn	EnFr	FrEn
Baseline1	17.82	25.65	21.40	25.29	21.28	23.13
This work	20.52	28.72	21.74	25.59	22.41	26.82

the performance of word alignment, our experiments show that the benefits from XLM also extend to translation quality.

Case Studies: Fig. 5 shows a comparison of word alignments derived from different En→Fr systems. "*Baseline1*" incorrectly aligns several target words. "*Baseline2*" can correct many of the alignment errors. However, the French word "situation" is still not correctly aligned to the English word "status" because "status" is not a frequent translation of "situation" in the bilingual training data (the bilingual corpus only contains 316 co-occurring instances of these two words). In contrast, "status" and "situation" appeared 226,247 and 435,609 times in the monolingual text of English and French, providing much more opportunities for learning their relationship [30]. Our method leverages monolingual data of both the source and target languages by pre-training a XLM model and use it for the initialization, as a result, "situation" is correctly aligned to the source word "status".

VI. RELATED WORK

Traditional alignment methods, such as GIZA++ [25], [31] and FastAlign [50], can give accurate word alignments. Unfortunately, they cannot produce immediate word alignment along with NMT's decoding procedure, word alignment is produced only when all the words in the target sentence are present, making them difficult to use synchronously with Transformer's decoding, which is important to downstream tasks (e.g., the dictionary-guided translation scenarios). Also, GIZA++ (as a over-the-shelf program) is hard to be utilized just as an aligner.

In recurrent neural network (RNN) based NMT [4], [5], attention corresponds to word alignment well. Some previous work has shown that translation quality benefits from supervising its attention with external word alignment [51], [52].

In Transformer, recent work try to improve alignment in a variety of ways. On leveraging external alignment as supervision for providing source aligned word during training, Alkhouli *et al.* (2018) [10] add a special attention head whose source context is computed only over the aligned source words. Li *et al.* (2019) [17] use additional parameters to predict the aligned source word during training, which is used to infer during decoding. Our work does not rely on external word alignment learned from bilingual data, which are limited in quantity and quality. In fact our method can make use of large amounts of monolingual data, which can provide much richer knowledge.

Ding *et al.* (2019) [22] propose methods to interpret word alignment from Transformer without using additional parameters and external supervisions. The method works well in convolution neural network (CNN) based NMT [6], but the performance is lacked in the Transformer because of the aforementioned gap between the attention mechanism and word alignment. Zenkel *et al.* (2019) [24] propose to add a separate alignment layer to the Transformer architecture and learn to focus its attention weights on relevant source words for a given target word in an unsupervised way. We augment the original word prediction loss of alignment layer with an additional alignment concentration loss, which models the natural characteristics of word alignment and is consistent with the features used in traditional methods.

As is in previous works, our work takes representations from Transformer's encoder and decoder for modeling alignment in Transformer. What's different in our work is that monolingual data is first used to provide better representations for learning word alignment in the Transformer. In addition, none of the above methods make use of the natural characteristics of word alignment [25], which is specially considered in our method.

Pre-training methods are widely used in language understanding tasks, which can transfer knowledge from rich-resource pre-training task to the low-resource downstream tasks. ELMo [53] generalizes contextual word representations by combining hidden states and word embeddings produced by a Bi-LSMT [5] language modeling task, OpenAI GPT [28] pre-train Transformer based language model instead of Bi-LSMT. To learn better bi-directional representations, BERT [27] uses Transformer encoder to train masked language model (MLM) task, XLM [29] extents BERT to enable training on cross-lingual corpora consists of monolingual data of different languages. MASS [54] predict the masked tokens in the decoder side

through a standard Transformer encoder-decoder architecture, which injects the pre-trained model with parameters of both the encoder and decoder.

These pre-training models are commonly used as language representations in NMT training on bilingual data. In our work, we make use of the representations learned from large amount of monolingual data for enhancing the correlation between source and target words, which is specially modeled in the alignment layer.

VII. Conclusion

We proposed methods to improve word alignment in Transformer on two dimensions: a) parameter initialization from a pre-trained cross-lingual language model, to leverage large amounts of monolingual data for learning robust contextualized word representations, and b) regularization of the training objective to favor more concentrated word alignment, as a way to incorporate prior knowledge of desirable alignment. On benchmark data sets of three language pairs, our methods consistently achieve lower alignment error rate compared to previous works on improving the Transformer's word alignment. Surprisingly, even without optimizing attention weights towards test sets, our methods outperform FastAlign in some cases, which includes test set during training. Besides, when using test sets for parameter updating, which is the same setting as GIZA++, our methods can achieve better results than GIZA++ on certain test sets. In future work, we will continue to investigate ways along these two dimensions to make better use of monolingual data and prior knowledge in neural word alignment. We will also study the effect of different methods which can map sub-word alignment to word alignment.

Acknowledgment

The authors would like to thank the anonymous reviewers for their detailed and constructed comments. Min Zhang is the corresponding author. Thanks to Niyu Ge for the guidance in paper writing.

References

[1] R. Zens, F. J. Och, and H. Ney, "Phrase-based statistical machine translation," *Lecture Notes Comput. Sci.*, vol. 11, no. 2, pp. 18–32, 2002.

[2] P. Koehn, F. Och, and D. Marcu, "Statistical phrase-based translation," in *Proc. Human Lang. Technol. Conf0 North Amer. Chapter Assoc. Comput. Linguistics*, Edmonton, Canada, May 2003, pp. 127–133.

[3] D. Chiang, "Hierarchical phrase-based translation," *Comput. Linguistics*, vol. 33, no. 2, pp. 201–228, 2007.

[4] K. Cho *et al.*, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[5] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. 3rd Int. Conf. Learn. Representations*, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2015.

[6] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. 34th Int. Conf. Mach. Learn.-Volume 70*, 2017, pp. 1243–1252.

[7] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Advances Neural Inf. Process. Syst. 30: Annu. Conf. Neural Inf. Process. Syst.*, 4-9 Dec. 2017, Long Beach, CA, USA, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds., 2017, pp. 5998–6008.

[8] P. Arthur, G. Neubig, and S. Nakamura, "Incorporating discrete translation lexicons into neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1557–1567.

[9] R. Chatterjee, M. Negri, M. Turchi, M. Federico, L. Specia, and F. Blain, "Guiding neural machine translation decoding with external knowledge," in *Proc. 2nd Conf. Mach. Transl.*, 2017, pp. 157–168.

[10] T. Alkhouli, G. Bretschner, and H. Ney, "On the alignment problem in multi-head attention-based neural machine translation," in *Proc. 3rd Conf. Mach. Translation: Res. Papers*, 2018, pp. 177–185.

[11] C. Hokamp and Q. Liu, "Lexically constrained decoding for sequence generation using grid beam search," in *Proc. 55th Annual Meeting of the Association for Computational Linguistics*, vol. 1: Long Papers, R. Barzilay and M. Kan, Eds., Vancouver, Canada, Jul. 30–Aug. 4, 2017, pp. 1535–1546.

[12] M. Post and D. Vilar, "Fast lexically constrained decoding with dynamic beam allocation for neural machine translation," in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, Jun. 2018, vol. 1 (Long Papers), pp. 1314–1324.

[13] K. Song, Y. Zhang, H. Yu, W. Luo, K. Wang, and M. Zhang, "Code-switching for enhancing nmt with pre-specified translation," in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Language Technologies*, 2019, vol. 1 (Long and Short Papers), pp. 449–459.

[14] G. Dinu, P. Mathur, M. Federico, and Y. Al-Onaizan, "Training neural machine translation to apply terminology constraints," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 3063–3068.

[15] Y. Wang *et al.*, "Sogou neural machine translation systems for wmt17," in *Proc. 2nd Conf. Mach. Transl.*, 2017, pp. 410–415.

[16] M. T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," *Bulletin Univ. Agricultural Sci. Veterinary Medicine Cluj-Napoca. Veterinary Medicine*, vol. 27, no. 2, pp. 82–86, 2014.

[17] X. Li, G. Li, L. Liu, M. Meng, and S. Shi, "On the word alignment from neural machine translation," in *Proc. 57th Conf. Assoc. Comput. Linguistics*, 2019, pp. 1293–1303.

[18] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," in *Proc. 54th Annual Meeting of the Association for Computational Linguistics*, vol. 1: Long Papers, Berlin, Germany, Aug. 7–12, 2016.

[19] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, Aug. 7-12, 2016, Berlin, Germany, Vol. 1, Long Papers. The Association for Computer Linguistics.

[20] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proc. 55th Annual Meeting of the Association for Computational Linguistics*, vol. 1: Long Papers, R. Barzilay and M. Kan, Eds., Vancouver, Canada, Jul. 30–Aug. 4, 2017, pp. 1073–1083.

[21] R. Weng, H. Zhou, S. Huang, L. Li, Y. Xia, and J. Chen, "Correct-and-memorize: Learning to translate from interactive revisions," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 10-16, 2019, S. Kraus, Ed., 2019, pp. 5255–5263.

[22] S. Ding, H. Xu, and P. Koehn, "Saliency-driven word alignment interpretation for neural machine translation," in *Proc. 4th Conf. Mach. Translation*, Florence, Italy, Aug. 1-2, 2019, vol. 1, Research Papers.

[23] S. Garg, S. Peitz, U. Nallasamy, and M. Paulik, "Jointly learning to align and translate with transformer models," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, Hong Kong, China, Nov. 3-7, 2019, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., 2019, pp. 4452–4461.

[24] T. Zenkel, J. Wuebker, and J. DeNero, "Adding interpretable attention to neural translation models improves word alignment," *CoRR*, vol. abs/1901.11359, 2019.

[25] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Comput. Linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[26] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Comput. Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, Minneapolis, MN, USA, Jun. 2-7, 2019, vol. 1, (Long and Short Papers), J. Burstein, C. Doran, and T. Solorio, Eds., pp. 4171–4186..

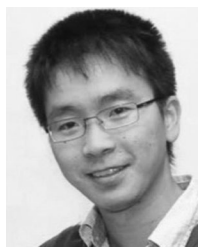
[28] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018. [Online]. Available: https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsu pervised/languageunderstanding paper.pdf

[29] G. Lample and A. Conneau, "Cross-lingual language model pretraining," in *Proc. Advances Neural Inf. Process. Syst. 32: Annu. Conf. Neural Inf. Process. Syst.*, 8-14 Dec. 2019, pp. 70577067.
[30] G. Lample, A. Conneau, L. Denoyer, and M. Ranzato, "Unsupervised machine translation using monolingual corpora only," in *Proc. 6th Int. Conf. Learn. Representations*, Vancouver, BC, Canada, Apr. 30 - May 3, 2018, Conference Track Proceedings.
[31] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Comput. Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.
[32] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *CoRR*, vol. abs/1607.06450, 2016.
[33] O. Press and L. Wolf, "Using the output embedding to improve language models," in *Proc. 15th Conf. Euro. Chapter Assoc. Comput. Linguistics*, Valencia, Spain, Short Papers, M. Lapata, P. Blunsom, and A. Koller, Eds., Apr. 3-7, 2017, vol. 2, pp. 157–163.
[34] J. Crego *et al.*, "Systran's pure neural machine translation systems," 2016, *arXiv:1610.05540*.
[35] E. Hasler, A. De Gispert, G. Iglesias, and B. Byrne, "Neural machine translation decoding with terminology constraints," in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, NAACLHLT, New Orleans, Louisiana, USA, M. A. Walker, H. Ji, and A. Stent, Eds., Jun. 1-6, 2018, vol. 2 (Short Papers), pp. 506–512.
[36] P. Koehn and R. Knowles, "Six challenges for neural machine translation," in *Proc. 1st Workshop Neural Mach. Translation*, NMT@ACL, Vancouver, Canada, Aug. 4, 2017.
[37] B. Krause, E. Kahembwe, I. Murray, and S. Renals, "Dynamic evaluation of neural sequence models," in *Proc. 35th Int. Conf. Mach. Learn.* 2018, pp. 2771–2780.
[38] B. Krause, E. Kahembwe, I. Murray, and S. Renals, "Dynamic evaluation of transformer language models," vol. abs/1904.08378, 2019.
[39] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. 54th Annu. Meeting Assoc. for Comput. Linguistics*, Berlin, Germany, Long Papers, Aug. 7-12, 2016, vol. 1.
[40] R. Sennrich *et al.*, "Nematus: a toolkit for neural machine translation," in *Proc. Softw. Demonstrations 15th Conf. Eur. Chapter Assoc. Comput. Linguistics*, Apr. 2017, pp. 65–68.
[41] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" in *Proc. Adv. Neural Inf. Process. Syst. 32: Annu. Conf. Neural Inf. Process. Syst.*, 8-14 Dec. 2019, pp. 14 014–14 024.
[42] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proc. 57th Conf. Assoc. Comput. Linguistics*, Florence, Italy, Jul. 28- Aug. 2, 2019, pp. 5797–5808.
[43] M. Ott *et al.*, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, Minneapolis, MN, USA, Jun. 2-7, 2019, pp. 48–53.
[44] F. J. Och and H. Ney, "Improved statistical alignment models," in *Proc. 38th Annu. Meeting Assoc. Comput. Linguistics*, 2000, pp. 440–447.
[45] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2002, pp. 311–318.
[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, Conference Track Proceedings, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 7-9, 2015.
[48] M. Junczys-Dowmunt, T. Dwojak, and R. Sennrich, "The amu-uedin submission to the wmt16 news translation task: Attention-based nmt models as feature functions in phrase-based smt," in *Proc. 1st Conf. Mach. Translation*, Aug. 11-12, Berlin, Germany, 2016, pp. 319325..
[49] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. Advances Neural Inf. Process. Syst.*, 2016, pp. 1019–1027.
[50] C. Dyer, V. Chahuneau, and N. A. Smith, "A simple, fast, and effective reparameterization of ibm model 2," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, 2013, pp. 644–648.
[51] L. Liu, M. Utiyama, A. Finch, and E. Sumita, "Neural machine translation with supervised attention," in *Proc. COLING, 26th Int. Conf. Comput. Linguistics, Proc. Conf.: Tech. Papers*, Dec. 11-16, 2016, pp. 3093–3102.
[52] H. Mi, Z. Wang, and A. Ittycheriah, "Supervised attentions for neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 2283–2288.
[53] M. E. Peters *et al.*, "Deep contextualized word representations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technologies*, 2018, pp. 2227–2237.
[54] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mass: Masked sequence to sequence pre-training for language generation," in *Proc. 36th Int. Conf. Mach. Learn.*, Jun. 9-15, 2019, pp. 5926–5936.

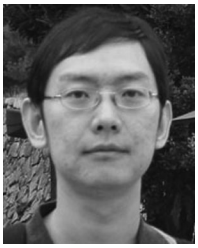
Kai Song received the MSc degree from Northeastern University, China, in 2014. He currently works as an Algorithm ExperT at Alibaba Group, he is also a Ph.D. Candidate Soochow University. Kai Song's research interests include machine translation and natural language processing.

Xiaoqing Zhou received the MSc degree from Soochow University, in 2020. He currently works as an Algorithm Engineer at OPPOLtd, His research interests include machine translation and natural language processing.

Heng Yu received the Ph.D. degree from Chinese Academy of Sciences, in 2014. He currently works as an Senior Algorithm expert at AlibabaGroup. His research interests include machine translation and natural language processing, intelligent computing and machine learning.



Zhongqiang Huang received the Bachelor degree at University of Science and Technology of China, in 2004. He received his MSc degree at Purdue University, in 2007. He received his Ph.D. degree at University of Maryland College Park, in 2011. He currently works as an Senior Algorithm Expert at Alibaba Group. His research interests include machine translation and natural language processing, intelligent computing and machine learning.



Yue Zhang received Ph.D. degree from the University of Oxford in Dec 2009, working on statistical Chinese processing. He received M.Sc. degree from the University of Oxford in Oct 2006, working on statistical machine translation from Chinese to English by parsing. He currently works as an Associate Professor at Westlake University. From Jul. 2012 to Aug. 2018, he worked as an Assistant Professor at Singapore University of Technology and Design (SUTD). Before joining SUTD, he worked as a Postdoctoral Research Associate at the University of Cambridge. He received undergraduate degree on Computer Science from Tsinghua University, China.

Weihua Luo received the Bachelor degree at Tsinghua University, in 1999. He received his MSc degree at Tsinghua University, in 2002. He received his Ph.D. degree at Chinese Academy of Sciences, in 2010. He currently works as an Senior Algorithm Expert at Alibaba Group. His research interests include machine translation and natural language processing, intelligent computing and machine learning.

Xiangyu Duan received his Ph.D. in Institute of Automation, Chinese Academy of Sciences, in 2008. He joined Soochow University as an Associate Professor in 2013. Before that, he was a Research Scientist of the Institute for Infocomm Research of Singapore from 2009 to 2013. In 2008 to 2009, he worked as a Postdoctoral Research Fellow in the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include machine translation, parsing, and machine learning.

Min Zhang received his Bachelor degree and Ph.D. degree in computer science from the Harbin Institute of Technology in 1991 and 1997, respectively. He joined the Soochow University in 2013 and is currently a Distinguished Professor at the university. From 1997 to 1999, he worked as a postdoctoral research fellow in Korean Advanced Institute of Science and Technology in South Korea. He began his academic and industrial career as a Researcher at Lernout & Hauspie Asia Paci?c (Singapore) in 1999. He joined Infotalk Technology (Singapore) as a Researcher in 2001 and became a Senior Research Manager in 2002. He joined the Institute for Infocomm Research (Singapore) in 2003. His current research interests include machine translation, natural language processing, information extraction, large scale text processing, intelligent computing and machine learning. He has authored 150 papers in leading journals and conferences. He is the Vice President of COLIPS, a Steering Committee Member of PACLIC, an Executive Member of AFNLP and a member of ACL.