# Investigation of Transformer-based Latent Attention Models for Neural Machine Translation

**Parnia Bahar**[1,2] bahar@cs.rwth-aachen.de
**Nikita Makarov**[1] nikita.makarov@rwth-aachen.de
**Hermann Ney**[1,2] ney@cs.rwth-aachen.de
[1]Human Language Technology and Pattern Recognition Group, Computer Science Department, RWTH Aachen University, 52074 Aachen, Germany
[2]AppTek GmbH, 52062 Aachen, Germany

**Abstract**

Current neural translation networks are based on an effective attention mechanism that can be considered as an implicit probabilistic notion of alignment. Such architectures do not guarantee a high quality alignment, even though alignments can easily be used for explainable machine translation. This work describes a latent variable attention model using the transformer architecture, where we carry out an approximate marginalization over alignments. We show that the alignment quality in transformer models can be improved by introducing a latent variable for the alignments. To study the effect of the latent model, we quantitatively and qualitatively analyze the extracted alignments from the multi-head attention. We demonstrate that this method slightly improves translation quality on four WMT 2018 shared translation tasks, as well as generating more focused alignments for better interpretability.

## 1 Introduction

Current state-of-the-art neural machine translation (NMT) systems are based on attention models (Bahdanau et al., 2015; Luong et al., 2015; Gehring et al., 2017; Vaswani et al., 2017), where "soft attention" is used to focus on the most relevant parts of the source sequence while generating target words. This distribution can be considered as an implicit probabilistic notion of alignment as an intermediate step of the translation model. However, it does not work in the same way as its analogous alignment in conventional statistical machine translation (SMT) (Koehn and Knowles, 2017).

Contrary to SMT, where a "hard alignment" is a transparent process that defines the correspondence between source and target words, it is often challenging for the fuzzy attention mechanism to extract a comprehensible alignment. Although the context vector, a weighted sum of the input states, includes the alignment information implicitly, the alignment model does not directly influence the final translation probability of a sentence. Hence, it is often hard to determine the contribution of the attention on the output. In this work,

- we incorporate the attention model into the direct hidden Markov model (HMM) formulation in the transformer architecture such that the translation posterior distribution has a direct dependency on the source positions.
- as to the interest in alignment based approaches, we intend to investigate whether the latent-based models are able to tackle the explainability problem of alignments especially

for the transformer model. Similar to how some approaches in explainable NMT use the attention (Stahlberg et al., 2018; Zenkel et al., 2019; Garg et al., 2019), we also incorporate it as a key interpretation component into our work.

From the statistical perspective, we revisit the formulation of the posterior probability of a given sentence by computing a separate translation score for every target-source word pair. Since marginalization is exponential in the order of the model, we only explore a zero-order assumption, i.e. there is no dependence between subsequent alignments. The marginalization over the latent variable becomes simple and efficient for zero-order models and can be easily applied in both training and decoding. Unlike higher order dependence (Alkhouli et al., 2016, 2018; Wang et al., 2018), no dynamic programming and no search for the alignment path is required, thus a simple beam search decoder is used. We also address the theoretical connection between our approach and IBM model 2 (Brown et al., 1993). Tackling the costly computation of the softmax for all source positions, we employ an approximation by taking the topK relevant positions. To the best of our knowledge, this work explores the first instance of a zero-order latent variable alignments using the transformer architecture in an end-to-end fashion for machine translation.

## 2 Related Works

Similar to IBM and HMM alignments (Brown et al., 1993; Vogel et al., 1996), there are stand-alone neural approaches based on introducing word alignments as hidden variables, such that either a full sum over the alignment path or a maximum approximation (Viterbi word alignment) is used. These models decompose the translation process into two parts, namely the alignment and lexicon models. Unlike (Wang et al., 2018, 2017) where they use two separate networks, one for the alignment and one for the lexicon model and they need to jointly train two networks batch-wise, our work is still a single network trained end-to-end similar to the transformer model. Due to first-order dependence, Wang et al. (2018) also apply the forward-backward algorithm to compute the posterior probabilities as true labels, which is not required in our model. Alkhouli et al. (2016) apply explicit hard alignments to generate a translation using a maximum approximation. Alkhouli and Ney (2017) compute attention weights using pre-defined alignments to bias soft attention toward hard alignments. Yang et al. (2013) use lexical and alignment models, where alignments have no dependence on the lexical context. Tamura et al. (2014) also introduce a lexicalized neural alignment model to generate word alignments for the phrase-based translation system. In this work, we use neither pre-computed alignments nor a phrase table.

As an extension to neural alignment models, Alkhouli et al. (2018) use the transformer architecture to train both alignment and lexicon models while augmenting the multi-head attention component with additional hard alignment information. In their work, they train two separate networks and combine them in decoding with a maximum approximation. In contrast, we train a single network and sum over source positions.

Inspired by the success of statistical alignment models, Tu et al. (2016); Cohn et al. (2016) propose fertility and coverage concepts by including dependence on previous attention weights. Zenkel et al. (2019) try to gain high quality alignments by recomputing the attention heads after each target word prediction. Peter et al. (2017); Li et al. (2018) employ the full target context to improve alignment accuracy. Given statistical alignments obtained from Giza++ toolkit (Och and Ney, 2003), there are some methods in which the soft attention weights are supervised to behave similarly to the conventional hard alignments. Chen et al. (2016); Mi et al. (2016); Liu et al. (2016); Garg et al. (2019) add an additional training objective term that is dependent on the pre-computed alignments like Giza++ to guide neural models in training.

Instead of an objective function, Alkhouli and Ney (2017) modify the attention energy

computation directly to have dependence on the Giza++ alignment and perform beam search over both the lexical and alignment hypotheses. Unlike the aforementioned methods, our approach does not require any pre-computed alignments, and we do not need to search for the alignment path. It means our zero-order model can be easily applied in both training and decoding.

There are also a number of recent works in which attention is used as a latent alignment by decomposing the joint distribution between an alignment model and a lexicon model (Wu et al., 2018; Shankar et al., 2018; Bahar et al., 2020). They mainly use a recurrent neural network (RNN) parameterization. The main difference of our work, is that we explore the use of transformer models. They either use character-level output or conduct the experiments on a small training set where the vocabularies appear to be small, whilst we evaluate our experiments using relatively large vocabularies. Shankar and Sarawagi (2019) extend the prior into an explicit posterior attention distribution. Deng et al. (2018) propose a non-differentiable approach, "hard attention", where an explicit dependence of a single input to output is taken by the REINFORCE algorithm (Xu et al., 2015).

In this work, we explore the soft attention as a latent variable alignment to directly calculate the posterior probability of a sentence. Our idea is a straight-forward application of an under-used idea from IBM word-based translation modeling, and makes clever use of existing transformer components to avoid adding parameters to the model. We avoid the temptation to create a separate distortion model, and just re-use attention heads. The solution is clean and probabilistically sound.

## 3 Background

In machine translation, given a source sequence $f_1^J = f_1, \ldots, f_j, \ldots, f_J$ of length $J$, with $f_j$ being a source word, the posterior probability of a target sequence $e_1^I = e_1, \ldots, e_i, \ldots, e_I$ of length $I$, is defined as $p(e_1^I|f_1^J)$. This conditional distribution, the so-called translation model, covers the alignment information between source and target words either implicitly or explicitly.

In most NMT methods, including both the recurrent-based attention (Bahdanau et al., 2015) and the transformer (Vaswani et al., 2017), the encoder scans the entire source sequence and generates a sequence of vector representation. Then, the decoder generates an output sequence conditioned on the precomputed encoder states. The posterior probability distribution of the target sequence is conditioned on the source $f_1^J$ and the target history $e_1^{i-1}$, as:

$$p(e_1^I|f_1^J) = \prod_{i=1}^{I} p(e_i|e_1^{i-1}, f_1^J) = \prod_{i=1}^{I} p\Big(e_i|e_1^{i-1}, c_i\big(e_1^{i-1}, h_1^J(f_1^J)\big)\Big) \tag{1}$$

Usually, the dependence on $f_1^J$ is modelled using a soft attention mechanism, where a weighted sum $c_i$ of the input representations $h_1^J$ is used inside a neural network layer. Thus, it is important to observe that there is no direct dependence on source position in Equation 1. In this equation, $h_1^J$ is the encoder's output, and $c_i$ is the output of an attention approach at target position $i$, summarizing the alignment information implicitly.

## 4 Latent Attention Model

Similar to (Wang et al., 2018; Wu et al., 2018; Shankar et al., 2018; Bahar et al., 2020), we introduce a word alignment as a sequence of latent variables that establishes a mapping from target position $i$ to source position $j$, i.e. $i \rightarrow b_i = j$ and decompose the posterior probability distribution of target sequence into two parts: alignment and lexicon models. To do so, we marginalize out all possible alignments in Equation 2. $\sum_{b_1^I}$ considers all possible alignment paths, such that $b_i$ aligns the target position $i$ to the source position $j$. Assuming the zero-order Markov assumption to simplify the dependencies of the alignment sequences, we decompose the

posterior probability into alignment and lexicon models, obtaining Equation 4. Here, we note that, assuming a first-order assumption leads to the first-order hidden Markov model. Performing the sum over all alignments would lead to a combinatorial problem. Due to the zero-order dependence, one can re-arrange the sum and the product in Equation 5 by the distributive property that leads to polynomial complexity. The mathematical proof of swapping between the sum and product can be found in (Brown et al., 1993).

Since the model combines a mixture of softmaxes, the exact marginalization over all source positions is infeasible for a large vocabulary. Hence, we choose the topK source positions with the highest alignment probabilities and compute the corresponding lexicon scores. By doing so, the computational complexity reduces to $\mathcal{O}(I \times K)$ (see Equation 6). We differentiate the topK approximation by the straight-through estimator (Bengio et al., 2013). We found empirically that $K = 6$ is enough and we only get marginal benefits for $K > 6$ (see Section 7.4 for more details).

$$p(e_1^I|f_1^J) = \sum_{b_1^I} p(e_1^I, b_1^I|f_1^J) = \sum_{b_1^I} \prod_{i=1}^{I} p(e_i, b_i|e_1^{i-1}, b_1^{i-1}, f_1^J) \tag{2}$$

$$= \sum_{b_1^I} \prod_{i=1}^{I} \underbrace{p(b_i|e_1^{i-1}, b_1^{i-1}, f_1^J)}_{\text{alignment modeling}} \cdot \underbrace{p(e_i|e_1^{i-1}, b_1^i, f_1^J)}_{\text{lexicon modeling}}$$

(exponential complexity, $J^I$) (3)

$$= \sum_{b_i} \prod_{i=1}^{I} \underbrace{p(b_i|e_1^{i-1}, f_1^J)}_{\text{alignment model}} \cdot \underbrace{p(e_i|e_1^{i-1}, b_i, f_1^J)}_{\text{lexicon model}} \tag{4}$$

$$= \prod_{i=1}^{I} \sum_{j=1}^{J} p(b_i = j|e_1^{i-1}, f_1^J) \cdot p(e_i|e_1^{i-1}, j, f_1^J)$$

(polynomial complexity, $I \times J$) (5)

$$\approx \prod_{i=1}^{I} \sum_{j \in topK} p(b_i = j|e_1^{i-1}, f_1^J) \cdot p(e_i|e_1^{i-1}, j, f_1^J)$$

(polynomial complexity, $I \times K$) (6)

**Relation to IBM Model 2**

We note the similarity between Equation 5 and IBM model 2 (in Equation 7) as both are zero-order models with respect to the alignments, however, our model has a dependence on target history.

$$p(e_1^I|f_1^J) = \prod_{i=1}^{I} \sum_{j=1}^{J} \underbrace{p(j|i, J, I)}_{\text{alignment}} \cdot \underbrace{p(e_i|f_j)}_{\text{lexicon}} \tag{7}$$

## 5 Neural Parameterization

In order to parameterize the individual components of the model, we use a network similar to the multihead self-attentive transformer. In the transformer architecture both the encoder and decoder are composed of stacked layers. In the encoder, multihead self-attentive components followed by a feed forward layer are used to encode the entire source sequence, obtaining a sequence of encoder states $h_1^{(L_{enc})}, \ldots, h_j^{(L_{enc})}, \ldots, h_J^{(L_{enc})}$. Here, $L_{enc}$ is the number of encoder layers.

The decoder contains an additional multihead attention layer, incorporating the encoder states and the decoder state $s_{i-1}^{(l_{dec-1})}$ in each decoder layer $l_{dec}$. Several attention heads are

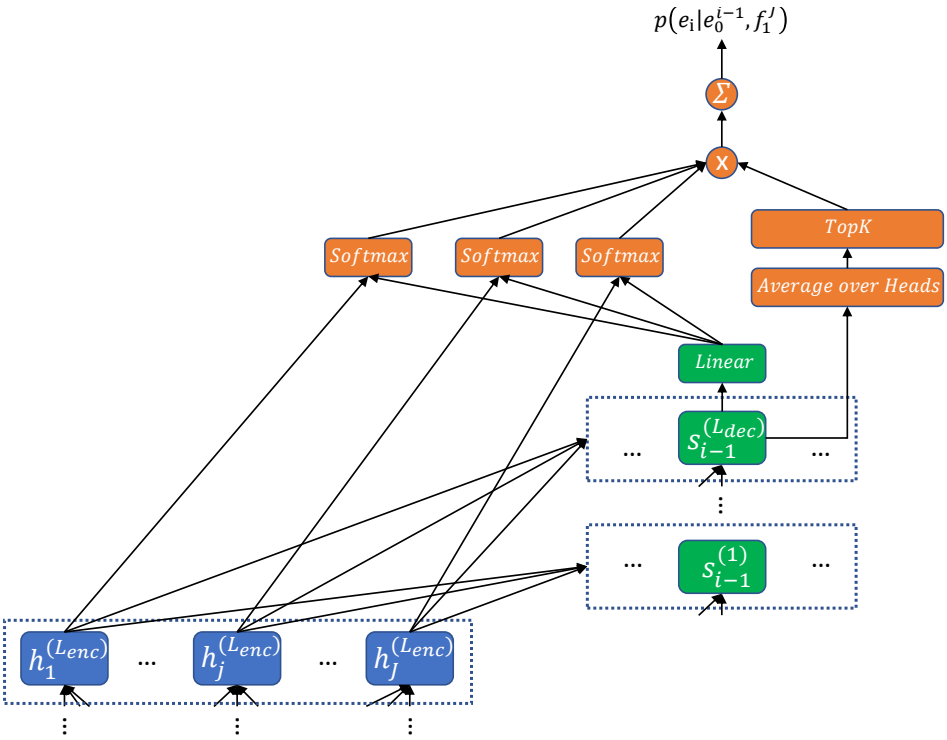



Figure 1: Overview of the transformer-based zero-order attention model. In the output layer, at each target step $i$, we add the decoder state $s_{i-1}^{(L_{dec})}$ with all the encoder state $h_1^J$ (or with topK approximation only the K best ones).

used to attend to the source positions. At every decoder layer, each attention head $n$, computes a normalized distribution over the source positions, $\alpha_n^{(l_{dec})}(j|i)$. These weights are used to compute the context vectors. The concatenation of these context vectors are fed through a feed-forward layer to form the decoder representation $s_{i-1}^{(l_{dec})}$. The representation from the last decoder layer is then passed through a softmax layer, which provides the final distribution over the vocabulary $p(e_i|f_1^J, e_1^{i-1})$. The latent attention model has two main components. Figure 1 shows an abstract of the model architecture.

**Lexicon Model:** The lexicon model is fairly straightforward to implement in the transformer. As shown in Figure 1, we combine the last decoder state of the top layer with the encoder states, as well as the last target word at each time step, Therefore, we have

$$p(e_i|e_1^{i-1}, j, f_1^J) = g\big(e_{i-1}, s_{i-1}^{(L_{dec})}, h_j^{(L_{enc})}\big) \tag{8}$$

where $g$ is a linear projection followed by the softmax. As seen in this equation, at each target step $i$, we add the decoder state $s_{i-1}$ with all the encoder state $h_1^J$ (or with topK approximation only the K best ones).

**Alignment Model:** On the other hand, the alignment model is not as trivial to implement. The first option is to use the existing attention weights of the transformer for the alignment, but the transformer has not only multiple attention layers, but also multiple heads per layer. The transformer's topology with multiple heads in a single layer makes the attention heads symmetrical, but the different layers might capture different alignment information (Garg et al.,

| | German | English | Chinese | English |
|---|---|---|---|---|
| Sentences | 5.9M | | 14.1M | |
| Running words | 160M | 157M | 314M | 427M |
| vocabularies | 45k | 33k | 38k | 32k |

Table 1: Corpus statistics.

2019). To form the alignment model, we average over the attention weights across all heads within each layer of the decoder at each time step, i.e.:

$$p^{(l_{dec})}(j|e_1^{i-1}, f_1^J) = \frac{1}{N}\sum_{n=1}^{N}\alpha_n^{(l_{dec})}(j|i) \tag{9}$$

where $N$ is the number of attention heads and $^{l_{dec}}$ is the decoder's $l$th layer. We have observed that this way the model has the ability to train the heads of selected layer (see Section 7.1). Other authors have found that the later attention layers of the transformer perform longer jumps and generally attend to more different source positions (Irie et al., 2019). Due to these properties and based on the results in Table 7.1, we focus our experiments on using the last layer's attention of the decoder, averaged over the heads, i.e.

$$p(j|e_1^{i-1}, f_1^J) = \frac{1}{N}\sum_{n=1}^{N}\alpha_n^{(L_{dec})}(j|i). \tag{10}$$

Once we compute the lexicon and alignment probabilities, we sum over $j$ positions as stated in Equation 6 with topK approximation. The model has almost the same number of parameters as our transformer baseline. A visualization of model architecture can be seen in Figure 1.

## 6 Experiments

We carry out the experiments on four WMT 2018 translation tasks[1]: German↔English and Chinese↔English. The corpora statistics are shown in Table 1. We do not employ any kind of synthetic or back-translated data in this work.

For German↔English, after tokenization and true-casing using the `Moses` toolkit (Koehn et al., 2007), we apply byte pair encoding (BPE) (Sennrich et al., 2016) with 50k merge operations. We use the original parallel data consisting of 5.9M sentence pairs (see Table 1).

For Chinese↔English, we apply sentence pieces model (SPM) (Kudo, 2018) with an n-best size of 30 and 32k merge operations. The original bilingual data consists of almost 26M samples. We filter out the noisy data by removing illegal characters and duplication, hence we end up with 14M pairs.

For De↔En and Zh↔En, we use the `newstest2015` and the `newsdev2017` as the development set respectively and the `newstest2017` and `newstest2018` as our test sets. The models are evaluated using case-sensitive BLEU (Papineni et al., 2002) computed by the official scripts of WMT campaign, i.e. `mteval-v13a`[2] and case-sensitive normalized TER (Snover et al., 2006) computed by `tercom`[3].

For our experiments, we train both the transformer and the RNN attention models. We follow a base transformer similar to (Vaswani et al., 2017) where we use 6 layers in both the

[1] http://statmt.org/wmt18/

[2] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v13a.pl

[3] http://www.cs.umd.edu/ snover/tercom/

encoder and the decoder with an internal dimension size of 512. We set the number of heads in the multi-head attention to 8. Layer normalization, dropout, and residual connections are applied. We use the same structure as our transformer baseline for the latent attention model with the almost the same number of parameters.

Given that the proposed latent method can be also applicable to the RNN attention architecture, we intend to apply this technique to the RNN attention to see if meaningful differences will be observed. However, we focus on our analysis of the difference between the transformer and the latent attention model. We train the RNN attention models similar to (Bahdanau et al., 2015), in which all words are projected into a $512$-dimensional embedding space. We use a 6-layer stacked bidirectional LSTM (BLSTM) (Hochreiter and Schmidhuber, 1997) with 1024 hidden cells to compute a sequence of encoder states $h_1^T$. The decoder is composed of one layer LSTM of size 1024 with an additive single-head attention layer with attention feedback (Bahar et al., 2017). In the RNN latent attention model, to form the alignment model, we use the attention weights itself, $\alpha(i|j)$, as our distribution, i.e.

$$p(j|e_1^{i-1}, f_1^J) = \alpha(i|j) = \nu^\top \tanh(h_j, s_{i-1}) \tag{11}$$

For the lexicon scores, similar to the transformer modeling, we combine the decoder state with the encoder states, as well as the last output token at each time step and write

$$p(e_i|e_1^{i-1}, j, f_1^J) = g(e_{i-1}, s_{i-1}, h_j) \tag{12}$$

We note the difference between equations 12 and 13 and direct dependency on source positions instead of a summary of all inputs as a single vector $c_i$. To compute the lexicon probabilities, we employ a softmax over the target vocabulary, $J$ times (or $K$ times using the approximation), while in the attention model, we only do it once as we summarize the encoder representations using the context vector $c_i$ without any dependence on $j$ positions.

$$p(e_i|e_1^{i-1}, f_1^J) = g(e_{i-1}, s_{i-1}, c_i) \tag{13}$$

For RNN-based models, we use a layerwise pre-training strategy (Zeyer et al., 2018) for the first epochs. We start using only the first layer in the encoder of the model and add new layers during the training progress. We observe that the layer-wise pretraining leads to a strong RNN attention baseline. The models are trained end-to-end using the Adam optimizer with a learning rate of $0.0003$, and a dropout of $10\%$ for the transformer-based systems, and a learning rate of $0.001$ and a dropout of $30\%$ for the RNN-based attention models. We employ a learning rate scheduling scheme, where we lower the learning rate with a decay factor of $0.9$ if the perplexity on the development set does not improve for several consecutive checkpoints. We remove sentences longer than $100$ tokens before batching them together. All batch sizes are specified to be as big as possible to fit in memory. A beam size of 12 is used in inference. We use our in-house implementation of sequence-to-sequence modeling in `RETURNN` (Zeyer et al., 2018). The code[4] and the configurations of the setups are available online[5].

## 7 Results

### 7.1 Averaging Attention Heads

Our latent model is flexible to learn one of the heads, all heads within a single layer or all heads across all layers. As described in Section 5, the transformer architecture leads to have

[4]https://github.com/rwth-i6/returnn

[5]https://github.com/rwth-i6/returnn-experiments/

| Layer | newsdev2017 | | newstest2017 | |
|---|---|---|---|---|
| | BLEU | TER | BLEU | TER |
| 1 | 21.9 | 62.7 | 23.4 | 61.2 |
| 2 | 22.5 | 61.9 | 23.8 | 60.6 |
| 3 | 22.6 | 61.9 | 23.8 | 60.6 |
| 4 | 21.9 | 62.6 | 23.8 | 60.8 |
| 5 | 22.5 | 62.0 | 24.2 | 60.3 |
| 6 | 22.6 | 61.9 | 24.2 | 60.5 |

Table 2: Results measured in BLEU [%] and TER [%] on the Zh→En task.

symmetrical attention heads within a layer, while different layers might learn different alignment patterns.

To examine which layer can be the most effective one for our latent attention model, we have chosen our alignment model obtained by layer-wise averaging of attention probabilities of various layers. Table 2 shows the BLEU and TER scores on the Zh→En task. As listed, for all selected layers, our model has the ability to learn a mixture of alignment and lexicon scores. For the rest of our experiments, we average over attention heads of the last layer (6th) for the latent attention model.

### 7.2 Translation Performance

In the second set of experiments, we compare the latent attention model with two baselines, both the transformer baseline and the RNN attention baseline. We additionally apply the RNN latent attention model on De→En and Zh→En tasks to see if the same behaviour will be observed. The results can be seen in Table 3. As shown, the latent attention model provides a small boost of $0.5\%$ BLEU and $0.2\%$ TER on Zh→En, $0.7\%$ BLEU and $0.6\%$ TER on En→Zh, $0.2\%$ BLEU and $0.2\%$ TER on De→En and $0.1\%$ BLEU and $0.4\%$ TER on En→De, on average. As expected, the transformer models give a better performance compared to the RNN attention models. Comparing the RNN latent attention model with its corresponding baseline shows very similar, yet more mixed results.

In theory, the latent model has more statistical capacity than the attention-based NMT. The attention-based NMT models are a deterministic interpolation of deterministic features (source encodings), whereas the latent attention model is a mixture model. With attention, an NMT system outputs exactly one distribution over target vocabulary per time step, whilst our model outputs $J$ (or $K$ when using the optimization) distributions, which are mixed probabilistically, yielding a multimodal marginal distribution. Our interpretation is that equipping a deterministic soft function with a probabilistic model might lead to better results. In this case, alignments have direct effects on the final translation scores and the lexicalized alignment model assigns more appropriate scores for the translation of target-source word pair.

In the third set of our evaluation, we compare our model with similar previous approaches. The only instance of latent model using the transformer architecture is (Alkhouli et al., 2018) where a maximum approximation is used instead of summation. Moreover, in their work, they train two independent networks whilst in our approach a single network is trained. Their model depends on pre-define alignments using GIZA++ (Och and Ney, 2003) and they hypothesis over alignments in search, whereas we do not need these steps. For De→En, our model outperforms theirs on `newstest2017` by 1.6% BLEU. We also compare with (Wang et al., 2018) where they have a full summation, however, a first-order assumption is employed. Our model consistently outperforms theirs on three tasks. We note that firstly their model is based on the RNN attention and secondly the number of parameters of their network is half of those of ours.

| Task | Model | newstest2017 | | newstest2018 | |
|---|---|---|---|---|---|
| | | BLEU | TER | BLEU | TER |
| Chinese→English | (Wang et al., 2018) | 20.2 | 63.7 | - | - |
| | transformer | 24.1 | 60.7 | 23.7 | - |
| | transformer latent attention | **24.2** | **60.5** | **24.5** | - |
| | RNN attention | 22.9 | 62.0 | 22.7 | - |
| | RNN latent attention | 22.6 | 61.2 | 23.0 | - |
| English→Chinese | transformer | 32.0 | 57.3 | 32.8 | - |
| | transformer latent attention | **32.9** | **56.7** | **33.3** | - |
| | RNN attention | 31.8 | 57.2 | 32.5 | - |
| German→English | (Alkhouli et al., 2018) | 32.1 | - | - | - |
| | (Wang et al., 2018) | 29.6 | - | - | - |
| | transformer | 33.5 | 55.1 | 40.4 | 46.8 |
| | transformer latent attention | **33.7** | **54.9** | **40.5** | **46.7** |
| | RNN attention | 32.6 | 54.9 | 39.6 | 46.8 |
| | RNN latent attention | 32.8 | 55.3 | 39.2 | 47.6 |
| English→German | (Wang et al., 2018) | 24.6 | - | - | - |
| | transformer | **26.5** | 64.8 | 39.0 | 50.7 |
| | transformer latent attention | **26.5** | 64.7 | **39.2** | **50.1** |
| | RNN attention | 26.3 | **64.4** | 38.9 | 50.7 |

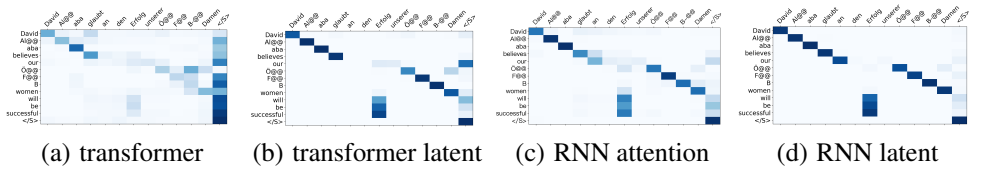
Table 3: Results measured in BLEU [%] and TER [%] on the test sets. The TER computation on newstest2018 fails for Chinese sets.

### 7.3 Attention Analysis

As stated in the introduction, we are interested in seeing how the attention weights in the transformer could be interpreted. As the model utilizes multi-head attention, we use Equation 10 to average over the attention heads to get a single score at each time step. The effect that we have observed, which is also reported by others (Alkhouli et al., 2018; Zenkel et al., 2019; Garg et al., 2019), is that the transformer heavily attends to the frequent words like end of sentence (EOS) token, often having the highest average attention value of the sequence, as seen in the examples in Figure 2(a) and 2(e). This effect leads to worse interpretation, as well as possibly worse performance since a large portion of the processing capabilities are spent on a token with limited information value. However, the intuition behind the original multi-head attention architecture is that it learns different alignment information, if one head learns bad alignment, other heads may learn good ones.

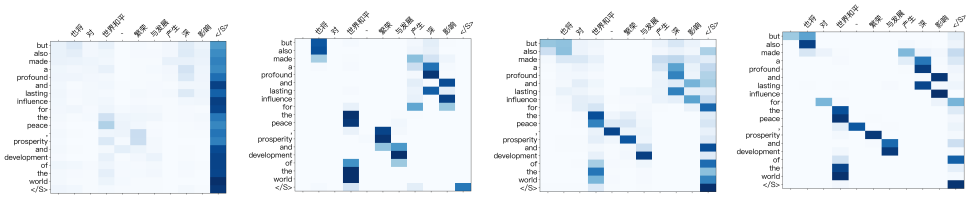
In our figures and experiments, we add in RNN attention models as well as the latent model with RNN attention architecture to verify that the attention issues are specific to the transformer and are not inherent to the soft attention mechanism. Similar to previous observations, the RNN attention performs relatively consistently across the datasets, with generally sharper attention than the transformer baseline. As our work was motivated by the drawbacks of the transformer's attention, we focus on our analysis of the difference between the transformer and the latent attention model.

**Alignment Error Rate** A large factor of the motivation for the attention latent model is that the attention distributions can be interpreted as alignments. We assume that the latent attention model might produce higher quality alignments. To verify this assumption, we use the RWTH German-English Golden Alignments which provides 504 manually word-aligned sentence pairs extracted from the Europarl corpus (Vilar et al., 2006). We compare the alignment error rate



(a) transformer (b) transformer latent (c) RNN attention (d) RNN latent

De→En examples



(e) transformer (f) transformer latent (g) RNN attention (h) RNN latent

Zh→En examples

Figure 2: Comparison of attention distributions on different tasks. On the x-axis is the source and on the y-axis is the target.

| Model | Europarl AER[%] | |
|---|---|---|
| | last heads | all heads |
| transformer | 71.4 | 57.1 |
| transformer latent attention | 57.8 | 54.8 |
| RNN attention | 51.5 | |
| RNN latent attention | 49.0 | |

Table 4: AER [%] of different models, while the EOS and the full stop have been ignored.

(AER) of the models as a key quantitative metrics for the analysis of alignments in Table 4. To calculate the alignments for the transformer-based models, we use the maximum of the attention weights as the alignment at each time step. We interpret the alignments from the transformer baseline and the latent attention model by a) averaging over the attention heads in the last layer, and b) averaging over the attention heads of the all layers and then taking the maximum value to determine which source token corresponds to each target token.

It is important to highlight that the references are on a word level whereas the alignments from these models are sub-word based. To make the AER more comparable with traditional systems, we merge the BPEs together, where we use the average of all tokens. We also note that, the transformer baseline attends many of its heads to the most frequent tokens (e.g. EOS token, and the full stop token). To mitigate this effect on the AER, we ignore these two positions for our computations. The results show that the transformer performs significantly worse than the attention latent model on the last layer, being 71.4% vs 57.8%. On the other hand, averaging over all layers considerably improves the AER for the transformer to 57.1%, whilst only marginally improving for the latent attention model to 54.8%. This improvement implies that the transformer performs the most interpretable attention not at the last layer, but at intermediate layers. The finding is supported by the observation of other groups as well (Voita et al., 2019). The RNN Attention model performs better than both other approaches at

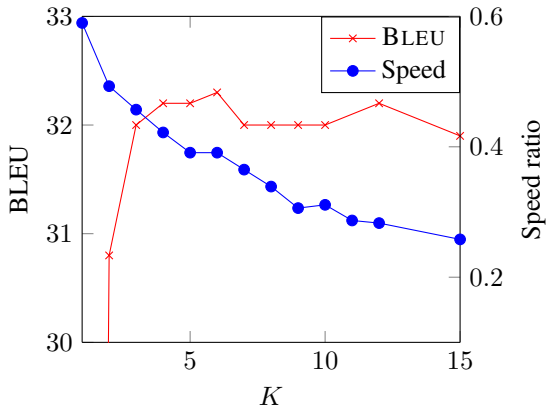



Figure 3: Speed and translation quality in BLEU vs. different values of $K$ on the development set (`newstest2015`) of the De→En task.

51.5%, signifying that the transformer's attention is still far away from being easily interpreted as alignments. The RNN latent model is also superior its baseline by 1.5% in AER. The alignment quality of the proposed model is poorer compared to conventional alignment models such as IBM Models or other works in the literature (Zenkel et al., 2019; Garg et al., 2019). The main reason is that in our model, we do not have full dependence on the whole target sequence in contrast to the conventional alignment models neither on the current target word $e_i$. Since, we do left-to-right decoding, we keep the dependence only on the predecessor words for an efficient and simple search.

**Qualitative Attention Analysis** A large factor of the motivation for the attention latent model is that the attention distributions can be interpreted as alignments. Based on the quantitative results, we extract some examples from `newstest2018` sets of different tasks of where the latent attention models performs better than their corresponding baseline. In Figures 2(a)-2(h), we see the examples side by side where the transformer attends to the EOS token, whereas the latent attention model has sharper attention distributions. An interesting component of these examples is that the latent attention model correctly attends to non-monotonic sub-sequences. The same behaviour can be seen for the RNN-based models, where the latent model results in a sharper distribution (cf. Figures 2(c) and 2(d) for De→En and Figures 2(g) and 2(h) for Zh→En).

Although in our model, the alignment's quality gets better both qualitatively and quantitatively, the final translation seems to be comparative with our baselines. The improvement in alignment quality does not carry over to better translations, as it is not surprising given prior work in MT (Koehn and Knowles, 2017). Thus, one important question still remains and that whether a high quality alignment is relevant and informative for the final translation quality?

### 7.4 Effect of K

We also plot the effect of $K$ applied in the topK approximation versus the corresponding BLEU score, as well as the speed ratio in Figure 3 on De→En development set. As it is shown, the BLEU score goes up from 30.8 to 32.3 by changing $K$ from 2 to 6 respectively, and then it statures without any further significant improvements. This implies that small values of $K$ are good enough for training, and we use values around $K = 6$ for our other training runs. We also show the speed effect with respect to $K$. Here, the speed ratio is defined as a proportion between

| Model | $K$=1 | $K$=2 | $K$=3 | $K$=6 | $K$=10 | all |
|---|---|---|---|---|---|---|
| latent attention | 30.7 | 31.9 | 32.2 | 32.2 | 32.2 | 32.2 |

Table 5: Results measured in BLEU [%] on the development set (`newstest2015`) with respect to different values of $K$ during inference. The latent attention model has been trained using $K = 6$.

the speed of the transformer baseline and of the latent model with different $K$ values on average of sub-epochs. Intuitively, as $K$ increases, the training time goes up. But it is not linear with respect to $K$.

Similar to (Shankar et al., 2018), we also investigate whether the gain is due to the softmax bottleneck as well as to check if the larger values of $K$ improves the performance during inference. To do so, we train the model with $K = 6$ and deploy it using different $K$ values in inference. The results are listed in Table 5. The output has only a single softmax vector, when $K = 1$. Therefore, we assume that the latent attention model encounters the same bottleneck as the transformer. As expected, the performance drops compared to the case of $K = 6$, which means that the model gives some gains behind the ensembling. Moreover, good performance requires $K = 6$ and we do not get benefit for $K > 6$. The exact marginalization does not help as well. Applying the same $K$ value in both training and inference meets the model's requirements.

### 7.5 Complexity

In order to compute the lexicon scores, we employ a softmax over the target vocabulary $V$, $K$ times. The time complexity of the latent model is $\mathcal{O}(I \times K \times V)$ compared to $\mathcal{O}(I \times V)$ for the transformer. Thus, the latent attention model can become very slow for a large vocabulary. Using GPU and optimized matrix operations, the computational cost is significantly compensated by parallelization as long as everything fits in memory. The model is about 2.7 times slower than the transformer on a single GPU. Moreover, the space complexity of the model on the output layer is $\mathcal{O}(K \times V)$, rather than $\mathcal{O}(V)$ for the baseline, which requires to decrease the batch size. This concludes the necessity of multiple GPU training.

## 8 Conclusion and Future Work

We have investigated the use of a zero-order latent variable attention model based on both the transformer and the RNN attention architecture. The restructured model makes use of a lexicalized alignment with the aim to have more focused attention weights, leading to better explainability. Our results on four WMT 2018 translation tasks show that the model slightly outperforms over the transformer model on average over all tasks. These are the first experiments using the latent variable model on both the transformer, as well as larger vocabulary tasks in comparison to previously conducted experiments. We also believe that due to the highly similar composition of our models to previously done latent attention experiments, we can conclude that their approaches would face the same results as shown here, if applied to the transformer.

As future work, we plan to investigate the model with more elaborate dependencies, as well as further exploration on first-order dependence with respect to the alignments. Additionally, we intend to approximate the softmax for large vocabularies to further speed up the latent method. We also would like to explore how the transformer performs so well, even though it attends so much to the EOS token. Finally, we want to survey further statistical modelling approaches that improve the explainability of the transformer model.

## References


Alkhouli, T., Bretschner, G., and Ney, H. (2018). On the alignment problem in multi-head attention-based neural machine translation. In *The Third Conference on Machine Translation, WMT, Belgium, Brussels, October 31 - November 1*, pages 177–185.

Alkhouli, T., Bretschner, G., Peter, J., Hethnawi, M., Guta, A., and Ney, H. (2016). Alignment-based neural machine translation. In *The First Conference on Machine Translation, WMT, Berlin, Germany, August 11-12*, pages 54–65.

Alkhouli, T. and Ney, H. (2017). Biasing attention-based recurrent neural networks using external alignment information. In *The Second Conference on Machine Translation, WMT, Copenhagen, Denmark, September 7-8*, pages 108–117.

Bahar, P., Makarov, N., Zeyer, A., Schlüter, R., and Ney, H. (2020). Exploring a zero-order direct hmm based on latent attention for automatic speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 7854–7858, Barcelona, Spain.

Bahar, P., Rosendahl, J., Rossenbach, N., and Ney, H. (2017). The rwth aachen machine translation systems for iwslt 2017. In *International Workshop on Spoken Language Translation*, pages 29–34, Tokyo, Japan.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations (ICLR)*.

Bengio, Y., Léonard, N., and Courville, A. C. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Brown, P. F., Pietra, S. D., Pietra, V. J. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Chen, W., Matusov, E., Khadivi, S., and Peter, J. (2016). Guided alignment training for topic-aware neural machine translation. *CoRR*, abs/1607.01628.

Cohn, T., Hoang, C. D. V., Vymolova, E., Yao, K., Dyer, C., and Haffari, G. (2016). Incorporating structural alignment biases into an attentional neural translation model. In *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies NAACL HLT, San Diego, USA, June 12-17*, pages 876–885.

Deng, Y., Kim, Y., Chiu, J., Guo, D., and Rush, A. M. (2018). Latent alignment and variational attention. In *Advances in Neural Information Processing Systems 31: Neural Information Processing Systems, NeurIPS, 3-8 December, Montréal, Canada.*, pages 9735–9747.

Garg, S., Peitz, S., Nallasamy, U., and Paulik, M. (2019). Jointly learning to align and translate with transformer models. In *The 2019 Conference on Empirical Methods in Natural Language Processing EMNLP, Hong Kong*.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML, Sydney, NSW, Australia, 6-11 August 2017*, pages 1243–1252.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Irie, K., Zeyer, A., Schlüter, R., and Ney, H. (2019). Language modeling with deep transformers. In Kubin, G. and Kacic, Z., editors, *The 20th Annual Conference of the International Speech Communication Association (Interspeech), Graz, Austria, 15-19 September*, pages 3905–3909.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *The 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180.

Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *The First Workshop on Neural Machine Translation, NMT@ACL, Vancouver, Canada, August 4*, pages 28–39.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In *The 56th Annual Meeting of the Association for Computational Linguistics*, pages 66–75, Melbourne, Australia.

Li, X., Liu, L., Tu, Z., Shi, S., and Meng, M. (2018). Target foresight based attention for neural machine translation. In *2018 Conference of the North American Chapter of the Association for Computational Linguistics, NAACL-HLT, New Orleans, USA, June 1-6*, pages 1380–1390.

Liu, L., Utiyama, M., Finch, A. M., and Sumita, E. (2016). Neural machine translation with supervised attention. In *The 26th International Conference on Computational Linguistics Proceedings of the Conference COLING, Osaka, Japan, December 11-16*, pages 3093–3102.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Mi, H., Wang, Z., and Ittycheriah, A. (2016). Supervised attentions for neural machine translation. In *2016 Conference on Empirical Methods in Natural Language Processing*, pages 2283–2288, Austin, Texas.

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

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a Method for Automatic Evaluation of Machine Translation. In *The 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Peter, J., Nix, A., and Ney, H. (2017). Generating alignments using target foresight in attention-based neural machine translation. *Prague Bull. Math. Linguistics*, 108:27–36.

Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *The 54th Annual Meeting of the Association for Computational Linguistics ACL, Berlin, Germany, August 7-12*.

Shankar, S., Garg, S., and Sarawagi, S. (2018). Surprisingly easy hard-attention for sequence to sequence learning. In *The 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, October 31 - November 4*, pages 640–645.

Shankar, S. and Sarawagi, S. (2019). Posterior attention models for sequence to sequence learning.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A Study of Translation Edit Rate with Targeted Human Annotation. In *The 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Stahlberg, F., Saunders, D., and Byrne, B. (2018). An operation sequence model for explainable neural machine translation. In *The Workshop: Analyzing and Interpreting Neural Networks for NLP, Brussels, Belgium, November 1*, pages 175–186.

Tamura, A., Watanabe, T., and Sumita, E. (2014). Recurrent neural networks for word alignment model. In *The 52nd Annual Meeting of the Association for Computational Linguistics, ACL, Baltimore, USA, June 22-27*, pages 1470–1480.

Tu, Z., Lu, Z., Liu, Y., Liu, X., and Li, H. (2016). Modeling coverage for neural machine translation. In *54th Annual Meeting of the Association for Computational Linguistics ACL, Aug. 7-12, Berlin, Germany*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30: Neural Information Processing Systems, 4-9 December 2017, Long Beach, USA*, pages 6000–6010.

Vilar, D., Popovic, M., and Ney, H. (2006). Aer: Do we need to "improve" our alignments? In *International Workshop on Spoken Language Translation*, pages 205–212, Kyoto, Japan.

Vogel, S., Ney, H., and Tillmann, C. (1996). Hmm-based word alignment in statistical translation. In *16th International Conference on Computational Linguistics, Proceedings of the Conference, COLING, Copenhagen, Denmark, August 5-9*, pages 836–841.

Voita, E., Talbot, D., Moiseev, F., Sennrich, R., and Titov, I. (2019). Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *The 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy.

Wang, W., Alkhouli, T., Zhu, D., and Ney, H. (2017). Hybrid neural network alignment and lexicon model in direct hmm for statistical machine translation. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 125–131, Vancouver, Canada.

Wang, W., Zhu, D., Alkhouli, T., Gan, Z., and Ney, H. (2018). Neural hidden markov model for machine translation. In *The 56th Annual Meeting of the Association for Computational Linguistics ACL, Melbourne, Australia , July 15-20*.

Wu, S., Shapiro, P., and Cotterell, R. (2018). Hard non-monotonic attention for character-level transduction. In *The 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), Brussels, Belgium, October 31 - November 4*, pages 4425–4438.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *The 32nd International Conference on Machine Learning, ICML, Lille, France, 6-11 July*, pages 2048–2057.

Yang, N., Liu, S., Li, M., Zhou, M., and Yu, N. (2013). Word alignment modeling with context dependent deep neural network. In *The 51st Annual Meeting of the Association for Computational Linguistics ACL, Sofia, Bulgaria, 4-9 August*, pages 166–175.

Zenkel, T., Wuebker, J., and DeNero, J. (2019). Adding interpretable attention to neural translation models improves word alignment. *CoRR*, abs/1901.11359.

Zeyer, A., Alkhouli, T., and Ney, H. (2018). RETURNN as a generic flexible neural toolkit with application to translation and speech recognition. In *Proceedings of ACL, Melbourne, Australia, July 15-20*, pages 128–133.