Learning When to Speak: Latency and Quality Trade-offs for Simultaneous Speech-to-Speech Translation with Offline Models

Liam Dugan[1,2], *Anshul Wadhawan*[1], *Kyle Spence*[2], *Chris Callison-Burch*[1], *Morgan McGuire*[2], *Victor Zordan*[2]

[1]University of Pennsylvania, [2]Roblox

{ldugan, anshulw, ccb}@seas.upenn.edu, {kspence, vbzordan, morgan}@roblox.com

Abstract

Recent work in speech-to-speech translation (S2ST) has focused primarily on offline settings, where the full input utterance is available before any output is given. This, however, is not reasonable in many real-world scenarios. In latency-sensitive applications, rather than waiting for the full utterance, translations should be spoken as soon as the information in the input is present.

In this work, we introduce a system for simultaneous S2ST targeting real-world use cases. Our system supports translation from 57 languages to English with tunable parameters for dynamically adjusting the latency of the output—including four policies for determining when to speak an output sequence. We show that these policies achieve offline-level accuracy with minimal increases in latency over a Greedy (wait-k) baseline. We open-source our evaluation code and interactive test script to aid future SimulS2ST research and application development.

Index Terms: Simultaneous Speech-to-Speech Translation

1. Introduction

Speech-to-Speech Translation (S2ST) is a popular task that has seen significant recent advancements in end-to-end [1] and textless scenarios [2]. However, the task of adapting such models to simultaneous applications has been relatively under-explored despite the clear practical applications like cross-lingual voice chat and live interpretation.

While many such applications would benefit from simultaneous S2ST, end-to-end and textless approaches remain challenging especially for low-resource languages [3]. In our work, we bypass these current research challenges by developing a robust cascaded SimulS2ST system. Our system consists of a SimulST (source speech to target text) component and a text-to-speech (TTS) component (target text to target speech).

Inspired by Papi et al. [4], we use an off-the-shelf offline ST model (OpenAI's Whisper [5]) and query it in an online fashion to produce accurate translations with low latency instead of training a model specifically for SimulST. We evaluate the trade-off between latency and quality for four prototyped policies for determining when to speak a given output utterance. We release our multi-threaded pipeline code and evaluations to aid future research and development in SimulS2ST[1].

2. System Design

In Figure 1, we show a diagram of our system. At a high level, the system works as follows: Let $S = s_1 \dots s_N$ be the input sequence of N speech frames. On each iteration we retrieve a chunk of size $w < N$ ($s_c \dots s_{c+w}$) from the input and append it to the frame buffer $F = s_f \dots s_{f'}, s_{f'+1} \dots s_{f'+w}$. We then give the frame buffer and current spoken transcription $T = t_1 \dots t_t$ as input to the ST model and generate the output text sequence $\hat{T} = t_{t+1} \dots t_p$. This output is given to the policy $\mathcal{P}$ which determines whether or not we should speak the sequence. If so, we use the TTS model to speak $\hat{T}$, append $\hat{T}$ to T, and clear the frame buffer. Otherwise, we discard $\hat{T}$ and wait for the next chunk of input speech.

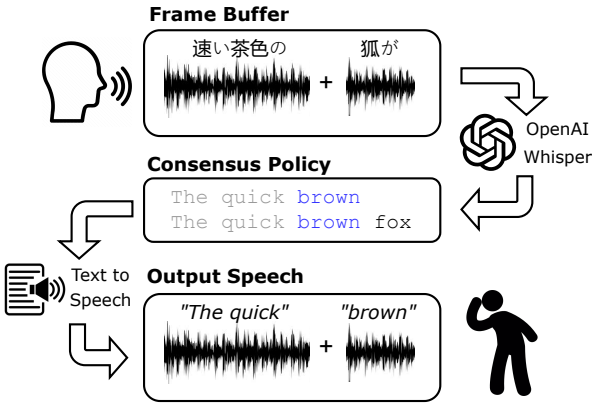

Figure 1: *An example of our cascaded SimulS2ST system. Speech segments from the frame buffer are passed to Whisper. The translation is output according to the selected policy.*

We implement the SimulST and TTS model on separate threads to prevent unnecessary execution dependencies and allow for lower output latency. Additionally, when accepting live microphone input, the recorder is also given a separate thread that runs in the background during processing. Our pipeline is written in a modular fashion to allow users to quickly prototype different policies and observe their effects on the latency and accuracy of the output.

3. System Evaluation

We evaluate our system on translation into English from four typologically diverse languages (Japanese, Spanish, Russian, and Arabic). For our offline ST model we use OpenAI's Whisper [5] and for our TTS model we query the ElevenLabs API [6]. We conduct our evaluations on a single NVIDIA RTX 2080 GPU and report metrics on a filtered subset of 75 examples of length 6 seconds or more from the dev set of CoVoST2 [7].

For accuracy, we compute the BLEU score between the spoken transcript T and the reference using the SacreBLEU

[1]http://github.com/liamdugan/speech-to-speech

Window Size (t)	Japanese → English		Spanish → English		Russian → English		Arabic → English	
	1s	2s	1s	2s	1s	2s	1s	2s
CAP ($\gamma = 0.9$)	20.7 (7.2)	21.1 (8.6)	38.3 (6.4)	41.5 (7.6)	36.6 (8.1)	37.4 (9.0)	18.8 (6.9)	19.5 (7.6)
CAP ($\gamma = 0.5$)	15.1 (4.9)	**18.8 (6.7)**	**25.8 (3.5)**	28.2 (5.4)	28.0 (4.1)	**31.8 (5.1)**	11.7 (4.1)	**13.7 (4.7)**
CP ($\alpha = 0.75$)	17.2 (6.8)	21.3 (9.6)	31.1 (4.6)	41.4 (7.9)	**27.2 (4.0)**	37.0 (8.5)	16.4 (6.2)	19.5 (8.7)
CP ($\alpha = 0.5$)	**15.3 (4.8)**	20.4 (8.8)	25.3 (4.1)	**35.6 (6.1)**	21.6 (3.3)	31.3 (6.2)	**11.0 (3.7)**	19.0 (7.3)
Greedy (wait-k)	5.7 (3.2)	10.0 (4.9)	8.6 (2.5)	15.2 (4.1)	12.8 (2.4)	17.3 (3.0)	3.8 (1.8)	5.9 (3.5)
Offline Policy	21.9 (9.5)	21.9 (9.5)	42.9 (9.6)	42.9 (9.6)	36.6 (10.0)	36.6 (10.0)	19.7 (9.1)	19.7 (9.1)

Table 1: *Performance scores - BLEU (Average Lagging in seconds) - for the four policies: Offline, Greedy, Confidence-Aware (CAP), and Consensus (CP) using Whisper Medium (769M params) [5]. Bolded numbers represent optimal latency-quality trade-off. While there is no consistent winner between CAP and CP, we see that languages more structurally similar to English consistently achieve better latency-quality trade-off. In particular, Spanish and Russian BLEU scores achieve near-Offline levels with minimal increases in latency over the Greedy policy.*

package [8]. We opt not to use ASR BLEU on the output speech due to the lack of precision caused by cascading ASR errors.

For our latency metric, we use Computation-Aware Average Lagging (AL_{CA}) [9] which is an adaptation of Average Lagging that takes into account the computational cost of generating an output. This is especially important for us given that we also bear a computational cost when speaking output and context switching between threads.

3.1. Policy Comparisons

We compare four simple policies for determining when to speak outputs. The first two are the baseline policies: the **Greedy Policy**[2] ($\mathcal{P}(\hat{T})$ = True) and **Offline Policy** ($\mathcal{P}(\hat{T})$ = False). These represent two ends of the latency-quality spectrum and help us gauge where the upper and lower bounds are for our system's quality and latency.

Next is the **Confidence-Aware Policy (CAP)**, where we return true if the average probability of the sequence returned by the ST model (i.e. the confidence of the model) is above some threshold γ. In our testing, we found that Whisper's `no_speech_prob` field gave better empirical results than the `avg_logprob` field, so we use that in our evaluation.

Finally, the **Consensus Policy (CP)** returns true only when the current transcript $\hat{T}_{k+1}$ and the previous transcript $\hat{T}_k$'s ratio of length to edit distance is under some threshold α. We run both this policy and the previous policy for two different parameter settings to show the capability of our system to adapt to different latency regimes.

4. Results

In Table 1 we report the results of our evaluation and find several surprising trends. First, we observe that languages more structurally similar to English tend to achieve lower latencies across all policies. In particular, we see that translations from Spanish are spoken almost a full second faster than translations from Japanese no matter the choice of policy.

Second, using our proposed policies we achieve substantial improvements in translation quality over our Greedy baseline while only sacrificing minimal extra latency. For example, translating Spanish to English using the Confidence-Aware Policy ($\alpha = 0.5$) achieves a 17-point improvement in BLEU score over wait-k with only 1 extra second of average lagging. Likewise in Russian, using the Consensus Policy ($\gamma = 0.75$) increases the BLEU score by nearly 15 points over wait-k while increasing the latency by just 1.6 seconds.

Finally, we observe that no single policy performs the best across all languages. Thus practitioners must tune their systems on a per-language basis for optimal latency-quality tradeoffs.

[2]This is equivalent to the wait-k policy from SimulST literature [9]

5. Conclusion

While SimulS2ST is still an active area of exploration, it already has significant practical utility for enhancing communication. We provide a customizable baseline system for this task that allows users to dynamically tune policy parameters, directly influencing the latency-quality trade-off of their system. Our evaluations show that these policy parameters achieve comparable accuracy to offline models while substantially improving latency over a wait-k baseline. We hope our system assists industry professionals and researchers alike in developing, benchmarking, and prototyping future SimulS2ST systems.

6. References

[1] Y. Jia, M. T. Ramanovich, T. Remez, and R. Pomerantz, "Translatotron 2: High-quality direct speech-to-speech translation with voice preservation," in *International Conference on Machine Learning*. PMLR, 2022, pp. 10 120–10 134.

[2] A. Lee, H. Gong, P.-A. Duquenne, H. Schwenk, P.-J. Chen, C. Wang, S. Popuri, Y. Adi, J. Pino, J. Gu, and W.-N. Hsu, "Textless speech-to-speech translation on real data," 2022.

[3] H. Inaguma, S. Popuri, I. Kulikov, P.-J. Chen, C. Wang, Y.-A. Chung, Y. Tang, A. Lee, S. Watanabe, and J. Pino, "Unity: Two-pass direct speech-to-speech translation with discrete units," 2022.

[4] S. Papi, M. Gaido, M. Negri, and M. Turchi, "Does simultaneous speech translation need simultaneous models?" *arXiv preprint arXiv:2204.03783*, 2022.

[5] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022.

[6] ElevenLabs, "Elevenlabs API," https://api.elevenlabs.io/docs, 2023, accessed: 2023-04-10.

[7] C. Wang, A. Wu, and J. Pino, "Covost 2 and massively multilingual speech-to-text translation," 2020.

[8] M. Post, "A call for clarity in reporting BLEU scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, Oct. 2018, pp. 186–191. [Online]. Available: https://www.aclweb.org/anthology/W18-6319

[9] X. Ma, J. Pino, and P. Koehn, "Simulmt to simulst: Adapting simultaneous text translation to end-to-end simultaneous speech translation," *arXiv preprint arXiv:2011.02048*, 2020.