On The Alignment Problem In Multi-Head Attention-Based Neural Machine Translation

Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney
Human Language Technology and Pattern Recognition Group
Computer Science Department
RWTH Aachen University
D-52056 Aachen, Germany
`<surname>@i6.informatik.rwth-aachen.de`

Abstract

This work investigates the alignment problem in state-of-the-art multi-head attention models based on the transformer architecture. We demonstrate that alignment extraction in transformer models can be improved by augmenting an additional alignment head to the multi-head source-to-target attention component. This is used to compute sharper attention weights. We describe how to use the alignment head to achieve competitive performance. To study the effect of adding the alignment head, we simulate a dictionary-guided translation task, where the user wants to guide translation using pre-defined dictionary entries. Using the proposed approach, we achieve up to 3.8% BLEU improvement when using the dictionary, in comparison to 2.4% BLEU in the baseline case. We also propose alignment pruning to speed up decoding in alignment-based neural machine translation (ANMT), which speeds up translation by a factor of 1.8 without loss in translation performance. We carry out experiments on the shared WMT 2016 English→Romanian news task and the BOLT Chinese→English discussion forum task.

1 Introduction

Attention-based neural machine translation (NMT) (Bahdanau et al., 2015) uses an attention layer to determine which part of the input sequence to focus on during decoding. This component eliminates the need for explicit alignment modeling. In conventional phrase-based statistical machine translation (Koehn et al., 2003), word alignment is modeled explicitly, making it clear which word or phrase is being translated. The lack of explicit alignment use in attention-based models makes it harder to determine which target words are generated using which source words. While this is not necessarily needed for translation itself, alignments can be useful in certain applications, e.g. when the customer wants to enforce specific translation of certain words.

One simple solution is to use maximum attention weights to extract the alignment, but this can result in wrong alignments in the case where the maximum attention weight is not pointing to the word being translated. Such cases are not uncommon in NMT, making the use of attention weights as alignment replacement non-trivial (Chatterjee et al., 2017; Hasler et al., 2018). Alignment extraction is even less clear for transformer models (Vaswani et al., 2017), which currently produce state-of-the-art results. These models use multiple attention components for each of the stacked decoder layers. In this work we focus our study on these models since they usually outperform single-attention-head recurrent neural network (RNN) attention models.[1]

Alignment-based NMT (Alkhouli et al., 2016) uses neural models trained using explicit hard alignments to generate translation. These systems include explicit alignment modeling, making them more convenient for tasks where the source-to-target alignment is needed. However, it is not clear whether these systems are able to compete with strong attention-based NMT systems. Alkhouli and Ney (2017) present results for alignment-based neural machine translation (ANMT) using models trained on CPUs, limiting them to small models of 200-node layers, and they only investigate RNN models. Wang et al. (2018) present results using only one RNN encoder layer, and do not include attention layers in their models. In this work, we investigate the performance of large and deep state-of-the-art transformer models. We keep the multi-head attention component and propose to augment it with an additional alignment head, to

[1] The transformer models won in most of the WMT 2018 news translation tasks: `http://matrix.statmt.org`.

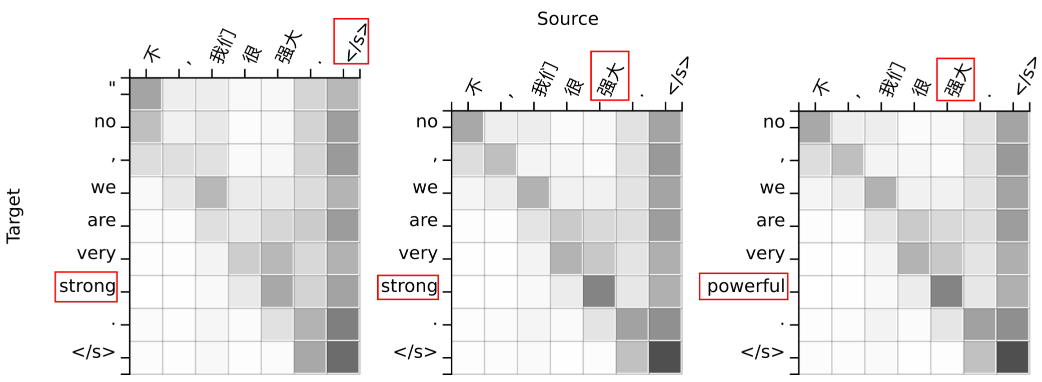

Figure 1: An example from the Chinese→English system. The figures illustrate the accumulated attention weights of the baseline transformer model (left), the alignment-assisted transformer model (middle), and the alignment-assisted model guided by a dictionary entry. We simulate a scenario where the user wants to translate the Chinese word "强大" to "powerful". Both the baseline and alignment-assisted transformer models generate the translation "strong" instead. To enforce the translation, we use the maximum attention weight to determine the source word being translated. Left: The maximum attention of the baseline case incorrectly points to the sentence end when translating the designated Chinese word, therefore we cannot enforce the translation in this case. Middle: The alignment looks sharper because the system has an augmented alignment head. In this case the maximum attention is pointing to the correct Chinese word. Right: using the maximum attention, the translation "strong" is successfully replaced with the translation "powerful" as suggested by the user using our proposed alignment-assisted transformer.

combine the benefits of the two. We demonstrate that we can train these models to achieve competitive results in comparison to strong state-of-the-art baselines. Moreover, we demonstrate that this variant has clear advantage in tasks that require alignments such as dictionary-guided translation.

Translation in NMT can be performed without explicit alignment. However, there are tasks where translation needs to be constrained given specific user requirements. Examples include interactive machine translation, and scenarios where customers demand domain-specific words or phrases to be translated according to a pre-defined dictionary. We demonstrate that the explicit use of alignment in ANMT can be leveraged to generate guided translation. Figure (1) illustrates an example. The figures are generated using attention weights averaged over all attention components in each system.

The contribution of this work is as follows. First, we propose a method to integrate alignment information into the multi-head attention component of the transformer model (Section 3.1). We describe how such models can be trained to maintain the strong baseline performance while also using external alignment information (Section 3.3). We also introduce alignment models that use self-attentive layers for faster evaluation (Section 3.2). Second, we introduce alignment pruning during search to speed up evaluation without affecting translation quality (Section 4). Third, we describe how to extract alignments from multi-head attention models (Section 5), and demonstrate that alignment-assisted transformer systems perform better than baseline systems in dictionary-guided translation tasks (Section 7). We present speed and performance results in Section 6.

2 Related Work

Alignment-based neural models have explicit dependence on the alignment information either at the input or at the output of the network. They have been extensively and successfully applied on top of conventional phrase-based systems (Sundermeyer et al., 2014; Tamura et al., 2014; Devlin et al., 2014). In this work, we focus on using the models directly to perform standalone neural machine translation.

Alignment-based neural models were proposed in (Alkhouli et al., 2016) to perform neural machine translation. They mainly used feedforward alignment and lexical models in decoding. Alkhouli and Ney (2017) used recurrent models instead, and presented an attention component biased using external alignment information. In this work, we explore the use of transformer models in ANMT instead of recurrent models.

Deriving neural models for translation based on the hidden Markov model (HMM) framework can also be found in (Yang et al., 2013; Yu et al., 2017). Alignment-based neural models were also applied to perform summarization and morphological inflection (Yu et al., 2016). Their work used a monotonous alignment model, where training was done by marginalizing over the alignment hidden variables, which is computationally expensive. In this work, we use non-monotonous alignment models. In addition, we train using pre-computed Viterbi alignments which speeds up neural training. In (Yu et al., 2017), alignment-based neural models were used to model alignment and translation from the target to the source side (inverse direction), and a language model was included in addition. They showed results on a small translation task. In this work, we present results on translation tasks containing tens of millions of words. We do not include a language model in any of our systems.

There is plenty of work on modifying attention models to capture more complex dependencies. Cohn et al. (2016) introduce structural biases from word-based alignment concepts like fertility and Markov conditioning. These are internal modifications that leave the model self-contained. Our modifications introduce alignments as external information to the model. Arthur et al. (2016) include lexical probabilities to bias attention. Chen et al. (2016) and Mi et al. (2016) add an extra term dependent on the alignments to the training objective function to guide neural training. This is only applied during training but not during decoding. Our work makes use of alignments during training and also during decoding.

There are several approaches to perform constrained translation. One possibility is including this information in training, but this requires knowing the constraints at training time (Crego et al., 2016). Post-processing the hypotheses is another possibility, but this comes with the downside that offline modification of the hypotheses happens out of context. A third possibility is to do constrained decoding (Hokamp and Liu, 2017; Chatterjee et al., 2017; Hasler et al., 2018; Post and Vilar, 2018). This does not require knowledge of the constraints at training time, and it also allows dynamic changes of the rest of the hypothesis when the constraints are activated. We perform experiments where the translation is guided online during decoding. We focus on the case where translation suggestions are to be used when a word in the source sentence matches the source side of a pre-defined dictionary entry. We show that alignment-assisted transformer-based NMT outperforms standard transformer models in such a task.

3 Alignment-Based Neural Machine Translation

Alignment-based NMT divides translation into two steps: (1) alignment and (2) word generation. The system is composed of an alignment model and a lexical model that can be trained jointly or separately. During translation, the alignment is hypothesized first, and the lexical score is computed next using the hypothesized alignment (Alkhouli et al., 2016). Hence, each translation hypothesis has an underlying alignment used to generate it. The alignment model scores the alignment path.

Formally, given a source sentence $f_1^J = f_1 ... f_j ... f_J$, a target sentence $e_1^I = e_1 ... e_i ... e_I$, and an alignment sequence $b_1^I = b_1 ... b_i ... b_I$, where $j = b_i \in \{1, 2, ..., J\}$ is the source position aligned to the target position $i \in \{1, 2, ..., I\}$, we model translation using an alignment model and a lexical model:

$$\begin{aligned} p(e_1^I|f_1^J) &= \sum_{b_1^I} p(e_1^I, b_1^I|f_1^J) \qquad (1) \\ &\approx \max_{b_1^I} \prod_{i=1}^{I} \underbrace{p(e_i|b_i, b_1^{i-1}, e_1^{i-1}, f_1^J)}_{\text{lexical model}} \cdot \\ &\qquad \underbrace{p(b_i|b_1^{i-1}, e_1^{i-1}, f_1^J)}_{\text{alignment model}} . \end{aligned}$$

Both the lexical model and the alignment model have rich dependencies including the full source context f_1^J, the full alignment history b_1^{i-1}, and the full target history e_1^{i-1}. The lexical model has an extra dependence on the current source position b_i.

While previous work focused on RNN structures for the lexical and alignment models (Alkhouli and Ney, 2017), we use multi-head self-attentive transformer model structures instead. The next two subsections describe the structural details of these models.

3.1 Transformer-Based Lexical Model

In this work we propose to use lexical models based on the transformer architecture (Vaswani et al., 2017). This architecture has the following main components:

- self-attentive layers replacing recurrent layers. These layers are parallelizable due to the lack of sequential dependencies that recurrent layers have.
- multi-head source-to-target attention: several attention heads are used to attend to the source side. Each attention head computes a normalized probability distribution over the source positions. The attention heads are concatenated. Each decoder layer in the model has its own multi-head attention component.

We propose to condition the lexical model on the alignment information. We add a special alignment head

$$\alpha(j|b_i) = \begin{cases} 1, & \text{if} \quad j = b_i \\ 0, & \text{otherwise.} \end{cases}$$

defined for the source positions $j, b_i \in \{1, 2, ..., J\}$. This is a one-hot distribution that has a value of 1 at position j that matches the aligned position b_i. This head is then concatenated to the rest of the attention heads as shown in Figure (2). The one-hot alignment distribution is used similar to attention weights to weight the encoded source representations, effectively selecting the representation h_{b_i} which corresponds to the aligned word.

3.2 Self-Attentive Alignment Model

In this work we use self-attentive layers instead of RNN layers in the alignment model. This removes the sequential dependency of computing RNN activations and allows for parallelization. We replace the bidirectional RNN encoder of the alignment model by multi-head self-attentive layers as described in (Vaswani et al., 2017). We also use multi-head self-attentive layers to replace the RNN layers in the decoder part of the network. There are two main differences when comparing this self-attentive alignment model to the transformer architecture described in (Vaswani et al., 2017). (1) The output is a probability distribution over possible source jumps $\Delta_i = b_i - b_{i-1}$, that

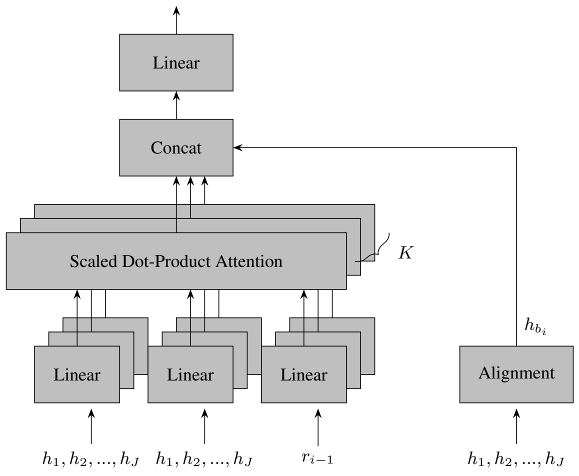

Figure 2: Alignment-assisted multi-head attention component. $h_1, h_2, ..., h_J$: the encoder states at all J source positions, h_{b_i}: the encoder state at the aligned source position b_i, r_{i-1}: the previous decoder state, K: number of attention heads. Removing the alignment block results in the default multi-head source-to-target attention component of (Vaswani et al., 2017).

is, the model predicts the likelihood of jumping from the previous source position b_{i-1} to the current source position b_i. (2) There is no multi-head source-to-target attention layer as in the transformer network. Rather, we use a single-head hard attention layer. This layer is not computed like attention weights, but it is constructed using the previous alignment point b_{i-1} using

$$\alpha(j|b_{i-1}) = \begin{cases} 1, & \text{if} \quad j = b_{i-1} \\ 0, & \text{otherwise.} \end{cases}$$

defined for the source positions $j, b_{i-1} \in \{1, 2, ..., J\}$. When multiplied by the source encodings, α effectively selects the source encoding $h_{b_{i-1}}$ of the previous aligned position. This is then summed up with the decoder state r_{i-1}.

3.3 Training

Our attempts to train the alignment-assisted transformer lexical model from scratch achieved suboptimal results. This could happen because the model could choose to over-rely on the alignment information, risking that the remaining attention heads would become useless, especially during the early stages of training. To overcome this, we first trained the transformer baseline parameters without the alignment information until convergence, and used the trained parameters to initial-

Algorithm 1 Alignment-Based Pruned Decoding

```
1:  procedure TRANSLATE($f_1^J$, $beamSize$, $threshold$)
2:      $hyps \leftarrow initHyp$    ▷init. set of partial hypotheses
3:      while GETBEST($hyps$) not terminated do
4:          ▷compute alignment distribution in batch mode
5:          $alignDists \leftarrow$ALIGNMENTDIST($hyps$)
6:          ▷hypothesize source alignment points
7:          $activePos \leftarrow \{\}$
8:          for $pos$ From $1$ to $J$ do
9:              ▷position computed if at least one
10:             ▷beam entry surpasses the threshold
11:             for $b$ From $1$ to $beamSize$ do
12:                 if $alignDists[b, pos] > threshold$ then
13:                     $activePos.Append(pos)$
14:                     break
15:         ▷evaluate all positions if none survived pruning
16:         if $activePos$ is empty then
17:             $activePos \leftarrow \{1, ...J\}$
18:         ▷compute lexical distributions of all
19:         ▷hypotheses in hyps in batch mode
20:         $lexDists \leftarrow$ LEXICALDIST($hyps$, $activePos$)
21:         ▷combine lexical and alignment scores
22:         $hyps \leftarrow Combine(lexDists, alignDists)$
23:         ▷prune to fit the beam
24:         $hyps \leftarrow Prune(hyps, beamSize)$
25:     ▷return the best scoring hypothesis
26:     return GETBEST($hyps$)
```

ize the alignment-assisted model training. This resulted in better systems compared to training from scratch. We were able to see significant perplexity improvements in the second stage of training indicating that the model was making use of the newly introduced information. Further details are discussed in Section 6.1.

4 Alignment Pruning

Alignment-based decoding requires hypothesizing alignment positions in addition to word translations. The algorithm is shown in Algorithm (1). Each lexical hypothesis has an underlying alignment hypothesis ($activePos$) that is used to compute it (line 20). This is done as a part of beam search. To speed up decoding, we compute the alignment model output first for all beam entries (line 5). This gives a distribution over the next possible source positions. We prune all source positions that have a probability below a fixed $threshold$ (lines 12–14). We only evaluate the lexical model for those positions that survive the threshold. If the pruning threshold is too aggressive to let any of the source positions survive, pruning is disabled for that time step (lines 16–17).

5 Alignment Extraction

We use attention weights to extract the alignments at each time step during decoding. We look up the source word having the maximum accumulated attention weight

$$j(i) = \underset{\hat{j} \in \{1...J\}}{\operatorname{argmax}} \left\{ \sum_{l=1}^{L} \sum_{k=1}^{K} \alpha_{i,k,l}(\hat{j}) \right\}$$

where K is the number of attention heads per decoder layer, L is the number of decoder layers, $\alpha_{i,k,l}(\hat{j})$ is the attention weight at source position $\hat{j} \in \{1, ..., J\}$ for target position i of the k-th head computed for the the l-th decoder layer. This is an extension of using maximum attention weights in single-head attention models (Chatterjee et al., 2017). In the alignment-assisted transformer, the aligned position is given by:

$$j(i, j') = \underset{\hat{j} \in \{1...J\}}{\operatorname{argmax}} \left\{ \sum_{l=1}^{L} \left(\sum_{k=1}^{K} \alpha_{i,k,l}(\hat{j}) + \alpha(\hat{j}|j') \right) \right\}$$

where $j' \in \{1, ..., J\}$ is the hypothesized source position during search, and $\alpha(\hat{j}|j')$ is the alignment indicator which is equal to 1 if $\hat{j} = j'$ and zero otherwise. This effectively gives a preference for the hypothesized position over all other positions. Note that the hypothesized positions are scored during translation using the alignment model described in Section 3.2.

6 Experiments

We run experiments on the WMT 2016 English→Romanian news task,[2] and on BOLT Chinese→English which is a discussion forum task. The corpora statistics are shown in Table (1).

All transformer models use 6 encoder and 6 decoder self-attentive layers. We use 8 scaled dot product attention heads and augment an additional alignment head to the source-to-target attention component. We use an embedding size of 512. The size of feedforward layers is 2048 nodes. We use source and target weight tying for the WMT English→Romanian task, and no tying for BOLT Chinese→English.

The structure of the RNN models is as follows. The English→Romanian lexical and alignment models use 1 bidirectional encoder layer. The

[2] `http://www.statmt.org/wmt16/`

	WMT 2016		BOLT	
	English	**Romanian**	**Chinese**	**English**
`Train` sentence pairs	604K		4.1M	
`Train` running words	15.5M	15.8M	80M	88M
`Dev` sentence pairs	1000		1845	
`Test` sentence pairs	1999		1124	
Vocabulary	92K	128K	380K	815K
Neural network vocabulary	50K	50K	50K	50K

Table 1: Corpora statistics.

			WMT En→Ro newstest2016			BOLT Zh→En test		
#	System	Layer size	PPL	BLEU[%]	TER[%]	PPL	BLEU[%]	TER[%]
	baselines							
1	Attention baseline	1000	10.2	24.7	58.9	8.0	20.0	65.6
2	Transformer baseline	2048	6.2	27.9	54.6	6.0	22.5	62.1
3	(Alkhouli and Ney, 2017)	200	-	24.8	58.1	-	-	-
	this work							
4	RNN Attention align.-biased	1000	7.2	26.4	56.1	5.6	19.6	62.3
5	Align.-assisted Transformer	2048	**5.0**	**28.1**	**54.3**	**4.7**	**22.7**	**61.8**

Table 2: Translation results for the WMT 2016 English→Romanian task and the BOLT Chinese→English task. We include the lexical model perplexities.

Chinese→English models have 1 bidirectional encoder and 3 stacked unidirectional encoder layers. All models use 2 decoder layers. The baseline attention models have similar structures. We use LSTM layers of 1000 nodes and embeddings of size 620. We train using the Adam optimizer (Kingma and Ba, 2015). All alignment models predict source jumps of maximum width of 100 source positions (forward and backward).

The alignments used during training are the result of IBM1/HMM/IBM4 training using GIZA++ (Och and Ney, 2003). All results are measured in case-insensitive BLEU[%] (Papineni et al., 2002). TER[%] scores are computed with *TER-Com* (Snover et al., 2006). We implement the models in Sockeye (Hieber et al., 2017), which allows efficient training of large models on GPUs.

6.1 Performance Comparison

Table (2) presents results on the two tasks. The RNN attention (row 1) and transformer (row 2) baselines are shown. The transformer baseline outperforms the attention baseline by a large margin. We also include the English→Romanian system of Alkhouli and Ney (2017). This is an alignment-based RNN attention system which uses 200-node layers. We also trained our own alignment-based RNN attention system using larger layers of 1000 nodes. This is shown in row 4. Our RNN system outperforms the previously published alignment-based results (row 3) by 1.6% BLEU and 2.0% TER. This is due to the increase in model size.

Our proposed alignment-assisted transformer system is shown in row 5. This system outperforms the RNN alignment-based system of row 4 by 1.7% BLEU on the English→Romanian task, establishing a new state-of-the-art result for alignment-based neural machine translation. We also achieve 3.1% BLEU improvement over our RNN alignment-biased attention system on the Chinese→English task. In comparison to the transformer baseline (row 2), the proposed system achieves similar performance on both tasks. We compare the development perplexity to check whether the lexical model makes use of the alignment information. Indeed, the baseline transformer development perplexity drops from 6.2 to 5.0 on English→Romanian and from 6.0 to 4.7

		WMT En→Ro			BOLT Zh→En		
#	Alignment	#entries	BLEU[%]	TER[%]	#entries	BLEU[%]	TER[%]
1	Transformer baseline	-	27.3	55.6	-	24.2	61.5
2	+ dictionary	3.1K	29.7	55.4	4.6K	25.5	61.0
3	Alignment-assisted Transformer	-	27.2	55.5	-	24.2	60.8
4	+ dictionary	3.1K	**31.0**	**53.0**	4.6K	**26.4**	**58.6**

Table 3: Improvements after using the dictionary of the development sets. The tokenized references of the English→Romanian and Chinese→English development sets have 26.7K and 46.6K running words respectively.

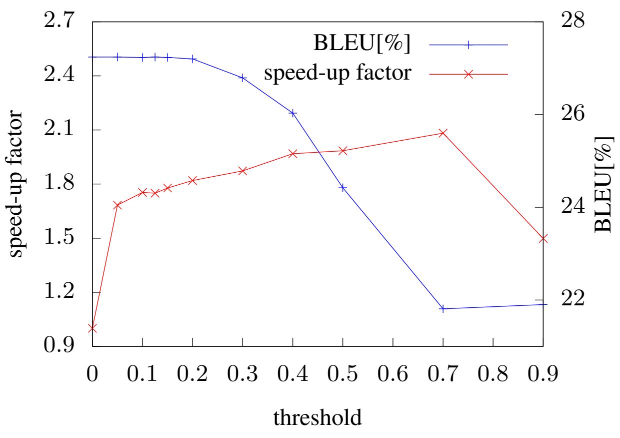

Figure 3: Speed up and translation quality in BLEU vs. pruning threshold on the WMT English→Romanian task.

on Chinese→English, indicating that the model is making use of the alignment information.

6.2 Decoding Speed Up

Figure (3) shows the speed-up factor and performance in BLEU over different threshold values. The speed-up factor is computed against the no-pruning case (i.e. threshold 0). The batch size used in these experiments is 5. We speed up translation by a factor of 1.8 without loss in translation quality at threshold 0.15. Higher threshold values result in more aggressive pruning and hence a degradation in translation quality. It is interesting to note that at threshold 0.05 we achieve a speed up of 1.7, implying that significant pruning happens at low threshold values. At high threshold values, speed starts to go down, since we have more cases where no alignment points survive the threshold, in which case pruning is disabled as discussed in Algorithm (1, lines 16–17).

7 Dictionary Suggestions

We evaluate the use of attention weights as alignments in a dictionary suggestion task, where a predefined dictionary of suggested one-to-one translations is given. We perform a relaxed form of constrained translation, i.e. we do not ensure that the suggestion will make it to the translation. To this end, we use attention weights to extract the alignments at each time step during decoding as described in Section 5. We look up the source word $f_{j(i)}$ having the maximum accumulated attention weight in the dictionary. If the word matches the source-side of a dictionary entry, we enforce the translation to match the dictionary suggestion $e(f_{j(i)})$ by setting an infinite cost for all but the suggested word.

We create a simulated dictionary using the reference side of the development set. We map the reference to the source words using IBM4 alignment. The development set is concatenated with the training data to obtain good-quality alignment. We exclude English stop words,[3] and only use source words aligned one-to-one to target words. We include up to 4 dictionary entries per sentence, and add reference translations only if they are not part of the baseline (i.e. unconstrained) translation, similar to (Hasler et al., 2018).

Table (3) shows results for the dictionary suggestions task described in Section (7). The English→Romanian dictionary covers 11.6% of the reference set, while the Chinese→English dictionary has 9.9% coverage. We observe larger improvement when using the dictionary entries in the alignment-assisted transformer system in comparison to the transformer baseline systems. Our system improves BLEU by 3.8%, while the baseline is improved only by 2.4% BLEU on the English→Romanian task. We also observe larger

[3]Long stop list: `https://www.ranks.nl/stopwords`

improvements in the Chinese→English case. This suggests that the maximum attention weights in alignment-assisted systems can point more accurately to the word being translated, allowing the use of more dictionary entries. As shown in Figure (1), the accumulated attention weights are sharper when the system has an augmented alignment head. This explains the larger improvements our systems achieve.

8 Conclusion

We proposed augmenting transformer models with an alignment head to help extract alignments in scenarios such as dictionary-guided translation. We demonstrated that the alignment-assisted systems can achieve competitive performance compared to strong transformer baselines. We also showed that the alignment-assisted systems outperformed standard transformer models when used for dictionary-guided translation on two tasks. Finally, we achieved a speed-up factor of 1.8 by pruning alignment hypotheses in alignment-based decoding while maintaining translation quality. In future work we plan to investigate alternative pruning methods like histogram pruning. We also plan to investigate the performance of alignment-assisted transformer models in constrained decoding settings, where the user demands specific translation of certain words.

Acknowledgments

This research has received funding from the European Research Council (ERC) (under the European Union's Horizon 2020 research and innovation programme, grant agreement No 694537, project "SEQCLAS") and the Deutsche Forschungsgemeinschaft (DFG; grant agreement NE 572/8-1, project "CoreTec"). Tamer Alkhouli was partly funded by the 2016 Google PhD fellowship for North America, Europe and the Middle East. The work reflects only the authors' views and none of the funding parties is responsible for any use that may be made of the information it contains.

References

Tamer Alkhouli, Gabriel Bretschner, Jan-Thorsten Peter, Mohammed Hethnawi, Andreas Guta, and Hermann Ney. 2016. Alignment-based neural machine translation. In *Proceedings of the First Conference on Machine Translation*, pages 54–65, Berlin, Germany.

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

Philip Arthur, Graham Neubig, and Satoshi Nakamura. 2016. Incorporating discrete translation lexicons into neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567, Austin, Texas.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*, San Diego, Calefornia, USA.

Rajen Chatterjee, Matteo Negri, Marco Turchi, Marcello Federico, Lucia Specia, and Frédéric Blain. 2017. Guiding neural machine translation decoding with external knowledge. In *Proceedings of the Second Conference on Machine Translation*, pages 157–168, Copenhagen, Denmark.

Wenhu Chen, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. 2016. Guided alignment training for topic-aware neural machine translation. In *Proceedings of the 2016 Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 121–134, Austin, Texas.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California.

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, et al. 2016. Systran's pure neural machine translation systems. *arXiv preprint arXiv:1610.05540*.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and Robust Neural Network Joint Models for Statistical Machine Translation. In *52nd Annual Meeting of the Association for Computational Linguistics*, pages 1370–1380, Baltimore, MD, USA.

Eva Hasler, Adrià De Gisper, 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*, pages 506–512, New Orleans, Louisiana, USA.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *arXiv preprint arXiv:1712.05690*.

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.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, San Diego, Calefornia, USA.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*, pages 127–133, Edmonton, Alberta.

Haitao Mi, Zhiguo Wang, and Abe Ittycheriah. 2016. Supervised attentions for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2283–2288, Austin, Texas.

Franz J. 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 the 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. 2014. Translation Modeling with Bidirectional Recurrent Neural Networks. In *Conference on Empirical Methods on Natural Language Processing*, pages 14–25, Doha, Qatar.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2014. Recurrent neural networks for word alignment model. In *52nd Annual Meeting of the Association for Computational Linguistics*, pages 1470–1480, Baltimore, MD, USA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Weiyue Wang, Derui Zhu, Tamer Alkhouli, Zixuan Gan, and Hermann Ney. 2018. Neural hidden markov model for machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 377–382. Association for Computational Linguistics.

Nan Yang, Shujie Liu, Mu Li, Ming Zhou, and Nenghai Yu. 2013. Word alignment modeling with context dependent deep neural network. In *51st Annual Meeting of the Association for Computational Linguistics*, pages 166–175, Sofia, Bulgaria.

Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomás Kociský. 2017. The neural noisy channel. In *Proceedings of the International Conference on Learning Representations*, volume abs/1611.02554.

Lei Yu, Jan Buys, and Phil Blunsom. 2016. Online segment to segment neural transduction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1307–1316, Austin, Texas.