BeaverTalk: Oregon State University's IWSLT 2025 Simultaneous Speech Translation System

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

Abstract

This paper discusses the construction, fine-tuning, and deployment of BeaverTalk[1], a cascaded system for speech-to-text translation as part of the IWSLT 2025 simultaneous translation task. The system architecture employs a VAD segmenter for breaking a speech stream into segments, Whisper Large V2 for automatic speech recognition (ASR), and Gemma 3 12B for simultaneous translation. Regarding the simultaneous translation LLM, it is fine-tuned via low-rank adaptors (LoRAs) for a conversational prompting strategy that leverages a single prior-sentence memory bank from the source language as context. The cascaded system participated in the English→German and English→Chinese language directions for both the low and high latency regimes. In particular, on the English→German task, the system achieves a BLEU of 24.64 and 27.83 at a StreamLAAL of 1837.86 and 3343.73, respectively. Then, on the English→Chinese task, the system achieves a BLEU of 34.07 and 37.23 at a StreamLAAL of 2216.99 and 3521.35, respectively.

1 Introduction

This paper covers Oregon State University's simultaneous translation system, BeaverTalk, for IWSLT 2025. The system constructed takes in a speech stream input and outputs text translation in a cascaded manner for two language pairs, those being English→German (en→de) and English→Chinese (en→zh). Unique to IWSLT 2025's simultaneous translation task (Abdulmumin et al., 2025), this system generates translation for unsegmented audio. Architecture-wise, the system includes a VAD speech segmenter (Team, 2024), breaking a speech stream into segments, Whisper Large V2 (Radford et al., 2022) performing automatic speech recognition (ASR), and a fine-tuned Gemma 3 12B model (Team et al., 2025) that performs context-aware conversational prompting to generate a simultaneous translation.

The simultaneous translation portion of this cascaded system is fine-tuned on OpenSubtitles v2018 (Lison et al., 2018) across both language pairs. Given the unsegmented source for this task, leveraging additional context is possible and likely to improve results, based on prior work (Papi et al., 2024). As such, our system utilizes a single-sentence memory bank for the source language as context. This memory bank required modifying the typical conversational prompting structure for simultaneous translation(Wang et al., 2024).

Although a fine-tuned Gemma 3 12B leveraging conversational prompting is a powerful model for simultaneous translation, its application in a cascaded architecture suffers from typical issues of error propagation (Tran et al., 2022; Zhou et al., 2024). As such, maximizing the capabilities of a powerful simultaneous translation LLM requires minimizing these errors in the preceding steps, consisting of the VAD segmenter and Whisper ASR model. As such, we conduct an extensive inference time hyperparameter search aimed at minimizing error propagation. From the joint contributions of our cascaded simultaneous translation system and minimization of error propagation, we achieve impressive results on the ACL 60/60 development set. For example, on the English→German language pair, our cascaded system achieves a BLEU of 24.64 and 27.83 at a streamLAAL of 1837.86 and 3343.73. Furthermore, on the English→Chinese task, our system achieves a BLEU of 34.07 and 37.23 at a streamLAAL of 2216.99 and 3521.35.

2 Task Description

Simultaneous translation, generally speaking, is the process of taking in some source context and making translation decisions to another language

[1]Our fine-tuning and evaluation code is available at https://github.com/OSU-STARLAB/BeaverTalk

Proceedings of the 22nd International Conference on Spoken Language Translation (IWSLT 2025), pages 301–308

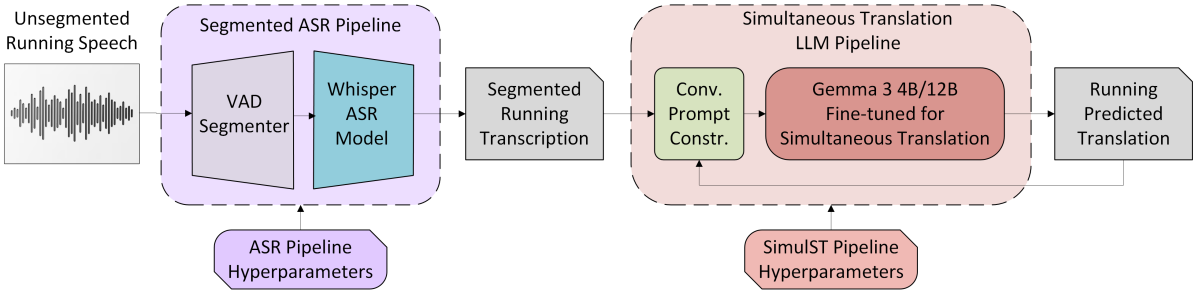

Figure 1: Depiction of the cascaded system described in this technical paper. Unsegmented source audio is taken in and fed into an ASR pipeline that segments the audio and then transcribes it into a hard text data modality. This segmented, running transcription is then fed into a simultaneous translation pipeline powered by Gemma 3. The transcription and current translation are fed into a conversational prompt constructor, adapted from prior work.

in a manner that does not rely on that source context being complete. For example, typical neural machine translation (NMT) might act on a sentence-to-sentence basis, taking in a source sentence and outputting a target sentence. In comparison, a simultaneous translation must balance the lagging factor of output translations (i.e., the time it takes from a piece of source context to a corresponding piece of the output translation) with translation quality, making translation decisions with only partial context.

As previously mentioned, the IWSLT 2025 simultaneous translation task (Abdulmumin et al., 2025) is fundamentally a speech-to-text task with two tracks governing what systems participants are expected to build: text-to-text where participants only construct a simultaneous agent for text data and prepend this system with an ASR model and speech-to-text where the simultaneous system takes in raw speech and outputs target translations in text without the need for a conversion to a text data modality. Our constructed system targets the text-to-text track, and since it is applied to the English→German (en→de) and English→Chinese (en→zh) language directions, it is restricted to predefined high and low latency regimes specified by the task. These two latency regimes, as specified below, are governed by non-computationally aware StreamLAAL in seconds (s):

- en→de: 0-2s (low), 2-4s (high);
- en→zh: 0-2.5s (low), 2.5-4s (high).

The required development set for en→de and en→zh is ACL 60/60. A blind test set is employed for final evaluations.

3 BeaverTalk: A System Description

Our simultaneous translation system consists of a cascaded architecture, which is divided into a VAD segmenter utilizing a Silero VAD model (Team, 2024), Whisper Large V2 (Radford et al., 2022), and a fine-tuned Gemma 3 (Team et al., 2025) for simultaneous translation. In the system, Gemma 3 was fine-tuned for a conversational prompting strategy (Wang et al., 2024), which is designed to mimic a streaming setting. Our complete system is provided in Figure 1.

Our choice of a cascaded architecture rather than an end-to-end system hinges on our desire to (1) leverage the language modeling capabilities of an LLM to overcome contextual obstacles faced during simultaneous translation, (2) take advantage of an LLMs context understanding capabilities to harness prior sentence context in a stream of data, and (3) benchmark the fine-tuning and multilingual capabilities of the recent Gemma 3 (Team et al., 2025). To provide a deeper understanding of our designed system, we will first explain the fine-tuning approach for our translation LLM to enable conversational prompting, followed by a deeper explanation of our cascaded translation system.

3.1 SFT Conversational Prompting

We conduct supervised fine-tuning (SFT) for our Gemma 3 LLM for translation using a conversational prompting strategy that leverages a prior sentence memory bank as context. The prompting strategy is designed whereby provided a source sequence $S = [s_1, s_2, ..., s_{|S|}]$ and a target sequence $T = [t_1, t_2, ..., t_{|T|}]$ the prompt will interleave subsequences from S and T leveraging delimiting tokens to separate the subsequences. Now suppose we had prior sentence context $C = [c_1, c_2, ..., c_{|C|}]$,

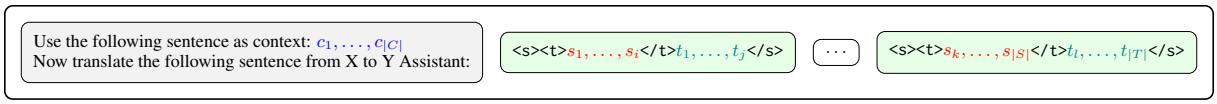

Figure 2: An example conversational prompt for translating for source language X to target language Y using source and target sequences $S = [s_1, s_2, ..., s_{|S|}]$ and $T = [t_1, t_2, ..., t_{|T|}]$ with context $C = [c_1, c_2, ..., c_{|C|}]$.

then an example of a conversational prompt constructed from these components is provided in Figure 2.

Aside from the prior sentence memory bank, which we inject into our prompt, our conversational prompting follows a similar implementation to Wang et al. (2024). The approach for generating this conversational prompting (the green region in Figure 2) can be broken into the following three steps:

1. Generate the alignments between words in the source and the target sequences. Unlike Wang et al. (2024), which uses fast-align (Dyer et al., 2013), we use the Itermax method from the SimAlign toolkit leveraging XLM-RoBERTa base to align words due to their work reporting better alignments (Jalili Sabet et al., 2020; Conneau et al., 2019).

2. Segment the graph into subsequences such that all the word dependencies for each target subsequence are available in or before the respective source subsequence. For example, assuming we did not perform step 3, in Figure 2 every word in the subsequence $t_1, ...t_j$ aligns with each word in $s_1, ..., s_i$.

3. Merge and shift subsequences to break the ideal alignments. Such a step is necessary to aid in making the LLM flexible for different variations of subsequences received during inference.

Once the prompt is constructed, we fine-tune our LLM using a causal language modeling objective using cross-entropy loss. We ensure that loss is only computed for tokens between the delimiting tokens </t>, not inclusive, and </s>, inclusive. Suppose our conversational prompt possesses K conversation intervals (ie. the number of times<s> appears in the prompt of Figure 2), where the beginning and end of each conversation interval are at index s_k and e_k. Then we can represent such a loss objective with Equation 1.

$$\mathcal{L} = \sum_{k=1}^{K} \sum_{i=s_k}^{e_k} \log p_\theta(t_i \mid s_{<i}) \qquad (1)$$

The purpose of such a loss is to ensure that the model learns to predict </s> whenever it has insufficient context at inference. In doing so our LLM learns a portion of the decision policy in conjunction with the translation objective.

3.2 Streaming Cascaded SimulST System

As we leverage a cascaded architecture we will break our explanation into (1) the Segmented ASR Pipeline, the part responsible for segmenting and transcribing a speech stream (shown in the left half of Figure 1), and (2) the Simultaneous Translation LLM Pipeline, the part responsible for translating the transcribed speech (shown in the right half of Figure 1).

3.2.1 Segmented ASR Pipeline

The first part of our Segmented ASR Pipeline is the VAD segmenter. As previously mentioned, it segments a speech signal. This segmentation is based on (1) the maximum segment duration, (2) the maximum unvoiced duration, and (3) the voice probability threshold. As the name implies, the maximum segment duration determines the maximum length of a valid segment. If the segment duration exceeds the maximum segment duration, it is cut. The maximum unvoiced duration and the voice probability threshold alternatively rely on one another to determine when to segment the speech input prior to reaching the maximum segment duration. The first part of this second facet of segmentation begins with the Silero VAD model (Team, 2024). This VAD model outputs a probability score of a specific sample containing audio of a speaker's voice. If the score falls below the voice probability threshold, it is determined that there is currently no voice from the speaker. When the score has been below the probability threshold for longer than the maximum unvoiced duration, the speech is segmented. Such a condition would be ideally met between pauses in speech or at the end of a sentence.

The Whisper ASR model (Radford et al., 2022) interacts with the VAD segmenter by receiving the segmented audio inputs. The Whisper portion of

the Segmented ASR Pipeline is designed to have Whisper transcribe the audio input using a stable transcription policy, leveraging a context mechanism. The stable transcription policy followed aims to create consistent, accurate transcriptions. It works by committing a transcription to a stable transcription buffer once it repeats a transcription for a given audio interval. For example, if on the first interval Whisper transcribes the sequence s_1, s_2, s_3, s_4 and then on the second interval it transcribes $s_1, s_2, s_3, s'_4, s_5, s_6$, only the s_1, s_2, s_3 will be committed to the stable transcription. Once committed as a stable transcription, it becomes available to the Simultaneous Translation LLM Pipeline for translation. To further improve transcription quality, Whisper is also provided with additional context from a context buffer. The context buffer is designed to provide Whisper with the transcript from the previous segment. However, if the previous segment exceeds the cutoff threshold, the number of context words is limited to be equal to the cutoff threshold.

3.2.2 Simultaneous Translation LLM Pipeline

As previously explained in Section 3.1, our Gemma 3 (Team et al., 2025) based simultaneous translation model follows a conversational prompting strategy utilizing a prior sentence memory bank as context. It is designed to firstly place the running stream of transcription chunks from the Segmented ASR Pipeline into a buffer upon receipt. Such a buffer will retain already translated portions of the transcript so long as the sentence these translated portions are associated with has yet to be completed.

Once the buffer has been extended, it is passed to a Spacy sentence tokenizer, which splits the buffer of words into sentences. Upon splitting the buffer into sentences, the pipeline will enter a translation generation loop, where a translation action will occur if one of two conditions is met. These conditions consist of (1) the length of untranslated words in the buffer has exceeded a prespecified minimum chunk size or (2) the sentence tokenizer has split the buffer into more than 1 sentence.

Once a translation action is triggered, the first step is to construct a conversational prompt. The conversational prompt is constructed identically to the one in Figure 2 by appending the new source subsequence from the oldest sentence in the buffer after a <t> delimiter following the previous translation action conversational prompt. In order to ensure the model understands it is a conversation phase after the source sequence, the </t> delimiter is appended. The new source subsequence length is equal to the untranslated word count present in the current sentence. We require such a condition as allowing for multiple sentences in the source subsequence would deviate from the fine-tuning setting, where only a single sentence was allowed at any time in the conversational prompt (a restriction by the dataset). Once constructed, the prompt is provided to the LLM to produce a translation until it outputs the delimiting token </s>. The output translation is added to a running translation to be reused in prompt construction for subsequent translation phases.

Upon completing the translation, if the sentence tokenizer determined there was more than a single sentence in the buffer, it would signify that the current sentence had completed translation. As such, the translated sentence transcript is cached to be used as context for the subsequent sentence. Additionally, its contents would be removed from the transcription buffer. Further translation phases would occur if conditions (1) and (2) were once again met.

4 Experimental Setup

The dataset of choice for fine-tuning is OpenSubtitles v2018. This corpus is particularly noisy (e.g. some Chinese translations are almost entirely English, mismatched translations to transcriptions, etc.), rendering it difficult to achieve reasonable initial results. Given that, some cleaning of the dataset is required. This occurs in four steps, the third of which only occurs for the en→zh dataset split:

1. Filtering all samples on length such that the source and memory bank sequence are greater than or equal to 25 characters.

2. Filtering all samples with '...', '[', ']', '(', ')', or consisting of only capital letters and replacing '- ' with empty space.

3. Filtering all samples in the en→zh language split that contain English words in the target column.

4. Filtering all remaining samples via CometKiwi (Rei et al., 2022) with a thresholding score of 0.6 to ensure semantic similarity between the transcription and

Table 1: Comparison table for simultaneous translation experiments, organized by language pair and model size.

Language Pair	Model Size	Latency Regime	BLEU ↑	StreamLAAL ↓
en–de	4B	low	23.64	1958.22
		high	25.22	3503.46
	12B	low	24.64	1837.86
		high	27.83	3343.73
en–zh	4B	low	32.81	2249.32
		high	34.62	3190.68
	12B	low	34.07	2216.99
		high	37.23	3521.35

the reference translation. This is meant to minimize the likelihood of a mismatched transcription and translation.

The fine-tuning pipeline that employs the aforementioned dataset is based on frameworks for simultaneous translation with LLMs provided in prior work (Agostinelli et al., 2024; Raffel et al., 2024), which were then adapted for unsegmented fine-tuning and evaluation.

Fine-tuning occurred on Gemma 3 4B/12B via LoRAs with quantization (Hu et al., 2021; Dettmers et al., 2023). The LoRA adapters were applied to all attention projections and all the feed-forward network linear projections. We chose a LoRA r of 64, a LoRA α of 16, and a LoRA dropout of 0.1. Our quantization quantized to 4-bit floating point via NormalFloat with a compute data type of bfloat16. We used the Paged 32-bit Adamw optimizer and an inverse sqrt learning rate scheduler with an effective batch size of 64, a learning rate of $2e^{-4}$, a weight decay of 0.1, a max gradient norm of 1, and a warm-up ratio of 0.03.

We evaluate the translation quality of our models using BLEU score with sacreBLEU (Post, 2018). The latency is reported using StreamLAAL (Papi et al., 2024). For the en→de language direction, the latency and BLEU scores are reported at the word level using the 13a tokenizer. Alternatively, for the en→zh language direction, the latency and BLEU scores are reported at the character level.

Our fine-tuning and evaluation for the Gemma 12B models was conducted on an NVIDIA H200. Alternatively, the Gemma 4B models were trained on a NVIDIA A40 and evaluated on a NVIDIA V100.

5 Results

5.1 Inference Hyperparameter Tuning

For our system, we tuned the maximum unvoiced duration (MUD), the voice probability threshold (VPT), the maximum segment duration (MSD), and the minimum chunk size (MCS). We immediately found that a maximum unvoiced duration greater than 0.1 s would deteriorate performance, so we kept that constant for our experimentation. Due to our 12B Gemma 3 model requiring an H200 to evaluate (a byproduct of memory requirements), we selected inference hyperparameters using a 4B Gemma 3 model, which could run on a V100. Such a choice is feasible due to the cascaded structure of our architecture. This is a byproduct of the maximum unvoiced duration, voice probability threshold, and maximum segment duration only influencing the quality of the transcriptions from the Segmented ASR Pipeline. If the transcription related hyperparameters are tuned properly, the Gemma 3 translation model will perform better irrespective of the model size.

We began our inference hyperparameter search by fixing our translation minimum chunk size to 3 words on the en→de language pair and 5 words on the en→zh language pair. These minimum chunk sizes were chosen to accommodate the en→de language pair, having a low latency cutoff of 2s, and the en→zh language pair, having a low latency cutoff of 2.5s. We then iteratively searched for the optimal maximum segment duration and voice probability threshold for both the low and high latency regimes. We report these results in Tables 2 and 3 for en→de and en→zh language pairs, respectively.

From observing, Table 2 we can see for the low latency regime of the en→de language pair the only selection of hyperparameters that fall below the 2s threshold is with a voice probability threshold of

Table 2: Simultaneous translation results for **en→de** organized by voice probability threshold (VPT) and maximum segment duration (MSD) with a minimum chunk size of 3.

VPT	MSD	BLEU ↑	StreamLAAL ↓
0.1	0.5	23.43	2079.14
0.1	1	24.86	2677.30
0.1	1.5	25.02	3114.15
0.3	0.5	23.36	2047.62
0.3	1	25.01	2477.67
0.3	1.5	25.05	2877.81
0.5	0.5	23.59	1940.58
0.5	1	24.96	2356.68
0.5	1.5	24.82	2804.01

Table 3: Simultaneous translation results for **en→zh** organized by voice probability threshold (VPT) and maximum segment duration (MSD) with a minimum chunk size of 5.

VPT	MSD	BLEU ↑	StreamLAAL ↓
0.1	0.5	33.16	2294.95
0.1	1	33.57	2979.20
0.1	1.5	34.11	3382.83
0.3	0.5	32.47	2307.56
0.3	1	33.36	2851.60
0.3	1.5	33.70	3206.28
0.5	0.5	32.81	2249.32
0.5	1	33.31	2728.01
0.5	1.5	34.62	3190.68

0.5 and a maximum segment duration of 0.5s. Alternatively, for the high latency regime, we select a voice probability threshold of 0.3 and a maximum segment duration of 1. We chose the maximum segment duration of 1s rather than 1.5s due to the increase in StreamLAAL. We report our final selected maximum unvoiced duration, voice probability threshold, and maximum segment duration in Table 4 for the en→de language pair.

Table 4: Inference hyperparameters for **en→de**.

Latency	MUD	VPT	MSD	MCS
low	0.1	0.5	0.5	3
high	0.1	0.3	1	7

On the high latency regime of the en→zh from observing Table 3 we chose a voice probability threshold of 0.5 with a maximum segment duration of 1.5s due to the high BLEU achieved. Then, for the low-latency regime, we chose a voice probability threshold of 0.5 and a maximum segment duration of 0.5 to align with our high-latency regime. We report our final selected maximum unvoiced duration, voice probability threshold, and maximum segment duration in Table 5 for the en→zh language pair.

Table 5: Inference hyperparameters for **en→zh**.

Latency	MUD	VPT	MSD	MCS
low	0.1	0.5	0.5	5
high	0.1	0.5	1.5	7

Using the optimal maximum unvoiced duration, voice probability threshold, and maximum segment duration from Tables 4 and 5 of our previous search, we iteratively step through a minimum chunk size of 1, 3, 5, and 7. We report the results for the BLEU and StreamLAAL for each given chunk size for both language pairs at the low and high latency regimes in Figure 3. Our final selected minimum chunk size for each latency regime is reported in Tables 2 and 3.

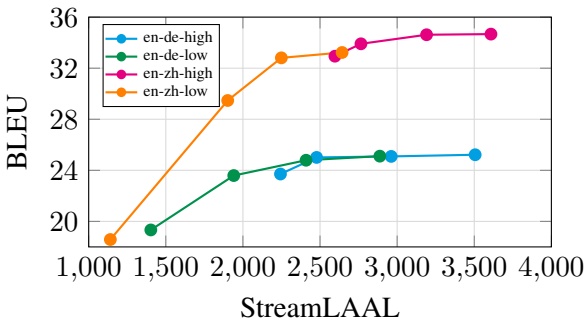

Figure 3: BLEU score plotted against StreamLAAL on the **en→de** and **en→zh** language pairs for minimum chunk sizes of 1, 3, 5, 7.

5.2 Quality and Latency Results on Dev set

We provide the quality and latency of our system in Table 1 on the ACL 60/60 dev set for the en→de and en→zh language pairs. For each language pair, we show the influence of model size on our system's BLEU score. From the results in Table 1, we can see that increasing the model size from 4B to 12B can offer approximately a 2 BLEU increase. We choose to submit the 12B Gemma 3 translation model version of our cascaded architecture to the IWSLT 2025 simultaneous track (Abdulmumin et al., 2025). On the en→de language pair, we achieve a BLEU of 24.64 and 27.83 on the low and high latency regimes. Then on the en→zh language

pair, we achieve a BLEU of 34.07 and 37.23 on the low and high latency regimes.

6 Conclusion

In this paper, we provide Oregon State University's system, BeaverTalk, designed for the low and high latency regimes of the en→de and en→zh language pairs as a part of the IWSLT 2025 simultaneous track. Our system consists of a cascaded architecture composed of a VAD speech segmenter, a Whisper ASR model, and a Gemma 3 translation LLM using conversational prompting. We provide an extensive inference hyperparameter search for our system and demonstrate its performance utilizing a 4B and 12B translation LLM. Our final submitted model, composed of the 12B translation LLM, demonstrates strong results on the en→de and en→zh language pairs for both the low and high latency categories.

Acknowledgements

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

References

Idris Abdulmumin, Victor Agostinelli, Tanel Alumäe, Antonios Anastasopoulos, Ashwin, Luisa Bentivogli, Ondřej Bojar, Claudia Borg, Fethi Bougares, Roldano Cattoni, Mauro Cettolo, Lizhong Chen, William Chen, Raj Dabre, Yannick Estève, Marcello Federico, Marco Gaido, Dávid Javorský, Marek Kasztelnik, and 30 others. 2025. Findings of the iwslt 2025 evaluation campaign. In *Proceedings of the 22nd International Conference on Spoken Language Translation (IWSLT 2025)*, Vienna, Austia (in-person and online). Association for Computational Linguistics. To appear.

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.

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. *CoRR*, abs/1911.02116.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115.

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.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

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.

Pierre Lison, Jörg Tiedemann, and Milen Kouylekov. 2018. OpenSubtitles2018: Statistical rescoring of sentence alignments in large, noisy parallel corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Sara Papi, Marco Gaido, Matteo Negri, and Luisa Bentivogli. 2024. Streamatt: Direct streaming speech-to-text translation with attention-based audio history selection. *Preprint*, arXiv:2406.06097.

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

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *arXiv preprint*.

Matthew Raffel, Victor Agostinelli, and Lizhong Chen. 2024. Simultaneous masking, not prompting optimization: A paradigm shift in fine-tuning LLMs for simultaneous translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18302–18314, Miami, Florida, USA. Association for Computational Linguistics.

Ricardo Rei, Marcos Treviso, Nuno M Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José GC De Souza, Taisiya Glushkova, Duarte M Alves, Alon Lavie, and 1 others. 2022. Cometkiwi: Ist-unbabel 2022 submission for the quality estimation shared task. *arXiv preprint arXiv:2209.06243*.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard

Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Silero Team. 2024. Silero vad: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier. https://github.com/snakers4/silero-vad.

Viet Anh Khoa Tran, David Thulke, Yingbo Gao, Christian Herold, and Hermann Ney. 2022. Does joint training really help cascaded speech translation? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4480–4487, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

Giulio Zhou, Tsz Kin Lam, Alexandra Birch, and Barry Haddow. 2024. Prosody in cascade and direct speech-to-text translation: a case study on korean wh-phrases. *arXiv preprint arXiv:2402.00632*.