Efficient Monotonic Multihead Attention

Xutai Ma, **Anna Sun**, **Siqi Ouyang**[†], **Hirofumi Inaguma**, **Paden Tomasello**

FAIR at Meta, [†]UC Santa Barbara

We introduce the Efficient Monotonic Multihead Attention (EMMA), a state-of-the-art simultaneous translation model with numerically-stable and unbiased monotonic alignment estimation. In addition, we present improved training and inference strategies, including simultaneous fine-tuning from an offline translation model and reduction of monotonic alignment variance. The experimental results demonstrate that the proposed model attains state-of-the-art performance in simultaneous speech-to-text translation on the Spanish and English translation task.

Date: November 30, 2023
Correspondence: xutaima@meta.com
Code: https://github.com/facebookresearch/seamless_communication.

1. Introduction

Simultaneous translation is a task focusing on reducing the latency of the machine translation system. In this approach, a simultaneous translation model initiates the translation process even before the speaker completes their sentence. This type of model plays a pivotal role in various low-latency scenarios, such as personal travels and international conferences, where people desire seamless and real-time translation experiences. In contrast to an offline model, which processes the entire input sentence and generates translation output in a single step, a simultaneous translation model operates on a partial input sequence. The simultaneous model incorporates a policy mechanism to determine when the model should generate translation output. The policy is characterized by two actions: `read` and `write`. While the `write` action indicates that the model should generate a partial translation, the `read` action introduces a pause in the generation process, allowing the model to acquire additional input information. The simultaneous policy can be either rule-based or learned through training processes.

In recent times, within the domain of learned policies, a specific category known as monotonic attention-based policies (Raffel et al., 2017; Chiu & Raffel, 2018; Arivazhagan et al., 2019), particularly Transformer-based Monotonic Multihead Attention (MMA) (Ma et al., 2019b) has demonstrated state-of-the-art performance in simultaneous text-to-text translation tasks. Monotonic attention provides an unsupervised policy learning framework that is based on an estimation of monotonic alignment during training time. Despite MMA's remarkable achievements in text-to-text translation, its adaptation to speech input faces certain challenges. Ma et al. (2020b) revealed that the MMA model fails to yield significant improvements when applied to speech input over simple wait-k baseline. Ma et al. (2020b) attributes the suboptimal performance of MMA on speech input to the granularity and continuity characteristics of the speech encoder states.

In this paper, we further investigate the adaptation of monotonic attention on speech translation. We demonstrate the two primary factors underlying the sub-optimal performance. The first is the numerical instability and introduction of bias during monotonic alignment estimation, which comes from techniques introduced by Raffel et al. (2017). The second is the significant variance in monotonic alignment estimation, especially in the later part of the sentence, due to the continuous nature of encoder states. To address these challenges, we propose Efficient Monotonic Multihead Attention (EMMA). Specifically, the key contributions of this work include:

- A novel numerically stable, unbiased monotonic alignment estimation, with yields state-of-the-art performance on both simultaneous text-to-text and speech-to-text translation.
- A new monotonic alignment shaping strategy, including new latency regularization and monotonic

alignment variance reduction

- An enhanced training scheme, involving fine-tuning the simultaneous model based on a pre-trained offline model.

2. Background

2.1 Notation

Given matrices A and B, we annotate the operations used in later chapters, along with their implementation in PyTorch toolkit in Table 1.

Notation	Definition	PyTorch
$A_{i,j}$	Index i-th row and j-th column in matrix A	`A[i, j]`
$A_{i,:}$	Index i-th row of A as a vector	`A[[i], :]`
$A_{:,j}$	Index j-th column of A as a vector	`A[:, [j]]`
$A \odot B$	Element-wise product (Hadamard roduct)	`A * B`
AB	Matrix multiplication	`torch.bmm(A, B)`
$\texttt{comprod}_l(\text{A})$	Cumulative product on the l-th dimension	`torch.cumprod(A, dim=l)`
$\texttt{comsum}_l(\text{A})$	Cumulative summation on the l-th dimension	`torch.cumsum(A, dim=l)`
$\texttt{triu}_b(\text{A})$	Upper triangle of A with a offset of b	`torch.triu(A, diagonal=b)`
$J_{N\times M}$	A matrix with size of N by M, filled with 1	`torch.ones(N, M)`
$\texttt{roll}_k$	Shift matrix by k elements, on last dimension	`A.roll(k, dims=[-1])`

Table 1 - Matrix operations and their implementation in PyTorch

2.2 Simultaneous Translation

Denote $\mathbf{X}$ and $\hat{\mathbf{Y}}$ the input and output sequences of a translation system. $\mathbf{X}$ is text token for text input and speech encoder states for speech input. We introduce the concept of a delay sequence denoted as $\mathbf{D}$, where each element d_i is the length of input utilized in generating the corresponding output element $\hat{y}_i$. It is essential to note that $\mathbf{D}$ forms a strictly monotonic non-decreasing sequence.

In a simultaneous translation system, there exists $\hat{\mathbf{Y}}$ such that $d_i < |\mathbf{X}|$. Meanwhile, an offline translation means $d_i = |\mathbf{X}|$ for all i. The measurement of $\mathbf{D}$ varies with input and output media. In this paper, d_i is measured in number of tokens for text input and seconds for speech input.

There are two aspects to evaluate simultaneous speech translation system: quality and latency. While the quality evaluation is same as offline system, for latency evaluation, we use the most commonly used metric Average Lagging (AL) Ma et al. (2019a), defined as

$$\text{AL} = \frac{1}{\tau(|\mathbf{X}|)} \sum_{i=1}^{\tau(|\mathbf{X}|)} d_i - d_i^* \tag{1}$$

where $\tau(|\mathbf{X}|) = \min\{i|d_i = |\mathbf{X}|\}$ is the index of the first target translation when the policy first reaches the end of the source sentence. d_i^* is the ideal policy defined as

$$d_i^* = (i-1) \cdot \frac{|\mathbf{X}|}{|\mathbf{Y}|} \tag{2}$$

where $\mathbf{Y}$ is the reference translation. As suggested by Ma et al. (2020b), $|\mathbf{X}|$ is measured in number of source words for text input and in number of seconds of source speech for speech input.

2.3 Monotonic Attention

Monotonic attention models (Raffel et al., 2017; Chiu & Raffel, 2018; Arivazhagan et al., 2019; Ma et al., 2019b) have a learnable policy based on monotonic alignment estimation during training time. At a given

time when $i-1$-th target translation has been predicted and j-th source input has been processed, a stepwise probability, denoted as $p_{i,j}$, describes the likelihood the model is going to `write` the i-th prediction rather than `read` the next input. Specifically, it is defined as

$$p_{i,j} = P(\text{action} = \texttt{write} | i, j; \theta_p) = \texttt{Sigmoid}\left(\mathcal{N}_{\theta_p}\left(s_{i-1}, h_j\right)\right) \tag{3}$$

where $\mathcal{N}_{\theta_p}$ is the policy network, s_{i-1} is $i-1$-th decoder state and h_j is j-th encoder state.

Raffel et al. (2017) proposed an closed form estimation of alignments between source and target $\alpha_{i,j}$ from $p_{i,j}$ during training:

$$\alpha_{i,:} = p_{i,:} \odot \texttt{cumprod}_2(1 - p_{i,:}) \odot \texttt{cumsum}_2\left(\alpha_{i-1,:} \odot \frac{1}{\texttt{cumprod}_2(1 - p_{i,:})}\right) \tag{4}$$

While Raffel et al. (2017) handle the hard alignment between source and target, Chiu & Raffel (2018) introduce monotonic chunkwise attention (MoChA), which enables soft attention in a chunk following the moving attention head. Arivazhagan et al. (2019) further proposed monotonic infinite lookback attention (MILk), in which a soft attention is computed over all the previous history. Given the energy $u_{i,j}$ for the i-th decoder state and the j-th encoder state, an expected soft attention is calculated in Equation 5:

$$\beta_{i,j} = \sum_{k=j}^{|\mathbf{X}|}\left(\frac{\alpha_{i,k}\exp(u_{i,j})}{\sum_{l=1}^{k}\exp(u_{i,l})}\right) \tag{5}$$

where β instead of α is then used in training. Arivazhagan et al. (2019) also introduce latency augmented training for latency control. Ma et al. (2019b) further extend the monotonic attention to multihead attention (MMA) for Transformer models. The design of MMA is to enable every attention head as individual monotonic attention.

3. Efficient Monotonic Multihead Attention

In this section, we will discuss three key factors of Efficient Monotonic Multihead Attention (EMMA): numerically stable estimation, alignment shaping, and streaming finetuning. It is noteworthy that the monotonic alignment estimation, denoted as α, discussed in this section is based on a single attention head. Following the same design as Ma et al. (2019b), the same estimation of α is applied to every attention head in MMA as integrated into the Transformer (Vaswani et al., 2017) model. Notably, only the infinite lookback (Arivazhagan et al., 2019) variant of monotonic attention is applied.

3.1 Numerically Stable Estimation

Similar to Raffel et al. (2017), the objective of the monotonic estimation is to calculate the expected alignment $\alpha_{i,j}$, from stepwise `write` action probability $p_{i,j}$. Nevertheless, the numerical instability arises from the denominator in Equation 4, particularly when dealing with the multiplication of several small probabilities. To address this issue, we introduce a innovative numerically stable approach for estimation of monotonic attention.

In accordance with the approach proposed by Raffel et al. (2017), the monotonic alignment between the i-th target item and the j-th source item can be represented as:

$$\alpha_{i,j} = p_{i,j}\sum_{k=1}^{j}\alpha_{i-1,k}\prod_{l=k}^{j-1}(1 - p_{i,l}) \tag{6}$$

This expression can be reformulated in a matrix multiplication format:

$$\alpha_{i,:} = p_{i,:} \odot \alpha_{i-1,:}\mathbf{T}(i) \tag{7}$$

where $\mathbf{T}(i)$ represents a transition matrix, with each of its elements defined as:

$$\mathbf{T}(i)_{m,n} = \begin{cases} \prod_{l=m}^{n-1}(1-p_{i,l}) & m < n \\ 1 & m = n \\ 0 & m > n \end{cases} \tag{8}$$

$\mathbf{T}(i)_{m,n}$ indicates the probability of the policy at the $(i-1)$-th target step consecutively skipping from the m-th to the n-th inputs. Moreover, the transition matrix can be further expressed as

$$\mathbf{T}(i) = \texttt{triu}_0\left(\texttt{cumprod}_2(1-\texttt{triu}_1\left(J_{|X|\times 1}\texttt{roll}_1(p_{i,:})\right))\right) \tag{9}$$

The operations within the equation can be efficiently executed in parallel on GPUs, as elaborated in Table 1. Reframing Equation 7, we arrive at the following expression:

$$\alpha_{i,:} = p_{i,:} \odot \alpha_{i-1,:}\texttt{triu}_0\left(\texttt{cumprod}_2(1-\texttt{triu}_1\left(J_{|X|\times 1}\texttt{roll}_1(p_{i,:})\right))\right) \tag{10}$$

It is noteworthy that this estimation process is also closed-form, with the desirable properties of being numerically stable and unbiased, as it does not require a denominator as the product of probabilities within the equation. A comprehensive derivation of this closed-form estimation is provided in Appendix A.

3.2 Alignment Shaping

When training the infinite lookback variant of monotonic attention, it is necessary to add latency regularization in order to prevent the model from learning a trivial policy. Without latency regularization, the optimal policy to minimize cross-entropy loss is to `read` the entire the sequence before starting the translation. Therefore, we applied latency and variance regularizations to control the tradeoff between translation quality and latency of the learned simultaneous translation policy.

The latency of the alignment describes how much partial input information is needed by the model to generate part of the translation. The reduction of latency is commonly achieved by introducing a regularization term derived from the estimated alignment. Consistent with prior work, such as Arivazhagan et al. (2019); Ma et al. (2019b), the expected delays $\bar{\mathbf{D}} = \bar{d}_1, ..., \bar{d}_{|\mathbf{Y}|}$ are estimated from the expected alignment α during training time. The expected delay of target token y_i, denoted as $\bar{d}_i$, is computed as

$$\bar{d}_i = \mathtt{E}[j|i] = \sum_{k=1}^{|\mathbf{X}|} k\alpha_{i,k} \tag{11}$$

Given a latency metric $\mathcal{C}$, the loss term is then computed as

$$\mathcal{L}_{\text{latency}} = \mathcal{C}(\bar{\mathbf{D}}) \tag{12}$$

The variance of the alignment characterizes the certainty of an estimation. It is noteworthy that an alignment estimation can be low latency but high variance. For instance, a random walk policy, which yields a monotonic alignment of linear latency, has a huge variance on the estimation. Arivazhagan et al. (2019) proposed a method to reduce the uncertainty by introducing a Gaussian noise to the input of stepwise probability network. Nevertheless, empirical results show that this method is not efficient, especially when applied to speech translation models. Therefore, we propose an alternative regularization-based strategy.

Denote the $\mathcal{V} = \bar{v_1}, ..., \bar{v}_{|\mathbf{Y}|}$ as the expected variances of the monotonic alignment. The expected variance of target token y_i, denoted as $\bar{v}_i$, can be expressed as

$$\bar{v}_i = \mathtt{E}[(j - \mathtt{E}[j|i])^2|i] = \mathtt{E}[j^2|i] - \mathtt{E}[j|i]^2 = \sum_{k=1}^{|\mathbf{X}|} k^2\alpha_{i,k} - \left(\sum_{k=1}^{|\mathbf{X}|} k\alpha_{i,k}\right)^2 \tag{13}$$

We then introduce the alignment variance loss as the following:

$$\mathcal{L}_{\text{variance}} = \sum_{i=1}^{|\mathbf{Y}|} \bar{v}_i \tag{14}$$

To further reduce the alignment variance, we proposed an enhanced stepwise probability network as

$$p_{i,j} = \texttt{Sigmoid}\left(\frac{\texttt{FFN}_s(s_{i-1})^T\texttt{FFN}_h(h_j) + b}{\tau}\right) \tag{15}$$

$\texttt{FFN}_s$ and $\texttt{FFN}_s$ serve as energy projections, constructed using multi-layer feedforward networks, which increase the expressive capability of stepwise probability network over linear projection adopted by prior monotonic work. b is a learnable bias, initialized by a negative value. Its purpose is to schedule an easier the policy optimization process from the offline policy. τ is the temperature factor, to encourage polarized output from stepwise probability network.

Finally, we optimize the model with the following objective

$$\mathcal{L}(\theta) = -\log(\mathbf{Y}|\mathbf{X}) + \lambda_{\text{latency}}\mathcal{L}_{\text{latency}} + \lambda_{\text{variance}}\mathcal{L}_{\text{variance}} \tag{16}$$

where λ_{latency} and $\lambda_{\text{variance}}$ are the loss weights.

3.3 Simultaneous Fine-tuning

In most prior work on simultaneous translation, the model is usually trained from scratch. However, this approach often requires substantial resources when dealing with extensive or multilingual scenarios. For instance, it can be a significant challenge to retrain a simultaneous model with the configuration from recent large-scale multilingual models, such as Whisper or SeamlessM4T. To leverage the recent advancements achieved with large foundational translation models and enhance the adaptability of the simultaneous translation model, we introduce a method for Simultaneous Fine-tuning.

Denote the an arbitrary offline encoder-decoder translation model as $\mathcal{M}(\theta_e^o, \theta_d^o)$, with θ_e representing the encoder parameters and θ_d representing the decoder parameters. The simultaneous model is denoted as $\mathcal{M}(\theta_e, \theta_d, \theta_p)$, where θ_p denotes the policy network. Simultaneous fine-tuning involves initializing θ_e with θ_e^o and θ_d with θ_d^o. During the training process, the encoder parameters θ_e remain fixed, and optimization is only performed on θ_d and θ_p. This design is motivated by the assumption that the generative components of the model, namely θ_e and θ_d, should closely resemble those of the offline model. In simultaneous setting, they are adapted to partial contextual information.

3.3.1 Streaming Inference

We used `SimulEval` (Ma et al., 2020a) to build the inference pipeine. The overall inference algorithm is illustrated in Algorithm 1. For streaming speech input, we update the whole encoder every time a new speech chunk is received by the model. Then, we run the decoder to generate a partial text translation based on the policy.

4. Experimental Setup

We evaluate proposed models on speech-to-text translation task. The models are evalauted with the `SimulEval` (Ma et al., 2020a) toolkit. Evaluation of the models focuses on two factors: quality and latency. The quality is measured by detokenized BLEU, using the SacreBLEU (Post, 2018) toolkit. Latency evaluation is measured by Average Lagging (AL) (Ma et al., 2019a) We follow the simultaneous fine-tuning strategy introduced in Section 3.3. The simultaneous model is initialized from an offline translation model. Detailed information regarding the tasks, evaluation datasets employed in this study, and the performance of the offline model are presented in Table 2

For speech-to-text (S2T) translation task, we establish two experimental configurations: bilingual and multilingual.

The bilingual setup aims to demonstrate the model's potential when provided with a extensive corpus of training data. We trained one model for each direction, `spa-eng` and `eng-spa`. The multilingual task

[1]Average on 100 language directions

Algorithm 1 EMMA Inference

Input: $\mathbf{X}$: Input streaming speech.
Input: $\mathbf{Y}$: Output text.
Input: t_{EMMA} : Decision threshold for EMMA

$i \leftarrow 1$, $j \leftarrow 0$, $k \leftarrow 0$
$s_0 \leftarrow \texttt{TextDecoder}(y_0)$
while $y_{i-1} \neq \texttt{EndOfSequence}$ **do**
  $j \leftarrow j + 1$
  $h_{\leq j} \leftarrow \texttt{SpeechEncoder}(\mathbf{X}_{\leq j})$
  while $y_{i-1} \neq \texttt{EndOfSequence}$ **do**
    $p \leftarrow 1$
    for StepwiseProbabilty in all attention head **do**
      $p \leftarrow \texttt{min}(p, \texttt{StepwiseProbabilty}(h_j, s_{i-1}))$
    if $p < t_{\text{EMMA}}$ **then**
      Break
    else
      $y_i, s_i \leftarrow \texttt{TextDecoder}(s_{<i}, h_{\leq j})$
      $k \leftarrow k + 1$
      $i \leftarrow i + 1$

Task	Evaluation Set	Language	BLEU
Bilingual	mTedX	spa-eng	37.1
	Must-C	eng-spa	38.1
Multilingual	Fleurs	100-eng	28.8 [1]

Table 2 - Offline model performance

demonstrates the model's capacity for rapid adaptation in offline-to-simultaneous transition, from an existing large scale multilingual translation model, SeamlessM4T (Seamless Communication et al., 2023).

In the bilingual setting, we follow the data setting from Inaguma et al. (2023). In the multilingual setting, we use the speech-to-text data from labeled and pseudo-labeled data in Seamless Communication et al. (2023)

In the bilingual S2T setup, we initialize the offline model with a pre-trained wav2vec 2.0 encoder (Baevski et al., 2020) and mBART decoder (Liu et al., 2020). Subsequently, we initialize the simultaneous model based on this pre-trained offline model. The bilingual model is trained on supervised on semi-unsupervised data. In the multilingual setting, we initialize the simultaneous model with the S2T part of an offline SeamlessM4T model, trained with the same labeled and pseudo-labeled data, and evaluate the model on `100-eng` directions.

5. Related Work

Recent research has focused on the neural end-to-end approach, anticipating that a simpler system can reduce errors between subsystems and enhance overall efficiency in direct translation. Initially applied to text translation, this approach extended to speech-to-text tasks, showing competitiveness against cascade approaches. Duong et al. (2016) introduced an attention-based sequence-to-sequence structure for speech-to-text, using a recurrent neural network (RNN) based encoder-decoder architecture. Despite the novelty, there was a significant quality downgrade compared to cascaded approaches. Subsequent studies Berard et al. (2016); Weiss et al. (2017); Bansal et al. (2018); Bérard et al. (2018) added convolutional layers, significantly improving end-to-end model performance. Leveraging the success of the Transformer in text translation Vaswani et al. (2017), Di Gangi et al. (2019) and Inaguma et al. (2020) applied it to speech translation, achieving further improvements in quality and training speed.

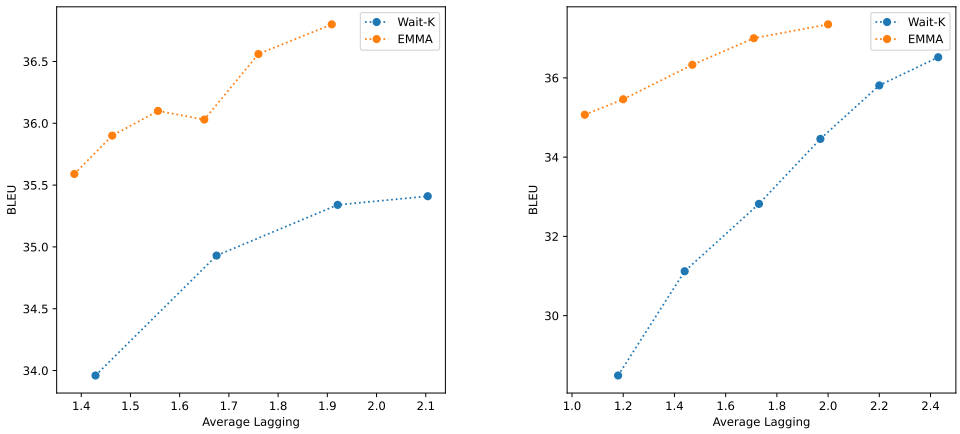
Simultaneous translation policies are categorized into three groups. The first category consists of predefined

context-free rule-based policies. Cho & Esipova (2016) proposed a Wait-If-* policy for offline simultaneous decoding, later modified by Dalvi et al. (2018) for consecutive prediction. Another variation, the Wait-k policy, was introduced by Ma et al. (2019a), where the model alternates between reading k inputs and performing read-write operations. The second category involves a learnable flexible policy with an agent, applying reinforcement learning. Examples include Grissom II et al. (2014), who used a Markov chain-based agent for phrase-based machine translation, and Gu et al. (2017), who introduced an agent that learns translation decisions from interaction with a pre-trained neural machine translation model. The third category features models using monotonic attention, replacing Softmax attention and leveraging closed-form expected attention. Notable works include Raffel et al. (2017), Chiu & Raffel (2018), Arivazhagan et al. (2019), and Ma et al. (2019b), demonstrating advancements in online linear time decoding and translation quality improvements.

6. Results

6.1 Quality-Latency Trade-off

We present the quality-latency trade-off on the bilingual setting. Figure 1 shows the BLEU score under different latency settings. We can see that the EMMA model significantly outperforms the Wait-k model on all the latency regions in both directions.



(a) Spanish to English **(b)** English to Spanish

Figure 1 - The quality-latency trade-off for bilingual model.

We present the quality-latency trade-off on the multilingual setting, in Table 3. SeamlessM4T-EMMA can achieve decent translation quality in a much shorter training time compared with training from scratch.

	t_{EMMA}	BLEU	AL
SeamlessM4T		28.8	-
SeamlessM4T-EMMA	0.5	26.0	1.75
	0.7	26.4	1.88
	0.4	25.9	1.68
	0.6	26.2	1.81

Table 3 - Average translation quality and latency under multilingual setting with different latency decision thresholds t_{EMMA}.

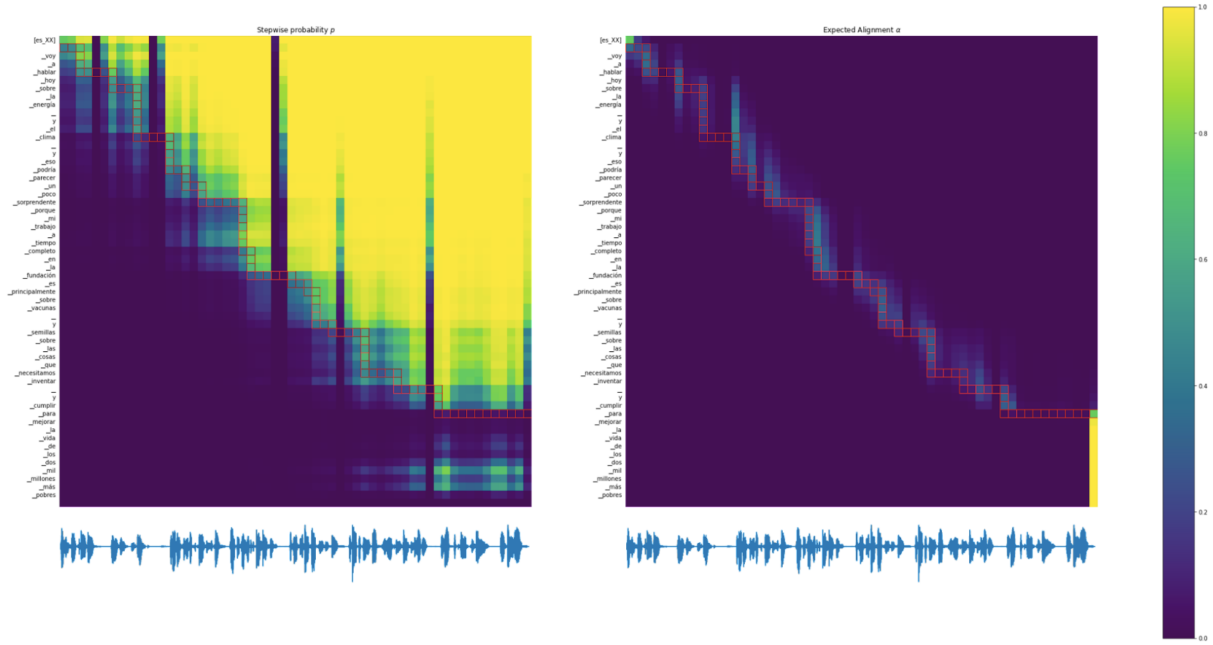


Figure 2 - Visualization of monotonic alignment. Left side is stepwise probability p and right side is estimated alignment α.

6.2 Visualization

We also visualize the policy learned from EMMA as Figure 2. We can see that with the guidance of latency and variance regularization, the model can learn a monotonic policy aligning input speech and output text.

7. Conclusion

We propose Efficient Monotonic Multihead Attention (EMMA), with a novel numerically stable alignment estimation. We fine-tune our model initialized from an offline model to speed up training and improve the performance. We evaluate our model in bilingual and multilingual settings, and observe improvement over the baseline in both.

References

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. doi: 10.18653/v1/P19-1126. URL https://www.aclweb.org/anthology/P19-1126.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12449–12460. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/92d1e1eb1cd6f9fba3227870bb6d7f07-Abstract.html.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. Low-Resource Speech-to-Text Translation. In *Interspeech 2018*, pp. 1298–1302. ISCA, September 2018. doi: 10.21437/Interspeech.2018-1326. URL http://www.isca-speech.org/archive/Interspeech_2018/abstracts/1326.html.

Alexandre Berard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. Listen and Translate: A Proof of Concept for End-to-End Speech-to-Text Translation. *NIPS Workshop on End- to-end Learning for Speech and Audio Processing*, December 2016. URL http://arxiv.org/abs/1612.01744. arXiv: 1612.01744.

A. Bérard, L. Besacier, A. C. Kocabiyikoglu, and O. Pietquin. End-to-End Automatic Speech Translation of Audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6224–6228, April 2018. doi: 10.1109/ICASSP.2018.8461690. ISSN: 2379-190X.

Chung-Cheng Chiu and Colin Raffel. Monotonic Chunkwise Attention. February 2018. URL https://openreview.net/forum?id=Hko85plCW.

Kyunghyun Cho and Masha Esipova. Can neural machine translation do simultaneous translation? *arXiv:1606.02012 [cs]*, June 2016. URL http://arxiv.org/abs/1606.02012. arXiv: 1606.02012.

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.

Mattia Antonino Di Gangi, Matteo Negri, Roldano Cattoni, Roberto Dessi, and Marco Turchi. Enhancing Transformer for End-to-end Speech-to-Text Translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pp. 21–31, Dublin, Ireland, August 2019. European Association for Machine Translation. URL https://www.aclweb.org/anthology/W19-6603.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. An Attentional Model for Speech Translation Without Transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 949–959, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1109. URL https://www.aclweb.org/anthology/N16-1109.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. Don't 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, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1140. URL https://www.aclweb.org/anthology/D14-1140.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. Learning to Translate in Real-time with Neural Machine Translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 1053–1062, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/E17-1099.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. ESPnet-ST: All-in-One Speech Translation Toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 302–311, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-demos.34. URL https://www.aclweb.org/anthology/2020.acl-demos.34.

Hirofumi Inaguma, Sravya Popuri, Ilia Kulikov, Peng-Jen Chen, Changhan Wang, Yu-An Chung, Yun Tang, Ann Lee, Shinji Watanabe, and Juan Pino. UnitY: Two-pass direct speech-to-speech translation with discrete units. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15655–15680, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.872. URL https://aclanthology.org/2023.acl-long.872.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual Denoising Pre-training for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020. doi: 10.1162/tacl_a_00343. URL https://aclanthology.org/2020.tacl-1.47. Place: Cambridge, MA Publisher: MIT Press.

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 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1289. URL https://www.aclweb.org/anthology/P19-1289.

Xutai Ma, Juan Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. Monotonic Multihead Attention. September 2019b. URL https://openreview.net/forum?id=Hyg96gBKPS.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. SIMULEVAL: An Evaluation Toolkit for Simultaneous Translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing: System Demonstrations, pp. 144–150, Online, October 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.19. URL https://www.aclweb.org/anthology/2020.emnlp-demos.19.

Xutai Ma, Juan Pino, and Philipp Koehn. SimulMT to SimulST: Adapting Simultaneous Text Translation to End-to-End Simultaneous Speech Translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 582–587, Suzhou, China, December 2020b. Association for Computational Linguistics. URL https://aclanthology.org/2020.aacl-main.58.

Matt Post. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 186–191, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6319. URL https://aclanthology.org/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*, ICML'17, pp. 2837–2846, Sydney, NSW, Australia, August 2017. JMLR.org.

Seamless Communication, Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, Paul-Ambroise Duquenne, Hady Elsahar, Hongyu Gong, Kevin Heffernan, John Hoffman, Christopher Klaiber, Pengwei Li, Daniel Licht, Jean Maillard, Alice Rakotoarison, Kaushik Ram Sadagopan, Guillaume Wenzek, Ethan Ye, Bapi Akula, Peng-Jen Chen, Naji El Hachem, Brian Ellis, Gabriel Mejia Gonzalez, Justin Haaheim, Prangthip Hansanti, Russ Howes, Bernie Huang, Min-Jae Hwang, Hirofumi Inaguma, Somya Jain, Elahe Kalbassi, Amanda Kallet, Ilia Kulikov, Janice Lam, Daniel Li, Xutai Ma, Ruslan Mavlyutov, Benjamin Peloquin, Mohamed Ramadan, Abinesh Ramakrishnan, Anna Sun, Kevin Tran, Tuan Tran, Igor Tufanov, Vish Vogeti, Carleigh Wood, Yilin Yang, Bokai Yu, Pierre Andrews, Can Balioglu, Marta R. Costa-jussà, Onur Celebi, Maha Elbayad, Cynthia Gao, Francisco Guzmán, Justine Kao, Ann Lee, Alexandre Mourachko, Juan Pino, Sravya Popuri, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, Paden Tomasello, Changhan Wang, Jeff Wang, and Skyler Wang. Seamlessm4t—massively multilingual & multimodal machine translation. *ArXiv*, 2023.

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*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. Sequence-to-Sequence Models Can Directly Translate Foreign Speech. In *Interspeech 2017*, pp. 2625–2629. ISCA, August 2017. doi: 10.21437/Interspeech.2017-503. URL http://www.isca-speech.org/archive/Interspeech_2017/abstracts/0503.html.

Appendix

A. Numerically Stable Estimation

Intuitively the α can be estimated from dynamic programming:

$$\alpha_{i,j} = p_{i,j} \sum_{k=1}^{j} \alpha_{i-1,k} \prod_{l=k}^{j-1} (1 - p_{i,l}) \tag{17}$$

While Equation (4) gives an closed form and parallel estimation of alignment, the denominator in the equation can cause instability and alignment vanishing in the training. We rewrite Equation (17) as

$$\alpha_{i,:} = p_{i,:} \odot \alpha_{i-1,:} \mathbf{T}(i) \tag{18}$$

where $\mathbf{T}(i)$ a transition matrix and each of its elements are defined as:

$$\mathbf{T}(i)_{m,n} = \begin{cases} \prod_{l=m}^{n-1}(1 - p_{i,l}) & m < n \\ 1 & m = n \\ 0 & m > n \end{cases} \tag{19}$$

$\mathbf{T}(i)_{m,n}$ is the probability of a `read` from x_m to x_n with y_i without `write`. Denote $t^i_{m,n} = \prod_{l=m}^{n}(1 - p_{i,l})$ We can see that if we manage to have $\mathbf{T}(i)$, then the $\alpha_{i,:}$ can be simply computed through matrix multiplication.

Define the probability of jumping from x_m to x_n with our write a new token y_i:

then we can expand $\mathbf{T}(i)$ as

$$\mathbf{T}(i) = \begin{pmatrix} 1 & t^i_{1,2} & t^i_{1,3} & t^i_{1,4} & \dots & t^i_{1,|X|} \\ 0 & 1 & t^i_{2,3} & t^i_{2,4} & \dots & t^i_{2,|X|} \\ 0 & 0 & 1 & t^i_{3,4} & \dots & t^i_{3,|X|} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \end{pmatrix}_{|X| \times |X|} \tag{20}$$

It can be further expressed as

$$\mathbf{T}(i) = \texttt{triu}_0 \left(\begin{pmatrix} 1 & t^i_{1,2} & t^i_{1,3} & t^i_{1,4} & \dots & t^i_{1,|X|} \\ 1 & 1 & t^i_{2,3} & t^i_{2,4} & \dots & t^i_{2,|X|} \\ 1 & 1 & 1 & t^i_{3,4} & \dots & t^i_{3,|X|} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & 1 & 1 & 1 & \dots & 1 \end{pmatrix}_{|X| \times |X|} \right) \tag{21}$$

$$= \texttt{triu}_0 \left(\texttt{cumprod}_2 (1 - \mathbf{P}^{ext}(i)) \right) \tag{22}$$

where $\texttt{triu}_b(\cdot)$ is function to extract the upper triangle of a matrix with an offset b [2], and $\texttt{cumprod}_2$ means that the computation is along the second dimension. Additionally, the extended probability matrix $\mathbf{P}^{ext}_i$ is

[2] See `torch.triu`

defined as

$$\mathbf{P}^{ext}(i) = \begin{pmatrix} 0 & p_{i,1} & p_{i,2} & \dots & p_{i,|X|-1} \\ 0 & 0 & p_{i,2} & \dots & p_{i,|X|-1} \\ 0 & 0 & 0 & \dots & p_{i,|X|-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & p_{i,|X|-1} \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix}_{|X|\times|X|} \tag{23}$$

$$= \texttt{triu}_1 \left(\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}_{|X|\times 1} \begin{pmatrix} p_{i,|X|} & p_{i,1} & \dots & p_{i,|X|-1} \end{pmatrix}_{1\times|X|} \right) \tag{24}$$

$$= \texttt{triu}_1 \left(J_{|X|\times 1} \texttt{roll}_1(p_{i,:}) \right) \tag{25}$$

Where $J_{|X|\times 1}$ is an all one matrix with a size of $|X|$ by 1, [3] and $\texttt{roll}_k$ is the function to shift matrix by k elements [4].

In summary, we can rewrite Equation (17) as

$$\alpha_{i,:} = p_{i,:} \odot \alpha_{i,:} \texttt{triu}_0 \left(\texttt{cumprod}_2 (1 - \texttt{triu}_1 \left(J_{|X|\times 1} \texttt{roll}_1(p_{i,:}) \right)) \right) \tag{26}$$

A code snippet of implementation of EMMA in PyTorch is shown as follows:

```
def monotonic_alignment(p):
    bsz, tgt_len, src_len = p.size()

    # Extension probablity matrix
    p_ext = p.roll(1, [-1]).unsqueeze(-2).expand(-1, -1, src_len, -1).triu(1)

    # Transition matrix
    T = (1 - p_ext).comprod(-1).triu()

    alpha = [p[:, [0]] * T[:, [0]]

    for i in range(1, tgt_len):
        alpha.append(p[:, [i]] * torch.bmm(alpha[i - 1], T[:, i]))

    return torch.cat(alpha[1:], dim=1)
```

[3] $J_{|X|\times 1}\texttt{roll}_1(p_{i,:})$ can be achived by `torch.expand` function.
[4] See `torch.roll`