INFUSING FUTURE INFORMATION INTO MONOTONIC ATTENTION THROUGH LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Simultaneous neural machine translation (SNMT) models start emitting the target sequence before they have processed the source sequence. The recent adaptive policies for SNMT use monotonic attention to perform *read/write* decisions based on the partial source and target sequences. The lack of sufficient information might cause the monotonic attention to take poor *read/write* decisions, which in turn negatively affects the performance of the SNMT model. On the other hand, human translators make better *read/write* decisions since they can anticipate the immediate future words using linguistic information and domain knowledge. In this work, we propose a framework to aid monotonic attention with an external language model to improve its decisions. We conduct experiments on the MuST-C English-German and English-French speech-to-text translation tasks to show the effectiveness of the proposed framework. It improves the quality-latency trade-off over the state-of-the-art monotonic multihead attention.

1 INTRODUCTION

Simultaneous Neural Machine Translation (SNMT) addresses the problem of real-time interpretation in machine translation. A typical application of real-time translation is conversational speech or live video caption translation. In order to achieve live translation, an SNMT model alternates between reading from the source sequence and writing to the target sequence using either a fixed or an adaptive *read/write* policy.

The fixed policies (Ma et al., 2019a) may introduce too much delay for some examples or not enough for others. The recent works focus on training adaptive policies using techniques such as monotonic attention. There are several variants of monotonic attention: hard monotonic attention (Raffel et al., 2017), monotonic chunkwise attention (Chiu & Raffel, 2018), monotonic infinite lookback attention (MILk) (Arivazhagan et al., 2019), and monotonic multihead attention (MMA) (Ma et al., 2019b). These monotonic attention processes can anticipate target words using only the available prefix source and target sequence. However, human translators anticipate the target words using their language expertise (linguistic anticipation) and contextual information (extra-linguistic anticipation) (Vandepitte, 2001) as well.

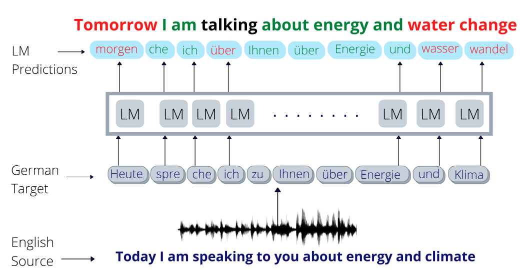


Figure 1: The finetuned XLM-RoBERTa language model predicts German words using the prefix as input (Green: Correct, Red: Incorrect, Black: Neutral).

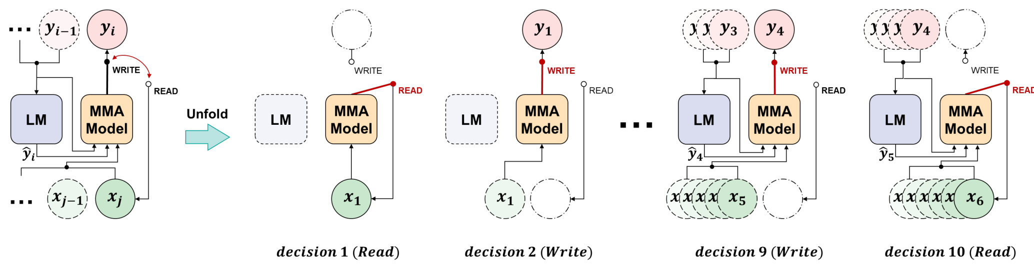


Figure 2: Overview of the simultaneous translation model with future information.

Motivated by human translation experts, we aim to augment monotonic attention with future information using language models (LM) (Devlin et al., 2019; Conneau et al., 2019). The LMs can learn linguistic and extra-linguistic information (Petroni et al., 2019), and help to improve the performance of the SNMT model. We modify monotonic attention using the plausible future information obtained from LMs to improve the MMA model's *read/write* policy. As shown in Figure 1, at each step, the LM takes the prefix target (and source, for cross-lingual LM) sequence and predicts the plausible future information. We hypothesize that aiding the monotonic attention with this future information might help the SNMT model improve the latency-quality trade-off.

The main contributions of this paper are: (1) A novel monotonic attention mechanism to leverage the plausible future information (2) Improved latency-quality trade-offs compared to the state-of-the-art MMA models on MuST-C (Di Gangi et al., 2019) English-German and English-French speech-to-text translation tasks. (3) Analyses on how our proposed monotonic attention achieves superior performance over MMA. We also analyze the performance of the proposed framework with custom and general-purpose LM.

2 Model

2.1 Monotonic Attention

The source and the target sequences are represented as $\mathbf{x} = \{x_1, x_2, \cdots, x_S\}$ and $\mathbf{y} = \{y_1, y_2, \cdots, y_T\}$, with S and T being the length of the source and the target sequences. The simultaneous machine translation models (SNMT) produce the target sequence concurrently with the growing source sequence. In other words, the probability of predicting the target token $y_i \in \mathbf{y}$ depends on the partial source and target sequences ($x_{\leq j} \in \mathbf{x}, y_{<i} \in \mathbf{y}$). In this work, we consider sequence-to-sequence based SNMT model in which each target token y_i is generated as follows:

$$h_j = \mathcal{E}(x_{\leq j}) \tag{1}$$
$$s_i = \mathcal{D}(y_{<i}, c_i = \mathcal{A}(h_j)) \tag{2}$$
$$y_i = Output(s_i) \tag{3}$$

where $\mathcal{E}(.)$ and $\mathcal{D}(.)$ are the Transformer (Vaswani et al., 2017) encoder and decoder layers, c_i is a context vector and $\mathcal{A}$ represents the attention mechanism. In offline MT, the context vector is computed using a soft attention mechanism (Bahdanau et al., 2015). In monotonic attention based SNMT, the context vector is computed as follows:

$$e_{i,j} = MonotonicEnergy(y_{i-1}, h_j) \tag{4}$$
$$p_{i,j} = Sigmoid(e_{i,j}) \tag{5}$$
$$z_{i,j} \sim Bernoulli(p_{i,j}) \tag{6}$$

When generating a new target token y_i, the decoder chooses whether to *read/write* based on Bernoulli selection probability $p_{i,j}$. When $z_{i,j} = 1$, we set $t_i = j$, $c_i = h_j$ (*write*) and generate the target token y_i; otherwise, we set $t_i = j + 1$ (*read*) and repeat Eq. 4 to 6. Here t_i refers to the index of the encoder entry needed to produce the i^{th} target token. Instead of hard alignment of $c_i = h_j$, Raffel

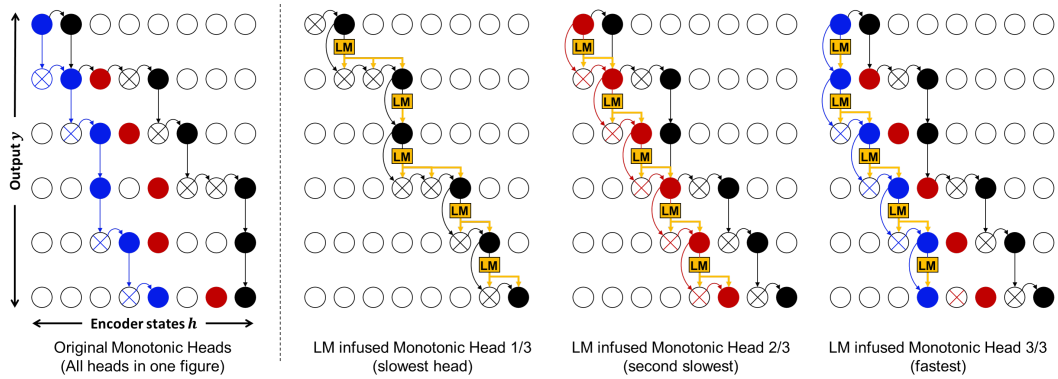


Figure 3: Overview of the monotonic multihead attention with future information.

et al. (2017) compute an expected alignment (α) in a recurrent manner and propose a closed-form parallel solution.

Arivazhagan et al. (2019) propose Monotonic Infinite Lookback Attention (MILk), which combines the soft attention with monotonic attention to attend all the encoder states from the beginning of the source sequence till t_i for each y_i. Ma et al. (2019b) extend MILk to monotonic multihead attention (MMA) to integrate it into the Transformer model.

The MMA model implements the monotonic energy function in Eq. 4 through scaled-dot product attention. For a Transformer model with L decoder layers and H attention heads per layer, the energy function of a h-th head encoder-decoder attention in the l-th decoder layer is computed as follows:

$$e_{i,j}^{l,h} = \left(\frac{h_j W_{l,h}^K (\mathbf{y}_{i-1} W_{l,h}^Q)^T}{\sqrt{d_k}}\right)_{i,j} \tag{7}$$

$$p_{i,j}^{l,h} = Sigmoid(e_{i,j}^{l,h}) \tag{8}$$

where $W_{l,h}^K$ and $W_{l,h}^Q$ are the projection matrices for h_j and $\mathbf{y}_{i-1}$, $\mathbf{y}_{i-1}$ is the representation of the previous output token and d_k is the dimension of the attention head.

The MMA attention for each head is calculated as follows:

$$u_{i,o}^{l,h} = \left(\frac{h_o W_{l,h}^K (y_{i-1} W_{l,h}^Q)^T}{\sqrt{d_k}}\right)_{i,j}, o \in 1, 2, \cdots, t_i \tag{9}$$

$$\beta_{i,j} = \sum_{k=j}^{|x|} \left(\frac{\alpha_{i,k} exp(u_{i,j})}{\sum_{n=1}^{k} exp(u_{i,n})}\right) \tag{10}$$

$$c_i = \sum_{j=1}^{|x|} \beta_{i,j} h_j \tag{11}$$

The attention mechanisms in MILk and MMA encourage the model to output the target token with limited source information by adding latency loss metrics to the training objective. Please refer to Arivazhagan et al. (2019) and Ma et al. (2019b) for more details.

2.2 Monotonic Attention with Future Information

The monotonic attention described in Section 2.1 performs anticipation based only on the currently available source and target information. We augment this anticipation process using future information which is extracted using LMs. LMs are known to inherently provide linguistic information (Petroni et al., 2019), while the extra-linguistic information is also obtained when the LM is finetuned on a particular task. To incorporate the future information, we propose the following modifications to the monotonic attention.

2.2.1 Future Representation Layer

At every decoding step i, the previous target token y_{i-1} is equipped with a plausible future token $\hat{y}_i$ as shown in the Figure 2. Since the token $\hat{y}_i$ comes from an LM possibly with a different tokenizer and vocabulary set, applying the model's tokenizer and vocabulary might split the token $\hat{y}_i$ further into multiple sub-tokens $\{\hat{y}_i^1, \hat{y}_i^2, \cdots, \hat{y}_i^m\}$. To get a single future token representation $\tilde{\mathbf{y}}_i \in \mathcal{R}^d$ from all the sub-tokens, we apply a sub-token summary layer as follows:

$$\tilde{\mathbf{y}}_i = \Gamma(\{\hat{y}_i^1, \hat{y}_i^2, \cdots, \hat{y}_i^m\}) \tag{12}$$

The Γ represents a general sequence representation layer such as a Transformer encoder layer or a simple normalized sum of sub-token representations.

We enrich $\tilde{\mathbf{y}}_i$ at every layer l of the decoder block by applying a residual-feed forward network similar to the final sub-layer in the transformer decoder block.

$$\tilde{\mathbf{y}}_i^l = FFN(\tilde{\mathbf{y}}_i^{l-1}) \tag{13}$$

2.2.2 Monotonic Energy Layer with Future Information

We can add the plausible future information to the output layer or append it to the target token representation y_{i-1}. However, the MMA *read/write* decisions happen in Eq. 7, therefore, we integrate $\tilde{\mathbf{y}}_i$ into the Eq. 7.

This way of integration of future information allows the model to condition the LM output usage on the input speech. Hence, it can choose to discard incorrect information. The integration is carried out by modifying Eq. 4 - Eq. 8 in the following way:

First, we compute the monotonic energy for future information using the enriched future token representation $\tilde{\mathbf{y}}_i^l$ available at each layer:

$$\tilde{e}_{i,j}^l = \left(\frac{h_j \tilde{W}_l^K (\tilde{\mathbf{y}}_i \tilde{W}_l^Q)^T}{\sqrt{d}} \right)_{i,j} \tag{14}$$

where $\tilde{W}_l^K$ and $\tilde{W}_l^Q$ are the projection matrices for h_j and $\tilde{\mathbf{y}}_i$, and d_k is the dimension of the attention head of Eq. 7.

We integrate the future monotonic energy function into Eq. 8 as follows:

$$\tilde{p}_{i,j}^{l,h} = \Omega(e_{i,j}^{l,h}, \tilde{e}_{i,j}^l), \tag{15}$$

Ω represents a general modulation operator and can be replaced with feature-wise linear modulation (Perez et al., 2018) or multiplicative or additive operations. As shown in Figure 3, during training, each head in the monotonic energy $e_{i,j}^{l,h}$ sees the same future monotonic energy $\tilde{e}_{i,j}^l$, since during the inference of multihead monotonic attention, the *write* operations depend on the slowest head.

After computing $\tilde{p}_{i,j}^{l,h}$, we compute c_i using Eq. 11. The overall process is described in Algorithm 1. In our experiments, we use the MMA-Infinite Lookback attention model, but our algorithm can be easily extended for MMA-H by modifying the context vector computation to choose only one encoder state.

2.2.3 Inference

During the inference time, the *start* token does not contain any future information. After predicting the first target token, for every subsequent prediction of target token y_i, we invoke the LM to predict

Algorithm 1: Monotonic Attention with Future Information

1 **Input**: Training examples, $E = \{x^n, y^n\}_{n=1}^{N}$, hyperparameters such as learning rate (α), λ (latency), language model (LM)
2 Extract plausible future information using LM, $\hat{y}_i^n = LM(y_1^n, \cdots, y_{i-1}^n) \quad \forall i, n$
3 Append each $\hat{y}^n$ information to target sequence y^n. New target sequence $\bar{y}^n = \{(y_1^n, \hat{y}_1^n), (y_2^n, \hat{y}_2^n), \cdots, (y_{|y|}^n, \emptyset)\}$
4 Modified training examples are $\bar{E} = \{x^n, \bar{y}^n\}_{n=1}^{N}$
5 Initialize model parameters θ
6 **while** *training is not done* **do**
7   Sample a training example(s) from $\bar{E}$
8   compute $\{h_1, \cdots, h_{|x|}\}$ for x (Eq. 1)
9   tokenize y and $\hat{y}_i$
10   Run a sub-token summary layer (Eq. 12) on $\hat{y}$) , to obtain $\tilde{\mathbf{y}}_i$ for each $\mathbf{y}_{i-1}$
11   **for** *each decoding step* **do**
12   Compute monotonic energy and future monotonic energy for $\mathbf{y}_{i-1}$, $\tilde{\mathbf{y}}_i$ (Eq. 7 and 14)
13   Compute *read/write* decision (Eq. 15)
14   compute context vector c_i (Eq. 11), and output target token (Eq. 2, 3)
15   **end**
16   Compute the latency loss along with negative log likelihood
17   Update θ with gradient descent
18 **end**
19 Return: θ

the next plausible future token. Whenever new source information arrives, we run step 8 of Algorithm 1. Similarly, for every new target token prediction, we extract new plausible future information from LM using the available predicted sequence, and then we run step 9 to 14 of the Algorithm 1.

3 Experiments

3.1 Datasets and Metrics

We conduct our experiments on the English(En)-German(De) and English(En)-French(Fr) speech-to-text (ST) translation tasks. These tasks are more involved compared to the text-to-text translation task since these are low resource tasks. Moreover, due to different input-output modalities, the difference in source and target sequence lengths is larger compared to the text-to-text task. We use the EnDe, and EnFr portions of the MuST-C dataset. More details about the datasets have been provided in the Appendix.

The speech sequence is represented using 80-dimensional log mel features extracted using the Kaldi (Povey et al., 2011) toolkit with a 25ms window size and 10ms shift. The target sequence is represented as subwords using a SentencePiece (Kudo & Richardson, 2018) model with a unigram vocabulary of size 10,000. We evaluate the performance of the models on both the latency and quality aspects. We use Average Lagging(AL) as our latency metric and case-sensitive detokenized SacreBLEU (Post, 2018) to measure the translation quality, similar to Ma et al. (2020b). The best models are chosen based on the dev set results and reported results are computed using the MuST-C tst-COMMON test sets.

3.2 Implementation Details

Our base model is adopted from Ma et al. (2020b) and the initial implementation is taken from Fairseq[1] repository. In the text-to-text case, each encoder state corresponds to a vocabulary unit. Hence, the *read/write* decisions are taken for each encoder state. For SNMT, each encoder state represents only 40ms of speech, assuming a sub-sampling factor of 4 from the convolutional layers. We use a pre-decision ratio of 7 (speech segment size of 280ms), which means that the simultaneous

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

read/write decisions are made after every seven encoder states, which roughly corresponds to a word (Ma et al., 2020b). Since we train MMA-IL (Ma et al., 2019b) models, we set $\lambda_{var} = 0$ for all our experiments as it was not reported to be helpful for models with infinite lookback. We use λ or $\lambda_{latency}$ to refer to the hyperparameter corresponding to the weighted average (λ_{avg}) in MMA. The values of this hyperparameter λ are chosen from the set $\{0.01, 0.05, 0.1\}$. The Γ layer in Eq. 12 computes the normalized sum of the sub-token representations. For SLM, it simply compute the embedding since it shares the same vocabulary set. The Ω layer in Eq. 15 performs the additive operation to add the energies corresponding to previous output token y_{i-1} and the prediction $\hat{y}_i$. All the models were trained by simulating 8 GPU settings on a single NVIDIA v100 GPU.

3.3 Models

We train a baseline model based on Ma et al. (2020b), called the MMA model. The base MMA model encoder and decoder embedding dimensions are set to 392, whereas our proposed model's encoder and decoder embeddings are set to 256 to have similar parameters ($\approx 39M$) for a fair comparison. Apart from the encoder and decoder embedding dimension, all other hyperparameter settings and training procedures are similar for all the reported models. We train two MMA models based on two different LMs used for extracting future information. Further details have been provided in Section 3.4.

We follow the similar training process as Ma et al. (2020b). We train an English ASR model using the source speech data. Next, we train a simultaneous model without the latency loss (setting $\lambda_{latency} = 0$) after initializing the encoder from the English ASR model. After this step, we finetune the simultaneous model for different λs. This training process is repeated for all the reported models and for each task.

3.4 Language Model

We use two different language models to train our proposed LM-based simultaneous speech translation model. Firstly, we use the pretrained XLM-RoBERTa (Conneau et al., 2019) model from Huggingface Transformers[2] model repository. It is a multilingual (more than 100 languages) LM trained on a wide variety of cross-lingual tasks. The model contains 550M parameters with 24 layers, 1,024 hidden-states. Since the LM output can be very open-ended and might not directly suit/cater to our task and dataset, we finetune the head of the model using the target MuST-C text data corresponding to each task.

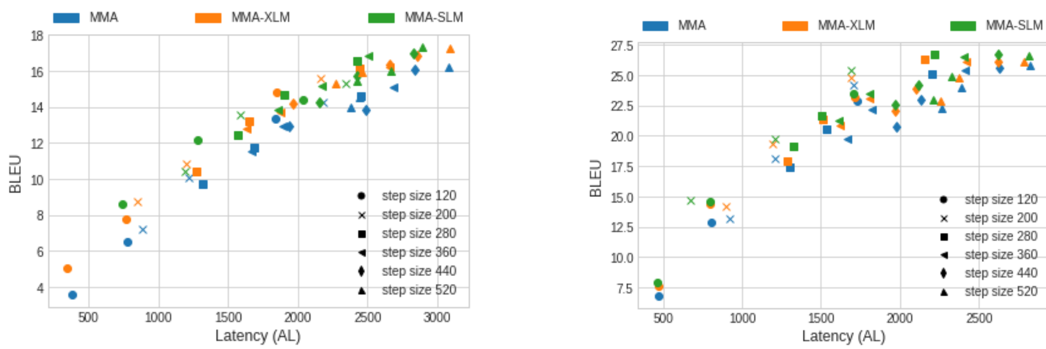
We also train a smaller language model (SLM), which contains 6 Transformer decoder layers, 512 hidden-states and 24M parameters. We use the MuST-C data along with additional data augmentation (refer to Appendix) to reduce overfitting. Although trained on a much smaller monolingual dataset, it helps to remove the issues related to vocabulary mismatch as discussed in the Section 2.2.1. This LM has lower inference time and has higher accuracy on the next token prediction task as compared to XLM, 30.15% vs. 21.5% for German & 31.65% vs. 18.45% for French. More details about LMs have been provided in the Appendix. The models trained using these LMs are referred to as MMA-XLM and MMA-SLM.

3.5 Results

In this section, we provide the results for MMA, MMA-XLM, and MMA-SLM in the form of latency-quality trade-off curves. We also provide analysis for the results in the later part of the section.

Latency-Quality Curves The hyperparameter corresponding to the average latency loss allows us to train separate models for different latency regimes. Moreover, the quality and latency for a particular model can also be varied by controlling the speech segment size during the inference. Speech segment size or step size refers to the duration of speech (in ms) processed corresponding to each *read* decision. We vary these hyperparameters for all the three models, namely MMA, MMA-XLM and MMA-SLM. The evaluation was carried out using SimulEval toolkit (Ma et al., 2020a).

[2]https://huggingface.co/transformers/



(a) EnDe Task

(b) EnFr Task

Figure 4: BLEU vs Average Lagging results for MMA, MMA-XLM and MMA-SLM models for different speech segment step sizes.

The BLEU-AL curves for all the models have been provided in Figure 4. We vary the step sizes from 120 ms to 520 ms in order to get performances corresponding to different latency regimes. We can observe that the LM-based models using both XLM and SLM provide a significant performance improvement over the baseline MMA model. We observe improvements in the range of 1-2 BLEU scores consistently across all the latency regimes.

LM anticipation vs Latency In order to measure the relative weight given to the predictions from the LM, we compare the norm of the monotonic energies corresponding to the LM predictions e_{pred} (Eq. 14) and the previous output tokens e_{output} (Eq. 7). Let us define LM prediction weight as

$$LM_{pw} = \left(\frac{\|e_{pred}\|}{\|e_{output}\|} \right) \tag{16}$$

In Figure 5, we plot the variation of LM_{pw}(averaged) vs. λ. For the sake of these plots, we also computed LM_{pw} for 2 additional values of λ, $\{0.001, 0.005\}$ We can observe that as the latency requirements become more and more strict, the model starts to give more weightage to the predictions coming from the LM. In other words, as the need for anticipation increases, the model starts to rely more on the LM predictions.

Effect of LM Size on Latency-Quality We train several SLM models with varying sizes in our experiments and choose the best model based on the top-1 accuracy. As we increase the number of layers in the LM model from 2 to 4 to 6 layers, the SLM and the proposed MMA with future information models have shown performance improvements. However, increasing the number of layers greater than 6 does not yield any performance improvements. We also notice this degradation of performance with the XLM model while varying the number of hidden layers in the LM head. The best performances which are reported in Figure 4 are observed with the 6 layer SLM model.

Effect of LM Size on CAAL As we can observe from the Figure 4, the results for MMA-XLM and MMA-SLM are very close. Not only does MMA-SLM perform slightly better than MMA-XLM, but the prediction computation time is also much lower since XLM is much deeper than SLM. The average time taken to compute one token during prediction for SLM is 6.1ms as compared to 24.78ms for XLM. However, XLM is more suitable for multilingual settings since we can use same LM for all the languages.

In order to account for the computation time incurred by the model, we also use the Computation Aware Average Latency (CAAL) introduced in Ma et al. (2020b). AL (non-computation aware) is measured in terms of the duration of speech listened to before generating target token, while CAAL uses the wall-clock time, which also factors in the time elapsed due to the model complexity. In Figure 6, we provide the CAAL for various λ values for approximately similar BLEU. For a given value of AL, LM based MMA models have a higher CAAL when compared to the MMA model. This gap is expected and occurs due to the time taken for the LM to compute the predictions. However,

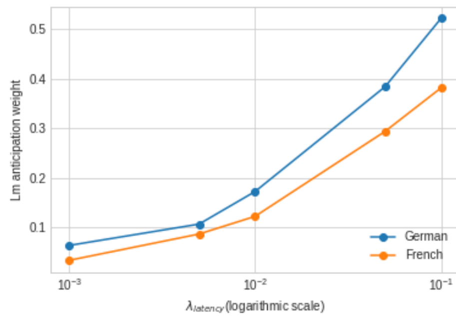
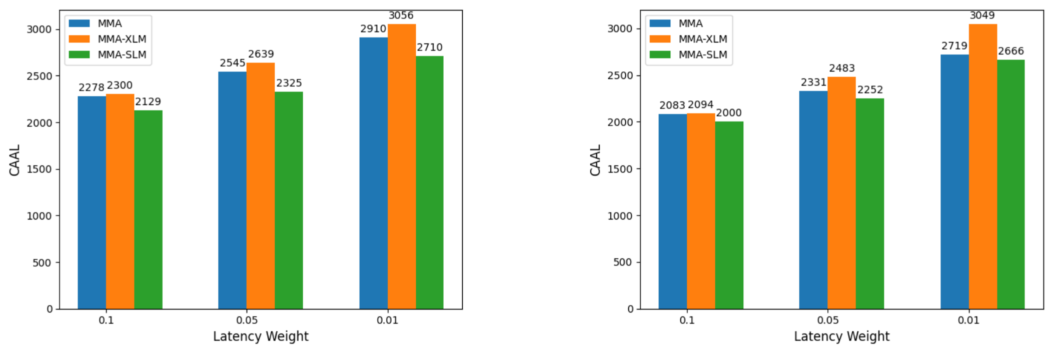


Figure 5: LM prediction weight vs λ



(a) EnDe Task

(b) EnFr Task

Figure 6: Computational Aware Average Latency of MMA, MMA-XLM, MMA-SLM with similar BLEU scores and different latencies = $\{0.1, 0.05, 0.01\}$

both MMA-XLM and MMA-SLM improve the latency-quality trade-off and hence reduce the AL for a given BLEU. As observed, MMA-SLM has lesser CAAL as compared to MMA since the extra computation time is balanced by the reductions in AL due to the algorithmic improvements. MMA-XLM, on the other hand, has a slightly higher CAAL. English speakers utter 6.2 syllables per sec (Pellegrino et al., 2011), which means 160ms/syllable. Both German and French readers read roughly 5 syllables/sec, 200ms/syllable (Trauzettel-Klosinski et al., 2012). Considering this human perception speed, a gap of 6ms or 24ms per sub-word (which might contain multiple syllables) should not cause acute deterioration in the user experience.

4 Related Work

The earlier works in streaming simultaneous translation such as Cho & Esipova (2016); Gu et al. (2016); Press & Smith (2018) lack the ability to anticipate the words with missing source context. Ma et al. (2019a) established a more sophisticated approach by integrating their *read/write* agent directly into MT. Similar to Dalvi et al. (2018), they employ a fixed agent that first reads k source tokens and then proceeds to alternate between *write* and *read* until the source tokens are finished. Recently, the adaptive policies based on several variants of monotonic attention for SNMT have been explored: 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 improves upon the wait-k training with an attention that can adapt how it will wait based on the current context. Monotonic multihead attention (MMA) (Ma et al., 2019b) extends MILk to transformer-based models.

Gulcehre et al. (2015; 2017) propose shallow fusion of language models (LMs) into text-to-text machine translation (MT) by combining LM and MT scores at inference time using a log-linear model.

However, this has a mismatch between training and inference of MT. To overcome this drawback, Stahlberg et al. (2018) integrate the LM scores during MT training. They use a pretrained LM and train the MT system to optimize the combined score of LM and MT on the training set. Sriram et al. (2018) explored a similar idea for Automatic Speech Recognition (ASR) using a gating network for controlling the relative contribution of the LM. These techniques allow the main sequence to sequence model (either ASR or MT) to focus on modeling the source sentence, while the LM controls the target side generation. Wu et al. (2020) implicitly uses future information during training of SNMT systems by simultaneously training different wait-k systems. The translation model is jointly trained with a controller model that decides which k is optimal to use for training a particular batch of examples. However, they do not use any explicit future information during training and inference.

End-to-end speech translation (Indurthi et al., 2020; Sperber & Paulik, 2020) has recently made great progress and even surpassed the cascaded models (ASR followed by MT) (Ney, 1999; Post et al., 2013). Ren et al. (2020) investigate how to adapt the simultaneous models for speech-to-text translation tasks. Han et al. (2020) use meta-learning algorithm MAML (Finn et al., 2017) to improve wait-k based simultaneous speech-to-text models. Ma et al. (2020b) explore the usage of monotonic multihead attention for speech translation by introducing a pre-decision module.

5 Conclusion

In this work, we provide a generic framework to integrate plausible future information into simultaneous models. This information helps to improve the anticipation for monotonic attention based models. We rely on language models to extract future information and propose a new monotonic attention mechanism to infuse into simultaneous models. We conduct several experiments on low resource speech-to-text translation tasks to show the effectiveness of proposed approach. We have achieved superior quality-latency trade-off compared to the state-of-the-art monotonic multihead attention. In the future work, we plan to extend the proposed framework to the text-to-text simultaneous translation and analyze different future information fusion mechanisms.

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.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. 2015.

Chung-Cheng Chiu and Colin Raffel. Monotonic chunkwise attention. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hko85plCW.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

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.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 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), pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. MuST-C: a Multilingual Speech Translation Corpus. 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)*, pp. 2012–2017, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1202. URL https://www.aclweb.org/anthology/N19-1202.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 06–11 Aug 2017. URL http://proceedings.mlr.press/v70/finn17a.html.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor OK Li. Learning to translate in real-time with neural machine translation. *arXiv preprint arXiv:1610.00388*, 2016.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On using monolingual corpora in neural machine translation, 2015.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, and Yoshua Bengio. On integrating a language model into neural machine translation. *Computer Speech and Language*, 45:137–148, September 2017. ISSN 0885-2308. doi: 10.1016/j.csl.2017.01.014.

Hou Jeung Han, Mohd Abbas Zaidi, Sathish Reddy Indurthi, Nikhil Kumar Lakumarapu, Beomseok Lee, and Sangha Kim. End-to-end simultaneous translation system for IWSLT2020 using modality agnostic meta-learning. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pp. 62–68, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.iwslt-1.5. URL https://www.aclweb.org/anthology/2020.iwslt-1.5.

S. Indurthi, H. Han, N. K. Lakumarapu, B. Lee, I. Chung, S. Kim, and C. Kim. End-end speech-to-text translation with modality agnostic meta-learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7904–7908, 2020.

Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. 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. 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2072. URL https://www.aclweb.org/anthology/N18-2072.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

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, 2019a.

Xutai Ma, Juan Pino, James Cross, Liezl Puzon, and Jiatao Gu. Monotonic multihead attention, 2019b.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. Simuleval: An evaluation toolkit for simultaneous translation. *CoRR*, abs/2007.16193, 2020a. URL https://arxiv.org/abs/2007.16193.

Xutai Ma, Juan Pino, and Philipp Koehn. Simulmt to simulst: Adapting simultaneous text translation to end-to-end simultaneous speech translation. *arXiv preprint arXiv:2011.02048*, 2020b.

Hermann Ney. Speech translation: Coupling of recognition and translation. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258)*, volume 1, pp. 517–520. IEEE, 1999.

François Pellegrino, Christophe Coupé, and Egidio Marsico. A cross-language perspective on speech information rate. *Language*, pp. 539–558, 2011.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16528.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? 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)*, pp. 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1250. URL https://aclanthology.org/D19-1250.

Matt Post. A call for clarity in reporting bleu scores. *arXiv preprint arXiv:1804.08771*, 2018.

Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. Improved speech-to-text translation with the fisher and callhome spanish–english speech translation corpus. In *International Workshop on Spoken Language Translation (IWSLT 2013)*, 2013.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number CONF. IEEE Signal Processing Society, 2011.

Ofir Press and Noah A Smith. You may not need attention. *arXiv preprint arXiv:1810.13409*, 2018.

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, and Douglas Eck. Online and linear-time attention by enforcing monotonic alignments. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2837–2846. PMLR, 06–11 Aug 2017. URL http://proceedings.mlr.press/v70/raffel17a.html.

Yi Ren, Jinglin Liu, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. SimulSpeech: End-to-end simultaneous speech to text translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 3787–3796, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.350. URL https://www.aclweb.org/anthology/2020.acl-main.350.

Matthias Sperber and Matthias Paulik. Speech translation and the end-to-end promise: Taking stock of where we are. *arXiv preprint arXiv:2004.06358*, 2020.

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. Cold fusion: Training seq2seq models together with language models. In *Proc. Interspeech 2018*, pp. 387–391, 2018. doi: 10.21437/Interspeech.2018-1392. URL http://dx.doi.org/10.21437/Interspeech.2018-1392.

Felix Stahlberg, James Cross, and Veselin Stoyanov. Simple fusion: Return of the language model. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 204–211, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6321. URL https://www.aclweb.org/anthology/W18-6321.

Susanne Trauzettel-Klosinski, Klaus Dietz, IReST Study Group, et al. Standardized assessment of reading performance: The new international reading speed texts irest. *Investigative ophthalmology & visual science*, 53(9):5452–5461, 2012.

Sonia Vandepitte. Anticipation in conference interpreting: a cognitive process. *Alicante Journal of English Studies / Revista Alicantina de Estudios Ingleses*, 0(14):323–335, 2001. ISSN 2171-861X. doi: 10.14198/raei.2001.14.18. URL https://raei.ua.es/article/view/2001-n14-anticipation-in-conference-interpreting-a-cognitive-process.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Xueqing Wu, Yingce Xia, Lijun Wu, Shufang Xie, Weiqing Liu, Jiang Bian, Tao Qin, and Tie-Yan Liu. Learn to use future information in simultaneous translation, 2020.

A LANGUAGE MODELS

As mentioned earlier, we train two different language models (LMs) and use them to improve the anticipation in monotonic attention based Simultaneous models.

A.1 XLM-ROBERTA(XLM-R)

XLM-R Large model [3] was trained on the 100 languages CommonCrawl corpora total size of 2.5TB with 550M parameters from 24 layers, 1024 hidden states, 4096 feed-forward hidden-states, and 16 heads. Total number of parameters is 558M. We finetune the head of the XLM-R LM model using the Masked Language Modeling objective which accounts for 0.23% of the total model parameters, i.e., 1.3M parameters.

A.2 SMALLER LANGUAGE MODEL

Since the LM predictions are computed serially during inference, the time taken to compute the LM token serves as a bottleneck to the latency requirements. To reduce the LM computation time, we train a smaller Language Model (SLM) from scratch using the Causal Language Modeling objective. SLM is composed of 6 Transformer decoder blocks, 512 hidden-states, 2048 feed-forward hidden-states & 8 attention heads. It alleviates the need for the sub-token summary layer since it shares the vocabulary and tokenization with the MMA models. The train examples are at the sentence level, rather than forming a block out of multiple sentences(which is the usual case for Language Models).

Since the target texts contain lesser than 250k examples, we use additional data augmentation techniques to upsample the target data. We also use additional data to avoid overfitting on the MuST-C target text. Details have been provided in A.2.1.

A.2.1 DATA AUGMENTATION

Up-Sampling: To boost the LM performance and mitigate overfitting, we use contextual data augmentation (Kobayashi, 2018) to upsample the MuST-C target text data by substituting and inserting words based on LM predictions. We use the NLPAUG [4] package to get similar words based on contextual embeddings. From the Hugging Face Repository, we use two different pretrained BERT (Devlin et al., 2019) models for German *bert-base-german-dbmdz-cased* & *bert-base-german-dbmdz-uncased* and *bert-base-fr-cased* for French. We upsample German to 1.13M examples and French to 1.38M examples.

Additional Data: We also use additional data to avoid overfitting. For German we use the Newscrawl(WMT 19) data which includes 58M examples. For French, we use Common Crawl and Europarl to augment 4M extra training examples.

We observe that both upsampling and data augmentation help us to reduce the overfitting on the MuST-C dev set.

A.3 TOKEN PREDICTION

For each output token, the LM prediction is obtained by feeding the prefix upto that token to the LM model. These predictions are pre-computed for training and validation sets. This ensures parallelization and avoids the overhead to run the LM simultaneously during the training process. During inference, the LM model is called every time a new output token is written.

B DATASET

The MuST-C dataset comprises of English TED talks, the translations and transcriptions have been aligned with the speech at sentence level. Dataset statistics have been provided in the Table 1.

[3]https://huggingface.co/xlm-roberta-large
[4]https://pypi.org/project/nlpaug/

Task	# Hours	# Sentences			# Talks	# Words	
		Train	Dev	Test		Source	Target
English-German	408	225k	1,423	2,641	2,093	4.3M	4M
English-French	492	269k	1,412	2,632	2,510	5.2M	5.4M

Table 1: Dataset Statistics(# - Number of)

Hyperparameter	MMA	MMA-XLM/CLM
encoder layers	12	12
encoder embed dim	292	256
encoder ffn embed dim	2048	2048
encoder attention heads	4	4
decoder layers	6	6
decoder embed dim	292	256
decoder ffn embed dim	2048	2048
monotonic ffn embde dim	–	2048
decoder attention heads	4	4
dropout	0.1	0.1
optimizer	adam	adam
adam-β	(0.9, 0.999)	(0.9, 0.999)
clip-norm	10.0	10.0
lr scheduler	inverse sqrt	inverse sqrt
learning rate	0.0001	0.0001
warmup-updates	4000	4000
label-smoothing	0.0	0.0
max tokens	40000	40000
conv layers	2	2
conv stride	(2,2)	(2,2)

Table 2: Model Hyperparameters

C Hyperparameters

The details regarding the hyperparameters for the model have been provided in Table 2.