Accurate Online Posterior Alignments for Principled Lexically-Constrained Decoding

Anonymous ACL submission

Abstract

Online alignment in machine translation refers to the task of aligning a target word to a source word when the target sequence has only been partially decoded. Good online alignments facilitate important applications such as lexically constrained translation where user-defined dictionaries are used to inject lexical constraints into the translation model. We propose a novel posterior alignment technique that is truly online in its execution and superior in terms of alignment error rates compared to existing methods. Our proposed inference technique jointly considers alignment and token probabilities in a principled manner and can be seamlessly integrated within existing constrained beam-search decoding algorithms. On five language pairs, including two distant language pairs, we achieve consistent drop in alignment error rates. When deployed on three lexically constrained translation tasks, we achieve significant improvements in BLEU specifically around the constrained positions. We show that our alignment guided constrained inference yields additional benefits of fluency with negligible additional computational costs.

1 Introduction

Online alignment seeks to align a target word to a source word at the decoding step when the word is output in an auto-regressive neural translation model [Kalchbrenner and Blunsom, 2013, Cho et al., 2014, Sutskever et al., 2014]. This is unlike the more popular offline alignment task that assumes the presence of the entire target sentence [Och and Ney, 2003]. State of the art methods of offline alignment based on matching of whole source and target sentences are not applicable for online alignment [Jalili Sabet et al., 2020, Dou and Neubig, 2021], where we need to commit on the alignment of a target word based on only the generated prefix thus far.

An important application of online alignment is lexically constrained translation which allows injection of domain-specific terminology and other phrasal constraints during decoding [Hasler et al., 2018, Hokamp and Liu, 2017, Alkhouli et al., 2018, Crego et al., 2016]. Other applications include preservation of markups between the source and target [Müller, 2017], and supporting source word edits in summarization [Shen et al., 2019]. These applications need to infer the specific source token which aligns with output token. Thus, alignment and translation is to be done simultaneously.

Existing online alignment methods can be categorized into Prior and Posterior alignment methods. Prior alignment methods [Garg et al., 2019, Song et al., 2020] extract alignment based on the attention at time step t when outputting token y_t. The attention probabilities at time-step t are conditioned on tokens output before time t. Thus, the alignment is estimated *prior* to observing y_t. Naturally, the quality of alignment can be improved if we condition on the target token y_t [Shankar and Sarawagi, 2019]. This motivated Chen et al. [2020] to propose a posterior alignment method where alignment is calculated from the attention probabilities at the next decoder step $t + 1$. While alignment quality improved as a result, their method is not truly online since it does not generate alignment *synchronously* with the token. The delay of one step makes it difficult and cumbersome to incorporate terminology constraints during beam decoding.

We propose a truly online posterior alignment method that provides higher alignment accuracy than existing online methods, while also being synchronous. Because of that we can easily integrate posterior alignment to improve lexicon-constrained translation in state of the art constrained beam-search algorithms such as VDBA [Hu et al., 2019]. We propose a principled joint distribution over token and alignment probability to score constraint placement. Our method provides higher BLEU around the constrained span both compared to the ad hoc inference proposed in Chen et al. [2021] and VDBA that ignores source alignment.

Contributions

- A truly online posterior alignment method that integrates into existing NMT sytems via a trainable light-weight module.
- Higher online alignment accuracy on five language pairs including two distant language pairs.
- Principled method of modifying VDBA to incorporate posterior alignment probabilities in lexically-constrained decoding.
- Significant improvement in BLEU around constrained span, while yielding more fluent translations than VDBA that ignores alignments.

2 Posterior Online Alignment

Given a sentence $\mathbf{x} = x_1, \ldots, x_S$ in the source language and a sentence $\mathbf{y} = y_1, \ldots, y_T$ in the target language, an alignment $\mathcal{A}$ between the word strings is a subset of the Cartesian product of the word positions [Brown et al., 1993, Och and Ney, 2003]: $\mathcal{A} \subseteq \{(s,t) : s = 1, \ldots, S; t = 1, \ldots, T\}$ such that the aligned words can be considered translations of each other. An online alignment at time-step t commits on alignment of the t^{th} output token conditioned only on $\mathbf{x}$ and $\mathbf{y}_{<t} = y_1, y_2, \ldots y_{t-1}$. Additionally, if token y_t is also available we call it a posterior online alignment. We seek to embed online alignment with existing NMT systems. We will first briefly describe the architecture of state of the art NMT systems. We will then elaborate on how alignments are computed from attention distributions in prior work and highlight some limitations, before describing our proposed approach.

2.1 Background

Transformer-based models have become a ubiquitous choice for neural machine translation [Vaswani et al., 2017]. Transformers adopt the popular encoder-decoder paradigm used for sequence-to-sequence modeling [Cho et al., 2014, Sutskever et al., 2014, Bahdanau et al., 2015]. The encoder and decoder are both multi-layered networks with each layer consisting of a multi-headed self-attention and a feedforward module. The decoder layers additionally make use of multi-headed attention to encoder states. We elaborate on this attention mechanism next since it plays an important role in alignments.

2.1.1 Decoder-Encoder Attention in NMTs

The encoder transforms the S input tokens into a sequence of token representations $\mathbf{H} \in \mathbb{R}^{S \times d}$. Each decoder layer (indexed by $\ell \in \{1, \ldots, L\}$) computes multi-head attention over $\mathbf{H}$ by aggregating outputs from a set of η independent attention heads. The attention output from a single head $n \in \{1, \ldots, \eta\}$ in decoder layer ℓ is computed as follows. Let the output of the self-attention sub-layer in decoder layer ℓ at the t^{th} target token be denoted as $\mathbf{g}_t^{\ell}$. Using three projection matrices $\mathbf{W}_Q^{\ell,n}$, $\mathbf{W}_V^{\ell,n}$, $\mathbf{W}_K^{\ell,n} \in \mathbb{R}^{d \times d_n}$, the query vector $\mathbf{q}_t^{\ell,n} \in \mathbb{R}^{1 \times d_n}$ and key and value matrices, $\mathbf{K}^{\ell,n} \in \mathbb{R}^{S \times d_n}$ and $\mathbf{V}^{\ell,n} \in \mathbb{R}^{S \times d_n}$, are computed using the following projections: $\mathbf{q}_t^{\ell,n} = \mathbf{g}_t^{\ell}\mathbf{W}_Q^{\ell,n}$, $\mathbf{K}^{\ell,n} = \mathbf{H}\mathbf{W}_K^{\ell,n}$, and $\mathbf{V}^{\ell,n} = \mathbf{H}\mathbf{W}_V^{\ell,n}$.[1] These are used to calculate the attention output from head n, $\mathbf{Z}_t^{\ell,n} = P(\mathbf{a}_t^{\ell,n}|\mathbf{x}, \mathbf{y}_{<t})\mathbf{V}^{\ell,n}$, where:

$$P(\mathbf{a}_t^{\ell,n}|\mathbf{x}, \mathbf{y}_{<t}) = \text{softmax}\left(\frac{\mathbf{q}_t^{\ell,n}(\mathbf{K}^{\ell,n})^\intercal}{\sqrt{d}}\right) \quad (1)$$

For brevity, the conditioning on $\mathbf{x}, \mathbf{y}_{<t}$ is dropped and $P(\mathbf{a}_t^{\ell,n})$ is used to refer to $P(\mathbf{a}_t^{\ell,n}|\mathbf{x}, \mathbf{y}_{<t})$ in the following sections.

Finally, the multi-head attention output is given $[\mathbf{Z}_t^{\ell,1}, \ldots, \mathbf{Z}_t^{\ell,\eta}]\mathbf{W}^O$ where $[\,]$ denotes the column-wise concatenation of matrices and $\mathbf{W}^O \in \mathbb{R}^{d \times d}$ is an output projection matrix.

2.1.2 Alignments from Attention

Several prior work have proposed to extract word alignments from the above attention probabilities. For example Garg et al. [2019] propose a simple method called NAIVEATT that aligns a source word to the t^{th} target token using $\text{argmax}_j \frac{1}{\eta}\sum_{n=1}^{\eta} P(a_{t,j}^{\ell,n}|\mathbf{x}, \mathbf{y}_{<t})$. In NAIVEATT, we note that the attention probabilities $P(a_{t,j}^{\ell,n}|\mathbf{x}, \mathbf{y}_{<t})$ at decoding step t are not conditioned on the current output token y_t. The quality of the alignment would benefit from conditioning on y_t as well. This observation prompted Chen et al. [2020] to extract alignment of token y_t using attention $P(a_{t,j}^{\ell,n}|\mathbf{x}, \mathbf{y}_{\leq t})$ computed at time step $t+1$. The asynchronicity inherent to this shift-by-one approach (SHIFTATT) makes it difficult and more computationally expensive to incorporate lexical constraints during beam decoding.

[1] d_n is typically set to $\frac{d}{\eta}$ so that a multi-head attention layer does not introduce more parameters compared to a single head attention layer.

2.2 Our Proposed Method: PostAln

We propose PostAln that produces posterior alignments synchronously with the output tokens, while being more computationally efficient compared to previous approaches like ShiftAtt. We incorporate a lightweight alignment module to convert prior attention to posterior alignments in the same decoding step as the output. Figure 1 illustrates how this alignment module fits within the standard Transformer architecture.

The alignment module is placed at the penultimate decoder layer $\ell = L - 1$ and takes as input 1) the encoder output $\mathbf{H}$, 2) the output of the self-attention sub-layer of decoder layer ℓ, $\mathbf{g}_t^\ell$ and, 3) the embedding of the decoded token $\mathbf{e}(y_t)$. Like in standard attention it projects $\mathbf{H}$ to obtain a key matrix, but to obtain the query matrix it uses both decoder state $\mathbf{g}_t^\ell$ (that summarizes $\mathbf{y}_{<t}$) and $\mathbf{e}(y_t)$ to compute the posterior alignment $P(\mathbf{a}_t^{\text{post}})$ as:

$$P(\mathbf{a}_t^{\text{post}}) = \frac{1}{\eta}\sum_{n=1}^{\eta} \text{softmax}\left(\frac{\mathbf{q}_{t,\text{post}}^n (\mathbf{K}_{\text{post}}^n)^\intercal}{\sqrt{d}}\right),$$

$$\mathbf{q}_{t,\text{post}}^n = [\mathbf{g}_t^\ell, \mathbf{e}(y_t)]\mathbf{W}_{Q,\text{post}}^n, \ \mathbf{K}_{\text{post}}^n = \mathbf{H}\mathbf{W}_{K,\text{post}}^n$$

Here $\mathbf{W}_{Q,\text{post}}^n \in \mathbb{R}^{2d\times d_n}$ and $\mathbf{W}_{K,\text{post}}^n \in \mathbb{R}^{d\times d_n}$.

This computation is synchronous with producing the target token y_t, thus making it compatible with beam search decoding (as elaborated further in Section 3). It also accrues minimal computational overhead since $P(\mathbf{a}_t^{\text{post}})$ is defined using $\mathbf{H}$ and $\mathbf{g}_t^{L-1}$, that are both already cached during a standard decoding pass.

Note that if the query vector $\mathbf{q}_{t,\text{post}}^n$ is computed using only $\mathbf{g}_t^{L-1}$, without concatenating $\mathbf{e}(y_t)$, then we get prior alignments that we refer to as PriorAtt. In our experiments, we explicitly compare PriorAtt with PostAln to show the benefits of using y_t in deriving alignments while keeping the rest of the architecture intact.

2.2.1 Training

Our posterior alignment sub-layer is trained using alignment supervision, while freezing the rest of the translation model parameters. Specifically, we train a total of $3d^2$ additional parameters across the matrices $\mathbf{W}_{K,\text{post}}^n$ and $\mathbf{W}_{Q,\text{post}}^n$.

Since gold alignments are very tedious and expensive to create for large training datasets, alignment labels are typically obtained using existing techniques. We use bidirectional symmetrized ShiftAtt alignments, denoted by $S_{i,j}$ that refers to an alignment between the i^{th} target word and the j^{th} source word, as reference labels to train our alignment sub-layer. Then the objective (following Garg et al. [2019]) can be defined as:

$$\max_{\mathbf{W}_{Q,\text{post}}^n, \mathbf{W}_{K,\text{post}}^n} \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{S} S_{i,j} \log\left(P(a_{i,j}^{\text{post}}|\mathbf{x}, \mathbf{y}_{\le i})\right)$$

In Section 4, we will show that both posterior alignments and the above training have a huge impact on alignment accuracy.

Next, we demonstrate the role of posterior online alignments on an important downstream task.

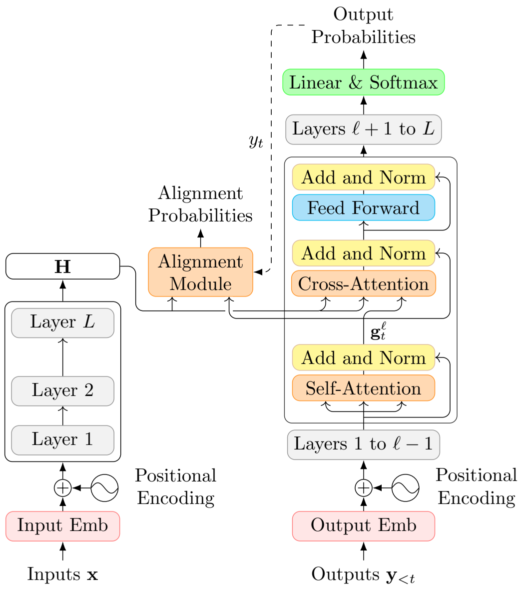

Figure 1: Our alignment module is an encoder-decoder attention sub-layer, similar to the existing cross-attention sub-layer. It takes as inputs the encoder output $\mathbf{H}$ as the key, and the concatenation of the output of the previous self-attention layer $\mathbf{g}_t^\ell$ and the currently decoded token y_t as the query, and outputs posterior alignment probabilities $\mathbf{a}_t^{\text{post}}$.

3 Lexicon Constrained Translation

In the lexicon constrained translation task, for each to-be-translated sentence $\mathbf{x}$, we are given a set of source text spans and the corresponding target tokens in the translation. A constraint $\mathcal{C}_j$ comprises of a pair $(\mathcal{C}_j^x, \mathcal{C}_j^y)$ where $\mathcal{C}_j^x = (p_j, p_j + 1 \ldots, p_j + \ell_j)$ indicates input token positions, and $\mathcal{C}_j^y = (y_1^j, y_2^j \ldots, y_{m_j}^j)$ denote target tokens that are translations of the input tokens $x_{p_j} \ldots x_{p_j+\ell_j}$. For the output tokens we do not know their positions in the target sentence. The different constraints are non-overlapping and each is expected to be used exactly once. The goal is to translate the given sentence $\mathbf{x}$ and satisfy as many constraints in $\mathcal{C} = \bigcup_j \mathcal{C}_j$ as possible while ensuring fluent and correct translations. Since the constraints do not specify target token position, it is natural to use online alignments to guide when a particular constraint is to be enforced.

3.1 Background: Constrained Decoding Methods

Existing inference algorithms for incorporating lexicon constraints differ in how pro-actively they enforce the constraints. A passive method is used in Song et al. [2020] where constraints are enforced only when the prior alignment is at a constrained source span. Specifically, if at decoding step t, $i = \text{argmax}_{i'} P(a_{t,i'})$ is present in some constraint $\mathcal{C}^x_j$, the output token is fixed to the first token y^j_1 from $\mathcal{C}^y_j$. Otherwise, the decoding proceeds as usual. Also, if the translation of a constraint $\mathcal{C}_j$ has started, the same is completed (y^j_2 through $y^j_{m_j}$) for the next $m_j - 1$ decoding steps before resuming unconstrained beam search. The pseudocode for this method is provided in Appendix D.

For the posterior alignment methods of Chen et al. [2020] this leads to a rather cumbersome inference [Chen et al., 2021]. First, at step t they predict a token $\hat{y}_t$, then start decoding step $t+1$ with $\hat{y}_t$ as input to compute the posterior alignment from attention at step $t+1$. If the maximum alignment is to the constrained source span $\mathcal{C}^x_j$ they *revise* the output token to be y^j_1 from $\mathcal{C}^y_j$, but the output score for further beam-search continues to be of $\hat{y}_t$. In this process both the posterior alignment and token probabilities are misrepresented since they are both based on $\hat{y}_t$ instead of the finally output token y^j_1. The decoding step at $t+1$ needs to be restarted after the revision. The overall algorithm continues to be normal beam-search, which implies that the constraints are not enforced pro-actively.

Many prior methods have proposed more proactive methods of enforcing constraints, including the Grid Beam Search (GBA, Hokamp and Liu [2017]), Dynamic Beam Allocation (DBA, Post and Vilar [2018]) and Vectorized Dynamic Beam Allocation (VDBA, Hu et al. [2019]). The latest of these, VDBA, is efficient and available in public NMT systems [Ott et al., 2019, Hieber et al., 2020]. Here multiple *banks*, each corresponding to a particular number of completed constraints, are maintained. At each decoding step, a hypothesis can either start a new constraint and move to a new bank or continue in the same bank (either by not starting a constraint or progressing on a constraint mid-completion). This allows them to achieve near 100% enforcement. However, VDBA enforces the constraints by considering only the target tokens of the lexicon and totally ignores the alignment of these tokens to the source span. This could lead to constraints being placed at unnatural locations leading to loss of fluency. Examples appears in Table 4 where we find that VDBA just attaches the constrained tokens at the end of the sentence.

3.2 Our Proposal: Align-VDBA

We modify VDBA with alignment probabilities to better guide constraint placement. The score of a constrained token instead of being only the token probability, is now the joint probability of the token, and the probability of the token being aligned with the corresponding constrained source span. Formally, if the current token y_t is a part of the j^{th} constraint *i.e.* $y_t \in \mathcal{C}^y_j$, the generation probability of y_t, $P(y_t|\mathbf{x}, \mathbf{y}_{<t})$ is scaled by multiplying with the alignment probabilities of y_t with $\mathcal{C}^x_j$, the source span for constraint i. Thus, the updated probability is given by:

$$\underbrace{P(y_t, \mathcal{C}^x_j|\mathbf{x}, \mathbf{y}_{<t})}_{\text{Joint Prob}} = \underbrace{P(y_t|\mathbf{x}, \mathbf{y}_{<t})}_{\text{Token Prob}} \underbrace{\sum_{r \in \mathcal{C}^x_j} P(a^{\text{post}}_{t,r}|\mathbf{x}, \mathbf{y}_{\leq t})}_{\text{Src Align. Prob.}} \tag{2}$$

$P(y_t, \mathcal{C}^x_j|\mathbf{x}, \mathbf{y}_{<t})$ denotes the joint probability of outputting the constrained token and the alignment being on the corresponding source span. Since the supervision for the alignment probabilities was noisy, we found it useful to recalibrate the alignment distribution using a temperature scale T, so that the recalibrated probability is $\propto \Pr(a^{\text{post}}_{t,r}|\mathbf{x}, \mathbf{y}_{\leq t})^{\frac{1}{T}}$. We used $T = 2$ which corresponds to taking the square-root of the estimated alignment probability.

We present the pseudocode of our modification (steps 5 and 6, in blue) to DBA in Algorithm 1. Other details of the algorithm including the handling of constraints and the allocation steps (step 10) are involved and we refer the reader to Post and Vilar [2018] and Hu et al. [2019] to understand these details. The point of this code is to show that our proposed posterior alignment method can be easily incorporated into these algorithms so as to provide a more principled scoring of constrained hypothesis in a beam than the ad hoc revision-based method of Chen et al. [2021]. Additionally, pos-

Algorithm 1 Align-VDBA: Modifications to DBA shown in blue. (Adapted from Post and Vilar [2018])

Inputs beam: K hypothesis in beam, scores: $K \times |V_T|$ matrix of scores where scores$[k, y]$ denotes the score of k^{th} hypothesis extended with token y at this step, constraints: $\{(\mathcal{C}_j^x, \mathcal{C}_j^y)\}$
candidates $\leftarrow$ [$(k, y$, scores$[k, y]$, beam$[k]$.constraints.add$(y))$] for k, y in ARGMAX_K(scores)
for $1 \leq k \leq K$ **do** ▷ Go over current beam
  for all $y \in V_T$ that are unmet constraints for beam$[k]$ **do** ▷ Expand new constraints
    alignProb $\leftarrow \Sigma_{\text{constraint\_xs}(y)}$ POSTALN(k, y) ▷ Modification in blue (Eqn (2))
    candidates.append($(k, y$, scores$[k, y] \times$ alignProb), beam$[k]$.constraints.add(y)))
    candidates.append($(k, y$, scores$[k, y]$, beam$[k]$.constraints.add(y))) ▷ Original DBA Alg.
  w = ARGMAX(scores$[k, :]$)
  candidates.append($(k, w$, scores$[k, w]$, beam$[k]$.constraints.add(w))) ▷ Best single word
newBeam $\leftarrow$ ALLOCATE(candidates, K)

terior alignments lead to better placement of constraints than in the original VDBA algorithm.

4 Experiments

We first compare our proposed posterior online alignment method on quality of alignment against existing methods in Section 4.2, and in Section 4.3, we demonstrate the impact of the improved alignment on the lexicon-constrained translation task.

4.1 Setup

We deploy the `fairseq` toolkit [Ott et al., 2019] and use `transformer_iwslt_de_en` preconfigured model for all our experiments. Other configuration parameters include: Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, a learning rate of $5\mathrm{e}{-4}$ with 4000 warm-up steps, an inverse square root schedule, weight decay of $1\mathrm{e}{-4}$, label smoothing of 0.1, 0.3 probability dropout and a batch size of 4500 tokens. The transformer models are trained for 50,000 iterations. Then, the alignment module is trained for 10,000 iterations, keeping the other model parameters fixed. A joint byte pair encoding (BPE) is learned for the source and the target languages with 10k merge operation [Sennrich et al., 2016] using `subword-nmt`[2].

All experiments were done on a single 11GB Nvidia GeForce RTX 2080 Ti GPU on a machine with 64 core Intel Xeon CPU and 755 GB memory. The vanilla Transformer models take between 15 to 20 hours to train for different datasets. Starting from the alignments extracted from these models, the POSTALN alignment module trains in about 3 to 6 hours depending on the dataset.

[2] https://github.com/rsennrich/subword-nmt

	de-en	en-fr	ro-en	en-hi	ja-en
Training	1.9M	1.1M	0.5M	1.6M	0.3M
Validation	994	1000	999	25	1166
Test	508	447	248	90	1235

Table 1: Number of sentence pairs for the five datasets used. Note that gold alignments are available only for a handful of sentence pairs in the test set.

4.2 Alignment Task

We evaluate online alignments on ten translation tasks spanning five language pairs. Three of these are popular in alignment papers [Zenkel et al., 2019]: German-English (de-en), English-French (en-fr), Romanian-English (ro-en). These are all European languages that follow the same subject-verb-object (SVO) ordering. We also present results on two distant language pairs (English-Hindi and English-Japanese) that follow a SOV word order which is different from the SVO word order of English. Data statistics are shown in Table 1 and more details of the datasets are described in Appendix B.

Evaluation Method: For evaluating alignment performance, it is necessary that the target sentence is exactly the same as for which the gold alignments are provided. Thus, for the alignment experiments, we force the output token to be from the gold target and only infer the alignment. We then report the Alignment Error Rate (AER) [Och and Ney, 2000] between the gold alignments and the predicted alignments for different methods. Though our focus is online alignment, for comparison to previous works, we also report results on bidirectional symmetrized alignments in Appendix C.

Methods compared: We compare our method with both existing statistical alignment models,

Method	Delay	de-en		en-fr		ro-en		en-hi		ja-en	
		de→en	en→de	en→fr	fr→en	ro→en	en→ro	en→hi	hi→en	ja→en	en→ja
Statistical Methods (Not Online)											
GIZA++ [Och and Ney, 2003]	End	18.9	19.7	7.3	7.0	27.6	28.3	35.9	36.4	41.8	39.0
FastAlign [Dyer et al., 2013]	End	28.4	32.0	16.4	15.9	33.8	35.5	-	-	-	-
No Alignment Training											
NaiveAtt [Garg et al., 2019]	0	32.4	40.0	24.0	31.2	37.3	33.2	50.5	52.9	62.2	63.5
ShiftAtt [Chen et al., 2020]	+1	20.0	22.9	14.7	20.4	26.9	27.4	38.6	42.3	53.6	48.6
With Alignment Training											
PriorAtt	0	23.4	25.8	14.0	16.6	29.3	27.2	38.5	35.5	52.7	50.9
ShiftAET [Chen et al., 2020]	+1	15.8	**19.5**	10.3	**10.4**	22.4	23.7	31.9	33.3	42.5	**41.9**
PostAln [Ours]	0	**15.5**	**19.5**	**9.9**	**10.4**	**21.8**	**23.2**	**31.8**	**32.4**	**41.2**	42.2

Table 2: AER for German-English, English-French, Romanian-English, English-Hindi, Japanese-English language pairs. The delay column indicates the decoding step at which the alignment of the target token is available. NaiveAtt, PriorAtt and PostAln are the only true online methods that output alignment at the same time step (delay=0), while ShiftAtt and ShiftAET output one decoding step later.

namely GIZA++ [Och and Ney, 2003] and FastAlign [Dyer et al., 2013], and recent Transformer-based alignment methods of Garg et al. [2019] (NaiveAtt) and Chen et al. [2020] (ShiftAtt and ShiftAET). Chen et al. [2020] also propose a variant of ShiftAtt called ShiftAET that employs the same idea of delaying computations by one time-step as in ShiftAtt, and additionally includes a learned attention sub-layer to compute alignment probabilities. As mentioned in Section 2.2, we also present results on PriorAtt which is similar to PostAln but does not use $\mathbf{y}_t$.

Results: The alignment results are shown in Table 2. First, AERs using statistical methods FastAlign and GIZA++ are shown. Here, for fair comparison, the IBM models used by GIZA++ are trained on the same sub-word units as the Transformer models and sub-word alignments are converted to word level alignments for AER calculations. (Even with deep learning based translation models gaining popularity, GIZA++ has remained a state-of-the-art technique for word alignments, although it is not online.) Next, we present alignment results for two vanilla Transformer models - NaiveAtt and ShiftAtt - that do not train a separate alignment module. The high AER of NaiveAtt shows that attention-as-is is very distant from alignment but posterior attention is closer to alignments than prior. Next we look at methods that train alignment-specific parameters: PriorAtt, a prior attention method; ShiftAET and PostAln, both posterior alignment methods. We observe that with training even PriorAtt has surpassed non-trained posterior. The posterior attention methods outperform the prior attention methods by a large margin, with a difference of 4.0 to 8.0 points between the posterior and prior alignment methods. Within each group, the methods with a trained alignment module outperform the ones without by a huge margin. PostAln performs better or matches the performance of ShiftAET while avoiding the one-step delay in alignment generation. We observe that PostAln has the lowest AER in nine out of ten cases in Table 2. Even on the distant languages, PostAln achieves significant reductions in error. For example, for ja→en we achieve a 1.3 AER reduction compared to ShiftAET which is not a truly online method. Figure 2 uses two examples to illustrate the superior alignments of PostAln compared to NaiveAtt and PriorAtt.

4.3 Impact of PostAln on Lexicon-Constrained Translation

We next depict the impact of improved AERs from our posterior alignment method on a downstream lexicon-constrained translation task. Following previous work [Hokamp and Liu, 2017, Post and Vilar, 2018, Song et al., 2020, Chen et al., 2020, 2021], we extract constraints using the gold alignments and gold translations. Up to three constraints of up to three words each are used for each sentence. Spans correctly translated by a greedy decoding are not selected as constraints.

Metrics: We report BLEU [Papineni et al., 2002] scores, Constraint Satisfaction Rate (CSR), and the time required to translate all test sentences as reported by others [Song et al., 2020]. Additionally to evaluate the appropriateness of constraint placement, we compute the BLEU of spans consisting of the constraints and a window of a few words, specifically three, on both sides of the constraint. We call this measure SpanBLEU. All numbers are averages over five different sets of randomly sam-

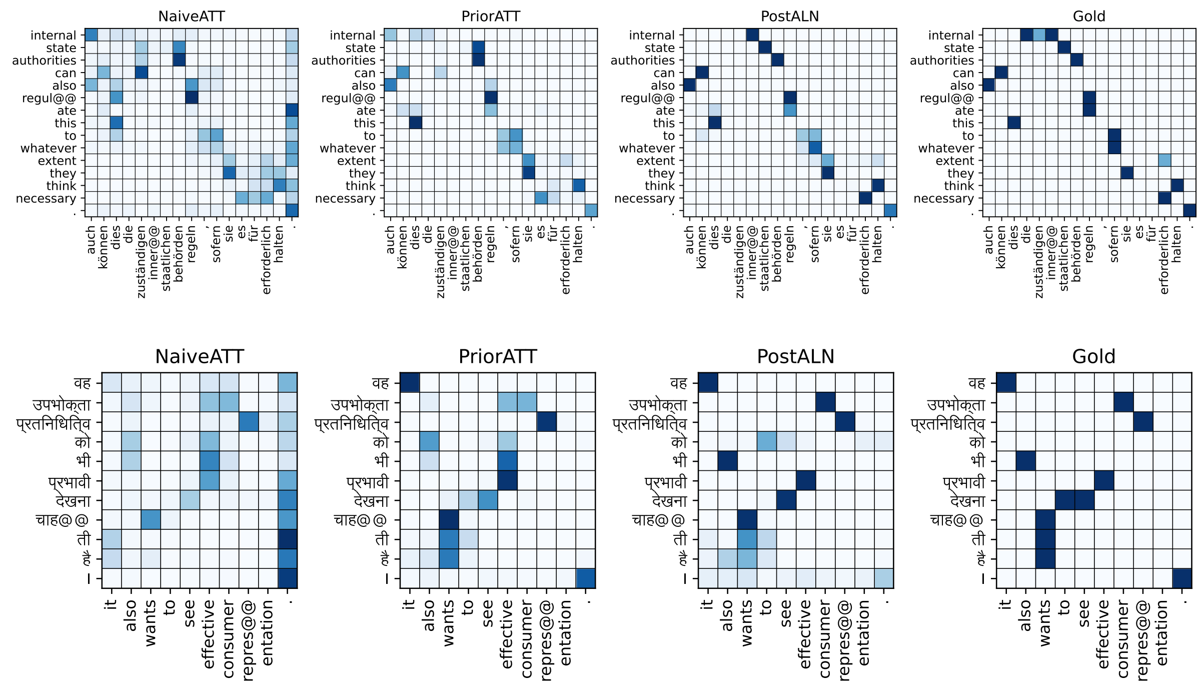

Figure 2: Alignments for de→en (top-row) and en→hi (bottom-row) by NAIVEATT, PRIORATT, and POSTALN. Note that POSTALN is most similar to Gold alignments in the last column.

	de→en				en→fr				ro→en			
Method	SpanBLEU	CSR	BLEU	Time(s)	SpanBLEU	CSR	BLEU	Time(s)	SpanBLEU	CSR	BLEU	Time(s)
No constraints	-	4.00	32.9	79	-	7.39	34.6	79	-	7.85	33.3	61
NAIVEATT	28.6	84.41	36.0	98	31.4	87.29	37.1	100	27.0	83.86	35.2	87
PRIORATT	36.8	94.21	37.1	104	39.0	92.49	38.2	108	32.1	86.01	35.9	90
SHIFTATT	39.5	96.77	37.6	208	41.9	93.62	38.0	160	34.5	89.97	35.9	150
SHIFTAET	41.0	97.75	37.8	223	42.6	93.92	38.1	165	35.5	91.43	36.2	157
POSTALN	41.4	97.78	37.8	177	42.2	93.66	38.1	126	35.2	90.47	36.1	111
VDBA	45.6	98.74	38.0	197	48.5	99.33	38.6	112	37.8	98.65	36.3	108
Align-VDBA	**46.1**	99.02	37.9	233	**49.2**	99.20	38.7	130	**38.5**	98.58	36.6	125

Table 3: Constrained translation results showing SpanBLEU, CSR (Constraint Satisfaction Rate), BLEU scores and total decoding time (in seconds) for the test set. Align-VDBA has the highest SpanBLEU on all datasets.

pled constraint sets. We show the standard deviation of the metrics across these runs in the Appendix E. The beam-size is set to five by default but for de→en we use ten since it provided significantly higher BLEU scores. Results for beam-size 5 for de→en appear in the Appendix E.

Methods Compared: First we compare all the alignment methods presented in Section 4.2 on the constrained translation task using the alignment based token-replacement algorithm of Song et al. [2020] described in Section 3.1. Next, we present a comparison between VBDA [Hu et al., 2019] and our modification Align-VDBA.

Results: Table 3 shows that VDBA and our Align-VDBA that pro-actively enforce constraints have a much higher CSR and higher SpanBLEU compared to the other lazy constraint enforcement methods. Within the lazy methods, those based on posterior alignment provide higher BLEU than prior alignment. POSTALN performs as well as SHIFTAET, with an almost equal BLEU (difference ≤ 0.1) and CSR (difference $\leq 1\%$). But, by avoiding the additional decoder pass for each token, it is more than 20% faster. On average, Align-VDBA has a 0.6 point greater SpanBLEU compared to VDBA. It also has a greater BLEU, on average, than VDBA and statistically comparable CSRs (less than 1 constraint on average). In Table 4, we compare some example translations produced by VDBA vs Align-VDBA. We observe instances where VDBA places constraints at the end of the translated sentence (e.g., "pusher", "de-

Constraints	(gesetz zur, **law also**), (dealer, **pusher**)
Gold	of course, if a drug addict becomes a **pusher**, then it is right and necessary that he should pay and answer before the **law also**.
VDBA	certainly, if a drug addict becomes a dealer, it is right and necessary that he should be brought to justice before the **law also pusher**.
Align-VDBA	certainly, if a drug addict becomes a **pusher**, then it is right and necessary that he should be brought to justice before the **law also**.
Constraints	(von mehrheitsverfahren, **of qualified**)
Gold	... whether this is done on the basis of a vote or of consensus, and whether unanimity is required or some form **of qualified** majority.
VDBA	... whether this is done by means **of qualified** votes or consensus, and whether unanimity or form of majority procedure apply.
Align-VDBA	... whether this is done by voting or consensus, and whether unanimity or form **of qualified** majority voting are valid.
Constraints	(zustimmung der, **strong backing of**)
Gold	... which were adopted with the **strong backing of** the ppe group and the support of the socialist members.
VDBA	... which were then adopted with broad agreement from the ppe group and with the **strong backing of** the socialist members.
Align-VDBA	... which were then adopted with **strong backing of** the ppe group and the support of the socialist members.
Constraints	(den usa, **the usa**), (sicherheitssystems an, **security system that**), (entwicklung, **development**)
Gold	matters we regard as particularly important are improving the working conditions between the weu and the eu and the **development** of a european **security system that** is not dependent on **the usa** .
VDBA	we consider **the usa** 's european security system to be particularly important in improving working conditions between the weu and the eu and developing a european **security system that** is independent of the united states **development** .
Align-VDBA	we consider the **development** of the **security system that** is independent of **the usa** to be particularly important in improving working conditions between the weu and the eu .

Table 4: Anecdotes showing constrained translations produced by VDBA vs. Align-VDBA.

velopment") unlike Align-VDBA. It is also interesting to see that in some cases where constraints contain frequent stop words (like of, the, etc.) appearing multiple times in the translated sentence, VDBA picks the token in the wrong position to tack on the constraint (e.g., "strong backing of", "of qualified") while Align-VDBA places the constraint correctly.

5 Related Work

Online Prior Alignment from NMTs: Zenkel et al. [2019] find alignments using a single-head attention submodule, optimized to predict the next token. Garg et al. [2019] and Song et al. [2020] supervise a single alignment head from the penultimate multi-head attention with prior alignments from GIZA++ alignments or FastAlign. Bahar et al. [2020] and Shankar et al. [2018] treat alignment as a latent variable and impose a joint distribution over token and alignment while supervising on the token marginal of the joint distribution.

Online Posterior Alignment from NMTs: Shankar and Sarawagi [2019] first identify the role of posterior attention for more accurate alignment. However, their NMT was a single-headed RNN. Chen et al. [2020] implement posterior attention in a multi-headed Transformer but they incur a delay of one step between token output and alignment. We are not aware of any prior work that extracts truly online posterior alignment in modern NMTs.

Offline Alignment Systems: Several recent methods apply only in the offline setting: Zenkel et al. [2020] extend an NMT with an alignment module; Nagata et al. [2020] frame alignment as a question answering task; and Jalili Sabet et al. [2020], Dou and Neubig [2021] leverage contextual embeddings from pretrained multilangual models.

Lexicon Constrained Translation: Hokamp and Liu [2017] and Post and Vilar [2018], Hu et al. [2019] modify beam search to ensure that target phrases from a given constrained lexicon are present in the translation. These methods ignore alignment with the source but ensure high success rate for appearance of the target phrases in the constraint. Song et al. [2020] and Chen et al. [2021] do consider source alignment but they do not enforce constraints leading to lower CSR. Dinu et al. [2019] and Lee et al. [2021] propose alternative training strategies for constraints, whereas we focus on working with existing models. Recently, non autoregressive methods have been proposed for enforcing target constraints but they require that the constraints are given in the order they appear in the target translation [Susanto et al., 2020].

6 Conclusion

In this paper we proposed a simple architectural modification to modern NMT systems to obtain accurate online alignments. The key idea that led to high alignment accuracy was conditioning on the output token. Further, our designed alignment module enables such conditioning to be performed synchronously with token generation. This property led us to Align-VDBA, a principled decoding algorithm for lexically constrained translation based on joint distribution of target token and source alignments. Future work includes harnessing such joint distributions for other forms of constraints, for example, nested constraints that arise when translating structured documents.

References

T. Alkhouli, G. Bretschner, and H. Ney. On the alignment problem in multi-head attention-based neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 177–185, Brussels, Belgium, Oct. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6318. URL https://aclanthology.org/W18-6318.

P. Bahar, N. Makarov, and H. Ney. Investigation of transformer-based latent attention models for neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 7–20, Virtual, Oct. 2020. Association for Machine Translation in the Americas. URL https://aclanthology.org/2020.amta-research.2.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993. URL https://aclanthology.org/J93-2003.

G. Chen, Y. Chen, and V. O. Li. Lexically constrained neural machine translation with explicit alignment guidance. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12630–12638, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17496.

Y. Chen, Y. Liu, G. Chen, X. Jiang, and Q. Liu. Accurate word alignment induction from neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 566–576, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.42. URL https://aclanthology.org/2020.emnlp-main.42.

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL https://aclanthology.org/W14-4012.

J. Crego, J. Kim, G. Klein, A. Rebollo, K. Yang, J. Senellart, E. Akhanov, P. Brunelle, A. Coquard, Y. Deng, S. Enoue, C. Geiss, J. Johanson, A. Khalsa, R. Khiari, B. Ko, C. Kobus, J. Lorieux, L. Martins, D.-C. Nguyen, A. Priori, T. Riccardi, N. Segal, C. Servan, C. Tiquet, B. Wang, J. Yang, D. Zhang, J. Zhou, and P. Zoldan. Systran's pure neural machine translation systems, 2016.

S. Ding, H. Xu, and P. Koehn. 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, Aug. 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5201. URL https://aclanthology.org/W19-5201.

G. Dinu, P. Mathur, M. Federico, and Y. Al-Onaizan. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1294. URL https://aclanthology.org/P19-1294.

Z.-Y. Dou and G. Neubig. Word alignment by fine-tuning embeddings on parallel corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online, Apr. 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.eacl-main.181.

C. Dyer, V. Chahuneau, and N. A. Smith. 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, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/N13-1073.

S. Garg, S. Peitz, U. Nallasamy, and M. Paulik. 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 4453–4462, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1453. URL https://aclanthology.org/D19-1453.

E. Hasler, A. de Gispert, G. Iglesias, and B. Byrne. 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, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2081. URL https://aclanthology.org/N18-2081.

F. Hieber, T. Domhan, M. Denkowski, and D. Vilar. Sockeye 2: A toolkit for neural machine translation. In *Proceedings of the 22nd Annual Conference of the European Association for Machine

Translation, pages 457–458, Lisboa, Portugal, Nov. 2020. European Association for Machine Translation. URL https://aclanthology.org/2020.eamt-1.50.

C. Hokamp and Q. Liu. 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, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1141. URL https://aclanthology.org/P17-1141.

J. E. Hu, H. Khayrallah, R. Culkin, P. Xia, T. Chen, M. Post, and B. Van Durme. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1090. URL https://aclanthology.org/N19-1090.

M. Jalili Sabet, P. Dufter, F. Yvon, and H. Schütze. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.147. URL https://aclanthology.org/2020.findings-emnlp.147.

N. Kalchbrenner and P. Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1176.

P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *International Workshop on Spoken Language Translation (IWSLT) 2005*, 2005. URL https://www.isca-speech.org/archive/iwslt_05/papers/slt5_068.pdf.

A. Kunchukuttan, P. Mehta, and P. Bhattacharyya. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://aclanthology.org/L18-1548.

G. Lee, S. Yang, and E. Choi. Improving lexically constrained neural machine translation with source-conditioned masked span prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 743–753, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.94. URL https://aclanthology.org/2021.acl-short.94.

J. Martin, R. Mihalcea, and T. Pedersen. Word alignment for languages with scarce resources. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 65–74, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0809.

R. Mihalcea and T. Pedersen. An evaluation exercise for word alignment. In *Proceedings of the HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pages 1–10, 2003. URL https://aclanthology.org/W03-0301.

M. Müller. Treatment of markup in statistical machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 36–46, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4804. URL https://aclanthology.org/W17-4804.

M. Nagata, K. Chousa, and M. Nishino. A supervised word alignment method based on cross-language span prediction using multilingual BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 555–565, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.41. URL https://aclanthology.org/2020.emnlp-main.41.

G. Neubig. The Kyoto free translation task, 2011. URL http://www.phontron.com/kftt.

F. J. Och and H. Ney. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, Oct. 2000. Association for Computational Linguistics. doi: 10.3115/1075218.1075274. URL https://aclanthology.org/P00-1056.

F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003. doi: 10.1162/089120103321337421. URL https://aclanthology.org/J03-1002.

M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*

(Demonstrations), pages 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4009. URL https://aclanthology.org/N19-4009.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

M. Post and D. Vilar. 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, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1119. URL https://aclanthology.org/N18-1119.

R. Sennrich, B. Haddow, and A. Birch. 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, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.

S. Shankar and S. Sarawagi. Posterior attention models for sequence to sequence learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BkltNhC9FX.

S. Shankar, S. Garg, and S. Sarawagi. Surprisingly easy hard-attention for sequence to sequence learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 640–645, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1065. URL https://aclanthology.org/D18-1065.

X. Shen, Y. Zhao, H. Su, and D. Klakow. Improving latent alignment in text summarization by generalizing the pointer generator. 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 3762–3773, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1390. URL https://aclanthology.org/D19-1390.

K. Song, K. Wang, H. Yu, Y. Zhang, Z. Huang, W. Luo, X. Duan, and M. Zhang. Alignment-enhanced transformer for constraining nmt with pre-specified translations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8886–8893, Apr. 2020. doi: 10.1609/aaai.v34i05.6418. URL https://ojs.aaai.org/index.php/AAAI/article/view/6418.

R. H. Susanto, S. Chollampatt, and L. Tan. Lexically constrained neural machine translation with Levenshtein transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3536–3543, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.325. URL https://aclanthology.org/2020.acl-main.325.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

D. Vilar, M. Popović, and H. Ney. Aer: Do we need to "improve" our alignments? In *International Workshop on Spoken Language Translation (IWSLT) 2006*, 2006. URL https://www-i6.informatik.rwth-aachen.de/publications/download/277/Vilar-IWSLT-2006.pdf.

T. Zenkel, J. Wuebker, and J. DeNero. Adding interpretable attention to neural translation models improves word alignment, 2019. URL https://arxiv.org/pdf/1901.11359.pdf.

T. Zenkel, J. Wuebker, and J. DeNero. End-to-end neural word alignment outperforms GIZA++. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1605–1617, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.146. URL https://aclanthology.org/2020.acl-main.146.

A Alignment Error Rate

Given gold alignments consisting of sure alignments $\mathcal{S}$ and possible alignments $\mathcal{P}$, and the predicted alignments $\mathcal{A}$, the Alignment Error Rate (AER) is defined as [Och and Ney, 2000]:

$$\text{AER} = 1 - \frac{|\mathcal{A} \cap \mathcal{P}| + |\mathcal{A} \cap \mathcal{S}|}{|\mathcal{A}| + |\mathcal{S}|}$$

Note that here $\mathcal{S} \subseteq \mathcal{P}$. Also note that since our models are trained on sub-word units but gold alignments are over words, we need to convert alignments between word pieces to alignments between words. A source word and target word are said to be aligned if there exists an alignment link between any of their respective word pieces.

B Description of the Datasets in Table 1

The European languages consist of parallel sentences for three language pairs from the Europarl Corpus and alignments from Mihalcea and Pedersen [2003], Och and Ney [2000]. Following previous works [Ding et al., 2019, Chen et al., 2020], the last 1000 sentences of the training data are used as validation data.

For English-Hindi, we use the dataset from Martin et al. [2005] consisting of 3440 training sentence pairs, 25 validation and 90 test sentences with gold alignments. Since training Transformers requires much larger datasets, we augment the training set with 1.6 million sentences from the IIT Bombay Parallel Corpus [Kunchukuttan et al., 2018].

For Japanese-English, we use The Kyoto Free Translation Task [Neubig, 2011]. It comprises roughly 330K training, 1166 validation and 1235 test sentences. As with other datasets, gold alignments are available only for the test sentences. The Japanese text is already segmented and we use it without additional changes. The gold alignments were provided by Mihalcea and Pedersen [2003] and Vilar et al. [2006].

C Bidirectional Symmetrized Alignment

We report AERs using bidirectional symmetrized alignments in Table 5 in order to provide fair comparisons to results in prior literature. The symmetrization is done using the *grow-diagonal* heuristic [Koehn et al., 2005, Och and Ney, 2000]. Since bidirectional alignments need the entire text in both languages, these are not online alignments.

Method	de-en	en-fr	ro-en	en-hi	ja-en
Statistical Methods					
GIZA++	18.6	5.5	26.3	35.9	39.7
FastAlign	27.0	10.5	32.1	-	-
No Alignment Training					
NAIVEATT	29.2	16.9	31.4	46.0	57.1
SHIFTATT	16.9	7.8	24.3	36.4	46.2
With Alignment Training					
PRIORATT	22.0	10.1	26.3	34.9	48.2
SHIFTAET	15.4	5.6	**21.0**	31.9	40.1
POSTALN	**15.3**	**5.5**	**21.0**	**30.9**	**39.5**

Table 5: AERs for bidirectional symmetrized alignments. POSTALN is consistently the best performing method.

D Alignment-based Token Replacement Algorithm

The pseudocode for the algorithm used in Song et al. [2020], Chen et al. [2021] and our non-VDBA based methods in Section 4.3 is presented in Algorithm 2. As described in Section 3.1, at each decoding step, if the source token having the maximum alignment at the current step lies in some constraint span, the constraint in question is decoded until completion before resuming normal decoding.

Though different alignment methods are represented using a call to the same ATTENTION function in Algorithm 2, these methods incur varying computational overheads. For instance, NAIVEATT incurs little additional cost, PRIORATT and POSTALN involve a multi-head attention computation. For SHIFTATT and SHIFTAET, an entire decoder pass is done when ATTENTION is called, thereby incurring a huge overhead as shown in Table 3.

E Additional Lexicon-Constrained Translation Results

Constrained translation results for de→en with beam-size 5 are shown in Table 6. The standard deviations for Table 3 are shown in Table 7.

Algorithm 2 k-best extraction with argmax replacement decoding.

Inputs: A $k \times |V_T|$ matrix of scores (for all tokens up to the currently decoded ones). k beam states.

```
function SEARCH_STEP(beam, scores)
    next_toks, next_scores ← ARGMAX_K(scores, k=2, dim=1)          ▷ Best 2 tokens for each beam
    candidates ← []
    for 0 ≤ h < 2 · k do
        candidate ← beam[h//2]
        candidate.tokens.append(next_toks[h//2, h%2])
        candidate.scores ← next_scores[h//2, h%2]
        candidates.append(candidate)
    attention ← ATTENTION(candidates)
    aligned_x ← ARGMAX(attention, dim=1)
    for 0 ≤ h < 2 · k do
        if aligned_x[h] ∈ C_i^x for some i and not candidates[h].inprogress then      ▷ Start constraint
            candidates[h].inprogress ← True
            candidates[h].constraintNum ← i
            candidates[h].tokenNum ← 0
        if candidates[h].inprogress then                          ▷ Replace token with constraint tokens
            candidates[h].tokens[-1] ← constraints[candidates[h].constraintNum][candidates[h].tokenNum]
            candidates[h].tokenNum ← candidates[h].tokenNum + 1
            if constraints[candidates[h].constraintNum].length == candidates[h].tokenNum then
                candidates[h].inprogress ← False                  ▷ Finish current constraint
    candidates ← REMOVE_DUPLICATES(candidates)
    newBeam ← TOP_K(candidates)
    return newBeam
```

Method	SpanBLEU	CSR	BLEU	Time(s)
No constraints	-	4.86	32.9	103
NAIVEATT	29.1	84.82	35.9	136
PRIORATT	36.9	94.22	37.1	150
SHIFTATT	39.2	96.88	37.5	246
SHIFTAET	40.7	97.65	37.6	257
POSTALN	**41.0**	97.56	37.7	195
VDBA	39.7	99.37	37.2	192
Align-VDBA	40.6	99.52	37.2	217

Table 6: Constrained translation results using a beam size of 5 for German-English.

F Layer Selection for Alignment Supervision of Distant Language Pairs

For the alignment supervision, we used alignments extracted from vanilla Transformers using the SHIFTATT method. To do so, however, we need to choose the decoder layers from which to extract the alignments. The validation AERs can be used for this purpose but since gold validation alignments are not available, Chen et al. [2020] suggest selecting the layers which have the best consistency between the alignment predictions from the two translation directions.

For the European language pairs, this turns out to be layer 3 as suggested by Chen et al. [2020]. However, for the distant language pairs Hindi-English and Japanese-English, this is not the case and layer selection needs to be done. The AER between the two translation directions on the validation set, with alignments obtained from different decoder layers, are shown in Tables 8 and 9.

Method	de→en				en→fr				ro→en			
	SpanBLEU	CSR	BLEU	Time(s)	SpanBLEU	CSR	BLEU	Time(s)	SpanBLEU	CSR	BLEU	Time(s)
No constraints	-	1.5	0.0	4.2	-	1.3	0.0	1.7	-	1.8	0.0	5.6
NAIVEATT	1.5	2.1	0.2	4.4	1.1	7.1	0.2	1.3	1.3	2.1	0.4	3.2
PRIORATT	1.7	1.3	0.4	3.5	1.8	0.4	0.0	5.3	1.1	1.8	0.4	2.8
SHIFTATT	1.1	0.6	0.4	9.5	1.3	1.6	0.2	1.9	1.3	1.2	0.2	5.7
SHIFTAET	1.3	0.6	0.3	17.9	1.2	1.4	0.2	3.0	2.0	0.9	0.3	7.0
POSTALN	1.6	0.8	0.4	8.5	1.9	1.6	0.2	5.5	1.1	1.7	0.6	1.8
VDBA	1.0	0.5	0.4	12.6	1.6	0.4	0.3	5.4	1.8	0.8	0.5	3.0
Align-VDBA	0.9	0.6	0.4	24.9	1.7	0.6	0.3	1.0	1.4	0.4	0.4	2.4

Table 7: Standard deviations of the metrics shown in Table 3 across five sets of randomly sampled constraint sets.

	1	2	3	4	5	6
1	65.5	55.8	56.1	95.2	94.6	96.6
2	59.2	47.5	**44.5**	95.1	91.9	95.8
3	62.6	52.1	48.3	93.7	91.4	95.2
4	88.6	83.3	82.1	89.9	88.0	90.3
5	91.6	87.7	88.5	91.4	88.8	90.2
6	93.5	91.1	92.5	92.5	90.5	90.7

Table 8: AER between en→hi and hi→en SHIFTATT alignments on the validation set for EnHi

	1	2	3	4	5	6
1	93.5	90.0	94.4	92.2	95.1	95.1
2	86.5	**58.7**	86.9	69.4	87.2	86.2
3	87.4	59.4	87.1	69.1	87.1	86.2
4	89.1	69.1	85.9	74.2	84.9	85.4
5	93.4	88.5	89.1	87.1	86.8	88.1
6	93.5	89.4	90.0	88.1	87.7	88.7

Table 9: AER between ja→en and en→ja SHIFTATT alignments on the validation set for JaEn