TRANSLLAMA: LLM-based Simultaneous Translation System

Roman Koshkin[†‡] **Katsuhito Sudoh**[†] **Satoshi Nakamura**[†]
[†]Nara Institute of Science and Technology, Japan
[‡]Okinawa Institute of Science and Tenchnology, Japan
roman.koshkin@oist.jp

Abstract

Decoder-only large language models (LLMs) have recently demonstrated impressive capabilities in text generation and reasoning. Nonetheless, they have limited applications in simultaneous machine translation (SiMT), currently dominated by encoder-decoder transformers. This study demonstrates that, after fine-tuning on a small dataset comprising causally aligned source and target sentence pairs, a pre-trained open-source LLM can control input segmentation directly by generating a special "wait" token. This obviates the need for a separate policy and enables the LLM to perform English-German and English-Russian SiMT tasks with BLEU scores that are comparable to those of specific state-of-the-art baselines. We also evaluated closed-source models such as GPT-4, which displayed encouraging results in performing the SiMT task without prior training (zero-shot), indicating a promising avenue for enhancing future SiMT systems. The code is available at https://github.com/RomanKoshkin/transllama.

1 Introduction

Unlike conventional sequential translation, in which the target text is produced after the end of the corresponding source sentence (or long phrase), in simultaneous machine translation (SiMT) the target text is produced with minimal delay, aiming for the best listener experience expected from professional conference interpreters. While recent years have seen tremendous progress in sentence-based machine translation, mainstream adoption of SiMT systems requires solving a range of technical problems. Perhaps the most important of them is that, much like human conference interpreters, SiMT systems must make optimal decisions about *when* (rather than *how*) to translate. In particular, naively translating each source word immediately results in compromised target quality, given that the meaning of a source word often makes sense only in the context of later words. And while waiting until the end of a sentence might seem a viable solution, in practice it would introduce unacceptable delays between the source and target message. Consequently, the development of an effective SiMT system necessitates striking a balance between these two opposite scenarios.

Existing approaches to maintaining an optimal quality-latency tradeoff in SiMT, conventionally called *policies*, fall into two broad categories: fixed and adaptive. The policy's role is to signal to a separately trained translation model *when* to produce a partial translation (aka WRITE action (Gu et al., 2017)) based of the partial input; at other times the input, which represents either text chunks from an upstream ASR system (in cascade SiMT systems) or speech embeddings (in end-to-end systems), is just read in (READ action). While with a fixed policy (Dalvi et al., 2018; Ma et al., 2019; Elbayad et al., 2020; Zhang and Feng, 2021), the decision to output translation is based on a simple heuristic, an adaptive policy (Arivazhagan et al., 2019; Ma et al., 2020b; Zhang and Feng, 2022) can be implemented as a separately trained model, for example an agent trained with reinforcement learning (RL) (Gu et al., 2017; Satija and Pineau, 2016).

To the best of our knowledge, state-of-the-art SiMT systems use encoder-decoder transformer architectures in a sequence-to-sequence paradigm. However, as of writing this paper the largest – and generally most expressive – language models are causal decoder-only architectures. We wanted to explore the utility of such models for SiMT tasks, focusing on the English-German and English-Russian language pairs, and specifically if they can be harnessed with minimal engineering effort.

Inspired by the recent success of LLMs in translation (Xu et al., 2024), as well as by their agential capabilities (Nascimento et al., 2023; Wang et al., 2024, 2023c) – here we propose TRANSLLAMA, a policy-free SiMT system, in which an off-the-shelf

Findings of the Association for Computational Linguistics: EMNLP 2024, pages 461–476
November 12-16, 2024 ©2024 Association for Computational Linguistics

pre-trained decoder-only LLM is fine-tuned on a dataset of causally aligned source and target sentences. The causality of the source is guaranteed by inserting one or more `<WAIT>` tokens into the target sentence to ensure that target content words never appear earlier than their closest equivalents in the source. We call our model policy-free, because as a result of fine-tuning on a causally aligned dataset the LLM becomes capable of deciding when to output translation and when to read in more of the source, without requiring a separate policy. At inference, the fine-tuned LLM is prompted with *part* of a source sentence concatenated with its corresponding (partial) translation and outputs one or more target tokens until either a full new word or a `<WAIT>` token is generated, which signals for more words to be read in. When extended with an off-the-shelf ASR model, in addition to text-to-text translation (T2TT), our system handles speech-to-speech translation (S2TT) tasks with quality (as measured by BLEU score (Papineni et al., 2002)) approaching that of some of the recently published baselines at comparable latencies.

Our main contributions are as follows:

1. We propose a way to fine-tune a pre-trained LLM for the SiMT task with direct supervision on a dataset of *causally aligned* source-target sentence pairs;

2. We demonstrate that an LLM can perform both simultaneous translation and input segmentation *without a separate policy*, with performance approaching or exceeding state of the art.

The rest of the paper is structured as follows. Section 2 offers a brief overview of most recent SiMT literature. In Section 3 we detail our system's architecture, fine-tuning data preparation and training procedure. In Section 4 we showcase its performance on `en-de` and `en-ru` language directions. We discuss directions for future work in Section 5 and limitations in Section 6.

2 Related Work

SiMT systems aim to deliver the best translation quality, usually measured with BLEU score (Papineni et al., 2002), while keeping its latency at an acceptable level. This quality-latency trade-off is controlled by the "policy", which decides *when* to translate (i.e. perform a WRITE action) and when to receive more input (i.e. perform a READ action). The various policies described in the literature can be broadly categorized into fixed and adaptive (Zhang et al., 2020). Fixed policies (e.g, *wait-k* (Ma et al., 2019)) are simple rules that determine the timing and order of WRITE and READ actions irrespective of the context. Early SiMT systems used *chunk-based* approaches (Fügen et al., 2007; Bangalore et al., 2012; Yarmohammadi et al., 2013; Sridhar et al., 2013), in which the input is split into sub-sentence phrases and translated independently of the previous chunk's context, which compromised translation quality. Attempting to overcome this limitation, Dalvi et al. (2018) proposed an *incremental decoding* approach, in which chunk translations incorporate previous context encapsulated by an RNN's hidden states. They showed that coupled with a simple segmentation strategy, their approach outperformed existing state of the art. On the other hand, adaptive policies (e.g. *wait-if* rules (Cho and Esipova, 2016)) make READ/WRITE actions more flexibly by taking account of the partial source and/or target. Adaptive policies can be implemented as separately trained agents (e.g. with reinforcement learning) (Grissom II et al., 2014; Gu et al., 2017; Satija and Pineau, 2016; Alinejad et al., 2018). In such policies, READ/WRITE actions can be taken based on attention (Raffel et al., 2017; Chiu and Raffel, 2018; Arivazhagan et al., 2019; Ma et al., 2020b), or stability of the model's outputs over n steps (so-called *local agreement* (Liu et al., 2020a; Ko et al., 2023; Polák et al., 2022)). More recent studies have also explored training the policy with binary search Guo et al. (2023) aiming to maximize the gain in translation quality per each token read, or cast the problem of deciding when to translate as a hidden Markov transformer (Zhang and Feng, 2023), in which hidden events correspond to the times at which to output translation.

Another promising line of work, related to the present study, aims to fine-tune encoder-decoder transformers, such as mBART (Liu et al., 2020b), originally pre-trained for sentence-level translation, for the SiMT task. For example, Fukuda et al. (2023); Kano et al. (2022) utilized fine-tuning on prefix-alignment data and Zhang et al. (2020) on meaningful units, achieving compelling performance on some language pairs.

Finally, in the course of writing this paper, we became aware of two concurrent projects which explored the use of fine-tuned LLMs for SiMT in

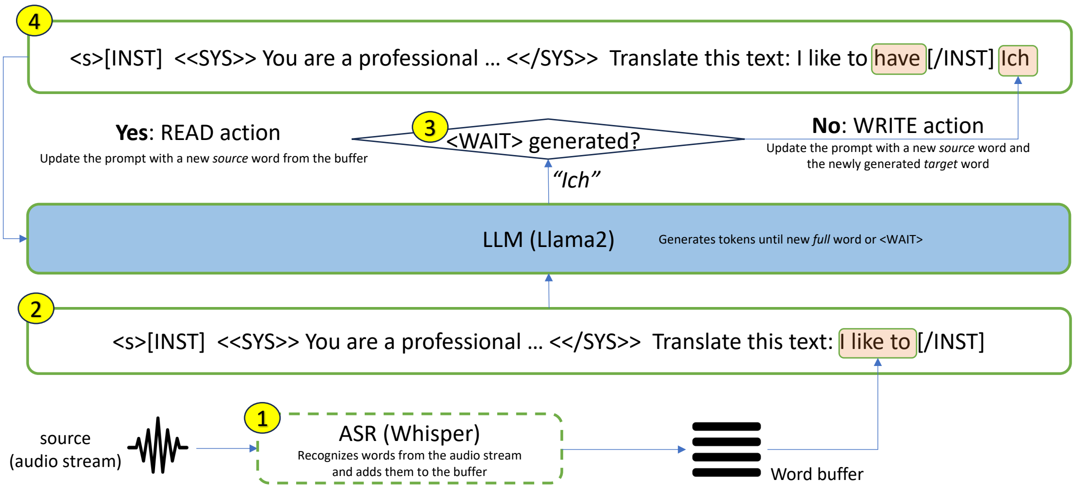

4
<s>[INST] <<SYS>> You are a professional ... <</SYS>> Translate this text: I like to have [/INST] Ich
Yes: READ action
Update the prompt with a new source word from the buffer
3
<WAIT> generated?
No: WRITE action
Update the prompt with a new source word and the newly generated target word
"Ich"
LLM (Llama2)
Generates tokens until new full word or <WAIT>
2
<s>[INST] <<SYS>> You are a professional ... <</SYS>> Translate this text: I like to [/INST]
source (audio stream)
1
ASR (Whisper)
Recognizes words from the audio stream and adds them to the buffer
Word buffer

Figure 1: Model overview. The source audio stream is processed with an ASR model (1), which saves each recognized word into the buffer. The initial prompt (2) is built with k source words ($k = 3$ in this example). When the buffer has 3 words, the initial prompt is fed into the LLM, which generates output tokens until either a <WAIT> token or a full word is generated ("Ich" in this example) (3). Then the prompt is updated with a new input ("have") and target ("Ich") word (WRITE action). Finally, the updated prompt (4) is fed back into the LLM. If <WAIT> is generated, the prompt is only updated with a new source word from the buffer (READ action).

conjunction with a modified local agreement (Wang et al., 2023a), and vanilla wait-k (Agostinelli et al., 2024) policies.

Distinct from previous work, we propose a *policy-free* approach, in which an LLM is fine-tuned for the SiMT task on *causally aligned* full source-target sentence pairs, which we describe below.

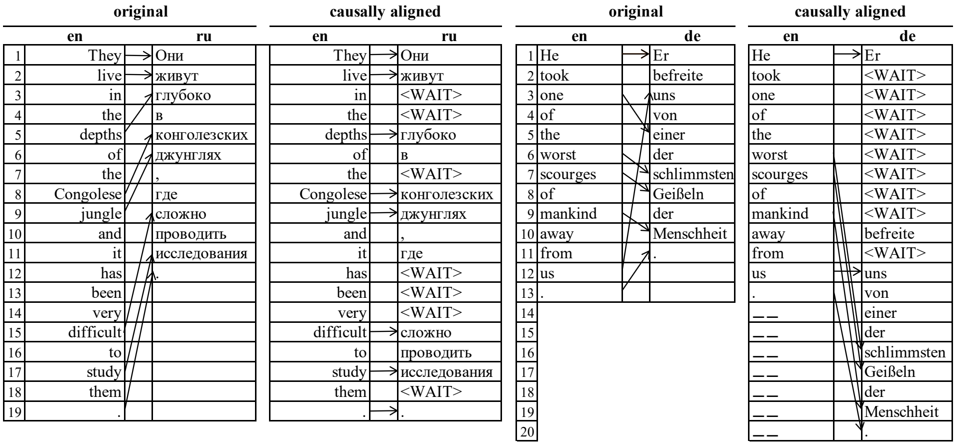
3 Method

Although the LLMs we consider in this paper are designed to process only text input, we add an ASR stage to enable it to also perform S2TT. Thus, we follow a cascaded approach shown in Fig. 1.

Causal alignment. Training SiMT models, including optimal segmentation policies, with direct supervision has remained a challenge (Guo et al., 2023) due to at least three reasons: (1) word order inconsistencies between the source and target, (2) omissions of words from the target that were present in the source, and/or (3) additions of words to the target not explicitly present in the source, making it difficult to establish unambiguous correspondences between each source and target words. This is less of a problem for offline translation models, because they are trained with direct supervision on pairs of *complete* source and target sentences, and both during training and inference the entire source context is revealed. However, it is not immediately clear how to use direct supervision for the SiMT task, in which the model must begin translation based on *partial* context. Nevertheless, we believe that direct supervision for the SiMT task is possible and propose a way to accomplish that with a *causally aligned* dataset. In such a dataset, a target word never appears before its corresponding (when such correspondence can be established) source word in time, which is defined as the number of words from the sentence start. In other words, in a causally aligned source-target sentence pair, source words are guaranteed to be causal relative to their corresponding target words. We illustrate this in Fig. 2.

Note that the causal alignment is not always perfect: due to the word length mismatch between the source and target, not all source words will have a corresponding target word, and vice versa, not every target word will have a corresponding word in the source. However, as we demonstrate below, fine-tuning an LLM on such a causally aligned dataset enabled us to achieve results comparable to some state-of-the-art baselines.

In order to causally align the source and target, we split each sentence using the `word_tokenize` function from the *nltk* package (Bird et al., 2009), treating punctuation marks as "words", then find the best correspondences between the source and target words with *SimAlign* (Jalili Sabet et al., 2020), and finally insert as many <WAIT> tokens into the target as appropriate. If after alignment the



	original en	original ru	causally aligned en	causally aligned ru
1	They	Они	They	Они
2	live	живут	live	живут
3	in	глубоко	in	<WAIT>
4	the	в	the	<WAIT>
5	depths	конголезских	depths	глубоко
6	of	джунглях	of	в
7	the	,	the	<WAIT>
8	Congolese	где	Congolese	конголезских
9	jungle	сложно	jungle	джунглях
10	and	проводить	and	,
11	it	исследования	it	где
12	has	.	has	<WAIT>
13	been		been	<WAIT>
14	very		very	<WAIT>
15	difficult		difficult	сложно
16	to		to	проводить
17	study		study	исследования
18	them		them	<WAIT>
19	.		.	.

	original en	original de	causally aligned en	causally aligned de
1	He	Er	He	Er
2	took	befreite	took	<WAIT>
3	one	uns	one	<WAIT>
4	of	von	of	<WAIT>
5	the	einer	the	<WAIT>
6	worst	der	worst	<WAIT>
7	scourges	schlimmsten	scourges	<WAIT>
8	of	Geißeln	of	<WAIT>
9	mankind	der	mankind	<WAIT>
10	away	Menschheit	away	befreite
11	from	.	from	<WAIT>
12	us		us	uns
13	.		.	von
14			_ _	einer
15			_ _	der
16			_ _	schlimmsten
17			_ _	Geißeln
18			_ _	der
19			_ _	Menschheit
20			_ _	.

Figure 2: Causal alignment. Two examples are shown: for en-ru (left) and en-de (right). If time is defined as the number of words from the beginning of the sentence, before alignment, some target words appear earlier than their corresponding English equivalents in the source. By inserting <WAIT> tokens, we can shift those target words into the future, thereby achieving causality for every content word. "_ _" are fillers added at the end of the source sentence if neccessary to match its length with that of the target. When a target word (e.g. "befreite") has no directly corresponding source word, it can be placed anywhere between the neighboring aligned words ("Er" and "uns").

target becomes longer than the source due to added <WAIT> tokens, we pad the source at the end with filler strings ensuring that the aligned source and target sentences have the same number of "words". These filler strings are only used for convenient batching and are dropped before tokenization.

Supervised Fine-Tuning (SFT). We fine-tune the LLAMA-2 13B and and 70B models Touvron et al. (2023) [1] to optimize the following objective:

$$\mathcal{L}_{\text{T2TT}} = -\sum_{t=1}^{|y|} \log p(y_t | y_{<t}, x_{\leq t}) \qquad (1)$$

where y_t is the next target token, $y_{<t}$ are previously generated (and committed) tokens and $x_{\leq t}$ and the partial source tokens revealed up to the time step t. Following (Touvron et al., 2023), we zero out the loss on tokens corresponding the to system message and source, only backpropagating on the target tokens.

We use batches of prompt-response pairs collated in the following way. Before tokenization, each aligned sentence-target pair selected from the causally aligned dataset is trimmed from the right to leave first l words, where $l \sim U(1, L)$ and L is the full length of the causally aligned source-target pair. After trimming, all the <WAIT> tokens except the last one (if present) are dropped, because they are never plugged back into the input and only serve the purpose of signaling for more words to be read in. Likewise, we drop all the fillers (if present) from the source. Finally, the system message, trimmed source and trimmed target are joined into the prompt (as shown in Fig. 3) and tokenized. Because there is no <WAIT> token in the LLAMA 2 tokenizer, we use 0 (which originally corresponds to the <UNK> token). Thus, the model is fine-tuned to either output the next token of a word or <WAIT>, if the partial source does not contain sufficient information needed to predict translation.

To save memory, we loaded the base model in 4-bit precision using the `bitsandbytes` library (Dettmers et al., 2022). This allowed us to fine-tune LLAMA 2 70B on one NVIDIA A100 80GB device. We fine-tune the base model with LoRA (Hu et al., 2022) with $r = 16$ and $\alpha = 32$ for 3 epochs with a batch size of 25 and gradient accumulation of 4 steps. We save model checkpoints every 10 steps and select the one with the lowest validation loss for inference. For optimization, we used the `paged_adamw_32bit` optimizer (Dettmers et al., 2022) with default parameters, and a learning rate schedule with a linear warm-up of 10 steps up to 0.00005, followed by a cosine decay. For parameter-efficient training, as well as for inference, we used the `transformers`[2] library.

Inference. At inference, given a prompt (Fig. 3) comprised of a system message, partial source

[1] We found that the LLAMA-2-CHAT variants (both 13B and 70B), when fine-tuned on our causally aligned dataset performed slightly, but consistently, worse than LLAMA-2, and we report the results for the latter model only.

[2] https://huggingface.co

and previously committed partial target, the LLM greedily generates one or more next tokens. We use modified wait-*k* (Ma et al., 2019), in which WRITE actions are only allowed when the length of the PARTIAL_SOURCE is equal or greater than k. Since k controls the tradeoff between quality and latency, we report results for different values of k. After a full new word – which may consist of several tokens – is generated, the prompt is updated by appending a new source word to the partial source and the newly generated word to the partial target. This process is repeated until the LLM generates the <EOS> token. All the generation parameters were at default, except top_p which we set to 0.7. We did not use beam search during generation.

After all the source words have been revealed, the input is no longer partial and no new words are added to it, but the generation process continues until <EOS>. Importantly, if the model generates the <WAIT> token, a new source word is read in, but the <WAIT> token itself is not appended to the partial target. We illustrate the inference process in Fig. 4 and Algorithm 1.

Algorithm 1 Inference process

```
partial_target = []
k = WAIT_K
while True:
    partial_source = SOURCE[:k]
    prompt = " ".join([
        SYS_MSG,
        partial_source,
        partial_target])

    # generate until next full word,
    # <EOS> or <WAIT>
    if k > len(SOURCE):
        suppress_wait = True
    else:
        suppress_wait = False

    next_word = model.generate(
        prompt, suppress_wait)

    if next_word == "<EOS>":
        break   # finish sentence
    elif next_word == "<WAIT>":
        k += 1  # READ action
    else:
        partial_target.append(next_word)
        k += 1  # WRITE action
```

Prompt structure. We follow a similar prompt structure as in Touvron et al. (2023) (Fig. 3). For the SYSTEM_MESSAGE we used the following text: *"You are a professional conference interpreter. Given an English text you translate it into* {TARGET_LANGUAGE} *as accurately and as concisely as possible, NEVER adding comments of your own. You output translation when the information available in the source is unambiguous, otherwise you output the wait token (*{WAIT_TOKEN}*), not flanked by anything else. It's important that you get this right."*. We note that while the system message is only necessary in zero-shot SiMT scenarios – which we discuss below – for consistency we still kept it in all the experiments reported here, including those involving supervised fine-tuning.

Automatic speech recognition. Given that the LLMs are designed to process text input, to enable S2TT we first need to extract text from input audio, for which we use Whisper [3] (Radford et al., 2023). Specifically, for each READ action, a new segment of audio, lasting 200 ms, is added to any previously read audio chunks and then processed by Whisper. This method of fixed audio windowing often results in partially clipped words. To address this, we discard the last word predicted by Whisper during each READ action unless the entire source audio has been read in. We note that this approach to online ASR is somewhat naive and has room for improvement – as indicated by a roughly 1 BLEU point decrease due to ASR-related errors (Fig. 9). Since our main objective was to assess the capability of LLMs to perform SiMT tasks, we leave exploring ways to decrease ASR errors to future work.

4 Results

Data. For supervised fine-tuning (SFT), validation and testing, we used MuST-C v2.0 (Di Gangi et al., 2019) for English-to-German (en-de) and English-to-Russian (en-ru) translation direction. We randomly selected 4000 sentences for training and 100 sentences for validation. However, since it is possible that the dataset that LLAMA2 was pre-trained on and MuST-C v2.0 (including its validation and test set) might have overlapping content, we also compiled another test set, which we call TED-TST-2023. This test set has a similar content type (TED talks) and follows the same format as the original MuST-C v2.0, but only includes talks posted after February 2023. The dataset has two parts: 102 source-target pairs for en-de and 102 for en-ru language pair. Unless indicated otherwise, we report the results obtained on this test set.

T2TT. We first analyzed the T2TT performance

[3] We used whisper-large-v2.

```
<s>[INST]
<<SYS>>
SYSTEM_MESSAGE
<</SYS>>
Translate this text: PARTIAL_SOURCE [/INST] PARTIAL_TARGET
```

Figure 3: Prompt structure. «SYS», «/SYS» and [INST], [/INST] are special strings used in Llama to mark the system message and instruction within the prompt.

PARTIAL_SOURCE	PARTIAL_TARGET	Prediction
I		<WAIT>
I like		Я
I like to	Я	люблю
I like to have	Я люблю	<WAIT>
I like to have tea	Я люблю	пить
I like to have tea in	Я люблю пить	чай
I like to have tea in the	Я люблю пить чай	<WAIT>
I like to have tea in the morning.	Я люблю пить чай	по
I like to have tea in the morning.	Я люблю пить чай по	утрам.
I like to have tea in the morning.	Я люблю пить чай по утрам.	<EOS>

Figure 4: An illustration of the inference process for the en-ru language pair. Assuming $k = 1$, given the prompt with one source and zero target words, the model first outputs <WAIT>, which signals for the next source word to be read in. At the next step, the model generates the first target word (Я), which is plugged into the prompt at the next step. This process continues until <EOS> is generated.

or our approach on the MuST-C dataset v2.0 (Di Gangi et al., 2019). To get a sense for the quality-latency tradeoff, we plot BLEU scores against several different values of k (because k is the only way to control the translation latency). The results, shown in Fig. 5, suggest that the LLM's size is a major factor determining the translation quality.

S2TT. We next test fine-tuned LLMs and compare them with two recently published S2TT baselines (Fukuda et al., 2023; Papi et al., 2023) (Fig. 6) and in zero-shot mode to OpenAI's GPT-3.5 and GPT-4 (Fig. 7). For as fair a comparison as possible, we ensured that average lagging (AL) of all of the models was below approximately 2000 ms. For Llama-2 models we set $k = 5$ (the other models' settings are listed in Appendix 5). The boxplots in Figs. 6, 7 and throughout are drawn based on data from 10 evaluation runs of the same model with the same parameters on sentence pairs sampled with replacement from TED-TST-2023. The results show a degradation of translation quality by approximately 1 BLEU score point compared to T2TT mode, which is to be expected due to ASR errors (Fig. 9).

Zero-shot T2TT. Can the LLMs perform the SiMT task zero-shot, that is without any prior fine-tuning? To answer this question, we used LLMs that had been fine-tuned with RLHF (reinforcement learning with human feedback) (Stiennon et al., 2020) for instruction following: open-source LLAMA2-CHAT, as well as GPT-3.5 (gpt-3.5-turbo-0613) and GPT-4 (gpt-4-0613), which were among the strongest closed-source LLMs available at the time of writing this paper. In general, with the notable exception of GPT-4, zero-shot performance was poor (Fig. 7). Inspection of the translations revealed that the models consistently failed to follow the prompt instruction, specifically, (1) generating output in English rather than the target language, (2) adding expressly prohibited explanatory comments, (3) restating or summarizing the task, or (4) explaining the reason for adding <WAIT> tokens). GPT-4 was surprisingly good, performing better than the supervised fine-tuned LLAMA2-70B, and we speculate that the performance of GPT-3.5 and GPT-4 could be further improved with SFT [4], more sophisticated

[4] SFT was not available for GPT-3.5 and GPT-4 at the time of writing this paper.

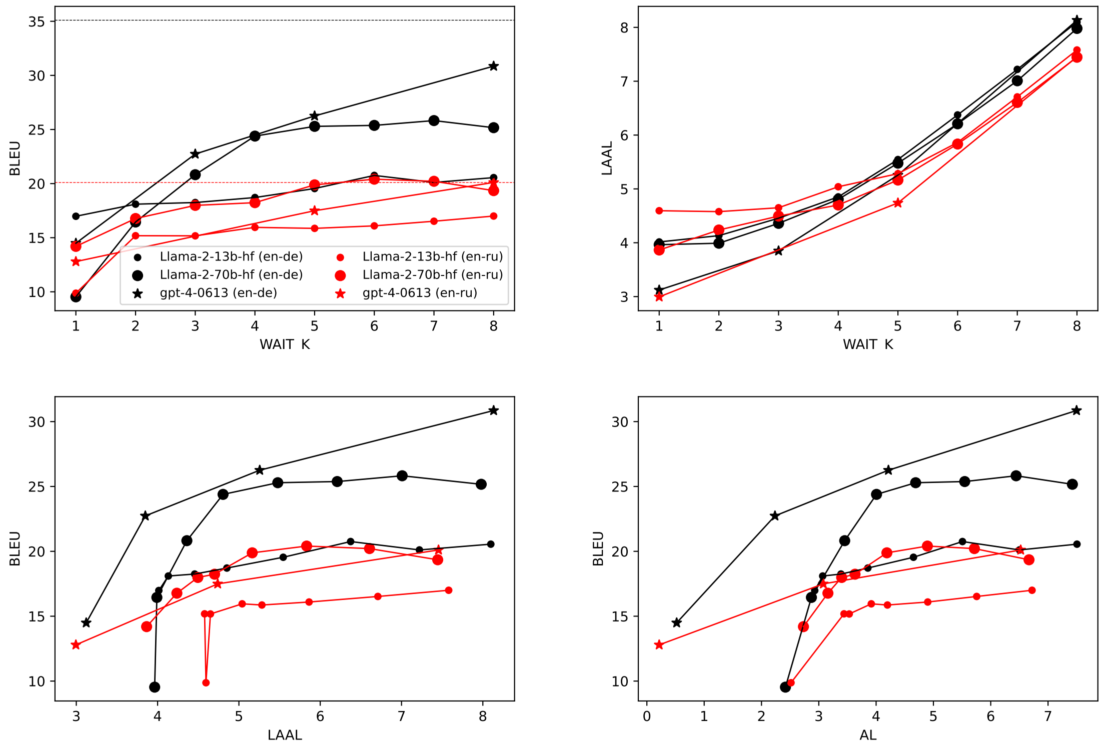


Figure 5: Dependence of latency and quality on k (top panels) and quality-latency tradeoff curves (bottom panels) for the T2TT mode on the MuST-C v2.0 dataset. Dashed lines mark GPT-4's sentence-level BLEU scores: black for en-de and red for en-ru. AL and LAAL are Average Lagging and Length-Adaptive Average Lagging, respectively.

generation strategies and prompt engineering.

Importance of wait tokens. To evaluate the utility of <WAIT> tokens, we conduct two ablation experiments. In the first experiment we consider a zero-shot translation scenario in which GPT-4 was not instructed to use <WAIT> tokens. In the second experiment, we suppress the generation of <WAIT> tokens in supervised fine-tuned LLMs. The results, as indicated in Table 1, reveal that GPT-4 demonstrates marginally inferior performance when $k \in \{1, 2\}$[5] when not instructed about <WAIT> tokens. However, it is important to note that in a zero-shot context, GPT-3.5 and GPT-4 seldom generated <WAIT> tokens (almost never for $k > 2$). Therefore, the directive to employ these tokens only produced a discernible impact for smaller values of k. By contrast, in the SFT scenario, suppressing <WAIT> tokens led to significantly decreased performance for both the 13B and 70B versions of LLAMA-2 (Table 1 (b, c)).

To gain insight into where LLAMA-2 tended to insert the <WAIT> token, we plot the distribution of words after which the SFT models generated this token. Fig. 8 shows that the model generated <WAIT> after function words[6] (129 times) rather than content words (103 times), indicating that it had learned to choose appropriately between READ and WRITE actions.

5 Conclusions and Future Directions

We have shown that with minimal fine-tuning and without resorting to sophisticated training techniques (e.g. checkpoint averaging (Fukuda et al., 2023)), an off-the-shelf pre-trained LLM can perform simultaneous translation and achieve encouraging results that rival some of the recent SiMT models. This opens interesting directions to be explored in future work, such as multilingual fine-tuning, self-instruct (Wang et al., 2023b) and human preference tuning (Ouyang et al., 2022).

There are several reasons to believe that we are far from unlocking the full potential of LLMs for SiMT. First, we followed the practice – standard in the SiMT literature – of evaluating the model on individual sentences randomly sampled from con-

[5] We did not study the role of <WAIT> tokens for $k > 2$, as GPT-4 almost never generates them for those values of k.

[6] Tagged with these Penn Treebank POS-tags: CC, DT, EX, IN, MD, PDT, POS, PRP$, RB, RBR, RBS, RP, TO, WDT, WP, WP$, WRB

k	w/ <WAIT>	w/o <WAIT>
1	15.23	14.88
2	17.17	15.66

(a)

k	w/ <WAIT>	w/o <WAIT>
1	14.76	10.80
2	14.97	11.94
4	17.42	15.67

(b)

k	w/ <WAIT>	w/o <WAIT>
1	17.17	4.64
2	16.83	7.84
4	19.24	14.80

(c)

Table 1: Removing the instruction to generate or suppressing the <WAIT> token degrades performance. The numbers indicate BLEU scores on TED-TST-2023 (en-de) in T2TT mode for GPT-4 (a), 13B (b) and 70B (c) Llama-2.

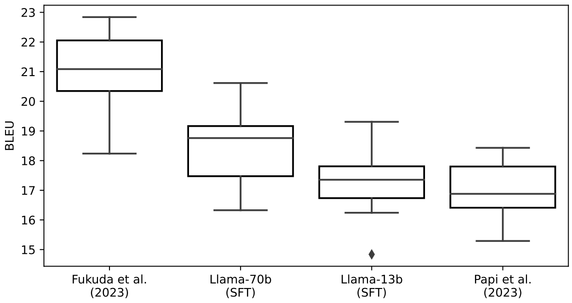


Figure 6: S2TT performance of SFT LLAMA-2 and two recently published models on the en-de language pair on TED-TST-2023. See also Table 3.

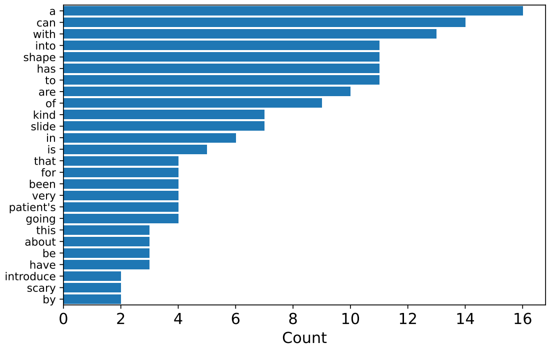


Figure 8: After fine-tuning, LLAMA-2 generates <WAIT> tokens predominantly after function words (especially articles and prepositions).

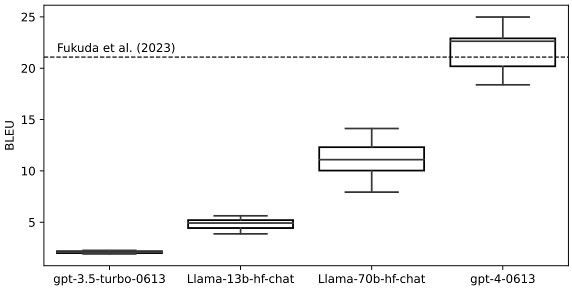


Figure 7: Zero-shot S2TT performance or our approach compared with GPT-3.5 and GPT-4 on the en-de language pair on TED-TST-2023.

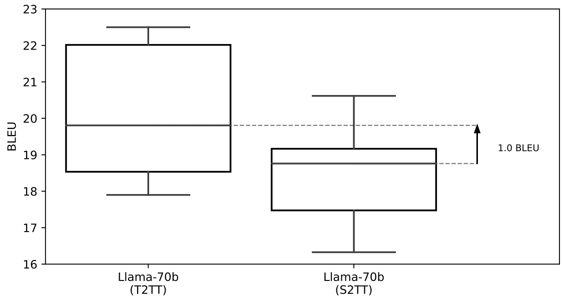


Figure 9: Performance decrease due to ASR-related errors. In T2TT mode, Llama2-70b performs by about 1 BLEU score point better than the same model on the same data in S2TT mode.

tinuous prose. However, many (if not the majority of) short sentences are ambiguous when taken out of context. Even human conference interpreters routinely prepare for an upcoming translation job, studying relevant materials, which means that they do not have to translate sentences taken out of context. For this reason, we believe that the most straightforward way to boost the performance of future LLM-based SiMT systems is to insert background information into the prompt. Second, the big difference in zero-shot performance between GPT-3.5 and GPT-4 suggests that size is likely the biggest factor determining the model's translation quality, and that further gains can be achieved once SFT becomes available for these closed-source models.

6 Limitations

Several performance bottlenecks impede the real-world application of our SiMT approach in its current form, notably the system message, whose length frequently exceeds that of the source sentence, leading to substantial slowdowns (refer to Table 2). Additionally, the ASR subsystem and weight quantization introduce further delays. To address these issues, instead of using a separate ASR model, future work might rely on an end-to-end approach similar to Fathullah et al. (2024), in which input audio is directly mapped into the LLM's embedding space, reducing the system's overall latency. Efficient quantization schemes, faster algorithms and hardware support for low bit-width arithmetic are also promising directions. Finally, because LLAMA-2 was trained predominantly on English text, its tokenizer represents English more efficiently than other languages. That is, fewer tokens on average are needed to encode a text in English than a text of the same length (in characters) in another, less represented, language. Thus, future LLMs pre-trained on a linguistically more balanced dataset, might be faster at inference in SiMT tasks.

Acknowledgements

The first author acknowledges financial support from KAKENHI grant JP23KJ2131 and Google.

References

Victor Agostinelli, Max Wild, Matthew Raffel, Kazi Fuad, and Lizhong Chen. 2024. Simul-LLM: A framework for exploring high-quality simultaneous translation with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10530–10541, Bangkok, Thailand. Association for Computational Linguistics.

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

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy. Association for Computational Linguistics.

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 437–445.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Colin Cherry and George Foster. 2019. Thinking slow about latency evaluation for simultaneous machine translation. *arXiv preprint arXiv:1906.00048*.

Chung-Cheng Chiu and Colin Raffel. 2018. Monotonic chunkwise attention. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. 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)*, pages 493–499, New Orleans, Louisiana. Association for Computational Linguistics.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems*, volume 35, pages 30318–30332. Curran Associates, Inc.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. 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)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2020. Efficient Wait-k Models for Simultaneous Machine Translation. In *Proc. Interspeech 2020*, pages 1461–1465.

Yassir Fathullah, Chunyang Wu, Egor Lakomkin, Junteng Jia, Yuan Shangguan, Ke Li, Jinxi Guo, Wenhan Xiong, Jay Mahadeokar, Ozlem Kalinli, Christian Fuegen, and Mike Seltzer. 2024. Prompting large language models with speech recognition abilities. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13351–13355.

Christian Fügen, Alex Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine translation*, 21:209–252.

Ryo Fukuda, Yuta Nishikawa, Yasumasa Kano, Yuka Ko, Tomoya Yanagita, Kosuke Doi, Mana Makinae, Sakriani Sakti, Katsuhito Sudoh, and Satoshi Nakamura. 2023. NAIST simultaneous speech-to-speech translation system for IWSLT 2023. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 330–340, Toronto, Canada (in-person and online). Association for Computational Linguistics.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. 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, and Yang Feng. 2023. Learning optimal policy for simultaneous machine translation via binary search. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2318–2333, Toronto, Canada. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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.

Yasumasa Kano, Katsuhito Sudoh, and Satoshi Nakamura. 2022. Simultaneous neural machine translation with prefix alignment. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 22–31, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Yuka Ko, Ryo Fukuda, Yuta Nishikawa, Yasumasa Kano, Katsuhito Sudoh, and Satoshi Nakamura. 2023. Tagged end-to-end simultaneous speech translation training using simultaneous interpretation data. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 363–375, Toronto, Canada (in-person and online). Association for Computational Linguistics.

Danni Liu, Gerasimos Spanakis, and Jan Niehues. 2020a. Low-Latency Sequence-to-Sequence Speech Recognition and Translation by Partial Hypothesis Selection. In *Proc. Interspeech 2020*, pages 3620–3624.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020b. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

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 Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2020b. Monotonic multihead attention. In *International Conference on Learning Representations*.

Nathalia Nascimento, Paulo Alencar, and Donald Cowan. 2023. Gpt-in-the-loop: Supporting adaptation in multiagent systems. In *2023 IEEE International Conference on Big Data (BigData)*, pages 4674–4683.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022. 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.

Sara Papi, Matteo Negri, and Marco Turchi. 2023. Attention as a guide for simultaneous speech translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13340–13356, Toronto, Canada. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Peter Polák, Ngoc-Quan Pham, Tuan Nam Nguyen, Danni Liu, Carlos Mullov, Jan Niehues, Ondřej Bojar, and Alexander Waibel. 2022. CUNI-KIT system for simultaneous speech translation task at IWSLT 2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 277–285, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR.

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2837–2846. JMLR.org.

Harsh Satija and Joelle Pineau. 2016. Simultaneous machine translation using deep reinforcement learning. In *ICML 2016 Workshop on Abstraction in Reinforcement Learning*.

Vivek Kumar Rangarajan Sridhar, John Chen, Srinivas Bangalore, Andrej Ljolje, and Rathinavelu Chengalvarayan. 2013. Segmentation strategies for streaming speech translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 230–238.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

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

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023b. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Zekun Wang, Ge Zhang, Kexin Yang, Ning Shi, Wangchunshu Zhou, Shaochun Hao, Guangzheng Xiong, Yizhi Li, Mong Yuan Sim, Xiuying Chen, Qingqing Zhu, Zhenzhu Yang, Adam Nik, Qi Liu, Chenghua Lin, Shi Wang, Ruibo Liu, Wenhu Chen, Ke Xu, Dayiheng Liu, Yike Guo, and Jie Fu. 2023c. Interactive natural language processing.

Haoran Xu, Young Jin Kim, Amr Sharaf, and Hany Hassan Awadalla. 2024. A paradigm shift in machine translation: Boosting translation performance of large language models. In *The Twelfth International Conference on Learning Representations*.

Mahsa Yarmohammadi, Vivek Kumar Rangarajan Sridhar, Srinivas Bangalore, and Baskaran Sankaran. 2013. Incremental segmentation and decoding strategies for simultaneous translation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1032–1036.

Ruiqing Zhang, Chuanqiang Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2020. Learning adaptive segmentation policy for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2280–2289, Online. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2021. Universal simultaneous machine translation with mixture-of-experts wait-k policy. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7306–7317, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022. Information-transport-based policy for simultaneous translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 992–1013, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2023. Hidden markov transformer for simultaneous machine translation. In *International Conference on Learning Representations*.

Appendix

A Examples of translation

A.1 en-de

Source: But in Antarctica, some scientists who are normally pretty levelheaded are getting a little bit freaked out, I would say, is a fair definition, about the lowest level ever, at this point in the year, of sea ice.

TransLLaMa: Aber in der Antarktis, einige Wissenschaftler, die normalerweise recht nüchtern sind, werden ein wenig verängstigt, würde ich sagen, ist eine gerechte Definition über den niedrigsten Stand, den wir jemals zu diesem Zeitpunkt im Jahr von Meereis hatten.

Back translation: But in Antarctica, some scientists who are normally quite sober are getting a little frightened, I would say, is a fair definition of the lowest level we've ever had of sea ice at this point in the year.

Source: I remember when there were so many fossil fuel delegates in Madrid, but by the time we got to Glasgow a year and a half ago, the delegates from the fossil fuel companies made up a larger group than the largest national delegation.

TransLLaMa: Ich erinnere mich an die Zeit, als es so viele Delegierte der fossilen Brennstoffindustrie in Madrid gab, aber als wir ein Jahr und ein halbes später in Glasgow waren, machten die Delegierten der fossilen Brennstoffunternehmen eine größere Gruppe aus als die größte nationale Delegation.

Back translation: I remember when there were so many fossil fuel industry delegates in Madrid, but when we were in Glasgow a year and a half later, the fossil fuel company delegates made up a larger group than the largest national delegation.

Source: And I'm a really positive person, so it's actually difficult to say this, but the way we do philanthropy right now, the way we've done it for decades, is broken.

TransLLaMa: Ich bin eine sehr positive Person, also ist es für mich eigentlich schwierig, das zu sagen, aber die Art und Weise, wie wir Philanthropie betreiben, die Art und Weise, wie wir es seit Jahrzehnten betreiben, ist kaputt.

Back translation: I'm a very positive person, so it's actually hard for me to say this, but the way we do philanthropy, the way we've done it for decades, is broken.

Source: Work norms, of course, differ around the world, but we replicated our original American result in South Korea, which is known by the numbers to be one of the hardest-working countries in the OECD, and in France, which is known for other strengths.

TransLLaMa: Natürlich unterscheiden sich die Arbeitsnormen in der Welt, aber wir konnten unser ursprüngliches Ergebnis in den USA in Südkorea wiederholen, das bekannt ist als eines der härtesten arbeitenden Länder in der OECD, und in Frankreich, das für andere Stärken bekannt ist.

Back translation: Of course, labor standards vary around the world, but we were able to replicate our original result in the US in South Korea, known as one of the hardest working countries in the OECD, and in France, known for other strengths.

A.2 en-ru

Source: A few years ago, I called for toothpaste from my hotel room, and it was delivered to me by a robot.

TransLLaMa: Несколько лет назад я попросила зубную пасту из моего номера в отеле и она была доставлена мне роботом.

Back translation: A few years ago I asked for toothpaste from my hotel room and it was delivered to me by robot.

Source: It conducted the first quantum video call, and it launched the first drone that can carry a passenger in air.

TransLLaMa: Она провела первую видеоконференцию с использованием квантовых технологий и запустила первый беспилотный самолет, способный перевозить пассажира в воздухе.

Back translation: She held the first video conference using quantum technology and launched the first unmanned aircraft capable of carrying a passenger in the air.

Source: And it's really gratifying to hear from people like Raymond who are using the technology we are building, and others, for so many wonderful things.

TransLLaMa: И это действительно приятно слышать от людей, как Раймонд, которые используют технологии, которые мы создаём, и другие, для таких замечательных вещей.

model	mode	quantization	system message	size, bn param.	RTF
Ours	T2TT	16-bit	no	13	1.7
Ours	T2TT	4-bit	no	13	2.2
Ours	T2TT	16-bit	yes	13	2.9
Ours	T2TT	4-bit	yes	13	4.2
Ours	S2TT	16-bit	no	13	5.9
Ours	S2TT	4-bit	no	13	7.6
Ours	S2TT	16-bit	yes	13	8.0
Ours	S2TT	4-bit	yes	13	9.3
Ours	T2TT	4-bit	no	70	14.6
Ours	T2TT	4-bit	yes	70	20.2
Ours	S2TT	4-bit	no	70	15.3
Ours	S2TT	4-bit	yes	70	23.9
GPT-4	T2TT	unknown	yes	unknown	1.5
GPT-4	S2TT	unknown	yes	unknown	4.8
Fukuda et al. (2023)	S2TT	16-bit	N/A	1.04	0.7
Papi et al. (2023)	S2TT	16-bit	N/A	0.176	1.4

Table 2: **Comparison of our system's inference times across varying sizes with selected baselines on en-de.** Real-time factor (RTF) is the ratio of the amount of time taken to process source audio to the length of the source audio itself. RTF less than one means the model is faster than real time. The RTF was calculated based on the known length of the audio corresponding to the source transcripts and the time to complete translation of that text. For T2TT mode, the source audio transcripts were fed directly to the LLM. We note that removing the system message from the prompt speeds up inference with no noticeable drop in quality for supervised fine-tuned models. Loading our model's weights with 16-bit (instead of 4-bit) quantization further accelerates inference. Finally, the use of ASR in S2TT mode substantially reduces system speed.

System	BLEU	LAAL	AL	AP	DAL
gpt-3.5-turbo-0613 (zero-shot)	2.08 (0.24)	2637.11 (252.79)	2574.98 (230.95)	0.35 (0.0)	2477.55 (146.26)
gpt-4-0613 (zero-shot)	21.82 (2.81)	2448.86 (74.74)	1998.63 (110.91)	0.94 (0.03)	2813.47 (69.48)
Llama-70b-hf (SFT)	18.41 (1.4)	2107.57 (59.68)	1619.64 (76.47)	0.84 (0.02)	2454.72 (67.84)
Llama-13b-hf (SFT)	17.07 (0.68)	2358.89 (34.11)	1880.76 (61.77)	0.88 (0.02)	2735.34 (40.88)
Papi et al. (2023)	17.01 (1.0)	2295.72 (41.54)	1867.1 (148.69)	0.77 (0.01)	3251.38 (139.12)
Fukuda et al. (2023)	21.08 (1.41)	2005.39 (71.04)	1397.33 (85.74)	0.9 (0.01)	3066.15 (122.01)

Table 3: **Mean performance metrics on en-de of Llama-2 (SFT) compared to some recent S2TT systems and GPT-3.5 and GPT-4 (zero-shot).** Then mean and standard deviation (in brackets) are computed over 10 runs of the same model on 102 source-target pairs sampled with replacement from TED-TST-2023. Here we report additional comparisons including latency performance measured using several different metrics, including Average Lagging (AL) (Ma et al., 2019), Length Adaptive Average Lagging (LAAL) (Papi et al., 2022), Average Proportion (AP) (Cho and Esipova, 2016) and Differentiable Average Lagging (DAL) (Cherry and Foster, 2019).

System	BLEU	LAAL	AL	AP	DAL
gpt-3.5-turbo-0613 (zero-shot)	0.14 (0.1)	2876.85 (240.03)	2861.22 (245.91)	0.28 (0.04)	2661.22 (231.0)
gpt-4-0613 (zero-shot)	16.86 (2.27)	2022.81 (20.3)	1584.38 (91.81)	0.82 (0.04)	2390.11 (23.65)
Llama-70b-hf (SFT)	20.96 (1.71)	2252.75 (49.77)	1937.76 (62.75)	0.9 (0.08)	2676.56 (62.11)
Llama-13b-hf (SFT)	16.9 (1.52)	2238.6 (48.38)	1917.46 (90.38)	0.87 (0.03)	2641.01 (45.73)

Table 4: **Mean performance metrics on en-ru of Llama-2 (SFT) compared to some recent S2TT systems and GPT-3.5 and GPT-4 (zero-shot).** Then mean and standard deviation (in brackets) are computed over 10 runs of the same model on 102 source-target pairs sampled with replacement from TED-TST-2023.

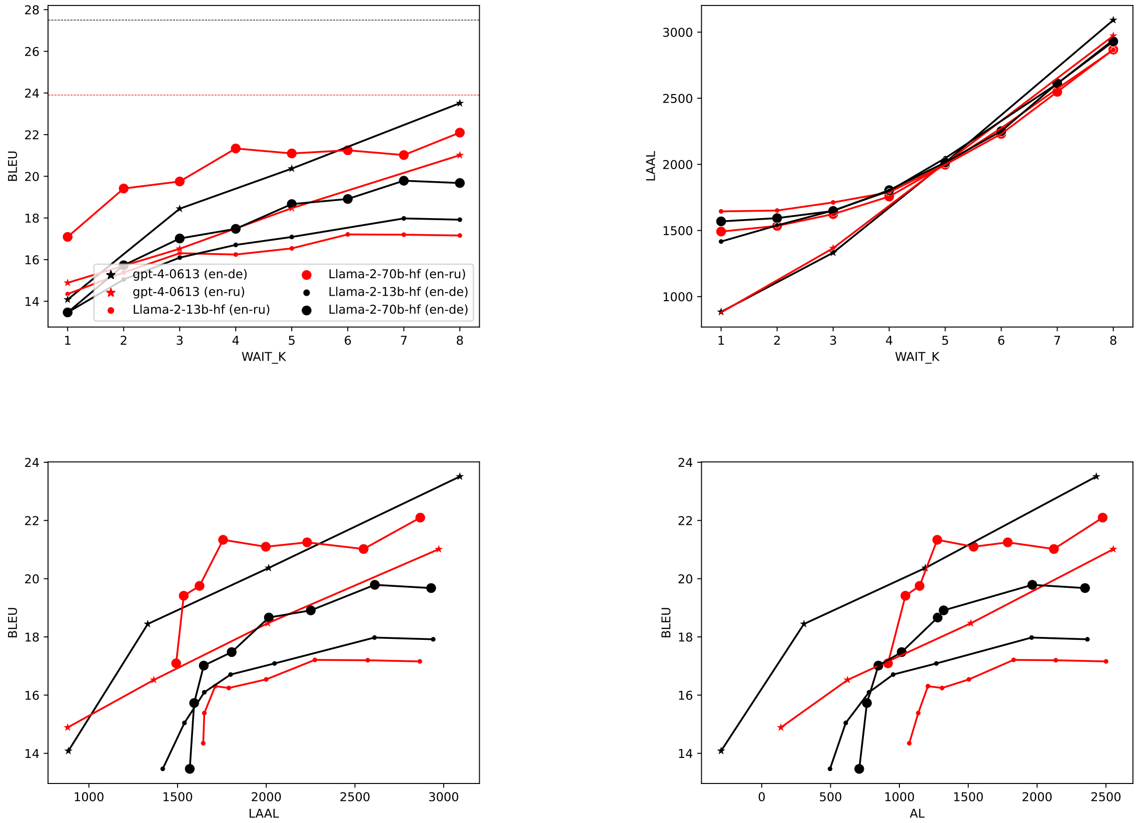


Figure 10: **Dependence of latency and quality on k (top panels) and quality-latency tradeoff curves (bottom panels) for the S2TT mode on the TED-TST-2023 dataset**. For reference, dashed lines indicated GPT-4's sentence-level (i.e. with k set to the sentence length) BLEU scores: black for en-de and red for en-ru.

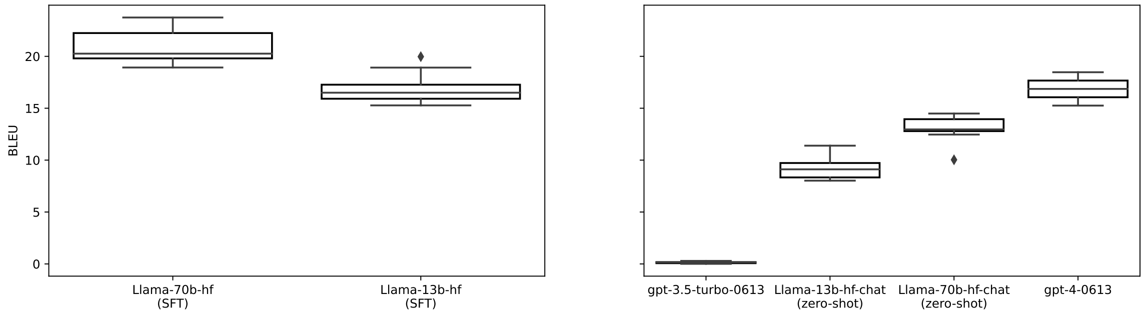


Figure 11: **S2TT en-ru performance of our method on TED-TST-2023**. Left panel: supervised fine-tuned LLAMA-2. Right panel: zero-shot S2TT performance of LLAMA-2-CHAT. All the runs were on TED-TST-2023, with $k = 5$ to ensure AL around 2000 ms. Each of the boxplots is drawn based on data from 10 evaluation runs on sentences randomly sampled with replacement from the test set.

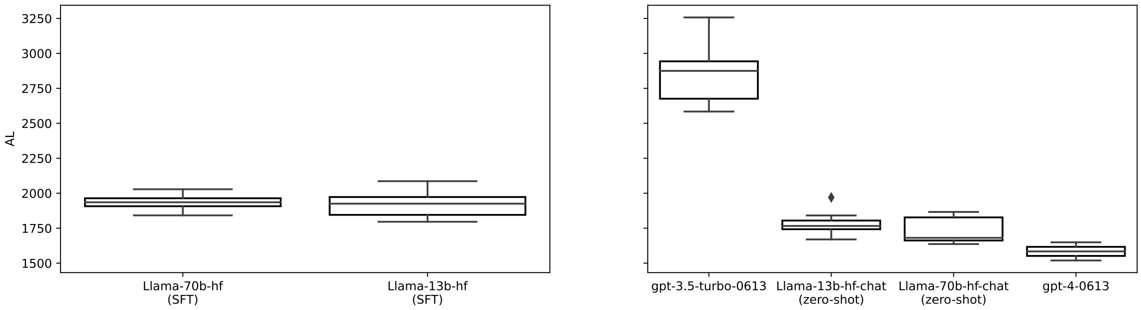


Figure 12: **Average lagging in S2TT mode for the English-Russian language pair**. Left panel: supervised fine-tuned LLAMA-2. Right panel: zero-shot S2TT performance of LLAMA-2-CHAT. All the runs were on TED-TST-2023, with $k = 5$ to ensure AL around 2000 ms. Each of the boxplots is drawn based on data from 10 evaluation runs on sentences randomly sampled with replacement from the test set.

baseline	Papi et al. (2023)	Fukuda et al. (2023)
parameters	`extract-attn-from-layer 5`	`source-segment-size 950`
	`frame-num 2`	`la-n 2`
	`attn-threshold 0.25`	`beam 5`
	`speech-segment-factor 8`	`sacrebleu-tokenizer 13a`

Table 5: **Parameters used for comparisons with baselines on the S2ST en-de task.** For Papi et al. (2023) we used the open-source implementation of the model (`https://github.com/hlt-mt/FBK-fairseq/tree/master/fbk_works`). For Fukuda et al. (2023) we obtained the source code of the model and weights on request from the authors. All the evaluations were run in *SimulEval* (Ma et al., 2020a) (`https://github.com/facebookresearch/SimulEval`).