Simultaneous Masking, Not Prompting Optimization: A Paradigm Shift in Fine-tuning LLMs for Simultaneous Translation

Matthew Raffel **Victor Agostinelli** **Lizhong Chen**
Oregon State University
{raffelm, agostinv, chenliz}@oregonstate.edu

Abstract

Large language models (LLMs) have achieved state-of-the-art performance in various language processing tasks, motivating their adoption in simultaneous translation. Current fine-tuning methods to adapt LLMs for simultaneous translation focus on prompting optimization strategies using either data augmentation or prompt structure modifications. However, these methods suffer from several issues, such as unnecessarily expanded training sets, computational inefficiency from dumping the key and value cache, increased prompt sizes, or restriction to a single decision policy. To eliminate these issues, in this work, we propose *SimulMask*, a new paradigm for fine-tuning LLMs for simultaneous translation. It utilizes a novel attention mask approach that models simultaneous translation during fine-tuning by masking attention for a desired decision policy. Applying the proposed SimulMask on a Falcon LLM for the IWSLT 2017 dataset, we have observed a significant translation quality improvement compared to state-of-the-art prompting optimization strategies on five language pairs while reducing the computational cost.

1 Introduction

Simultaneous translation refers to the process of producing a target output translation concurrently with an oncoming source input. In our increasingly interconnected world, where communication across languages in real-time is desired, simultaneous translation is becoming a requirement. As such, there is a need for machine learning models to fill this role.

Current literature has primarily focused on adapting end-to-end Transformer models (Vaswani et al., 2017) to overcome the difficulties of simultaneous machine translation (SimulMT) due to their reduced parameter counts and greater inference speed(Ma et al., 2020b). However, the recent successes of large language models (LLMs) (Touvron et al., 2023; Jiang et al., 2023; Almazrouei et al., 2023) has prompted preliminary research applying them to SimulMT through fine-tuning and inference techniques (Agostinelli et al., 2023; Wang et al., 2023; Koshkin et al., 2024; Wang et al., 2024; Guo et al., 2024). Unfortunately, most modern works have neglected the computational increases created by dumping the target sequence's key and value (KV) cache (Wang et al., 2024). Furthermore, there has yet to be a universal approach to fine-tuning LLMs for SimulMT that is not unnecessarily computationally expensive by either expanding the dataset through data augmentation, a process referred to as prefix fine-tuning (Agostinelli et al., 2023; Wang et al., 2023; Koshkin et al., 2024) or increasing the prompt length through prompt restructuring (Koshkin et al., 2024; Wang et al., 2024).

The lack of an efficient fine-tuning strategy of LLMs for SimulMT has led us to propose a new paradigm, referred to as *SimulMask*. SimulMask is a novel attention mask to model SimulMT during fine-tuning by redistributing the attention under a decision policy. By design, SimulMask is broadly applicable to both flexible and fixed decision policies, creating a path forward for future work to build upon it. Furthermore, if we avoid injecting positional information into the keys and values through a modified ALiBi (Press et al., 2021), SimulMask allows for KV caching during SimulMT without accuracy degradation.

To validate the efficacy of SimulMask, we fine-tuned and evaluated 1.3 billion parameter Falcon models pre-trained on the RefinedWeb dataset using SimulMask (Almazrouei et al., 2023; Penedo et al., 2023) and compared them against identical Falcon models that adopt existing prefix fine-tuning or prompt restructuring methods on the IWSLT 2017 dataset (Cettolo et al., 2017). From the results, we demonstrate models fine-tuned with SimulMask outperform prefix fine-tuning and prompt restructuring models at SimulMT for a given latency

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 18302–18314

regime with a reduced computational cost.

The main contributions of the paper include:

1. Providing insights on the shortcomings of current methods in adapting LLMs to SimulMT.
2. Proposing a novel attention masking approach to fine-tune SimulMT LLMs that enables efficient training and inference.
3. Demonstrating the efficacy of our approach in terms of translation quality and computational costs by evaluating them on multiple language pairs across varied latencies.

2 Background and Related Work

2.1 Masked Transformer Self-Attention

We briefly review self-attention functionality in Transformers (Vaswani et al., 2017) focusing on masking behavior in Equation 1. $\boldsymbol{M}$ is defined as an optional attention mask.

$$\boldsymbol{A} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T + \boldsymbol{M}}{\sqrt{d_{head}}}\right)\boldsymbol{V} \quad (1)$$

Critically, $\boldsymbol{M}$ functions by modeling context limitations or time-based dependencies that might exist during inference but do not exist during training/fine-tuning. For generative Transformer blocks or architectures, $\boldsymbol{M}$ is defined as a causal attention mask, where each entry, $\boldsymbol{M}_{ij}$, is represented by Equation 2 to avoid attending to the future.

$$\boldsymbol{M}_{ij} = \begin{cases} 0, & \text{if } j \leq i \\ -\infty, & \text{otherwise} \end{cases} \quad (2)$$

2.2 Simultaneous Translation

SimulMT is dictated by read-write decision policies, whereby a model will wait a specific amount of time before alternating between reading and writing in fixed or flexible intervals. One fixed decision policy for SimulMT that is broadly adopted as a common baseline to build on due to its effectiveness and simplicity is the wait-k policy (Ma et al., 2019). As the name suggests, the wait-k policy will wait for k words before alternating between writing and reading a word. Although effective in SimulMT, alternative adaptive policies have gained traction, which base reading and writing on an auxiliary model or a predefined set of rules (Cho and Esipova, 2016; Gu et al., 2017; Zheng et al., 2019). While capable of impressive results, such adaptive policies often incur additional computational costs.

A Transformer is trained for a SimulMT decision policy by masking attention scores in the encoder self-attention and the decoder cross-attention. In the case of the encoder self-attention, each source *token* is prevented from attending to future source tokens following a decision policy (Ma et al., 2019). An example mask is provided in Appendix A.1.

Alternatively, in decoder cross-attention, each target token is prevented from attending to future source *hidden states* following the decision policy (Papi et al., 2022a). Equation 3 expresses each entry of the decoder cross-attention mask, $\boldsymbol{M}_{tj}$.

$$\boldsymbol{M}_{tj} = \begin{cases} 0, & \text{if } j \leq f(t) \\ -\infty, & \text{otherwise} \end{cases} \quad (3)$$

In Equation 3, $f(t)$, is a decision policy function that denotes the cumulative number of source hidden states to read when predicting target token t.

2.3 Applying LLMs to SimulMT

LLMs have demonstrated remarkable performance on neural machine translation (NMT) (Moslem et al., 2023; Vilar et al., 2023; Xu et al., 2023; Zhang et al., 2023; Iyer et al., 2023). Such successes have prompted recent works to extend the reach of LLMs into the realm of SimulMT (Agostinelli et al., 2023; Wang et al., 2023; Koshkin et al., 2024; Wang et al., 2024; Guo et al., 2024). LLMs are especially promising for the field of SimulMT due to their strong understanding of language semantics and meaning. Intuitively, SimulMT LLMs inject holistic linguistic knowledge that could allow for correct translation decisions when facing difficult contextual obstacles (e.g., translating a verb in a target language without access to that verb in the source language).

Unfortunately, Equation 3 is no longer effective in modeling SimulMT for decoder-only LLMs as with the decoder of a classical Transformer. The reason is that Equation 3 is constructed specifically for the cross-attention calculation between keys exclusively from the source and queries exclusively from the target, as in Transformers. In contrast, LLMs perform self-attentions involving the prompt, the source, and the target concurrently. Equation 3 can no longer properly mask the source (keys) from the target (queries), due to the additional prompt and target sequences in the keys and the additional prompt and source sequences in the queries. Furthermore, Equation 3 does not enforce the autoregressive language modeling behavior of LLMs. As such, alternative means to model SimulMT have been proposed, leveraging prompting optimization.

3 Prompting Optimization Methods

Current methods of fine-tuning LLMs for SimulMT fall under prompting optimization. We define prompting optimization as either employing data augmentation to help with prompting (Koshkin et al., 2024; Wang et al., 2023; Agostinelli et al., 2023) or redefining the prompt structure (Wang et al., 2024; Koshkin et al., 2024) to somewhat simulate SimulMT.

3.1 Data Augmentation

Prompting optimization focusing on data augmentation resorts to subdividing each sentence in a dataset into multiple partial sentence pairs. These partial sentence pairs mimic SimulMT, as SimulMT produces outputs with a partial input. We label such a method as *prefix fine-tuning*, and although the high-level procedure is identical amongst current works, the algorithms employed to obtain these partial sentence pairs are unique. In the case of Agostinelli et al. (2023), each source-target sentence pair is subdivided according to the wait-k policy such that if we order the new samples from smallest to largest, each subsequent sentence pair will have one additional target word and source word so long as the end of the target or source is not reached. Upon completion there will be $\texttt{max}(|S| - (k-1), |T|)$ sentence pairs, where $|S|$ and $|T|$ are the original source and target sequence lengths. The approach requires the model to predict only the final target word in the sequence during fine-tuning.

Alternatively, Wang et al. (2023) randomly sampled a subset of sentence pairs from the dataset and truncated the source sentence to be 20% to 80% of the full length according to a uniform distribution. They obtained the respective target translations by prompting ChatGPT (gpt-3.5-turbo). The new truncated source-target sentence pairs were then added to the complete dataset, expanding it.

3.2 Prompt Restructuring

Prompting optimization that modifies the prompting structure adjusts the prompt to include the decision policy. In the case of Wang et al. (2024), a conversational prompting structure is adopted for the LLM, alternating between source and target subsequences of the original complete sequences using delimiting tokens to separate regions. For instance, if we have the source sequence $S = [s_1, s_2, ..., s_n]$ and the target sequence $T = [t_1, t_2, ..., t_m]$, then one potential conversational prompt expansion could be "<s>, [U], s_1, s_2, [A], t_1, t_2, </s>, ..., <s>, [U], s_n, [A], t_m, </s>", where the added <s>, </s>, [A], [U] are delimiting tokens. During fine-tuning, the choice of alternating subsequences is arrived at by attempting to maximize the relevant source context before each target sequence in the form of an oracle decision policy. For instance, the prompt will ensure an arbitrary target verb prediction only after the respective source verb is read. Some minor perturbations are added to the oracle decision policy to improve generalizability. Then, at inference, a prompt constructor provides the source sequence in fixed-size chunks.

Similarly, Koshkin et al. (2024) leverages prompt restructuring; however, it also employs prefix finetuning. Like the conversational prompting structure, it constructs a fine-tuning prompt by aligning words between the source and target sequence to mimic an oracle decision policy. However, it deviates from conversational prompting by ensuring the alignment using padding tokens in the target sequence. Then, the causally aligned sentence prompt is subdivided using a prefix fine-tuning strategy to expand the dataset with partially filled source-target sentence pairs. At inference, the LLM contains the decision policy outputting padding tokens whenever it requires more source context tokens.

4 Analysis and Shortcomings of Prompting Optimization Methods

Prompting optimization, while functional to a certain degree, is inherently deficient, possessing a host of fine-tuning and inference issues. These issues include a persistent fine-tuning-inference mismatch, consistent positional confusion in the target sequence, and high computational costs.

4.1 Fine-tuning/Inference Mismatch

A fine-tuning-inference mismatch is a mismatch between a LLM's fine-tuning and inference environments. For instance, fine-tuning a LLM for NMT where the entire sentence is available and deploying it for SimulMT where little of the sentence is available when beginning generation will create a massive inference time fine-tuning-inference mismatch. Furthermore, the LLM must be fine-tuned to accommodate KV caching, the process of caching the keys and values at inference to prevent recomputation. Overall, fine-tuning for SimulMT

aims to minimize the fine-tuning-inference mismatch.

Unfortunately, prefix fine-tuning precludes high-quality SimulMT with KV caching (Agostinelli et al., 2023; Koshkin et al., 2024; Wang et al., 2023) as with the continuously increasing prompt size, each key and value in the KV cache deviates more from the keys and values in its fine-tuning environment. For example, suppose we have a LLM in the middle of SimulMT using KV caching adhering to a wait-1 policy with the following prompting structure: "`Translate the following sentence:` $s_1, s_2, ..., s_{i+1}$ `[a]:` $t_1, t_2, ..., t_i$". Then, at the current write step, the query of t_i attends to the KV cache for `[a]:`$, t_1, t_2, ..., t_{i-1}$. By construction, each key and value in the KV cache was generated in a previous time step with a different subset of the source sequence $s_1, s_2, ..., s_i$. For instance, the keys and values for delimiting token `[a]:` when it predicted t_1 were conditioned only on s_1, whereas the keys and values for t_{i-1} when it predicted t_i were conditioned on $s_1, s_2, ..., s_i$. However, during prefix fine-tuning, the LLM was fine-tuned to predict t_{i+1} as if the KV cache for `[a]:`$, t_1, t_2, ..., t_{i-1}$ were each generated with the same subset of the source sequence $s_1, s_2, ..., s_i$. Such fine-tuning-inference mismatch is unsolved through conventional prompting structures.

Prompting restructuring also creates additional fine-tuning-inference mismatches. In Koshkin et al. (2024) and Wang et al. (2024), they all fine-tune for an oracle decision policy. However, at inference, such an oracle decision policy is not truly achievable, creating a mismatch. Furthermore, since the LLMs that leverage prompt restructuring encapsulate a specific oracle decision policy into their fine-tuning curriculum, extending them to alternative decision policies at inference is infeasible without incurring a mismatch. This calls for a new, flexible method adaptable to a range of decision policies while also eliminating the fine-tuning-inference mismatch.

4.2 Positional Confusion

Positional confusion describes the process whereby the relative and/or global positional information during SimulMT progressively becomes incorrect. Unfortunately, most SimulMT LLMs using KV caching suffer from this positional confusion (Agostinelli et al., 2023; Koshkin et al., 2024; Wang et al., 2023). The reason is that as the source sequence grows with SimulMT, the target sequence also shifts, necessitating the target sequence's positional information to follow suit. However, since KV caching is employed, the positional information held in the keys and values is not properly updated.

Aligning with our previous example, for the sequence portion "`[a]:` $t_1, t_2, ..., t_i$", after the first prediction step, the positional distance between s_1 and `[a]:` and between s_1 and t_1 would be 1 and 2, respectively. Then, after the second read, where the source sequence is s_1, s_2, the positional distance between s_1 and `[a]:` and t_1 would change to 2 and 3, respectively. However, while using KV caching, this positional distance would remain 1 and 2 in the keys and/or values for subsequent predictions, causing positional confusion. Continuing translation would see the increased gap between the true positional distance and the stale positional distance in the KV cache. Therefore, we need to identify an effective method to deal with positional confusion that is essential to prevent LLM hallucinations.

4.3 Computational Inefficiency

Avoiding KV caching and instead recomputing all the keys and values at each prediction step is the default solution for resolving the aforementioned fine-tuning-inference mismatch and positional confusion problems while employing prefix fine-tuning. Although effective from a translation quality standpoint, doing so incurs a large computational cost, an undesirable result for streaming tasks like SimulMT, where latency is equally important.

Outside of KV caching, the computational costs necessary for prefix fine-tuning methods are excessive. By subdividing each sample into multiple, the dataset drastically expands, contributing toward an increased cost to complete each epoch (Agostinelli et al., 2023; Koshkin et al., 2024; Wang et al., 2023). Such an increase causes the duration of each epoch to rise by upwards of a factor of 10 (as exemplified in Table 1 in Section 7.2). However, unlike normal methods of expanding a dataset through data augmentation, prefix fine-tuning does not actually add additional information. It is from this added computational burden that Agostinelli et al. (2023) and Wang et al. (2023) are forced to fine-tune with a subset of their entire prefix datasets.

Alternatively, methods of restructuring the prompt as in Koshkin et al. (2024) and Wang et al. (2024) have computational burdens of their own. For instance, Wang et al. (2024) requires adding

delimiting tokens in the prompt sequence, expanding the sequence length. Similarly, the requirement of padding tokens to induce a causal alignment between the source and target sequences, as in Koshkin et al. (2024), also expands the sequence length. Since the computational cost of the self-attention cost in the LLM scales quadratically with the sequence length, such a method is undesirable for both inference and fine-tuning.

Currently, no computationally efficient fine-tuning approach exists that enables computationally efficient inference. Identifying such a method is necessitated by the desire for low latency and high-quality translations and reducing the already high computational costs of fine-tuning LLMs.

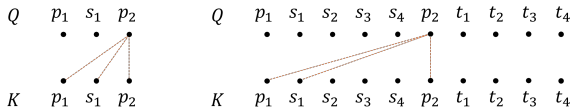
5 SimulMask: A Paradigm Shift

In this work, we propose *SimulMask*, which we believe could be a paradigm shift in fine-tuning LLMs for SimulMT that eschews current methods of prompting optimization. By restricting attention during fine-tuning, SimulMask efficiently solves the fine-tuning-inference mismatch and positional confusion problem. We demonstrate SimulMask through its application for the wait-k decision policy, but it should be noted that SimulMask broadly applies to various decision policies.

5.1 Inference Mirrored Attention

We first introduce the concept of *Inference Mirrored Attention* that cleverly models SimulMT during LLM fine-tuning. Under SimulMT, the latest translation token at each prediction step is conditioned only on the running source tokens. For conventional Transformers, specialized attention masks could achieve such conditioning; however, directly mapping these to LLMs is impossible since they fail to enforce autoregressive language modeling and cannot mask properly when the prompt, source, and target sequences are collectively included in the queries and keys. As such, prior works attempted to achieve such conditioning during fine-tuning using prompting optimization strategies littered with shortcomings. The proposed inference mirrored attention is aimed to model SimulMT with attention masks for LLMs by mirroring the attention during inference at fine-tuning according to the chosen decision policy.

As an example, suppose we model the attention for a wait-1 decision policy where the complete oracle input sequence is



(a) Inference. (b) Fine-tuning.

Figure 1: Inference Mirror Attention for matching attention during inference and fine-tuning for SimulMT.

"$p_1, s_1, s_2, s_3, s_4, p_2, t_1, t_2, t_3, t_4$". In the sequence, s_1, s_2, s_3, s_4 and t_1, t_2, t_3, t_4 are the 4-word source and target sequences and p_1 and p_2 are prompting regions. Then, at inference, by definition of the wait-1 policy, p_2 predicts t_1 while conditioned on the partial sequence p_1, s_1, p_2. As such, as shown in Figure 1a the query of p_2 attends to the keys of p_1, s_1, p_2. Thus, during fine-tuning, to eliminate the fine-tuning-inference mismatch, the query of p_2 should be limited to similarly attend to the keys of p_1, s_1, p_2 as shown in Figure 1b rather than the entire source sequence. For each successive prediction step, the previously predicted target word, t_i, predicts the next target word, t_{i+1} by conditioning on an extra source word, s_{i+1}, acquired from the previous read step. To mimic such behavior at fine-tuning, the query for t_i attends to identical keys as its inference step. The complete steps of this example are in Appendix A.2.

5.2 SimulMask

To achieve the above inference mirrored attention, we opt for an attention mask to restrict attention during fine-tuning to mimic an arbitrary decision policy during SimulMT. An attention mask is preferable to prompting optimization as it is flexible and directly extends the LLM causal attention mask. We call such an attention mask SimulMask.

As a demonstration, let us create a SimulMask for the wait-1 policy that extends our example from Section 5.1. As depicted in Figure 2, since the LLM is autoregressive, SimulMask begins with a causal attention mask from which attention is limited to be identical to the attention during SimulMT for the source sequence. Starting from the prompt p_2, from the example in Figure 1b, p_2 generates the first target token, t_1, conditioned on p_1, s_1, p_2. As such, SimulMask eliminates the attention between p_2 and s_2, s_3, s_4. Similarly, t_1 and t_2 are conditioned on p_1, s_1, s_2, p_2, t_1 and $p_1, s_1, s_2, s_3, p_2, t_1, t_2$, respectively. Thus, attention is eliminated between t_1 and s_3, s_4 and t_2 and s_4.

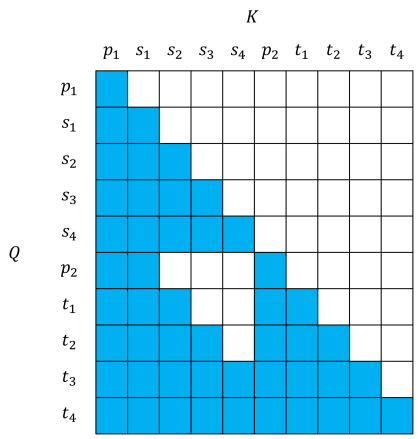


Figure 2: SimulMask for modeling SimulMT according to a wait-1 decision policy during fine-tuning.

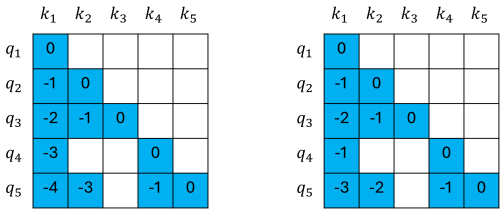
SimulMask is a flexible scheme that supports a range of decision policies. Since each decision policy performs read/write decisions differently and each limits attention differently, this requires a unique attention mask for every sentence. However, this can be done straightforwardly. The general procedure to construct a SimulMask for a given policy and sentence consists of the following steps:

1. Construct a causal attention mask using Equation 2 as a starting point for SimulMask.
2. Starting from the intersection between the query that predicts the first target token and the first source key, apply the sub-attention mask expressed in Equation 3. The sub-attention mask prevents the target queries from attending to source keys following the arbitrary decision policy.
3. Mask any non-source queries before the query predicting the first target token from attending to the source keys not included in the first read decision. Such a step is necessary to prevent the hidden states associated with these queries from holding information of the entire source sequence at later layers in the LLM.

As reported in Section 7.2, the computation for constructing an arbitrary SimulMask is minor, and since SimulMask is not applied during inference, it does not impact computational cost at deployment. Therefore, SimulMask is an efficient option for mimicking SimulMT during fine-tuning and providing low-latency translations at inference.

5.3 Positional Reordering

Since positional confusion during inference is a byproduct of retaining outdated positional information in either the keys or values, bypassing it requires providing a form of positional information without injecting it directly into the sequence or KV cache. One positioning method that satisfies such a constraint is the popular ALiBi, which supplies positional information through biases in attention (Press et al., 2021). The bias is applied to each query-key dot product row in the attention calculation as shown in Equation 4, where m is a head-specific constant.



(a) Original ALiBi. (b) Modified ALiBi.

Figure 3: ALiBi biases with SimulMask.

$$\boldsymbol{q}_i \boldsymbol{K}^T + \boldsymbol{M}_i + m \cdot [-(i-1), ..., -1, 0] \quad (4)$$

Though simple, ALiBi has demonstrated an ability to extrapolate to much larger sequence lengths than other state-of-the-art positional encodings, making it desirable for LLMs like Falcon (Penedo et al., 2023), BLOOM (Le Scao et al., 2023), etc.

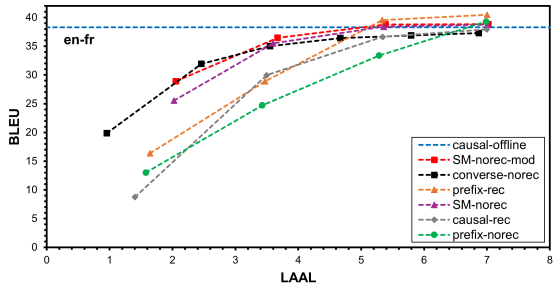
Unfortunately, ALiBi, by default, does not mesh with SimulMask as SimulMask removes attention between the target queries and source keys. This removed attention creates a gap in ALiBi biases during fine-tuning that are not present at inference. An example of such a gap is provided in Figure 3a, where both q_4 and q_5 have gaps in the position distance.

To eliminate the bias gap, we modify ALiBi by reducing the bias values of all query rows influenced by SimulMask. For each query row, the reduction in bias values is equivalent to the amount of attention removed along the row using SimulMask. Figure 3b provides an example of such a modification. In the case of q_4, it is no longer able to attend to k_2 and k_3; therefore, the bias on the right of the gap is reduced by 2. Together with the modified ALiBi, SimulMask eliminates positional confusion from the LLM during SimulMT.

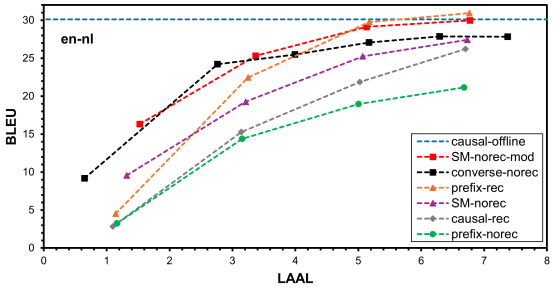
6 Experimental Setup

6.1 Fine-tuning

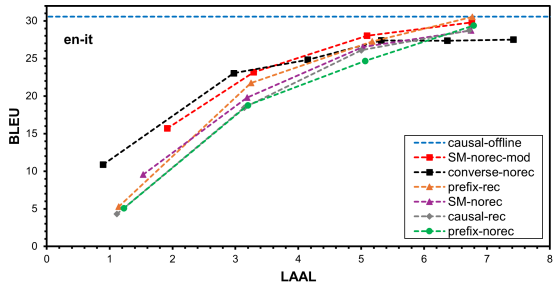
Our fine-tuning was conducted with the Simul-LLM framework (Agostinelli et al., 2023), which



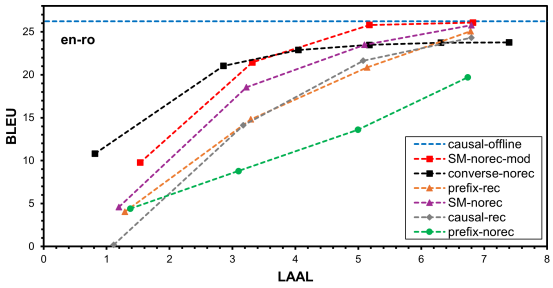
(a) English-French language pair results.



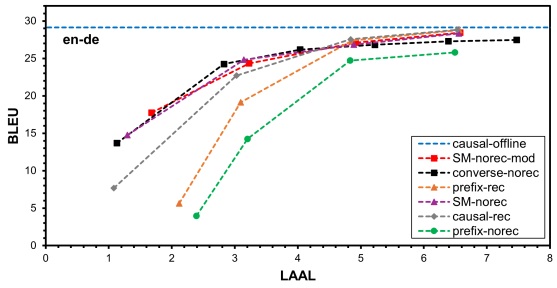
(b) English-Dutch language pair results.



(c) English-Italian language pair results.



(d) English-Romanian language pair results.



(e) English-German language pair results.

Figure 4: Translation quality plotted against latency for LLMs on the English-French, English-Dutch, English-Romanian, and English-German language pairs.

contains our publicly available code for SimulMask[1]. Each experiment used a 1.3 billion parameter Falcon model pre-trained on the RefinedWeb dataset (Penedo et al., 2023). We compared 7 schemes:

- **causal-offline**: Fine-tuned with a causal attention mask and evaluated for NMT (non-SimulMT).
- **causal-rec**: Fined-tuned with a causal attention mask and evaluated with recomputing the KV cache.
- **prefix-rec** and **prefix-norec**: Fined-tuned with prefix fine-tuning and evaluated with/without recomputing the KV cache.
- **converse-norec**: Fined-tuned with a conversational prompting strategy and evaluated without recomputing the KV cache.
- **SM-norec-mod** and **SM-norec**: Fined-tuned with SimulMask with/without modifying ALiBi and evaluated without recomputing the KV cache.

Appendix A.3 provides all model hyperparameters. Our fine-tuning experiments included the English-French (en-fr), English-Italian (en-it), English-Dutch (en-nl), English-Romanian (en-ro), and English-German (en-de) language pairs of the IWSLT 2017 dataset (Cettolo et al., 2017).

6.2 Evaluation

We evaluated translation quality and latency for SimulMT using Simul-LLM inference agents (Agostinelli et al., 2023) interfacing with the SimulEval toolkit (Ma et al., 2020a). The translation quality was determined using detokenized BLEU with SacreBLEU (Post, 2018) and chrF++ (Popović, 2017). Latency was determined using Length-Adaptive Average Lagging (LAAL) (Papi

[1] https://github.com/OSU-STARLAB/Simul-LLM

et al., 2022b). The computational cost of SimulMT was recorded with GFLOPs. All metrics were obtained on a single A10 GPU with bfloat16 precision. The models fine-tuned for the wait-k policy were evaluated at a wait-k four lower, for which they were fine-tuned, as suggested by (Ma et al., 2019).

7 Results

7.1 Translation Quality and Latency Results

In this section, we demonstrate the efficacy of fine-tuning with the proposed SimulMask compared with other schemes using BLEU scores and LAAL. All wait-k model evaluations are performed across wait-{1,3,5,7}, and the *converse-norec* is evaluated for a chunk size of 1, 3, 5, 7, 9, and 11. Figure 4 provides the BLEU translation quality and latency results on the English-French, English-Dutch, English-Italian, English-Romanian, and English-German language pairs. We provide the numerical BLEU and chrF++ translation quality results in Tables 3 and 4 of Appendix A.4.

Overall, throughout Figure 4, the proposed *SM-norec-mod* outperforms or matches the translation quality of *causal-rec*, *prefix-rec*, and *converse-norec* across all latencies. The only major exception occurs at wait-1, where *converse-norec* outperforms *SM-norec-mod* on the English-Romanian language pair. This overall excellent performance in terms of translation quality underscores the importance of the proposed method.

Furthermore, Figure 4 provides two ablation studies. The first ablation demonstrates the importance of modifying ALiBi with SimulMask for high-quality translations by comparing *SM-norec-mod* with *SM-norec*. For each wait-k value and language pair, *SM-norec-mod* outperforms *SM-norec*. Unsurprisingly, at higher wait-k values where the setting approaches NMT, the difference in BLEU scores becomes less pronounced between the models.

A secondary ablation is provided in Figure 4 by comparing *prefix-rec* and *prefix-norec*. Doing so demonstrates that translation quality increases by recomputing the KV cache across all wait-k values. Similarly, as with the previous ablation, the difference in the BLEU score becomes less pronounced for the higher wait-k values.

An interesting observation is that models evaluated at lower wait-k values have their LAAL deviate from their respective k to a greater degree than those evaluated at higher wait-k. Such an increase is a byproduct of the lower wait-k models generating longer predictions than their corresponding references. The increased generation length is a byproduct of the model hallucinating on sequences provided insufficient contexts. These hallucinations are most noticeable with *prefix-rec* and *prefix-norec* in Figure 4e.

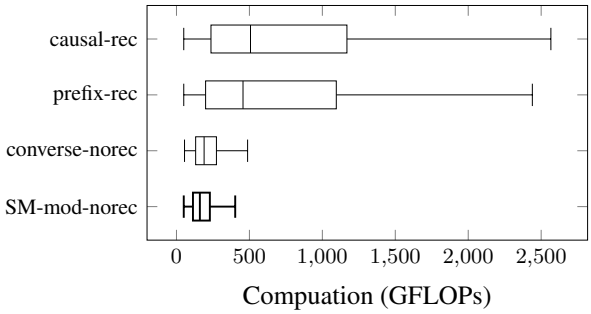


Figure 5: Box plots of the computational cost of each method in GFLOPs during inference.

7.2 Compuational Saving Results

Fine-tuning LLMs with SimulMask also features reduced training time compared with LLMs leveraging prefix fine-tuning or conversational prompting. For instance, this is reflected in the fine-tuning times for one epoch on an H100 GPU on the English-French dataset of the IWSLT 2017 dataset, as reported in Table 1 (Cettolo et al., 2017).

Fine-tuning Approach	Time (s)
Prefix Fine-tuning	9953
Conversational Prompting	1274
SimulMask	**1014**
Causal Mask	727

Table 1: Time to complete one epoch for different fine-tuning approaches on an H100.

Furthermore, we find that *SM-norec* is also more computationally efficient at inference than *prefix-rec* and *converse-norec*. We report these results in GFLOPs that are needed to complete a sentence translation in Figure 5. The data used to obtain the results was a random 1000 samples from the English-French split of the IWSLT 2017 test set (Cettolo et al., 2017). The models chosen either used wait-3 or a chunk size of 5.

By leveraging SimulMask during fine-tuning, we eschew the need to recompute the KV cache at inference. In doing so, SimulMask saves computation compared to *prefix-rec* and *causal-rec*.

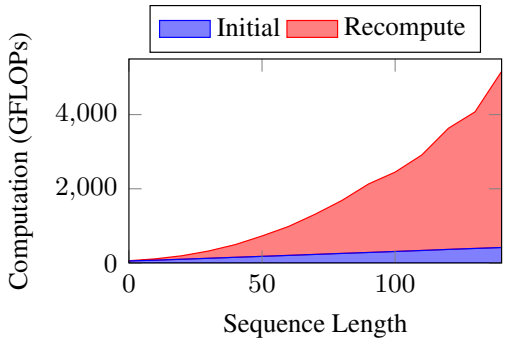


Figure 6: Separated computational cost in GFLOPs between initial (or required) computational cost and the cost of recomputing already emitted target words in a provided prompt during translation versus the sequence length of a given sample.

We demonstrate the proportions of computation in GFLOPs dedicated to re-computing the KV cache and processing/predicting initial tokens in Figure 6 (based on *prefix-rec*). The sequence length is the number of tokens in the predicted target and input source. As can be seen, it is critical to avoid re-computing KV cache, as achieved by SimulMask, to provide low latency translations, especially at longer sequence lengths.

8 Conclusion

In this work, we first examine current LLM fine-tuning approaches for SimulMT and identify their shortcomings. We then propose a new paradigm for fine-tuning LLMs for SimulMT that we call SimulMask, which avoids the shortcomings of previous methods. When employing SimulMask, the target sequence is prevented from attending to a portion of the source sequence according to an arbitrary decision policy modeling SimulMT. Through the application of SimulMask, we can efficiently fine-tune a LLM for SimulMT and reduce the computational costs of inference by eliminating the recomputation of the KV cache for the target sequence, unlike prior works. Furthermore, we can exceed or match the translation quality of prior works at all wait-k values across multiple language pairs.

Limitations

Given the translation quality benefits at a reduced computational cost of fine-tuning with SimulMask, it would be beneficial to evaluate the approach to larger and more powerful LLMs, adapting them for SimulMT. Also, while SimulMask is broadly applicable to various decision policies, our current evaluation was limited to only testing the effectiveness of SimulMask on the wait-k policy and did not evaluate alternative fixed or more flexible decision policies. Additionally, we did not explore simultaneous speech-to-text or speech-to-speech translation, which SimulMask has yet to be tested on.

Acknowledgments

This research was supported, in part, by the National Science Foundation grants 2223483 and 2223484.

References

Victor Agostinelli, Max Wild, Matthew Raffel, Kazi Asif Fuad, and Lizhong Chen. 2023. Simul-llm: A framework for exploring high-quality simultaneous translation with large language models. *arXiv preprint arXiv:2312.04691*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Jan Niehues, Sebastian Stüker, Katsuhito Sudoh, Koichiro Yoshino, and Christian Federmann. 2017. Overview of the IWSLT 2017 evaluation campaign. In *Proceedings of the 14th International Conference on Spoken Language Translation*, pages 2–14, Tokyo, Japan. International Workshop on Spoken Language Translation.

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

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

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

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. 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, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Shoutao Guo, Shaolei Zhang, Zhengrui Ma, Min Zhang, and Yang Feng. 2024. Sillm: Large language models for simultaneous machine translation. *arXiv preprint arXiv:2402.13036*.

Vivek Iyer, Pinzhen Chen, and Alexandra Birch. 2023. Towards effective disambiguation for machine translation with large language models. In *Proceedings of the Eighth Conference on Machine Translation*, pages 482–495, Singapore. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Roman Koshkin, Katsuhito Sudoh, and Satoshi Nakamura. 2024. Transllama: Llm-based simultaneous translation system. *arXiv preprint arXiv:2402.04636*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

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. 2019. 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*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020a. SIMULEVAL: An evaluation toolkit for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 144–150, Online. Association for Computational Linguistics.

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

Yasmin Moslem, Rejwanul Haque, John D. Kelleher, and Andy Way. 2023. Adaptive machine translation with large language models. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 227–237, Tampere, Finland. European Association for Machine Translation.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022a. Does simultaneous speech translation need simultaneous models? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 141–153, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022b. Over-generation cannot be rewarded: Length-adaptive average lagging for simultaneous speech translation. In *Proceedings of the Third Workshop on Automatic Simultaneous Translation*, pages 12–17, Online. Association for Computational Linguistics.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

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

Ofir Press, Noah A Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Vilar, Markus Freitag, Colin Cherry, Jiaming Luo, Viresh Ratnakar, and George Foster. 2023. Prompting PaLM for translation: Assessing strategies and performance. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15406–15427, Toronto, Canada. Association for Computational Linguistics.

Minghan Wang, Thuy-Trang Vu, Ehsan Shareghi, and Gholamreza Haffari. 2024. Conversational simulmt: Efficient simultaneous translation with large language models. *arXiv preprint arXiv:2402.10552*.

Minghan Wang, Jinming Zhao, Thuy-Trang Vu, Fatemeh Shiri, Ehsan Shareghi, and Gholamreza Haffari. 2023. Simultaneous machine translation with large language models. *arXiv preprint arXiv:2309.06706*.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. 2023. A paradigm shift in machine translation: Boosting translation performance of large language models. *arXiv preprint arXiv:2309.11674*.

Xuan Zhang, Navid Rajabi, Kevin Duh, and Philipp Koehn. 2023. Machine translation with large language models: Prompting, few-shot learning, and fine-tuning with QLoRA. In *Proceedings of the Eighth Conference on Machine Translation*, pages 468–481, Singapore. Association for Computational Linguistics.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simpler and faster learning of adaptive policies for simultaneous translation. *arXiv preprint arXiv:1909.01559*.

A Appendix

A.1 Encoder Attention Mask

An example of an encoder attention mask used to model simultaneous translation during training is provided in Figure 7. The attention mask in Figure 7 is designed for a source sequence length of 5 tokens where the first read step reads 2 tokens, the second read step reads 1 token, and the third read step reads 2 tokens.

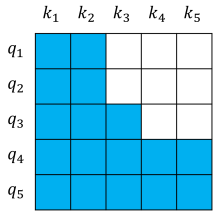


Figure 7: An attention mask to model simultaneous translation for a Transformer encoder during training.

A.2 Inference Mirrored Attention and SimulMask

Figure 8 provides the complete example of inference mirrored attention for the wait-1 policy explained in Section 5.1. To reiterate in Figure 8a, the query of p_2 attends to the keys of p_1, s_1, p_2. Thus, during fine-tuning, to eliminate the fine-tuning-inference mismatch, the query of p_2 is limited to similarly attend to the keys of p_1, s_1, p_2 as shown in Figure 8b rather than the entire source sequence. Then for the second and third decoding steps at inference, the queries of t_1 and t_2 attend to the keys of p_1, s_1, s_2, p_2 and p_1, s_1, s_2, s_3, p_2, respectively as shown in Figures 8c and 8e. Once again, to eliminate the fine-tuning-inference mismatch, the queries of t_1 and t_2 must attend to an identical set of keys as shown in Figures 8d and 8f.

A.3 Hyperparameters

The fine-tuning hyperparameters used for *SM-norec, causal-rec, causal-offline, prefix-rec, prefix-norec,* and *converse-norec* models are provided in Table 2.

The prompts used for the *SM-norec, causal-rec, causal-offline, prefix-rec,* and *prefix-norec* models consisted of the following format:

```
Translate the following sentence from
[SRC] to [TGT]: [SRC-Sentence]\n
Assistant: [TGT-Sentence]
```

Alternatively, the *converse-norec* model used the prompt:

```
Translate the following sentence from
[SRC] to [TGT]\nAssistant: <s><t>
[SRC-Chunk-1]</t>[TGT-Chunk-1]
</s><s><t>...<s><t>
[SRC-Chunk-n]</t>[TGT-Chunk-n]</s>
```

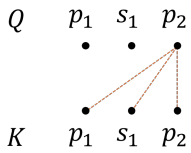
Our implementation for *converse-norec* followed Wang et al. (2024). However, we used the Itermax method from the SimAlign toolkit leveraging XLM-RoBERTa base (Conneau et al., 2019) to align words due to their work reporting better alignments than fast-align (Jalili Sabet et al., 2020; Dyer et al., 2013).

A.4 Extended Translation Results

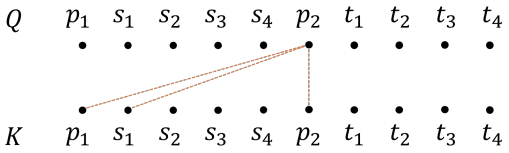
The translation quality results in Table 3 provide the numerical BLEU and LAAL values from Figure 4 for *prefix-rec*, *converse-norec*, and *SM-norec-mod*. Alternatively, Table 4 provide the numerical chrF++ values associated with each of these models from Figure 4.

A.5 Sequence Lengths

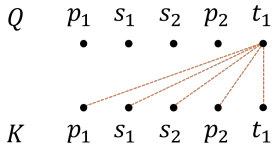
Figure 9 reports the number of occurrences on the English-French IWSLT2017 validation set (Cettolo et al., 2017) that the combined length of the source sequence and the predicted target sequence are within a specified range for prefix-rec at wait-3.



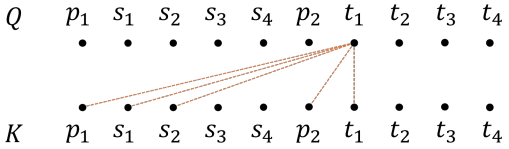
(a) Attention for the first prediction step.



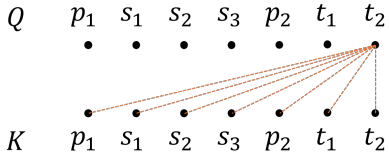
(b) Inference mirrored attention for the first prediction step.



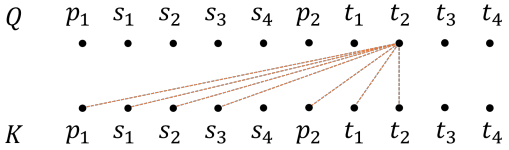
(c) Attention for the second prediction step.



(d) Inference mirrored attention for the second prediction step.



(e) Attention for the third prediction step.



(f) Inference mirrored attention for the third prediction step.

Figure 8: Attention during inference and finetuning for SimulMT.

Hyperparameter	Group 1	Group 2	Group 3	Group 4
Weight Precision	bfloat16	bfloat16	bfloat16	bfloat16
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning Rate	$2 \cdot 10^{-4}$	$2 \cdot 10^{-4}$	$2 \cdot 10^{-4}$	$2 \cdot 10^{-4}$
LR Scheduler	Inverse Sqrt	Inverse Sqrt	Inverse Sqrt	Inverse Sqrt
Weight Decay	0.1	0.1	0.1	0.1
Warmup Ratio	0.03	0.03	0.03	0.03
Max Gradient Norm	1	1	1	1
Max Sequence Length	512	512	512	512
Wait-k	5,7,9,11	5,7,9,11	-	-
Epochs	2	1	1	2
Batch size	64	1024	64	64
Attention heads	32	32	32	32
Layers	24	24	24	24
Hidden Size	2048	2048	2048	2048
Positional Encoding	Modified ALiBi	ALiBi	ALiBi	ALiBi
Attention Mask	SimulMask	Causal Mask	Causal Mask	Causal Mask
$\delta_{\max}$	-	-	10	-
β	-	-	0.5	-
$\rho_{\min}$	-	-	0.5	-
$\rho_{\max}$	-	-	0.9	-

Table 2: Fine-tuning hyperparameters for all models in Section 7. **Group 1:** *SM-norec, SM-norec-mod.* **Group 2:** *prefix-rec, prefix-norec.* **Group 3:** converse-norec. **Group 4:** *causal-rec, causal-offline.*

A.6 Licensing Information

The SimulEval toolkit is licensed under CC BY-SA-4.0 license (Ma et al., 2020a). The Simul-LLM framework and SimAlign toolkit are licensed under the MIT license (Agostinelli et al., 2023; Jalili Sabet et al., 2020). The IWSLT 2017 dataset is licensed under CC BY-NC-ND (Cettolo et al., 2017). The Falcon model we used from Hugging Face

Model	en-fr	en-nl	en-it	en-ro	en-de
SM-norec-mod (wait-1)	28.89 (2.05)	16.37 (1.53)	15.72 (1.92)	9.77 (1.53)	17.73 (1.68)
SM-norec-mod (wait-3)	36.48 (3.67)	25.31 (3.37)	23.17 (3.29)	21.44 (3.32)	24.34 (3.23)
SM-norec-mod (wait-5)	38.77 (5.39)	29.13 (5.13)	28.01 (5.09)	25.79 (5.18)	27.11 (4.92)
SM-norec-mod (wait-7)	38.85 (7.02)	29.98 (6.78)	29.81 (6.76)	26.07 (6.82)	28.43 (6.58)
prefix-rec (wait-1)	16.37 (1.64)	4.53 (1.14)	5.27 (1.12)	4.03 (1.29)	5.64 (2.12)
prefix-rec (wait-3)	28.90 (3.47)	22.44 (3.25)	21.74 (3.15)	14.82 (3.29)	19.13 (3.09)
prefix-rec (wait-5)	39.55 (5.33)	29.76 (5.17)	27.26 (5.03)	20.85 (5.14)	27.38 (4.86)
prefix-rec (wait-7)	40.48 (7.00)	30.95 (6.77)	30.52 (6.75)	25.06 (6.78)	28.79 (6.56)
converse-norec (chunk-1)	19.89 (0.95)	9.17 (0.65)	10.86 (0.89)	10.80 (0.82)	13.67 (1.13)
converse-norec (chunk-3)	31.94 (2.46)	24.22 (2.76)	23.04 (2.98)	21.02 (2.86)	24.26 (2.83)
converse-norec (chunk-5)	35.03 (3.55)	25.49 (3.99)	24.82 (4.15)	22.89 (4.05)	26.17 (4.04)
converse-norec (chunk-7)	36.42 (4.67)	27.09 (5.17)	27.40 (5.32)	23.48 (5.18)	26.82 (5.22)
converse-norec (chunk-9)	36.87 (5.79)	27.86 (6.29)	27.37 (6.37)	23.73 (6.31)	27.28 (6.39)
converse-norec (chunk-11)	37.27 (6.87)	27.84 (7.37)	27.52 (7.42)	23.76 (7.40)	27.47 (7.47)

Table 3: Translation quality and latency results in BLEU and LAAL.

Model	en-fr	en-nl	en-it	en-ro	en-de
SM-norec-mod (wait-1)	50.76 (2.05)	36.14 (1.53)	36.37 (1.92)	26.36 (1.53)	42.20 (1.68)
SM-norec-mod (wait-3)	58.42 (3.67)	47.60 (3.37)	45.17 (3.29)	43.41 (3.32)	50.64 (3.23)
SM-norec-mod (wait-5)	60.40 (5.39)	52.34 (5.13)	51.34 (5.09)	49.78 (5.18)	53.09 (4.92)
SM-norec-mod (wait-7)	60.59 (7.02)	53.74 (6.78)	53.30 (6.76)	50.18 (6.82)	53.73 (6.58)
prefix-rec (wait-1)	33.72 (1.64)	15.50 (1.14)	18.72 (1.12)	13.62 (1.18)	25.71 (2.16)
prefix-rec (wait-3)	49.64 (3.47)	44.26 (3.25)	42.63 (3.15)	28.24 (3.173)	40.15 (3.10)
prefix-rec (wait-5)	61.05 (5.33)	53.50 (5.17)	49.91 (5.03)	43.11 (5.04)	52.36 (4.86)
prefix-rec (wait-7)	61.91 (7.00)	55.07 (6.77)	54.28 (6.75)	49.09 (6.77)	53.46 (6.53)
converse-norec (chunk-1)	47.54 (0.95)	28.01 (0.65)	33.19 (0.89)	30.67 (0.74)	46.12 (1.24)
converse-norec (chunk-3)	56.45 (2.46)	47.13 (2.76)	46.93 (2.98)	44.04 (2.74)	52.35 (2.91)
converse-norec (chunk-5)	58.27 (3.55)	49.15 (3.99)	48.98 (4.15)	46.49 (3.96)	53.05 (4.09)
converse-norec (chunk-7)	58.98 (4.67)	50.69 (5.17)	51.37 (5.32)	47.97 (5.08)	53.11 (5.26)
converse-norec (chunk-9)	59.33 (5.79)	51.52 (6.29)	51.49 (6.37)	48.56 (6.22)	53.14 (6.38)
converse-norec (chunk-11)	59.47 (6.87)	52.11 (7.37)	51.47 (7.42)	48.85 (7.31)	52.97 (7.43)

Table 4: Translation quality and latency results in chrF++ and LAAL.[2]

(tiiuae/falcon-rw-1b) is licensed under Apache 2.0 (Penedo et al., 2023).

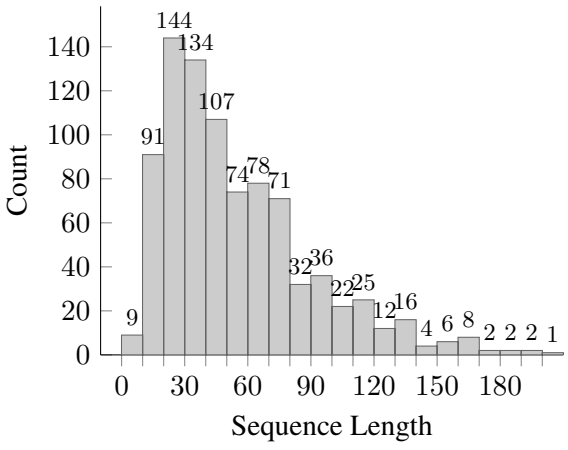


Figure 9: Histogram of the distribution of sequence lengths.

[2]The chrF++ results for *prefix-rec* and *converse-norec* on the English-German and English-Romanian language pairs were obtained with different fine-tuning and evaluation seeds than the models reported in Figure 4.