Accurate Word Alignment Induction from Neural Machine Translation

Yun Chen[*1], **Yang Liu**[2], **Guanhua Chen**[3], **Xin Jiang**[4], **Qun Liu**[4]
[1]Shanghai University of Finance and Economics, Shanghai, China
[2]Tsinghua University, Beijing, China
[3]The University of Hong Kong, Hong Kong, China
[4]Huawei Noah's Ark Lab, Hong Kong, China
yunchen@sufe.edu.cn, liuyang2011@tsinghua.edu.cn,
ghchen@eee.hku.hk, {jiang.xin, qun.liu}@huawei.com

Abstract

Despite its original goal to jointly learn to align and translate, prior researches suggest that Transformer captures poor word alignments through its attention mechanism. In this paper, we show that attention weights DO capture accurate word alignments and propose two novel word alignment induction methods SHIFT-ATT and SHIFT-AET. The main idea is to induce alignments at the step when the to-be-aligned target token is the decoder input rather than the decoder output as in previous work. SHIFT-ATT is an interpretation method that induces alignments from the attention weights of Transformer and does not require parameter update or architecture change. SHIFT-AET extracts alignments from an additional alignment module which is tightly integrated into Transformer and trained in isolation with supervision from symmetrized SHIFT-ATT alignments. Experiments on three publicly available datasets demonstrate that both methods perform better than their corresponding neural baselines and SHIFT-AET significantly outperforms GIZA++ by 1.4-4.8 AER points.[1]

1 Introduction

The task of word alignment is to find lexicon translation equivalents from parallel corpus (Brown et al., 1993). It is one of the fundamental tasks in natural language processing (NLP) and is widely studied by the community (Dyer et al., 2013; Brown et al., 1993; Liu and Sun, 2015). Word alignments are useful in many scenarios, such as error analysis (Ding et al., 2017; Li et al., 2019), the introduction of coverage and fertility models (Tu et al., 2016), inserting external constraints in interactive machine translation (Hasler et al., 2018; Chen et al., 2020) and providing guidance for human translators in computer-aided translation (Dagan et al., 1993).

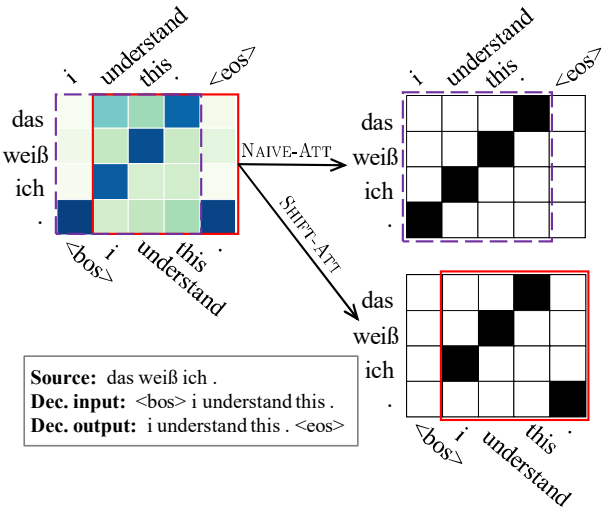


Figure 1: An example to compare our method SHIFT-ATT and the baseline NAIVE-ATT. The left is an attention map from the third decoder layer of the vanilla Transformer and the right are the induced alignments. SHIFT-ATT induces alignments for target word y_i at decoding step $i+1$ when y_i is the decoder input, while NAIVE-ATT at step i when y_i is the decoder output.

Word alignment is part of the pipeline in statistical machine translation (Koehn et al., 2003, SMT), but is not necessarily needed for neural machine translation (Bahdanau et al., 2015, NMT). The attention mechanism in NMT does not functionally play the role of word alignments between the source and the target, at least not in the same way as its analog in SMT. It is hard to interpret the attention activations and extract meaningful word alignments especially from Transformer (Garg et al., 2019). As a result, the most widely used word alignment tools are still external statistical models such as FAST-ALIGN (Dyer et al., 2013) and GIZA++ (Brown et al., 1993; Och and Ney, 2003).

*Corresponding author. Part of the work was done when Yun was in Huawei Noah's Ark Lab.

[1]Code can be found at https://github.com/sufe-nlp/transformer-alignment.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 566–576,
November 16–20, 2020.

Recently, there is a resurgence of interest in the community to study word alignments for the Transformer (Ding et al., 2019; Li et al., 2019). One simple solution is NAIVE-ATT, which induces word alignments from the attention weights between the encoder and decoder. The next target word is aligned with the source word that has the maximum attention weight, as shown in Fig. 1. However, such schedule only captures noisy word alignments (Ding et al., 2019; Garg et al., 2019). One of the major problems is that it induces alignment before observing the to-be-aligned target token (Peter et al., 2017; Ding et al., 2019). Suppose for the same source sentence, there are two alternative translations that diverge at decoding step i, generating y_i and y_i' which respectively correspond to different source words. Presumably, the source word that is aligned to y_i and y_i' should change correspondingly. However, this is not possible under the above method, because the alignment scores are computed before prediction of y_i or y_i'.

To alleviate this problem, some researchers modify the transformer architecture by adding alignment modules that predict the to-be-aligned target token (Zenkel et al., 2019, 2020) or modify the training loss by designing an alignment loss computed with full target sentence (Garg et al., 2019; Zenkel et al., 2020). Others argue that using only attention weights is insufficient for generating clean word alignment and propose to induce alignments with feature importance measures, such as leave-one-out measures (Li et al., 2019) and gradient-based measures (Ding et al., 2019). However, all previous work induces alignment for target word y_i at step i, when y_i is the decoder output.

In this work, we propose to induce alignment for target word y_i at step $i+1$ rather than at step i as in previous work. The motivation behind this is that the hidden states in step $i+1$ are computed taking word y_i as the input, thus they can incorporate the information of the to-be-aligned target token y_i easily. Following this idea, we present SHIFT-ATT and SHIFT-AET, two simple yet effective methods for word alignment induction. Our contributions are threefold:

• We introduce SHIFT-ATT (see Fig. 1), a pure interpretation method to induce alignments from attention weights of vanilla Transformer. SHIFT-ATT is able to reduce the Alignment Error Rate (AER) by 7.0-10.2 points over NAIVE-ATT and 5.5-7.9 points over FAST-ALIGN on three publicly available datasets, demonstrating that if the correct decoding step and layer are chosen, attention weights in vanilla Transformer are **sufficient** for generating accurate word alignment interpretation.
• We further propose SHIFT-AET , which extracts alignments from an additional alignment module. The module is tightly integrated into vanilla Transformer and trained with supervision from symmetrized SHIFT-ATT alignments. SHIFT-AET does not affect the translation accuracy and significantly outperforms GIZA++ by 1.4-4.8 AER points in our experiments.
• We compare our methods with NAIVE-ATT on dictionary-guided decoding (Alkhouli et al., 2018), an alignment-related downstream task. Both methods consistently outperform NAIVE-ATT, demonstrating the effectiveness of our methods in such alignment-related NLP tasks.

2 Background

2.1 Neural Machine Translation

Let $\mathbf{x} = \{x_1, ..., x_{|\mathbf{x}|}\}$ and $\mathbf{y} = \{y_1, ..., y_{|\mathbf{y}|}\}$ be source and target sentences. Neural machine translation models the target sentence given the source sentence as $p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})$:

$$p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta}) = \prod_{t=1}^{|\mathbf{y}|+1} p(y_t|y_{0:t-1}, \mathbf{x}; \boldsymbol{\theta}), \quad (1)$$

where $y_0 = \langle \text{bos} \rangle$ and $y_{|\mathbf{y}|+1} = \langle \text{eos} \rangle$ represent the beginning and end of the target sentence respectively, and $\boldsymbol{\theta}$ is a set of model parameters.

In this paper, we use Transformer (Vaswani et al., 2017) to implement the NMT model. Transformer is an encoder-decoder model that only relies on attention. Each decoder layer attends to the encoder output with multi-head attention. We refer to the original paper (Vaswani et al., 2017) for more model details.

2.2 Alignment by Attention

The encoder output from the last encoder layer is denoted as $\mathbf{h} = \{h_1, ..., h_{|\mathbf{x}|}\}$, and the hidden states at decoder layer l as $\mathbf{z} = \{z_1^l, ..., z_{|\mathbf{y}|+1}^l\}$. For decoder layer l, we define the head averaged encoder-decoder attention weights as $\boldsymbol{W}^l \in \mathbb{R}^{(|\mathbf{y}|+1)\times|\mathbf{x}|}$, in which the element $W_{i,j}^l$ measures the relevance between decoder hidden state z_i^l and encoder output h_j. For simplicity, below we use the term "attention weights" to denote the head averaged encoder-decoder attention weights.

Given a trained Transformer model, word alignments can be extracted from the attention weights. More specifically, we denote the alignment score matrix as $\boldsymbol{S} \in \mathbb{R}^{|\mathbf{y}|\times|\mathbf{x}|}$, in which the element $S_{i,j}$ is the alignment score of target word y_i and source word x_j. Then we compute $\boldsymbol{S}$ with:

$$S_{i,j} = W^l_{i,j} \quad (1 \leq i \leq |\mathbf{y}|, 1 \leq j \leq |\mathbf{x}|) \quad (2)$$

and extract word alignments $\boldsymbol{A}$ with maximum a posterior strategy following Garg et al. (2019):

$$A_{ij} = \begin{cases} 1 & \text{if } j = \text{argmax}_{j'} S_{i,j'} \\ 0 & \text{otherwise} \end{cases}, \quad (3)$$

where $A_{ij} = 1$ indicates y_i is aligned to x_j. We call this approach NAIVE-ATT. Garg et al. (2019) show that attention weights from the penultimate layer, i.e., $l = L - 1$, can induce the best alignments.

Although simple to implement, this method fails to obtain satisfactory word alignments (Ding et al., 2019; Garg et al., 2019). First of all, instead of the relevance between y_i and x_j, $W^l_{i,j}$ measures the relevance between decoder hidden state z^l_i and encoder output h_j. Considering that the decoder input is y_{i-1} and the output is y_i at step i, z^l_i may better represent y_{i-1} instead of y_i, especially for bottom layers. Second, since $W^l_{i,j}$ is computed before observing y_i, it becomes difficult for it to induce the aligned source token for the target token y_i, as discussed in Section 1.

As a result, it is necessary to develop novel methods for alignment induction. This method should be able to (i) take into account the relationship of z^l_i, y_i and y_{i-1}, and (ii) adapt the alignment induction with the to-be-aligned target token.

3 Method

In this section, we propose two novel alignment induction methods SHIFT-ATT and SHIFT-AET. Both methods adapt the alignment induction with the to-be-aligned target token by computing alignment scores at the step when the target token is the decoder input.

3.1 SHIFT-ATT: Alignment from Vanilla Transformer

Alignment Induction NAIVE-ATT (Garg et al., 2019) induces alignment for target token y_i at step i when y_i is the decoder output and defines the alignment score matrix with Eq. 2. They find the best layer l to extract alignments by evaluating the AER of all layers on the test set.

We instead propose to induce alignment for target token y_i at step $i+1$ when y_i is the decoder input. We define the alignment score matrix $\boldsymbol{S}$ as:

$$S_{i,j} = W^l_{i+1,j} \quad (1 \leq i \leq |\mathbf{y}|, 1 \leq j \leq |\mathbf{x}|). \quad (4)$$

This is because $W^l_{i+1,j}$ measures the relevance between z^l_{i+1} and h_j, and we use z^l_{i+1} and h_j to represent y_i and x_j respectively. With the alignment score matrix $\boldsymbol{S}$, we can extract word alignments $\boldsymbol{A}$ using Eq. 3. We call this method SHIFT-ATT. Fig. 1 shows an alignment induction example to compare NAIVE-ATT and SHIFT-ATT.

SHIFT-ATT uses z^l_{i+1} to represent the to-be-aligned target token y_i while NAIVE-ATT uses z^l_i. We argue using z^l_{i+1} is better. First, at bottom layers, we hypothesize that z^l_{i+1} could better represent the decoder input y_i than output y_{i+1}. Therefore we can use z^l_{i+1} with small l to represent y_i. Second, z^l_{i+1} is computed after observing y_i, indicating that SHIFT-ATT is able to adapt the alignment induction with the to-be-aligned target token.

Our proposed method involves inducing alignments from source-to-target and target-to-source vanilla Transformer models. Following Zenkel et al. (2019), we merge bidirectional alignments using the grow diagonal heuristic (Koehn et al., 2005).

Layer Selection Criterion To select the best layer l_b to induce alignments, we propose a surrogate layer selection criterion without manually labelled word alignments. Experiments show that this criterion correlates well with the AER metric.

Given parallel sentence pairs $\langle \mathbf{x}, \mathbf{y} \rangle$, we train a source-to-target model $\boldsymbol{\theta}_{\mathbf{x}\to\mathbf{y}}$ and a target-to-source model $\boldsymbol{\theta}_{\mathbf{y}\to\mathbf{x}}$. We assume that the word alignments extracted from these two models should agree with each other (Cheng et al., 2016). Therefore, we evaluate the quality of the alignments by computing the AER score on the validation set with the source-to-target alignments as the hypothesis and the target-to-source alignments as the reference. For each model, we can obtain L word alignments from L different layers. In total, we obtain $L \times L$ AER scores. We select the one with the lowest AER score, and its corresponding layers of the source-to-target and target-to-source models are the layers we will use to extract alignments at test time:

$$l_{\text{b},\mathbf{x}\to\mathbf{y}}, l_{\text{b},\mathbf{y}\to\mathbf{x}} = \underset{i,j}{\text{argmin}}\, \text{AER}(\boldsymbol{A}^i_{\mathbf{x}\to\mathbf{y}}, \boldsymbol{A}^j_{\mathbf{y}\to\mathbf{x}}).$$

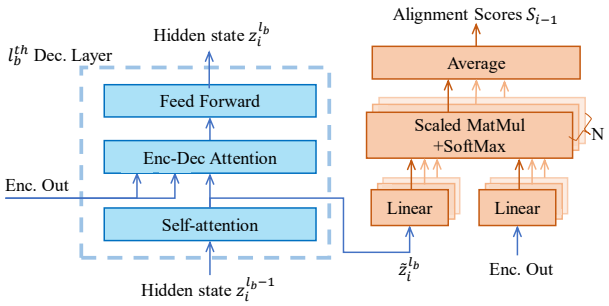


Figure 2: Illustration of the alignment module at decoding step i. The decoder input token is y_{i-1}, while the output token is y_i. The alignment module predicts S_{i-1}, the alignment scores corresponding to the input target token y_{i-1}. During the alignment module training process, parameters of the blue blocks are frozen, and only parameters of the orange blocks are updated.

3.2 SHIFT-AET: Alignment from Alignment-Enhanced Transformer

To further improve the alignment accuracy, we propose SHIFT-AET, a word alignment induction method that extracts alignments from Alignment-Enhanced Transformer (AET). AET extends the Transformer architecture with a separate alignment module, which observes the hidden states of the underlying Transformer at each step and predicts the alignment scores for the current decoder **input**. Note that this module is a plug and play component and it neither makes any change to the underlying NMT model nor influences the translation quality.

Fig. 2 illustrates the alignment module of AET at decoding step i. We add the alignment module only at layer l_b, the best layer to extract alignments with SHIFT-ATT. The alignment module performs multi-head attention similar to the encoder-decoder attention sublayer. It takes the encoder outputs $\mathbf{h} = \{h_1, ..., h_{|\mathbf{x}|}\}$ and the current decoder hidden state $\tilde{z}_i^{l_b}$ inside layer l_b as input and outputs S_{i-1}, the alignment score corresponding to target word y_{i-1}:

$$S_{i-1} = \frac{1}{N}\sum_n \text{softmax}\left(\frac{(\mathbf{h}\boldsymbol{G}_n^K)(\tilde{z}_i^{l_b}\boldsymbol{G}_n^Q)^\top}{\sqrt{d_k}}\right), \quad (5)$$

where $\boldsymbol{G}_n^K, \boldsymbol{G}_n^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$ are the key and query projection matrices for the n-th head, N is the number of attention heads and $d_k = d_{\text{model}}/N$. Since we only care about the attention weights, the value-related parameters and computation are omitted in this module.

To train the alignment module, we use the symmetrized SHIFT-ATT alignments extracted from vanilla Transformer models as labels. Specifically, while the underlying Transformer is pretrained and fixed (Fig. 2), we train the alignment module with the loss function following Garg et al. (2019):

$$\mathcal{L}_a = -\frac{1}{|\mathbf{y}|}\sum_{i=1}^{|\mathbf{y}|}\sum_{j=1}^{|\mathbf{x}|}(\hat{A}^p_{i,j} \odot \log S_{i,j}), \quad (6)$$

where $\boldsymbol{S} = \{S_1;...;S_{|\mathbf{y}|}\}$ is the alignment score matrix predicted by the alignment module, and $\hat{\boldsymbol{A}}^p$ denotes the normalized reference symmetrized SHIFT-ATT alignments.[2] In this way, we transfer the alignment knowledge implicitly learned in two vanilla Transformer models $\boldsymbol{\theta}_{\mathbf{x}\to\mathbf{y}}$ and $\boldsymbol{\theta}_{\mathbf{y}\to\mathbf{x}}$ into the alignment module of a single AET model.

Once the alignment module is trained, we extract alignment scores $\boldsymbol{S}$ from it given a parallel sentence pair and induce alignments $\boldsymbol{A}$ using Eq. 3.

Dataset	Train	Validation	Test
de-en	1.9M	994	508
fr-en	1.1M	1,000	447
ro-en	0.5M	999	248

Table 1: Number of sentences in each dataset.

4 Experiments

4.1 Settings

Dataset We follow previous work (Zenkel et al., 2019, 2020) in data setup and conduct experiments on publicly available datasets for German-English (de-en)[3], Romanian-English (ro-en) and French-English (fr-en)[4]. Since no validation set is provided, we follow Ding et al. (2019) to set the last 1,000 sentences of the training data before preprocessing as validation set. We learn a joint source and target Byte-Pair-Encoding (Sennrich et al., 2016) with 10k merge operations. Table 1 shows the detailed data statistics.

NMT Systems We implement the Transformer with fairseq-py[5] and use the `transformer_iwslt_de_en` model configuration following Ding et al. (2019). We train the models with a batch size of 36K tokens and set the maximum updates as 50K and 10K for

[2] We simply normalize rows corresponding to target tokens that are aligned to at least one source token of $\hat{\boldsymbol{A}}$.

[3] https://www-i6.informatik.rwth-aachen.de/goldAlignment/

[4] http://web.eecs.umich.edu/~mihalcea/wpt/index.html

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

Method	Inter.	Fullc	de-en			fr-en			ro-en		
			de→en	en→de	bidir	fr→en	en→fr	bidir	ro→en	en→ro	bidir
Statistical Methods											
FAST-ALIGN (Dyer et al., 2013)	-	Y	28.5	30.4	25.7	16.3	17.1	12.1	33.6	36.8	31.8
GIZA++ (Brown et al., 1993)	-	Y	18.8	19.6	17.8	7.1	7.2	6.1	27.4	28.7	26.0
Neural Methods											
NAIVE-ATT (Garg et al., 2019)	Y	N	33.3	36.5	28.1	27.5	23.6	16.0	33.6	35.1	30.9
NAIVE-ATT-LA (Garg et al., 2019)	Y	N	40.9	50.8	39.8	32.4	29.8	21.2	37.5	35.5	32.7
SHIFT-ATT-LA	Y	N	54.7	46.2	45.5	60.5	46.9	55.1	66.1	60.4	65.3
SMOOTHGRAD (Li et al., 2016)	Y	N	36.4	45.8	30.3	25.5	27.0	15.6	41.3	39.9	33.7
SD-SMOOTHGRAD (Ding et al., 2019)	Y	N	36.4	43.0	29.0	25.9	29.7	15.3	41.2	41.4	32.7
PD (Li et al., 2019)	Y	N	38.1	44.8	34.4	32.4	31.1	23.1	40.2	40.8	35.6
ADDSGD (Zenkel et al., 2019)	N	N	26.6	30.4	21.2	20.5	23.8	10.0	32.3	34.8	27.6
MTL-FULLC (Garg et al., 2019)	N	Y	-	-	20.2	-	-	7.7	-	-	26.0
Statistical + Neural Methods											
MTL-FULLC-GZ (Garg et al., 2019)	N	Y	-	-	16.0	-	-	**4.6**	-	-	23.1
Our Neural Methods											
SHIFT-ATT	Y	N	20.9	25.7	17.9	17.1	16.1	6.6	27.4	26.0	23.9
SHIFT-AET	N	N	15.8	19.2	**15.4**	9.9	10.5	4.7	22.7	23.6	**21.2**

Table 2: AER on the test set with different alignment methods. *bidir* are symmetrized alignment results. The column Inter. represents whether the method is an interpretation method that can extract alignments from a pretrained vanilla Transformer model. The column Fullc denotes whether full target sentence is used to extract alignments at test time. The lower AER, the better. We mark best symmetrized interpretation results of vanilla Transformer with underlines, and best symmetrized results among all with boldface.

Transformer and AET respectively. The last checkpoint of AET is used for evaluation. All models are trained in both translation directions and symmetrized with *grow-diag* (Koehn et al., 2005) using the script from Zenkel et al. (2019).[6]

Evaluation We evaluate the alignment quality of our methods with Alignment Error Rate (Och and Ney, 2000, AER). Since word alignments are useful for many downstream tasks as discussed in Section 1, we also evaluate our methods on dictionary-guided decoding, a downstream task of alignment induction, with the metric BLEU (Papineni et al., 2002). More details are in Section 4.3.

Baselines We compare our methods with two statistical baselines FAST-ALIGN and GIZA++ and nine other baselines:

• NAIVE-ATT (Garg et al., 2019): the approach we discuss in Section 2.2, which induces alignments from the attention weights of the penultimate layer of the Transformer.

• NAIVE-ATT-LA (Garg et al., 2019): the NAIVE-ATT method without layer selection. It induces alignments from attention weights averaged across all layers.

• SHIFT-ATT-LA: SHIFT-ATT method without layer selection. It induces alignments from attention weights averaged across all layers.

• SMOOTHGRAD (Li et al., 2016): the method that induces alignments from word saliency, which is computed by averaging the gradient-based saliency scores with multiple noisy sentence pairs as input.

• SD-SMOOTHGRAD (Ding et al., 2019): an improved version of SMOOTHGRAD, which defines saliency on one-hot input vector instead of word embedding.

• PD (Li et al., 2019): the method that computes the alignment scores from Transformer by iteratively masking each source token and measuring the prediction difference.

• ADDSGD (Zenkel et al., 2019): the method that explicitly adds an extra attention layer on top of Transformer and directly optimizes its activations towards predicting the to-be-aligned target token.

• MTL-FULLC (Garg et al., 2019): the method that trains a single model in a multi-task learning framework to both predict the target sentence and the alignment. When predicting the alignment, the model observes full target sentence and uses symmetrized NAIVE-ATT alignments as labels.

• MTL-FULLC-GZ (Garg et al., 2019): the same method as MTL-FULLC except using symmetrized GIZA++ alignments as labels. It is a statistical and neural method as it relies on GIZA++ alignments.

Among these nine baselines and our proposed methods, SMOOTHGRAD, SD-SMOOTHGRAD and PD induce alignments using feature importance measures, while the others from some form of attention weights. Note that the computation

[6] https://github.com/lilt/alignment-scripts

cost of methods with feature importance measures is much higher than those with attention weights.[7]

4.2 Alignment Results

Comparison with Baselines Table 2 compares our methods with all the baselines. First, SHIFT-ATT, a pure interpretation method for the vanilla Transformer, significantly outperforms FAST-ALIGN and all neural baselines, and performs comparable with GIZA++. For example, it outperforms SD-SMOOTHGRAD, the state-of-the-art method with feature importance measures to extract alignments from vanilla Transformer, by 8.7-11.1 AER points across different language pairs. The success of SHIFT-ATT demonstrates that vanilla Transformer has captured alignment information in an implicit way, which could be revealed from the attention weights if the correct decoding step and layer are chosen to induce alignments.

Second, the method SHIFT-AET achieves new state-of-the-art, significantly outperforming all baselines. It improves over GIZA++ by 1.4-4.8 AER across different language pairs, demonstrating that it is possible to build a neural aligner better than GIZA++ without using any alignments generated from statistical aligners to bootstrap training. We also find SHIFT-AET performs either marginally better (de-en and ro-en) or on-par (fr-en) when comparing with MTL-FULLC-GZ, a method that uses GIZA++ alignments to bootstrap training. We evaluate the model sizes: the number of parameters in vanilla Transformer and AET are 36.8M and 37.3M respectively, and find that AET only introduces 1.4% additional parameters to the vanilla Transformer. In summary, by supervising the alignment module with symmetrized SHIFT-ATT alignments, SHIFT-AET improves over SHIFT-ATT and GIZA++ with negligible parameter increase and without influencing the translation quality.

Comparison with Zenkel et al. (2020) Concurrent with our work, Zenkel et al. (2020) propose a neural aligner that can outperform GIZA++. Table 3 compares the performance of SHIFT-AET and the best method BAO-GUIDED (Birdir. Att. Opt. + Guided) in Zenkel et al. (2020). We observe that SHIFT-AET performs better than BAO-GUIDED in terms of alignment accuracy.

Method	de-en	fr-en	ro-en
BAO-GUIDED	16.3	5.0	23.4
SHIFT-AET	**15.4**	**4.7**	**21.2**

Table 3: Comparison of our method SHIFT-AET with BAO-GUIDED (Zenkel et al., 2020). We report the symmetrized AER on the test set.

Direction	zh→en	en→zh	bidir
GIZA++	19.6	23.3	18.5
NAIVE-ATT	36.9	40.3	28.9
SHIFT-ATT	28.1	27.3	20.2
SHIFT-AET	20.1	22.0	**17.2**

Table 4: AER on the test set of zh-en. *bidir* are symmetrized alignment results.

SHIFT-AET is also much simpler than BAO-GUIDED. The training of BAO-GUIDED includes three stages: (i) train vanilla Transformer in source-to-target and target-to-source directions; (ii) train the alignment layer and extract alignments on the training set with bidirectional attention optimization. This alignment extraction process is computational costly since bidirectional attention optimization fine-tunes the model parameters separately for each sentence pair in the training set; (iii) re-train the alignment layer with the extracted alignments as the guidance. In contrast, SHIFT-AET can be trained much faster in two stages and does not involve bidirectional attention optimization.

Similar with MTL-FULLC (Garg et al., 2019), BAO-GUIDED adapts the alignment induction with the to-be-aligned target token by requiring full target sentence as the input. Therefore, BAO-GUIDED is not applicable in cases where alignments are incrementally computed during the decoding process, e.g., dictionary-guided decoding (Alkhouli et al., 2018). In contrast, SHIFT-AET performs quite well on such cases (Section 4.3). Therefore, considering the alignment performance, computation cost and applicable scope, we believe SHIFT-AET is more appropriate than BAO-GUIDED for the task of alignment induction.

Performance on Distant Language Pair To further demonstrate the superiority of our methods on distant language pairs, we also evaluate our methods on Chinese-English (zh-en). We use NIST corpora[8] as the training set and v1-tstset released by TsinghuaAligner (Liu and Sun, 2015) as the test

[7] For each sentence pair, PD forwards once with $|\mathbf{x}| + 1$ masked sentence pairs as the input, while SMOOTHGRAD and SD-SMOOTHGRAD forward and backward once with m ($m = 30$ in Ding et al. (2019)) noisy sentence pairs as the input. In contrast, attention weights based methods forward once with one sentence pair as the input.

[8] The corpora include LDC2002E18, LDC2003E07, LDC2003E14, LDC2004T07, LDC2004T08 and LDC2005T06

Task	NAIVE-ATT	SHIFT-ATT	SHIFT-AET
de→en	33.7	34.3*	**34.8***
en→de	26.5	26.8	**28.0***

Table 5: Comparison of dictionary-guided decoding with different alignment methods. We report BLEU scores on the test set. Without dictionary-guided decoding, we obtain 32.3 and 24.2 BLEU on de→en and en→de translations respectively. "*" indicates the result is significantly better than that of NAIVE-ATT ($p<0.05$). All significance tests are measured by paired bootstrap resampling (Koehn, 2004)

set. The test set includes 450 parallel sentence pairs with manually labelled word alignments.[9] We use *jieba*[10] for Chinese text segmentation and follow the settings in Section 4.1 for data pre-processing and model training. The results are shown in Table 4. It presents that both SHIFT-ATT and SHIFT-AET outperform NAIVE-ATT to a large margin. When comparing the symmetrized alignment performance with GIZA++, SHIFT-AET performs better, while SHIFT-ATT is worse. The experimental results are roughly consistent with the observations on other language pairs, demonstrating the effectiveness of our methods even for distant language pairs.

4.3 Downstream Task Results

In addition to AER, we compare the performance of NAIVE-ATT, SHIFT-ATT and SHIFT-AET on dictionary-guided machine translation (Song et al., 2020), which is an alignment-based downstream task. Given source and target constraint pairs from dictionary, the NMT model is encouraged to translate with provided constraints via word alignments (Alkhouli et al., 2018; Hasler et al., 2018; Hokamp and Liu, 2017; Song et al., 2020). More specifically, at each decoding step, the last token of the candidate translation will be revised with target constraint if it is aligned to the corresponding source constraint according to the alignment induction method. To simulate the process of looking up dictionary, we follow Hasler et al. (2018) and extract the pre-specified constraints from the test set and its reference according to the golden word alignments. We exclude stop words, and sample up to 3 dictionary constraints per sentence. Each dictionary constraint includes up to 3 source tokens.

Table 5 presents the performance with different alignment methods. Both SHIFT-ATT and SHIFT-AET outperform NAIVE-ATT. SHIFT-AET obtains the best translation quality, improving over NAIVE-ATT by 1.1 and 1.5 BLEU scores on de→en and en→de translations, respectively. The results suggest the effectiveness of our methods in application to alignment-related NLP tasks.

[9] TsinghuaAligner labels the word alignments based on segmented Chinese sentences and does not provide the segmentation model. Therefore, we convert the manually labelled word alignments to our segmented Chinese sentences for evaluation.

[10] https://github.com/fxsjy/jieba

(a) Validation AER for Layer Selection

de→en / en→de	1	2	3	4	5	6
1	42.2	35.4	35.7	67.5	89.2	88.8
2	45.1	39.5	39.1	67.1	87.8	88.2
3	42.5	34.6	**34.2**	65.2	87.4	87.6
4	74.4	73.0	72.3	80.6	89.5	89.7
5	84.8	86.7	86.1	87.3	88.7	88.9
6	87.1	88.2	87.6	88.1	88.7	88.6

(b) Test AER for Verification

layer	1	2	3	4	5	6
de→en	31.5	22.7	**20.9**	55.7	80.5	81.5
en→de	27.4	31.3	**25.7**	68.5	83.4	85.1

Table 6: Layer selection criterion verification with SHIFT-ATT on de-en alignment. (a) For each cell, we induce hypothesis alignment from de→en translation and reference alignment from en→de translation. $l_{\text{b}} = 3$ for both translation directions in this table. (b) Test AER when inducing alignments from different layers. Layer 3 induces the best alignment for both translation directions, which verifies l_{b} selected in (a).

4.4 Analysis

Layer Selection Criterion To test whether the layer selection criterion can select the right layer to extract alignments, we first determine the best layer $l_{\text{b},\mathbf{x}\to\mathbf{y}}$ and $l_{\text{b},\mathbf{y}\to\mathbf{x}}$ based on the layer selection criterion. Then we evaluate the AER scores of alignments induced from different layers on the test set, and check whether the layers with the lowest AER score are consistent with $l_{\text{b},\mathbf{x}\to\mathbf{y}}$ and $l_{\text{b},\mathbf{y}\to\mathbf{x}}$. The experiment results shown in Table 6 verify that the layer selection criterion is able to select the best layer to induce alignments. We also find that the best layer is always layer 3 under our setting, consistent across different language pairs.

Relevance Measure Verification To investigate the relationship between z_i^l and y_{i-1}/y_i, we design an experiment to probe whether z_i^l contain the identity information of y_{i-1} and y_i, following Brunner et al. (2019). Formally, for decoder hidden state z_i^l, the input token is identifiable if there exists a func-

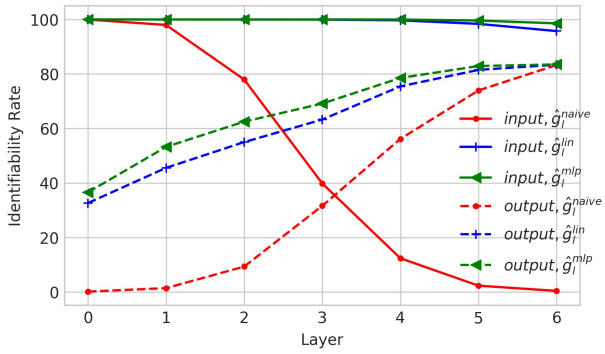


Figure 3: Identifiability rate of the input and output tokens for decoder hidden states at different layers.

tion g such that $y_{i-1} = g(z_i^l)$. We cannot prove the existence of g analytically. Instead, for each layer l we learn a projection function $\hat{g}_l$ to project from the hidden state space to the input token embedding space $\hat{y}_i^l = \hat{g}_l(z_i^l)$ and then search for the nearest neighbour y_k within the same sentence. We say that z_i^l can identify y_{i-1} if $k = i - 1$. Similarly, we follow the same process to identify the output token y_i. We report the identifiability rate defined as the percentage of correctly identified tokens.

Fig. 3 presents the results on the validation set of de→en translation. We try three projection functions: a naive baseline $\hat{g}_l^{\text{naive}}(z_i^l) = z_i^l$, a linear perceptron $\hat{g}_l^{\text{lin}}$ and a non-linear multi-layer perceptron $\hat{g}_l^{\text{mlp}}$. We observe the following points: (i) With trainable projection functions $\hat{g}_l^{\text{lin}}$ and $\hat{g}_l^{\text{mlp}}$, all layers can identify the input tokens, although more hidden states cannot be mapped back to their input tokens anymore in higher layers. (ii) Overall it is easier to identify the input token than the output token. For example, when projecting with mlp, all layers can identify more than 98% of the input tokens. However, for the output tokens, we can only identify 83.5% even from the best layer. Since z_i^l even may not be able to identify y_i, this observation partially verifies that it is better to represent y_i using z_{i+1}^l than z_i^l. (iii) At bottom layers, the input tokens remain identifiable and the output tokens are hard to identify, regardless of the projection function we use. This confirms our hypothesis that for small l, z_i^l is more relevant to y_{i-1} than y_i.

AER v.s. BLEU During training, vanilla Transformer gradually learns to align and translate. To analyze how the alignment behavior changes at different layers with checkpoints of different translation quality, we plot AER on the test set v.s. BLEU on the validation set for de→en translation. We compare NAIVE-ATT and SHIFT-ATT, which align the decoder output token (*align output*) and decoder input token (*align input*) to the source tokens based on current decoder hidden state, respectively.

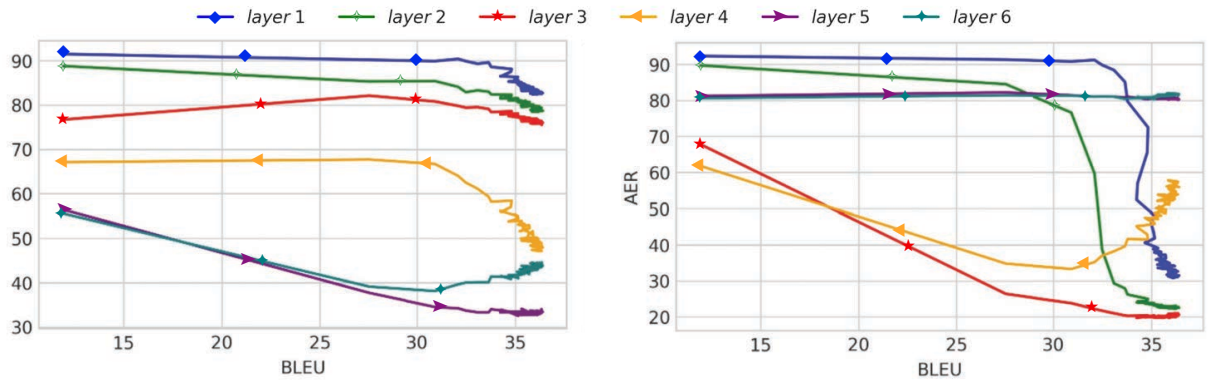
The experiment results are shown in Fig. 4. We observe that at the beginning of training, layers 3 and 4 learn to align the input token, while layers 5 and 6 the output token. However, with the increasing of BLEU score, layer 4 tends to change from aligning input token to aligning output token, and layer 1 and 2 begin to align input token. This suggests that vanilla Transformer gradually learns to align the input token from middle layers to bottom layers. We also see that at the end of training, layer 6's ability to align output token decreases. We hypothesize that layer 5 already has the ability to attend to the source tokens which are aligned to the output token, therefore attention weights in layer 6 may capture other information needed for translation. Finally, for checkpoints with the highest BLEU score, layer 5 aligns the output token best and layer 3 aligns the input token best.

Alignment Example In Fig. 5, we present a symmetrized alignment example from de-en test set. Manual inspection of this example as well as others finds that our methods SHIFT-ATT and SHIFT-AET tend to extract more alignment pairs than GIZA++, and extract better alignments especially for sentence beginning compared to NAIVE-ATT.

5 Related Work

Alignment induction from RNNSearch (Bahdanau et al., 2015) has been explored by a number of works. Bahdanau et al. (2015) are the first to show word alignment example using attention in RNNSearch. Ghader and Monz (2017) further demonstrate that the RNN-based NMT system achieves comparable alignment performance to that of GIZA++. Alignment has also been used to improve NMT performance, especially in low resource settings, by supervising the attention mechanisms of RNNSearch (Chen et al., 2016; Liu et al., 2016; Alkhouli and Ney, 2017).

There is also a number of other studies that induce word alignment from Transformer. Li et al. (2019); Ding et al. (2019) claim that attention may not capture word alignment in Transformer, and propose to induce word alignment with prediction difference (Li et al., 2019) or gradient-based measures (Ding et al., 2019). Zenkel et al. (2019) modify the Transformer architecture for better align-



(a) NAIVE-ATT (b) SHIFT-ATT

Figure 4: AER on the test set v.s. BLEU on the validation set on the de→en translation, evaluated with different checkpoints.

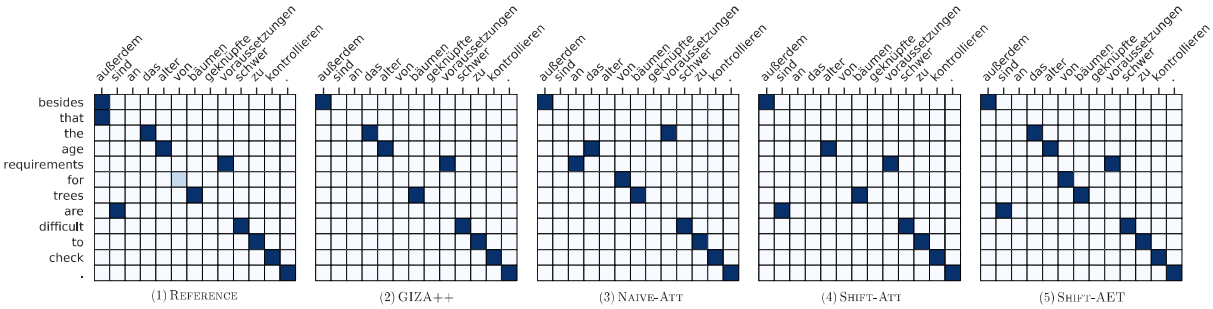


Figure 5: One example from the de-en alignment test set. Golden alignments are shown in (1), blue squares and light blue squares represent *sure* and *possible* alignments separately.

ment induction by adding an extra alignment module that is restricted to attend solely on the encoder information to predict the next word. Garg et al. (2019) propose a multi-task learning framework to improve word alignment induction without decreasing translation quality, by supervising one attention head at the penultimate layer with GIZA++ alignments. Although these methods are reported to improve over head average baseline, they ignore that better alignments can be induced by computing alignment scores at the decoding step when the to-be-aligned target token is the decoder input.

6 Conclusion

In this paper, we have presented two novel methods SHIFT-ATT and SHIFT-AET for word alignment induction. Both methods induce alignments at the step when the to-be-aligned target token is the decoder input rather than the decoder output as in previous work. Experiments on three public alignment datasets and a downstream task prove the effectiveness of these two methods. SHIFT-AET further extends Transformer with an additional alignment module, which consistently outperforms prior neural aligners and GIZA++, without influencing the translation quality. To the best of our knowledge, it reaches the new state-of-the-art performance among all neural alignment induction methods. We leave it for future work to extend our study to more downstream tasks and systems.

Acknowledgments

This work was supported by the National Key R&D Program of China (No. 2018YFB1005103), National Natural Science Foundation of China (No. 61925601), the Fundamental Research Funds for the Central Universities and the funds of Beijing Advanced Innovation Center for Language Resources (No. TYZ19005). We thank the anonymous reviewers for their insightful feedback on this work.

References

Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney. 2018. On the alignment problem in multi-head attention-based neural machine translation. In *Pro-*

ceedings of the Third Conference on Machine Translation: Research Papers, pages 177–185, Belgium, Brussels. Association for Computational Linguistics.

Tamer Alkhouli and Hermann Ney. 2017. Biasing attention-based recurrent neural networks using external alignment information. In *Proceedings of the Second Conference on Machine Translation*, pages 108–117.

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

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311.

Gino Brunner, Yang Liu, Damián Pascual, Oliver Richter, Massimiliano Ciaramita, and Rogert Wattenhofer. 2019. On identifiability in transformers. *arXiv e-prints*.

Guanhua Chen, Yun Chen, Yong Wang, and Victor O.K. Li. 2020. Lexical-constraint-aware neural machine translation via data augmentation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3587–3593. International Joint Conferences on Artificial Intelligence Organization. Main track.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. *AMTA 2016, Vol.*, page 121.

Yong Cheng, Shiqi Shen, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Agreement-based joint training for bidirectional attention-based neural machine translation. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2761–2767. AAAI Press.

Ido Dagan, Kenneth Church, and Willian Gale. 1993. Robust bilingual word alignment for machine aided translation. In *VERY LARGE CORPORA: ACADEMIC AND INDUSTRIAL PERSPECTIVES*.

Shuoyang Ding, Hainan Xu, and Philipp Koehn. 2019. Saliency-driven word alignment interpretation for neural machine translation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 1–12, Florence, Italy. Association for Computational Linguistics.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada. Association for Computational Linguistics.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4452–4461, Hong Kong, China. Association for Computational Linguistics.

Hamidreza Ghader and Christof Monz. 2017. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 30–39, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural machine translation decoding with terminology constraints. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 506–512, New Orleans, Louisiana. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 388–395.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *International Workshop on Spoken Language Translation (IWSLT) 2005*.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of*

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 681–691, San Diego, California. Association for Computational Linguistics.

Xintong Li, Guanlin Li, Lemao Liu, Max Meng, and Shuming Shi. 2019. On the word alignment from neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1293–1303, Florence, Italy. Association for Computational Linguistics.

Lemao Liu, Masao Utiyama, Andrew Finch, and Eiichiro Sumita. 2016. Neural machine translation with supervised attention. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3093–3102.

Yang Liu and Maosong Sun. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2295–2301.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 440–447. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Jan-Thorsten Peter, Arne Nix, and Hermann Ney. 2017. Generating alignments using target foresight in attention-based neural machine translation. *The Prague Bulletin of Mathematical Linguistics*, 108:27–36.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Kai Song, Kun Wang, Heng Yu, Yue Zhang, Zhongqiang Huang, Weihua Luo, Xiangyu Duan, and Min Zhang. 2020. Alignment-enhanced transformer for constraining nmt with pre-specified translations. In *AAAI*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, Berlin, Germany. Association for Computational Linguistics.

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

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *arXiv preprint arXiv:1901.11359*.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. End-to-end neural word alignment outperforms GIZA++. In *ACL*.