MONOTONIC MULTIHEAD ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Simultaneous machine translation models start generating a target sequence before they have encoded or read the source sequence. Recent approaches for this task either apply a fixed policy on a state-of-the art Transformer model, or a learnable monotonic attention on a weaker recurrent neural network-based structure. In this paper, we propose a new attention mechanism, Monotonic Multihead Attention (MMA), which extends the monotonic attention mechanism to multihead attention. We also introduce two novel and interpretable approaches for latency control that are specifically designed for multiple attentions heads. We apply MMA to the simultaneous machine translation task and demonstrate better latency-quality tradeoffs compared to MILk, the previous state-of-the-art approach. We also analyze how the latency controls affect the attention span and we motivate the introduction of our model by analyzing the effect of the number of decoder layers and heads on quality and latency. Code will be released upon publication.

1 INTRODUCTION

Simultaneous machine translation adds the capability of a live interpreter to machine translation: a simultaneous machine translation model starts generating a translation before it has finished reading the entire source sentence. Such models are useful in any situation where translation needs to be done in real time. For example, simultaneous models can translate live video captions or facilitate conversations between people speaking different languages. In a usual neural machine translation model, the encoder first reads the entire sentence, and then the decoder writes the target sentence. On the other hand, a simultaneous neural machine translation model alternates between reading the input and writing the output using either a fixed or learned policy.

Monotonic attention mechanisms fall into learned policy category. Recent work exploring monotonic attention variants for simultaneous translation include: hard monotonic attention (Raffel et al., 2017), monotonic chunkwise attention (MoChA) (Chiu & Raffel, 2018) and monotonic infinite lookback attention (MILk) (Arivazhagan et al., 2019). MILk in particular has shown better quality / latency trade-offs than fixed policy approaches, such as wait-k (Ma et al., 2019) or wait-if-* (Cho & Esipova, 2016) policies. MILk also outperforms hard monotonic attention and MoChA; while the other two monotonic attention mechanisms only consider a fixed reading window, MILk computes a softmax attention over all previous encoder states, which may be the key to its improved latency-quality tradeoffs. These monotonic attention approaches also provide a closed form expression for the expected alignment between source and target tokens, and avoid unstable reinforcement learning.

However, monotonic attention-based models, including the state-of-the-art MILk, were built on top of RNN-based models. RNN-based models have been outperformed by the recent state-of-the-art Transformer model (Vaswani et al., 2017), which features multiple encoder-decoder attention layers and multihead attention at each layer.

We thus propose monotonic multihead attention (MMA), which combines the strengths of multilayer multihead attention and monotonic attention. We propose two variants, Hard MMA (MMA-H) and Infinite Lookback MMA (MMA-IL). MMA-H is designed with streaming systems in mind where the attention span must be limited. MMA-IL emphasizes the quality of the translation system. We also propose two novel latency regularization methods. The first encourages the model to be faster by directly minimizing the average latency. The second encourages the attention heads to maintain similar positions, preventing the latency from being dominated by a single or a few heads.

The main contributions of this paper are:

1. We introduce a novel monotonic attention mechanism, monotonic multihead attention, which enables the Transformer model to perform online decoding. This model leverages the power of the Transformer and the efficiency of monotonic attention.
2. We demonstrate better latency-quality tradeoffs compared to the MILk model, the previous state-of-the-art, on two standard translation benchmarks, IWSLT15 English-Vietnamese (En-Vi) and WMT15 German-English (De-En).
3. We provide analyses on how our model is able to control the attention span and we motivate the design of our model with an ablation study on the number of decoder layers and the number of decoder heads.

2 MONOTONIC MULTIHEAD ATTENTION MODEL

In this section, we review the monotonic attention-based approaches in RNN-based encoder-decoder models. We then introduce the two types of Monotonic Multihead Attention (MMA) for Transformer models: MMA-H and MMA-IL. Finally we introduce strategies to control latency and coverage.

2.1 MONOTONIC ATTENTION

The hard monotonic attention mechanism (Raffel et al., 2017) was first introduced in order to achieve online linear time decoding for RNN-based encoder-decoder models. We denote the input sequence as $\mathbf{x} = \{x_1, ..., x_T\}$, and the corresponding encoder states as $\mathbf{m} = \{m_1, ..., m_T\}$, with T being the length of the source sequence. The model generates a target sequence $\mathbf{y} = \{y_1, ..., y_U\}$ with U being the length of the target sequence. At the i-th decoding step, the decoder only attends to one encoder state m_{t_i} with $t_i = j$. When generating a new target token y_i, the decoder chooses whether to move one step forward or to stay at the current position based on a Bernoulli selection probability $p_{i,j}$, so that $t_i \geq t_{i-1}$. Denoting the decoder state at the i-th, starting from $j = t_{i-1}, t_{i-1}+1, t_{i-1}+2, ...$, this process can be calculated as follows: [1]

$$
\begin{aligned}
e_{i,j} &= \text{MonotonicEnergy}(s_{i-1}, m_j) && (1)\\
p_{i,j} &= \text{Sigmoid}\left(e_{i,j}\right) && (2)\\
z_{i,j} &\sim \text{Bernoulli}(p_{i,j}) && (3)
\end{aligned}
$$

When $z_{i,j} = 1$, we set $t_i = j$ and start generating a target token y_i; otherwise, we set $t_i = j+1$ and repeat the process. During training, an expected alignment $\boldsymbol{\alpha}$ is introduced in order to replace the softmax attention. It can be calculated in a recurrent manner, shown in Equation 4:

$$
\begin{aligned}
\alpha_{i,j} &= p_{i,j} \sum_{k=1}^{j} \left(\alpha_{i-1,k} \prod_{l=k}^{j-1} (1 - p_{i,l}) \right) \\
&= p_{i,j} \left((1 - p_{i,j-1}) \frac{\alpha_{i,j-1}}{p_{i,j-1}} + \alpha_{i-1,j} \right)
\end{aligned}
\tag{4}
$$

Raffel et al. (2017) also introduce a closed-form parallel solution for the recurrence relation in Equation 5:

$$
\alpha_{i,:} = p_{i,:}\texttt{cumprod}(1 - p_{i,:})\texttt{cumsum}\left(\frac{\alpha_{i-1,:}}{\texttt{cumprod}(1 - p_{i,:})} \right) \tag{5}
$$

where $\texttt{cumprod}(\boldsymbol{x}) = [1, x_1, x_1x_2, ..., \prod_{i=1}^{|\boldsymbol{x}|-1} x_i]$ and $\texttt{cumsum}(\boldsymbol{x}) = [x_1, x_1 + x_2, ..., \sum_{i=1}^{|\boldsymbol{x}|} x_i]$. In practice, the denominator in Equation 5 is clamped into a range of $(\epsilon, 1]$ to avoid numerical instabilities introduced by `cumprod`. Although this monotonic attention mechanism achieves online linear time decoding, the decoder can only attend to one encoder state. This limitation can diminish translation quality because there may be insufficient information for reordering.

Moreover, the model lacks a mechanism to adjust latency based on different requirements at decoding time. To address these issues, Chiu & Raffel (2018) introduce Monotonic Chunkwise Attention

[1] Notice that during training, to encourage discreteness, Raffel et al. (2017) added a zero mean, unit variance pre-sigmoid noise to $e_{i,j}$.

(MoChA), which allows the decoder to apply softmax attention over a chunk (subsequence of encoder positions). Alternatively, Arivazhagan et al. (2019) introduce Monotonic Infinite Lookback Attention (MILk) which allows the decoder to access encoder states from the beginning of the source sequence. The expected attention for the MILk model is defined in Equation 6.

$$\beta_{i,j} = \sum_{k=j}^{|\boldsymbol{x}|} \left(\frac{\alpha_{i,k} \exp(u_{i,j})}{\sum_{l=1}^{k} \exp(u_{i,l})} \right) \tag{6}$$

2.2 Monotonic Multihead Attention

Previous monotonic attention approaches are based on RNN encoder-decoder models with a single attention and haven't explored the power of the Transformer model. [2] The Transformer architecture (Vaswani et al., 2017) has recently become the state-of-the-art for machine translation (Barrault et al., 2019). An important feature of the Transformer is the use of a separate multihead attention module at each layer. Thus, we propose a new approach, Monotonic Multihead Attention (MMA), which combines the expressive power of multihead attention and the low latency of monotonic attention.

Multihead attention allows each decoder layer to have multiple heads, where each head can compute a different attention distribution. Given queries Q, keys K and values V, multihead attention MultiHead(Q, K, V) is defined in Equation 7.

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ..., \text{head}_H) W^O \\ \text{where} \;\; \text{head}_h &= \text{Attention}\left(QW_h^Q, KW_h^K, VW_h^V, \right) \end{aligned} \tag{7}$$

The attention function is the scaled dot-product attention, defined in Equation 8:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{8}$$

There are three applications of multihead attention in the Transformer model:

1. The **Encoder** contains self-attention layers where all of the queries, keys and values come from previous layers.
2. The **Decoder** contains self-attention layers that allow each position in the decoder to attend to all positions in the decoder up to and including that position.
3. The **Decoder-encoder attention** contains multihead attention layers where queries come from the previous decoder layer and the keys and values come from the output of the encoder. Every decoder layer has an decoder-encoder attention.

For MMA, we assign each head to operate as a separate monotonic attention in decoder-encoder attention.

For a transformer with L decoder layers and H attention heads per layer, we define the selection process of the h-th head decoder-encoder attention in the l-th decoder layer as

$$e_{i,j}^{l,h} = \left(\frac{m_j W_{l,h}^K (s_{i-1} W_{l,h}^Q)^T}{\sqrt{d_k}} \right)_{i,j} \tag{9}$$

$$p_{i,j}^{l,h} = \text{Sigmoid}(e_{i,j}) \tag{10}$$

$$z_{i,j}^{l,h} \sim \text{Bernoulli}(p_{i,j}) \tag{11}$$

where $W_{l,h}$ is the input projection matrix, d_k is the dimension of the attention head. We make the selection process independent for each head in each layer. We then investigate two types of

[2] MILk was based on a strengthened RNN-based model called RNMT+. The original RNMT+ model (Chen et al., 2018) uses multihead attention, computes attention only once, and then concatenates that single attention layer to the output of each decoder layer block. However, the RNMT+ model used for MILk in Arivazhagan et al. (2019) only uses a single head.

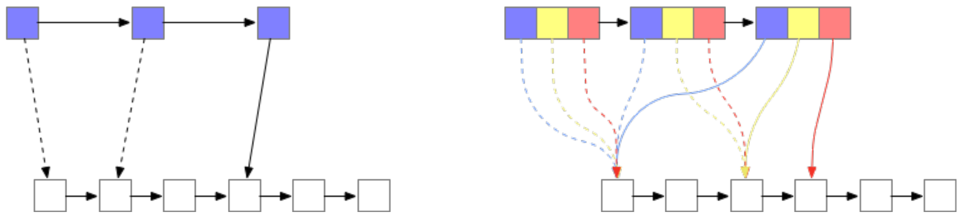

Figure 1: Monotonic Attention (Left) versus Monotonic Multihead Attention (Right). At each decoding step, MMA still has access to various encoder states.

MMA, MMA-H(ard) and MMA-IL(infinite lookback). For MMA-H, we use Equation 4 in order to calculate the expected alignment for each layer each head, given $p_{i,j}^{l,h}$. For MMA-IL, we calculate the softmax energy for each head as follows:

$$u_{i,j}^{l,h} = \text{SoftEnergy} = \left(\frac{m_j \hat{W}_{l,h}^{K} (s_{i-1} \hat{W}_{l,h}^{Q})^T}{\sqrt{d_k}} \right)_{i,j} \tag{12}$$

and then use Equation 6 to calculate the expected attention. Each attention head in MMA-H hard-attends to one encoder state. On the other hand, each attention head in MMA-IL can attend to all previous encoder states. Thus, MMA-IL allows the model to leverage more information for translation, but MMA-H may be better suited for streaming systems with stricter efficiency requirements.

At inference time, our decoding strategy is shown in Algorithm 1. For each l, h, at decoding step i, we apply the sampling processes discussed in subsection 2.1 individually and set the encoder step at $t_i^{l,h}$. Then a hard alignment or partial softmax attention from encoder states, shown in Equation 13, will be retrieved to feed into the decoder to generate the i-th token. The model will write a new target token only after all the attentions have decided to write – the heads that have decided to write must wait until the others have finished reading.

$$\begin{aligned} c_i^l &= \text{Concat}(c_i^{l,1}, c_i^{l,2}, ..., c_i^{l,H}) \\ \text{where } c_i^{l,h} &= f_{\text{context}}(\boldsymbol{h}, t_i^{l,h}) = \begin{cases} m_{t_i^{l,h}} & \text{MMA-H} \\ \dfrac{\sum_{j=1}^{t_i^{l,h}} \exp\left(u_{i,j}^{l,h}\right) m_j}{\sum_{j=1}^{t_i^{l,h}} \exp\left(u_{i,j}^{l,h}\right)} & \text{MMA-IL} \end{cases} \end{aligned} \tag{13}$$

Figure 1 illustrates a comparison between our model and the monotonic model with one attention head. Compared with the monotonic model, the MMA model is able to set attention to different positions so that it can still attend to previous states while reading each new token. Each head can adjust its speed on-the-fly. Some heads read new inputs, while the others can stay in the past to retain the source history information. Even with the hard alignment variant (MMA-H), the model is still able to preserve the history information by setting heads to past states. In contrast, the hard monotonic model, which only has one head, loses the previous information at the attention layer.

2.3 Latency Control

Effective simultaneous machine translation must balance quality and latency. At a high level, latency measures how many source tokens the model has to read until a translation is generated. The model we have introduced in subsection 2.2 is not able to control latency on its own. While MMA allows simultaneous translation by having a read or write schedule for each head, the overall latency is determined by the fastest head, i.e. the head that reads the most. It is possible that a head always reads new input without producing output, which would result in the maximum possible latency. Note that the attention behaviors in MMA-H and MMA-IL can be different. In MMA-IL, a head reaching the end of the sentence will provide the model with maximum information about the source sentence. On the other hand, in the case of MMA-H, reaching the end of sentence for a head only gives a hard alignment to the end-of-sentence token, which provides very little information to the decoder. Furthermore, it is possible that an MMA-H attention head stays at the beginning of sentence

Algorithm 1 MMA monotonic decoding. Because each head is independent, we compute line 3 to 10 in parallel

Input: Memory h of length T, $i=1, j=1, t_0^{l,h}=1, y_0=$ StartOfSequence.

1: **while** $y_{i-1} \neq$ EndOfSequence **do**
2:     **for** $l \leftarrow 1$ to L **do**
3:         **for** $h \leftarrow 1$ to H **do**
4:             **for** $j \leftarrow t_{i-1}^{l,h}$ to T **do**
5:                 $p_{i,j}^{l,h} = \text{Sigmoid}\left(\text{MonotonicEnergy}(s_{i-1,m_j})\right)$
6:                 **if** $p_{i,j}^{l,h} > 0.5$ **then**
7:                     $c_i^{l,h} = f_{\text{context}}(\boldsymbol{h}, t_i^{l,h})$
8:                     $t_i^{l,h} = j$
9:                     **Break**
10:             $c_i^l = \text{Concat}(c_i^{l,1}, c_i^{l,2}, ..., c_i^{l,H})$
11:         $s_i^l = \text{DecoderLayer}^l(s_{1:i-1}^l, s_{1:i-1}^{l-1}, c_i^l)$
12:     $y_i = \text{Output}(s_i^L)$
13:     $i = i+1$

without moving forward. Such a head would not cause latency issues but would degrade the model quality since the decoder would not have any information about the input. In addition, this behavior is not suited for streaming systems.

To address these issues, we introduce two latency control methods. The first one is weighted average latency, shown in Equation 14:

$$g_i^W = \frac{\exp(g_i^{l,h})}{\sum_{l=1}^{L}\sum_{h=1}^{H}\exp(g_i^{l,h})} g_i^{l,h} \tag{14}$$

where $g_i^{l,h} = \sum_{j=1}^{|\boldsymbol{x}|} j\alpha_{i,j}$. Then we calculate the latency loss with a differentiable latency metric $\mathcal{C}$.

$$L_{avg} \quad = \quad \mathcal{C}\left(\boldsymbol{g}^W\right) \tag{15}$$

Like Arivazhagan et al. (2019), we use the Differentiable Average Lagging. It is noticeable that, different from original latency augmented training in Arivazhagan et al. (2019), Equation 15 is not the expected latency metric given $\mathcal{C}$, but weighted average $\mathcal{C}$ on all the attentions. The real expected latency is $\hat{\boldsymbol{g}} = \max_{l,h}\left(\boldsymbol{g}^{l,h}\right)$ instead of $\bar{\boldsymbol{g}}$, but using this directly would only affect the speed of the fastest head. Equation 15, however, can control every head — the regularization has a much greater effect on the fast heads but also inhibits the slow heads from getting faster. However, for MMA models, we found that the latency of are mainly due to outliers that skip almost every token. The weighted average latency loss is not sufficient to control the outliers. We therefore introduce the head divergence loss, the average variance of expected delays at each step, defined in Equation 16:

$$L_{var} \quad = \quad \frac{1}{LH}\sum_{l=1}^{L}\sum_{h=1}^{H}\left(g_i^{l,h} - \bar{g}_i\right)^2 \tag{16}$$

where $\bar{g}_i = \frac{1}{LH}\sum g_i$ The final objective function is presented in Equation 17:

$$L(\theta) = -\log(\boldsymbol{y} \mid \boldsymbol{x}; \theta) + \lambda_{avg}L_{avg} + \lambda_{var}L_{var} \tag{17}$$

where λ_{avg}, λ_{var} are hyperparameters that control both losses. Intuitively, while λ_{avg} controls the overall speed, λ_{var} controls the divergence of the heads. Combining these two losses, we are able to dynamically control the range of attention heads so that we can control the latency and the reading buffer. For MMA-IL model, we used both loss terms; for MMA-H we only use L_{var}.

3 EXPERIMENTAL SETUP

3.1 EVALUATION METRICS

We evaluate our model using quality and latency. For translation quality, we use tokenized BLEU [3] for IWSLT15 En-Vi and detokenized BLEU with SacreBLEU (Post, 2018) for WMT15 De-En. For latency, we use three different recent metrics, **Average Proportion** (AP) (Cho & Esipova, 2016), **Average Lagging** (AL) (Ma et al., 2019) and **Differentiable Average Lagging** (DAL) (Arivazhagan et al., 2019) [4]. We remind the reader of the metric definitions in Appendix A.2.

3.2 DATASETS

Dataset	Train	Validation	Test
IWSLT15 En-Vi	133k	1268	1553
WMT15 De-En	4.5M	3000	2169

Table 1: Number of sentences in each split.

Dataset	RNN	Transformer
IWSLT15 En-Vi	25.6 [5]	28.70
WMT15 De-En	28.4 (Arivazhagan et al., 2019)	32.3

Table 2: Offline model performance with unidirectional encoder and greedy decoding.

We evaluate our method on two standard machine translation datasets, IWSLT14 En-Vi and WMT15 De-En. Statistics of the datasets can be found in Table 1. For each dataset, we apply tokenization with the Moses (Koehn et al., 2007) tokenizer and preserve casing.

IWSLT15 English-Vietnamese TED talks from IWSLT 2015 Evaluation Campaign (Cettolo et al., 2016). We follow the same settings from Luong & Manning (2015) and Raffel et al. (2017). We replace words with frequency less than 5 by <*unk*>. We use tst2012 as a validation set tst2013 as a test set.

WMT15 German-English We follow the setting from Arivazhagan et al. (2019). We apply byte pair encoding (BPE) (Sennrich et al., 2016) jointly on the source and target to construct a shared vocabulary with 32K symbols. We use newstest2013 as validation set and newstest2015 as test set.

3.3 MODELS

We evaluate MMA-H and MMA-IL models on both datasets. The MILK model we evaluate on IWSLT15 En-Vi is based on Luong et al. (2015) rather than RNMT+ (Chen et al., 2018). All our offline models use unidirectional encoders: the encoder self-attention can only attend to previous states. Offline model performance can be found in Table 2. For MMA models, we replace the encoder-decoder layers with MMA and keep other hyperparameter settings the same as the offline model. Detailed hyperparameter settings can be found in subsection A.1. We use the Fairseq library (Ott et al., 2019) [6] for our implementation. Code will be released upon publication.

[3] We acquire the data from https://nlp.stanford.edu/projects/nmt/, which is tokenized. We do not have the tokenizer which processed this data, thus we report tokenized BLEU for IWSLT15

[4] Latency metrics are computed on BPE tokens for WMT15 De-En – consistent with Arivazhagan et al. (2019) – and on word tokens for IWSLT15 En-Vi.

[5] Luong & Manning (2015) report a BLEU score of 23.0 but they didn't mention what type of BLEU score they used. This score is from our implementation on the data aquired from https://nlp.stanford.edu/projects/nmt/

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

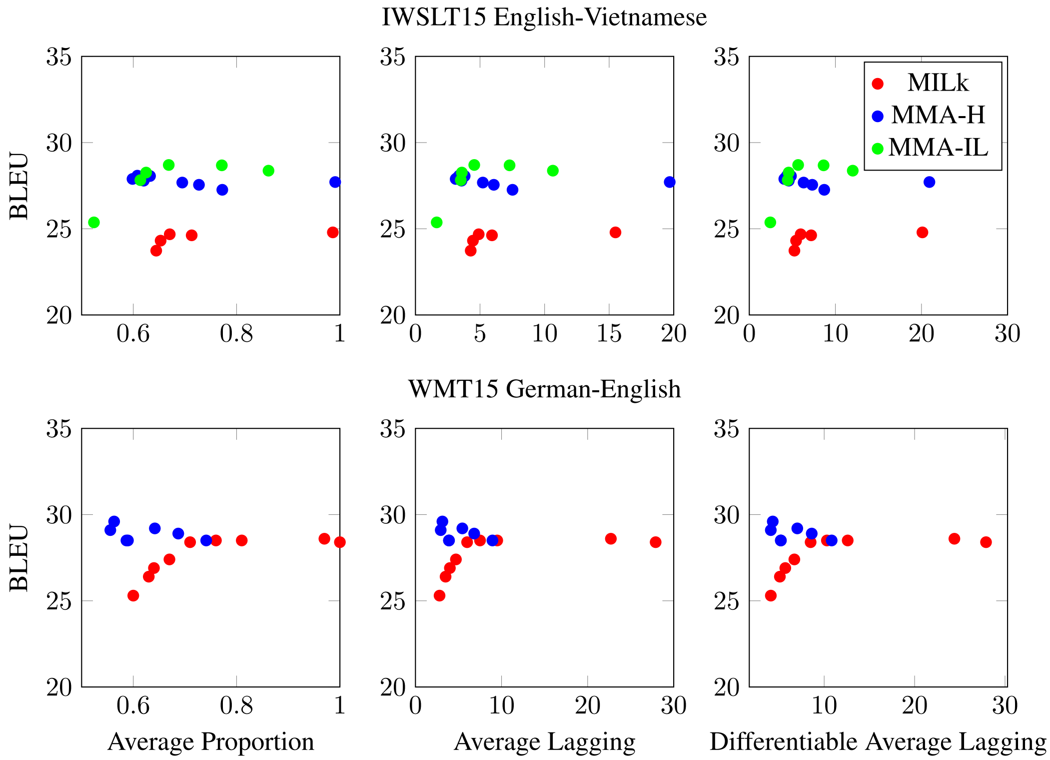

Figure 2: Latency-quality tradeoffs for MILk and MMA on IWSLT15 En-Vi and WMT15 De-En.

4 Results

In this section, we present the main results of our model in terms of latency-quality tradeoffs and two ablation studies. In the first one, we analyze the effect of the variance loss on the attention span. Then, we study the effect of the number of decoder layers and decoder heads on quality and latency.

4.1 Latency-Quality Tradeoffs

We plot the quality-latency curves for MMA-H and MMA-IL in Figure 2. The BLEU and latency scores on the test sets were generated by setting a latency range and selecting the checkpoint with best BLEU score on the validation set. We use differentiable average lagging (Arivazhagan et al., 2019) when setting the latency range. We found that for a given latency, our models obtain a better translation quality. It is interesting to observe that even MMA-H has a better latency-quality tradeoff than MILk even though each head only attends to only one state. Although MMA-H is not quite yet streaming capable since both the encoder and decoder self-attention have an infinite lookback, that model represents a good step in that direction.

4.2 Attention Span

In subsection 2.3, we introduced the attention variance loss to MMA-H in order to prevent outlier attention heads from increasing the latency or increasing the attention span. We have already evaluated the effectiveness of this method on latency in subsection 4.1. We also want to measure the difference between the fastest and slowest heads at each decoding step. We define the average attention span in Equation 18:

$$\bar{S} = \frac{1}{|\boldsymbol{y}|}\left(\sum_{i}^{|\boldsymbol{y}|} \max_{l,h} t_i^{l,h} - \min_{l,h} t_i^{l,h}\right) \tag{18}$$

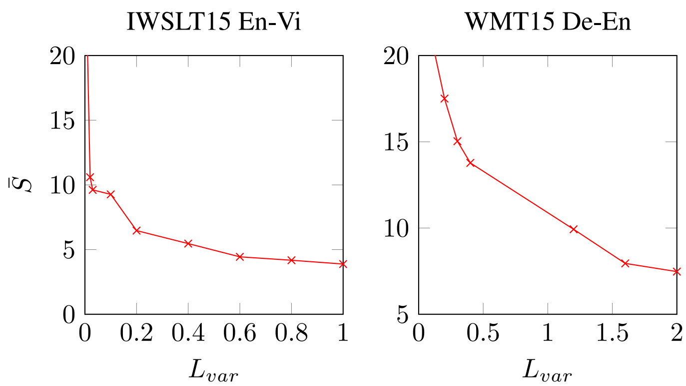

Figure 3: Effect of L_{var} on the average attention span. The variance loss works as intended by reducing the span with higher weights.

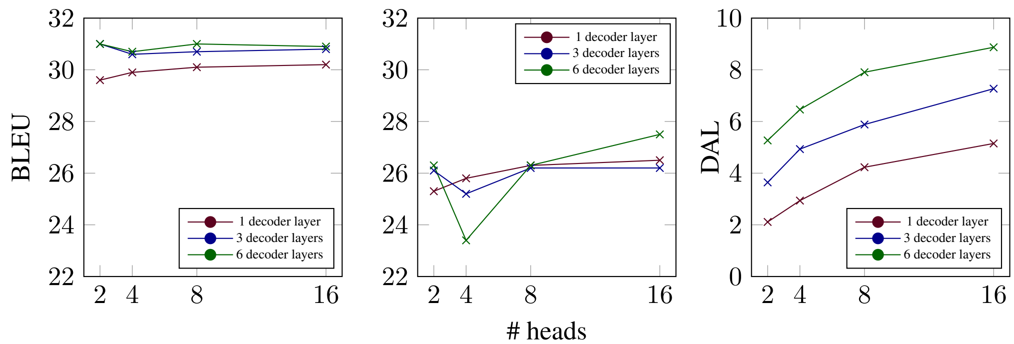

Figure 4: Effect of the number of decoder attention heads and the number of decoder attention layers on quality and latency, reported on the WMT13 validation set. For both the baseline and the proposed model, quality generally improves with the number of heads and the number of layers, which motivates the proposed model.

It estimates the reading buffer we need for streaming translation. We show the relation between the average attention span (averaged over the IWSLT and WMT test sets) versus L_{var} in Figure 3. As expected, the average attention span is reduced as we increase L_{var}.

4.3 Effect on number of layers and number of heads

One motivation to introduce MMA is to adapt the Transformer, which is the current state-of-the-art model for machine translation, to online decoding. Important features of the Transformer architecture include having a separate attention layer for each decoder layer block and multihead attention. In this section, we test the effect of these two components on both the offline baseline and MMA-H from a quality and latency perspective. We report quality as measure by detokenized BLEU and latency as measured by DAL on the WMT13 validation set in Figure 4. We set $\lambda_{avg} = 0$ and $\lambda_{var} = 0.2$. We can see that quality generally tends to improve with more layers and more heads for both the offline baseline and MMA-H, which motivates extending monotonic attention to the multilayer/multihead setting. We also note that latency increases accordingly. This is due to having fixed loss weights: when more heads are involved, we should increase λ_{var} to better control latency.

5 Related Work

Recent work on simultaneous machine translation falls into three categories. In the first one, models use a rule-based policy for reading input and writing output. Cho & Esipova (2016) propose a Wait-If-* policy to enable an offline model to decode simultaneously. Ma et al. (2019) propose a wait-k policy where the model first reads k tokens, then alternates reads and writes. Dalvi et al. (2018) propose an incremental decoding method, also based on a rule-based schedule. In the second category, models learn the policy with reinforcement learning. Grissom II et al. (2014) introduce a Markov chain to phrase-based machine translation models for simultaneous machine translation, in which they apply reinforcement learning to learn the read-write policy based on states. Gu et al. (2017) introduce an agent which learns to make decisions on when to translate from the interaction with a pre-trained neural machine translation model. Alinejad et al. (2018) propose a new operation PREDICT which predicts future source tokens to improve quality and minimize latency. Models from the last category leverage monotonic attention and replace the softmax attention with an expected attention calculated from a stepwise Bernoulli selection probability. Raffel et al. (2017) first introduce the concept of monotonic attention for online linear time decoding, where the attention only attends to one encoder state at a time. Chiu & Raffel (2018) extended that work to let the model attend to a chunk of encoder state. Arivazhagan et al. (2019) also make use of the monotonic attention but introduce an infinite lookback to improve the translation quality.

6 Conclusion

In this paper, we propose two variants of the monotonic multihead attention model for simultaneous machine translation. By introducing two new targeted loss terms which allow us to control both latency and attention span, we are able to leverage the power of the Transformer architecture to achieve better quality-latency trade-offs than the previous state-of-the-art model. We also present detailed ablation studies demonstrating the efficacy and rationale of our approach. By introducing these stronger simultaneous sequence-to-sequence models, we hope to facilitate important applications, such as high-quality real-time interpretation between human speakers.

References

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3022–3027, 2018.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1313–1323, Florence, Italy, July 2019. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/P19-1126`.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pp. 1–61, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5301. URL `https://www.aclweb.org/anthology/W19-5301`.

Mauro Cettolo, Niehues Jan, Stüker Sebastian, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. The iwslt 2016 evaluation campaign. In *International Workshop on Spoken Language Translation*, 2016.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, et al. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 76–86, 2018.

Chung-Cheng Chiu and Colin Raffel. Monotonic chunkwise attention. 2018. URL `https://openreview.net/pdf?id=Hko85plCW`.

Kyunghyun Cho and Masha Esipova. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*, 2016.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 493–499, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2079. URL https://www.aclweb.org/anthology/N18-2079.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. Dont until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on empirical methods in natural language processing (EMNLP)*, pp. 1342–1352, 2014.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor OK Li. Learning to translate in real-time with neural machine translation. In *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017*, pp. 1053–1062. Association for Computational Linguistics (ACL), 2017.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pp. 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P07-2045.

Minh-Thang Luong and Christopher D Manning. Stanford neural machine translation systems for spoken language domains. 2015.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3025–3036, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1289.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2837–2846. JMLR. org, 2017.

Rico Sennrich, Barry Haddow, and Alexandra 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)*, pp. 1715–1725, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

A Appendix

A.1 Hyperparameters

The hyperparameters we used for offline and monotonic transformer models are defined in Table 3.

Hyperparameter	WMT15 German-English	IWSLT English-Vietnamese
encoder embed dim	1024	512
encoder ffn embed dim	4096	1024
encoder attention heads	16	4
encoder layers	6	
decoder embed dim	1024	512
decoder ffn embed dim	4096	1024
decoder attention heads	16	4
decoder layers	6	
dropout	0.3	
optimizer	adam	
adam-β	$(0.9, 0.98)$	
clip-norm	0.0	
lr	0.0005	
lr scheduler	inverse sqrt	
warmup-updates	4000	
warmup-init-lr	1e-07	
label-smoothing	0.1	
max tokens	$3584 \times 8 \times 8 \times 2$	16000

Table 3: Offline and monotonic models hyperparameters.

A.2 Latency Metrics Definitions

Given the delays $\mathbf{g} = \{g_1, g_2, ..., g_{|\boldsymbol{y}|}\}$ of generating each target token, AP, AL and DAL are defined in Table 4.

Latency Metric	Calculation
Average Proportion	$\frac{1}{\lvert\boldsymbol{x}\rvert\lvert\boldsymbol{y}\rvert}\sum_{i=1}^{\lvert\boldsymbol{y}\rvert} g_i$
Average Lagging	$\frac{1}{\tau}\sum_{i=1}^{\tau} g_i - \frac{i-1}{\lvert\boldsymbol{y}\rvert/\lvert\boldsymbol{x}\rvert}$ where $\tau = \arg\max_i (g_i = \lvert\boldsymbol{x}\rvert)$
Differentiable Average Lagging	$\frac{1}{\lvert\boldsymbol{y}\rvert}\sum_{i=1}^{\lvert\boldsymbol{y}\rvert} g'_i - \frac{i-1}{\lvert\boldsymbol{y}\rvert/\lvert\boldsymbol{x}\rvert}$ where $g'_i = \begin{cases} g_i & i = 0 \\ \max(g_i, g'_{i-1} + \frac{\lvert\boldsymbol{y}\rvert}{\lvert\boldsymbol{x}\rvert}) & i < 0 \end{cases}$

Table 4: The calculation of latency metrics, given source $\boldsymbol{x}$, target $\boldsymbol{y}$ and delays $\boldsymbol{g}$