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 seven lexically constrained translation tasks, we achieve significant improvements in BLEU specifically around the constrained positions.

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). Our method (Align-VDBA) presents a significant departure from existing papers on alignment-guided constrained translation (Chen et al., 2020; Song et al., 2020) that employ a greedy algorithm with poor constraint satisfaction rate (CSR). For example, on a ja→en their CSR is 20 points lower than ours. Moreover, the latter does not benefit from larger beam sizes unlike VDBA-based methods that significantly improve with larger beam widths. Compared to Chen et al. (2020), our method improves average overall BLEU scores by 1.2 points and average BLEU scores around the constrained span by up to 9 points. In the evaluations performed in these earlier work, VDBA was not allocated the slightly higher beam size needed to pro-actively enforce constraints without compromising BLEU. Compared to Hu et al. (2019) (VDBA), this paper's contributions include online alignments and their use in more fluent constraint placement.

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 where we improve over the best existing in seven out of ten translation models.
- Principled method of modifying VDBA to incorporate posterior alignment probabilities in lexically-constrained decoding. VDBA enforces constraints ignoring source alignments, our change (Align-VDBA), leads to more fluent constraint placement.
- Establishing that VDBA-based pro-active constrained inference should be preferred over prevailing greedy alignment-guided inference (Chen et al., 2021; Song et al., 2020) when high constraint satisfaction rate (CSR) is important to the end-user. Further, VDBA and our Align-VDBA inference with beam size 10 provide 1.2 BLEU increase over these methods with the same beam size.

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 timestep 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

Transformers (Vaswani et al., 2017) 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 by $[\mathbf{Z}_t^{\ell,1}, \ldots, \mathbf{Z}_t^{\ell,\eta}]\mathbf{W}^O$ where $[\ ]$ denotes the column-wise concatenation of matrices and $\mathbf{W}^O \in \mathbb{R}^{d \times d}$ is an output projection matrix.

[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.

2.1.2 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 $\text{argmax}_j \frac{1}{\eta}\sum_{n=1}^{\eta} P(a_{t,j}^{\ell,n}|\mathbf{x},\mathbf{y}_{<t})$ where j indexes the source tokens. 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. Alignment quality 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.

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.

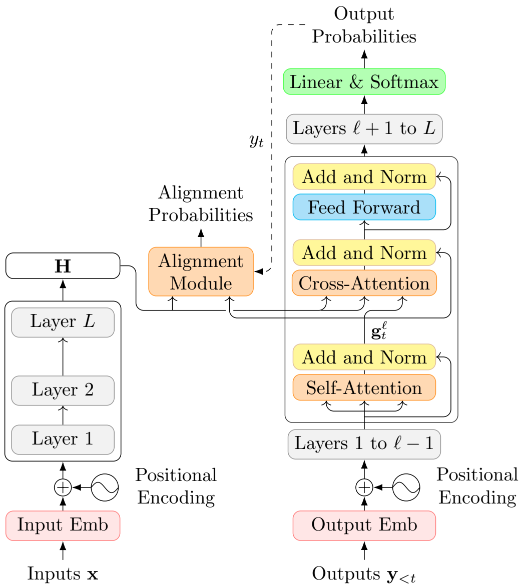

Figure 1: Our alignment module is an encoder-decoder attention sub-layer, similar to the existing cross-attention sub-layer. It takes as inputs the encoder output $\mathbf{H}$ as the key, and the concatenation of the output of the previous self-attention layer $\mathbf{g}_t^\ell$ and the currently decoded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

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.

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:

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

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

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 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

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 $\mathcal{C}_j^x$, the output token is fixed to the first token y_1^j from $\mathcal{C}_j^y$. Otherwise, the decoding proceeds as usual. Also, if the translation of a constraint $\mathcal{C}_j$ has started, the same is completed (y_2^j through $y_{m_j}^j$) for the next $m_j - 1$ decoding steps before resuming unconstrained beam search. The pseudocode for this method is provided in Appendix G.

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 $\mathcal{C}_j^x$ they *revise* the output token to be y_1^j from $\mathcal{C}_j^y$, 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_1^j. 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 pro-active 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 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.* $y_t \in \mathcal{C}_j^y$, 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 $\mathcal{C}_j^x$, the source span for constraint i. Thus, the updated probability is given by:

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

$P(y_t, \mathcal{C}_j^x|\mathbf{x}, \mathbf{y}_{<t})$ 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 $\propto \Pr(a_{t,r}^{\text{post}}|\mathbf{x}, \mathbf{y}_{\leq t})^{\frac{1}{T}}$. We used $T = 2$ i.e., square-

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

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

root of the 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, posterior 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 $5e{-}4$ with 4000 warm-up steps, an inverse square root schedule, weight decay of $1e{-}4$, 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`.

	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	140	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.

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.

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 (en-hi) and English-Japanese (ja-en), that follow a SOV word order which is different from the SVO word order of English. Data statistics are shown in Table 1 and details are in Appendix C.

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

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	49.1	53.8	62.2	63.5
ShiftAtt (Chen et al., 2020)	+1	20.0	22.9	14.7	20.4	26.9	27.4	35.3	38.6	53.6	48.6
With Alignment Training											
PriorAtt	0	23.4	25.8	14.0	16.6	29.3	27.2	36.4	35.1	52.7	50.9
ShiftAET (Chen et al., 2020)	+1	15.8	**19.5**	10.3	**10.4**	22.4	23.7	29.3	29.3	42.5	**41.9**
PostAln [Ours]	0	**15.5**	**19.5**	**9.9**	**10.4**	**21.8**	**23.2**	**28.7**	**28.9**	**41.2**	42.2

Table 2: AER for de-en, en-fr, ro-en, en-hi, ja-en language pairs. “Delay" indicates the decoding step at which the alignment of the target token is available. NaiveAtt, PriorAtt and PostAln are truly online and output alignment at the same time step (delay=0), while ShiftAtt and ShiftAET output one decoding step later.

our focus is online alignment, for comparison to previous works, we also report results on bidirectional symmetrized alignments in Appendix D.

Methods compared: We compare our method with both existing statistical alignment models, 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 delays computations by one time-step as in ShiftAtt, and additionally includes a learned attention sub-layer to compute alignment probabilities. 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. (GIZA++ has remained a state-of-the-art alignment technique and continues to be compared against.) 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 an improvement of 4.0 to 8.0 points. 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 (achieving the lowest AER in nine out of ten cases in Table 2) while avoiding the one-step delay in alignment generation. Even on the distant languages, PostAln achieves significant reductions in error. For ja→en, we achieve a 1.3 AER reduction compared to ShiftAET which is not a truly online method. Figure 2 shows an example 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) (Song et al., 2020), 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

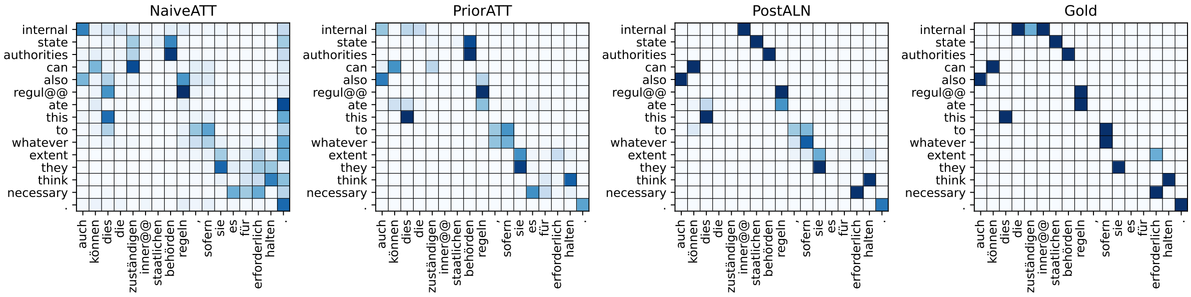

Figure 2: Alignments for de→en by NAIVEATT, PRIORATT, and POSTALN. Note that POSTALN is most similar to Gold alignments in the last column.

	de→en				en→fr				ro→en				en→hi				ja→en			
Method	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time
No constraints	0.0	4.6	32.9	87	0.0	8.7	34.8	64	0.0	8.8	33.4	47	0.0	6.3	19.7	21	0.0	8.8	18.9	237
NAIVEATT	28.7	86.1	36.6	147	36.5	88.0	38.3	93	33.3	92.3	36.5	99	22.5	88.4	23.6	27	15.1	75.9	20.2	315
PRIORATT	35.0	92.8	37.6	159	42.1	94.4	38.9	97	36.0	91.2	37.2	100	27.2	91.5	24.4	28	16.7	79.7	20.4	326
SHIFTATT	41.0	96.6	38.7	443	45.0	93.5	38.7	239	39.2	94.2	37.4	241	23.2	78.7	21.9	58	15.2	72.7	19.3	567
SHIFTAET	43.1	97.5	39.1	458	46.6	94.3	39.0	235	40.8	94.4	37.6	263	24.3	80.2	22.0	62	18.1	75.9	19.7	596
POSTALN	42.7	97.2	39.0	399	46.3	94.1	38.7	218	40.0	93.5	37.4	226	23.8	79.0	22.0	47	18.2	75.7	19.7	460
VDBA	**44.5**	98.9	38.5	293	51.9	98.5	39.5	160	43.1	99.1	37.9	165	29.8	92.3	24.5	49	24.3	95.6	21.6	494
Align-VDBA	**44.5**	98.6	38.6	357	**52.9**	98.4	39.7	189	**44.1**	98.9	38.1	203	**30.5**	91.5	24.7	70	**25.1**	95.5	21.8	630

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.

randomly sampled constraint sets. Standard deviations across all runs are listed in Appendix E. The beam size is set to ten by default; results for other beam-sizes appear in 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. For example, for ja→ en greedy methods can only achieve a CSR of 76% compared to 96% of the VDBA-based methods. In terms of overall BLEU too these methods provide an average increase in BLEU of 1.2 and an average increase in SpanBLEU of 5 points. On average, Align-VDBA has a 0.7 point greater SpanBLEU compared to VDBA. It also has a greater BLEU than VDBA on all the five datasets and statistically comparable CSRs (difference less than 1 constraint on average). Table 4 lists 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", “development") 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.

Dataset →	IATE.414		Wiktionary.727	
Method (Beam Size) ↓	CSR	BLEU (Δ)	CSR	BLEU (Δ)
Baseline (5)	76.3	25.8	76.9	26.0
Train-by-app. (5)	92.9	26.0 (+0.2)	90.7	26.9 (+0.9)
Train-by-rep. (5)	94.5	26.0 (+0.2)	93.4	26.3 (+0.3)
No constraints (10)	77.0	29.7	72.4	29.9
Align-VDBA (10)	99.8	30.8 (+1.1)	99.5	31.0 (+1.1)

Table 5: Constrained translation results on the two real world constraints from Dinu et al. (2019).

Real World Constraints: We also evaluate our method using real world constraints extracted from IATE and Wiktionary datasets by Dinu et al. (2019). In Table 5 we compare Align-VDBA with the soft-constraints method of Dinu et al. (2019) that requires special retraining to teach the model to copy constraints. We reproduced the numbers from their paper in the first three rows. Their baseline numbers are almost 4 BLEU points worse than our baseline since they used a smaller transformer NMT

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 with 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.

model, thus making running times incomparable. When we compare the increment Δ in BLEU over the respective baselines, Align-VDBA shows much greater gains of +1.1 vs. their +0.5. Also, Align-VDBA provides a much larger CSR of 99.6 compared to their 92. Results for other beam sizes and other methods appear in Appendix F.

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 and projecting HTML tags from source to target sentences.

References

Tamer Alkhouli, Gabriel Bretschner, and Hermann Ney. 2018. 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. Association for Computational Linguistics.

Parnia Bahar, Nikita Makarov, and Hermann Ney. 2020. 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. Association for Machine Translation in the Americas.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. 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*.

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

Guanhua Chen, Yun Chen, and Victor O.K. Li. 2021. Lexically constrained neural machine translation with explicit alignment guidance. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12630–12638.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020. 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. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. 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. Association for Computational Linguistics.

Josep Crego, Jungi Kim, Guillaume Klein, Anabel Rebollo, Kathy Yang, Jean Senellart, Egor Akhanov, Patrice Brunelle, Aurelien Coquard, Yongchao Deng, Satoshi Enoue, Chiyo Geiss, Joshua Johanson, Ardas Khalsa, Raoum Khiari, Byeongil Ko, Catherine Kobus, Jean Lorieux, Leidiana Martins, Dang-Chuan Nguyen, Alexandra Priori, Thomas Riccardi, Natalia Segal, Christophe Servan, Cyril Tiquet, Bo Wang, Jin Yang, Dakun Zhang, Jing Zhou, and Peter Zoldan. 2016. Systran's pure neural machine translation systems.

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

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. 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. Association for Computational Linguistics.

Zi-Yi Dou and Graham Neubig. 2021. 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. Association for Computational Linguistics.

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

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4453–4462, Hong Kong, China. Association for Computational Linguistics.

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

Felix Hieber, Tobias Domhan, Michael Denkowski, and David Vilar. 2020. 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. European Association for Machine Translation.

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

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin

Van Durme. 2019. 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. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. 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. Association for Computational Linguistics.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

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

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2018. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Gyubok Lee, Seongjun Yang, and Edward Choi. 2021. 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. Association for Computational Linguistics.

Joel Martin, Rada Mihalcea, and Ted Pedersen. 2005. 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. Association for Computational Linguistics.

Rada Mihalcea and Ted Pedersen. 2003. 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.

Mathias Müller. 2017. Treatment of markup in statistical machine translation. In *Proceedings of the Third Workshop on Discourse in Machine Translation*, pages 36–46, Copenhagen, Denmark. Association for Computational Linguistics.

Masaaki Nagata, Katsuki Chousa, and Masaaki Nishino. 2020. 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. Association for Computational Linguistics.

Graham Neubig. 2011. The Kyoto free translation task.

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

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

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. 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. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. 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. Association for Computational Linguistics.

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, New Orleans, Louisiana. Association for Computational Linguistics.

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

Shiv Shankar, Siddhant Garg, and Sunita Sarawagi. 2018. 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. Association for Computational Linguistics.

Shiv Shankar and Sunita Sarawagi. 2019. Posterior attention models for sequence to sequence learning. In *International Conference on Learning Representations*.

Xiaoyu Shen, Yang Zhao, Hui Su, and Dietrich Klakow. 2019. 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. Association for Computational Linguistics.

Kai Song, Kun Wang, Heng Yu, Yue Zhang, Zhongqiang Huang, Weihua Luo, Xiangyu Duan, and Min Zhang. 2020. Alignment-enhanced transformer for constraining nmt with pre-specified translations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8886–8893.

Raymond Hendy Susanto, Shamil Chollampatt, and Liling Tan. 2020. 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. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

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*, volume 30. Curran Associates, Inc.

David Vilar, Maja Popović, and Hermann Ney. 2006. AER: Do we need to “improve” our alignments? In *International Workshop on Spoken Language Translation (IWSLT) 2006*.

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

Thomas Zenkel, Joern Wuebker, and John DeNero. 2020. 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. Association for Computational Linguistics.

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):

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

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 SpanBLEU

Given a reference sentence, a predicted translation and a set of constraints, for each constraints, a segment of the sentence is chosen which contains the constraint and window size words (if available) surrounding the constraint words on either side. Such segments, called spans, are collected for the reference and predicted sentences in the test and BLEU is computed over these spans. If a constraint is not satisfied in the prediction, the corresponding span is considered to be the empty string. An example is shown in Table 6. Table 7 shows how SpanBLEU varies as a function of varying window size for a fixed English-French constraint set with beam size set to 10.

Window Size →	2	3	4	5	6	7	8
No constraints	0.0	0.0	0.0	0.0	0.0	0.0	0.0
NaiveAtt	34.4	32.0	30.4	29.5	29.4	29.5	29.7
PriorAtt	41.5	38.7	36.4	35.1	34.9	35.0	35.2
ShiftAtt	44.9	41.5	38.9	37.3	36.4	36.2	36.0
ShiftAet	47.0	43.2	40.4	38.7	38.0	37.6	37.4
PostAln	46.4	42.7	39.8	38.0	37.1	36.9	36.6
VDBA	54.9	50.5	46.8	44.6	43.5	43.0	42.6
Align-VDBA	56.4	51.7	47.9	45.6	44.4	43.7	43.3

Table 7: SpanBLEU vs Window Size for a constraint set of English-French with beam size 10.

C Description of the Datasets

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), Vilar et al. (2006). 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). We also add the first 50 sentences from the dev set of IIT Bombay Parallel Corpus with manually annotated alignments to the test set giving a total of 140 test sentences.

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 real world constraints datasets of Dinu et al. (2019) are extracted from the German-English WMT newstest 2017 task with the IATE dataset consisting of 414 sentences (451 constraints) and the Wiktionary 727 sentences (879 constraints). The constraints come from the IATE and Wiktionary termninology databases.

D Bidirectional Symmetrized Alignment

We report AERs using bidirectional symmetrized alignments in Table 8 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	43.8	57.1
ShiftAtt	16.9	7.8	24.3	30.9	46.2
With Alignment Training					
PriorAtt	22.0	10.1	26.3	32.1	48.2
ShiftAet	15.4	5.6	**21.0**	26.7	40.1
PostAln	**15.3**	**5.5**	**21.0**	**26.1**	**39.5**

Table 8: AERs for bidirectional symmetrized alignments. PostAln consistently performs the best.

Reference	we consider the **development** of a robust **security system** that is independent of the	
Prediction	we consider developing a robust **security system** which is independent of the	
	SpanBLEU (Window Size = 2)	
Cons. No	Reference Spans	Predicted Spans
1	consider the **development** of a	(empty sentence)
2	a robust **security system** that is	a robust **security system** which is
SpanBLEU = BLEU(Reference Spans, Predicted Spans)		

Table 6: An example SpanBLEU computation

		de→en				en→fr				ro→en				en→hi				ja→en			
Beam Size	Method	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time
5	No constraints	0.0	5.0	32.9	78	0.0	8.7	34.6	61	0.0	8.4	33.3	45	0.0	5.6	19.7	18	0.0	7.9	19.1	221
	NaiveATT	28.9	86.2	36.7	127	36.7	88.6	38.0	87	32.9	91.8	36.3	88	23.0	89.9	23.9	25	15.1	77.0	20.3	398
	PriorATT	35.3	93.0	37.7	136	42.2	94.7	38.6	89	36.0	91.6	37.0	89	27.6	91.7	24.7	26	16.8	80.2	20.6	353
	ShiftATT	41.0	96.7	38.7	268	45.2	93.8	38.4	167	39.2	94.4	37.2	160	23.8	81.8	22.0	42	15.1	72.6	19.3	664
	ShiftAET	43.1	97.6	39.1	291	46.5	94.8	38.6	165	40.8	94.7	37.5	163	24.5	83.6	22.1	44	18.0	76.5	19.6	583
	PostAln	42.7	97.3	39.0	252	46.1	93.9	38.5	151	39.8	93.5	37.3	141	23.3	79.7	21.7	39	17.9	75.3	19.6	469
	VDBA	39.6	99.4	37.8	203	45.9	99.5	38.5	109	36.6	99.2	36.7	117	27.3	96.6	24.2	37	22.1	96.9	20.9	397
	Align-VDBA	40.3	99.0	38.0	244	47.4	99.3	38.7	132	37.6	99.7	36.8	139	27.2	95.6	24.1	46	22.5	97.2	21.0	460
10	No constraints	0.0	4.6	32.9	87	0.0	8.7	34.8	64	0.0	8.8	33.4	47	0.0	6.3	19.7	21	0.0	8.8	18.9	237
	NaiveATT	28.7	86.1	36.6	147	36.5	88.0	38.3	93	33.3	92.3	36.5	99	22.5	88.4	23.6	27	15.1	75.9	20.2	315
	PriorATT	35.0	92.8	37.6	159	42.1	94.4	38.9	97	36.0	91.2	37.2	100	27.2	91.5	24.4	28	16.7	79.7	20.4	326
	ShiftATT	41.0	96.6	38.7	443	45.0	93.5	38.7	239	39.2	94.2	37.4	241	23.2	78.7	21.9	58	15.2	72.7	19.3	567
	ShiftAET	43.1	97.5	39.1	458	46.6	94.3	39.0	235	40.8	94.4	37.6	263	24.3	80.2	22.0	62	18.1	75.9	19.7	596
	PostAln	42.7	97.2	39.0	399	46.3	94.1	38.7	218	40.0	93.5	37.4	226	23.8	79.0	22.0	47	18.2	75.7	19.7	460
	VDBA	44.5	98.9	38.5	293	51.9	98.5	39.5	160	43.1	99.1	37.9	165	29.8	92.3	24.5	49	24.3	95.6	21.6	494
	Align-VDBA	44.5	98.6	38.6	357	52.9	98.4	39.7	189	44.1	98.9	38.1	203	30.5	91.5	24.7	70	25.1	95.5	21.8	630
20	No constraints	0.0	4.9	32.8	84	0.0	8.4	34.8	69	0.0	8.7	33.2	50	0.0	6.5	19.5	20	0.0	8.2	18.9	255
	NaiveATT	28.8	86.1	36.6	133	36.4	88.1	38.3	118	33.4	92.1	36.6	126	23.4	90.1	24.0	34	15.0	75.5	20.1	403
	PriorATT	34.9	92.6	37.4	128	42.0	94.5	38.9	123	35.9	91.0	37.3	121	27.1	92.2	24.6	33	16.6	79.5	20.4	423
	ShiftATT	40.9	96.4	38.7	398	45.7	94.2	39.0	378	39.1	94.0	37.3	409	23.0	77.5	21.8	82	15.2	72.3	19.2	827
	ShiftAET	43.1	97.1	39.0	395	47.1	95.0	39.2	404	40.5	93.9	37.5	403	24.0	79.5	21.9	80	17.9	76.0	19.6	872
	PostAln	42.7	97.0	39.0	354	46.8	94.9	39.1	351	39.6	93.0	37.3	376	23.5	77.6	21.8	73	18.0	75.3	19.6	687
	VDBA	45.1	97.7	38.4	337	52.5	95.7	39.7	250	43.8	96.2	38.0	268	28.7	86.8	23.6	82	24.3	93.6	21.9	780
	Align-VDBA	45.2	97.3	38.4	400	52.5	95.1	39.5	292	44.8	96.3	38.2	330	29.2	85.8	23.5	107	24.7	93.2	21.8	870

Table 9: Lexically Constrained Translation Results with different beam sizes. All numbers are average over 5 randomly sampled constraint sets and running times are in seconds.

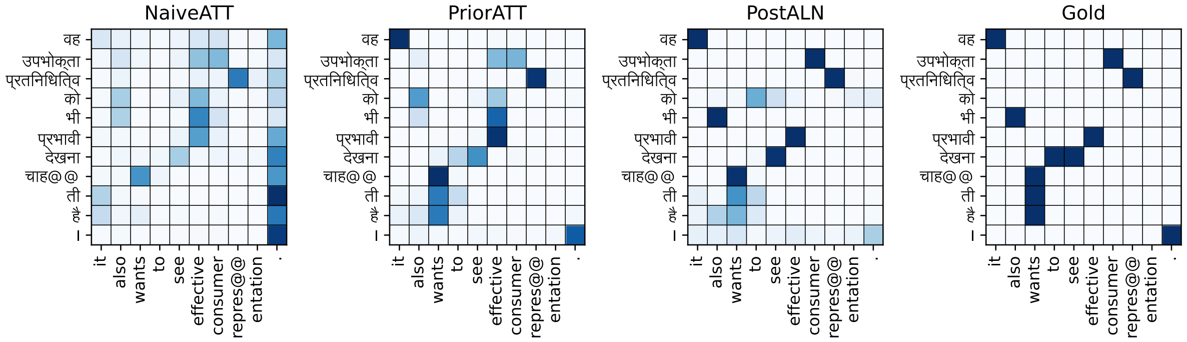

Figure 3: Alignments for en→hi by NaiveATT, PriorATT, and PostAln. Note that PostAln is most similar to Gold alignments in the last column.

E Additional Lexicon-Constrained Translation Results

Constrained translation results for beam sizes 5, 10 and 20 are shown in Table 9. The standard deviations for Table 3 are shown in Table 11.

F Additional Real World Constrained Translation Results

Results on the real world constrained translation datasets of Dinu et al. (2019) for all the methods in Table 3 with beam sizes 5, 10 and 20 are presented in Table 10.

	Dataset →	IATE.414				Wiktionary.727			
Beam Size	Method ↓	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time
5	No constraints	27.9	76.6	29.7	134	26.3	72.0	29.9	217
	NAIVEATT	29.2	96.9	29.2	175	29.0	95.3	29.1	341
	PRIORATT	31.2	97.1	29.7	198	32.2	95.9	29.9	306
	SHIFTATT	34.9	96.7	29.9	355	35.3	96.5	30.0	568
	SHIFTAET	35.2	96.3	30.0	378	35.8	97.1	30.2	637
	POSTALN	35.3	96.7	30.0	272	35.8	96.7	30.2	467
	VDBA	35.3	98.8	29.8	258	35.0	99.2	30.4	442
	Align-VDBA	35.4	99.8	29.8	280	35.1	99.3	30.3	534
10	No constraints	28.3	77.0	29.7	113	26.3	72.4	29.9	164
	NAIVEATT	28.9	97.3	29.1	145	29.2	95.3	29.1	269
	PRIORATT	31.3	96.9	29.5	155	32.3	96.0	29.9	260
	SHIFTATT	34.9	96.3	29.8	345	35.3	96.8	30.3	600
	SHIFTAET	35.2	95.9	29.9	350	35.9	97.2	30.4	664
	POSTALN	35.1	95.9	29.9	287	35.8	97.0	30.3	458
	VDBA	37.6	99.8	30.9	257	36.9	99.4	30.9	451
	Align-VDBA	37.5	99.8	30.8	353	37.3	99.5	31.0	540
20	No constraints	28.4	77.2	29.9	103	26.3	72.1	30.0	177
	NAIVEATT	28.9	96.9	29.0	188	29.1	95.4	29.3	325
	PRIORATT	31.3	96.9	29.6	203	32.6	96.4	30.1	338
	SHIFTATT	34.7	96.1	29.8	528	35.3	96.8	30.2	892
	SHIFTAET	35.0	95.8	29.9	539	36.1	97.3	30.4	923
	POSTALN	35.1	96.1	29.9	420	36.0	97.0	30.4	751
	VDBA	37.8	99.8	30.9	381	37.4	99.2	31.2	680
	Align-VDBA	37.9	99.8	30.9	465	38.0	99.5	31.3	818

Table 10: Additional results for the real world constraints for all methods and different beam sizes.

G 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.

H 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 12 and 13.

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 $0 \leq h < 2 \cdot k$ 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 $0 \leq h < 2 \cdot k$ do
        if aligned_x[h] $\in \mathcal{C}_i^x$ 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
```

	de→en				en→fr				ro→en				en→hi				ja→en			
Method	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time	Span BLEU	CSR	BLEU	Time
No constraints	0.0	0.6	0.0	8.9	0.0	2.2	0.0	0.8	0.0	1.7	0.0	2.0	0.0	1.8	0.0	2.4	0.0	0.7	0.0	28.1
NaiveATT	2.0	0.9	0.3	9.6	2.7	2.5	0.4	5.0	1.1	0.9	0.3	2.5	2.7	3.9	0.3	2.8	0.9	1.6	0.1	5.6
PriorATT	1.6	1.0	0.1	13.3	1.9	0.8	0.5	2.0	1.4	1.0	0.4	7.3	0.7	1.8	0.4	3.3	0.9	1.4	0.2	6.5
ShiftATT	1.6	0.6	0.3	35.7	2.8	1.3	0.4	20.2	1.5	1.0	0.6	14.8	2.3	3.9	0.5	6.1	0.4	1.4	0.1	7.0
ShiftAET	1.7	0.8	0.3	36.8	2.3	0.9	0.4	18.5	2.0	1.0	0.6	13.9	2.6	2.0	0.6	8.6	0.6	0.6	0.1	42.5
PostAln	1.8	0.6	0.3	12.8	2.3	0.9	0.4	9.9	1.5	1.1	0.6	26.7	2.6	2.6	0.6	5.0	0.6	1.0	0.1	11.0
VDBA	1.7	0.6	0.2	33.3	1.7	0.7	0.3	6.8	1.6	0.6	0.3	7.1	1.4	2.8	0.9	1.2	0.9	0.9	0.2	50.0
Align-VDBA	1.7	0.4	0.1	27.4	1.6	0.8	0.3	7.4	1.3	0.9	0.4	15.0	1.3	2.9	0.9	7.4	1.0	0.9	0.3	91.0

Table 11: 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 12: 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 13: AER between ja→en and en→ja ShiftATT alignments on the validation set for JaEn