Efficient and Direct Duplex Modeling for Speech-to-Speech Language Model

Anonymous preprint

Abstract

Spoken dialogue is the most intuitive form of human-computer interaction, yet current speech language models often remain constrained to turn-based exchanges, lacking real-time adaptability such as user barge-in. We propose a novel duplex speech to speech (S2S) architecture featuring continuous user inputs and codec agent outputs with channel fusion that directly models simultaneous user and agent streams. Using a pretrained streaming encoder for user input enables the first duplex S2S model without requiring speech pretrain. Separate architectures for agent and user modeling facilitate codec fine-tuning for better agent voices and halve the bitrate (0.6 kbps) compared to previous works. Experimental results show that the proposed model outperforms previous duplex models in reasoning, turn-taking, and barge-in abilities. The model requires significantly less speech data, as speech pretrain is skipped, which markedly simplifies the process of building a duplex S2S model from any LLMs. Finally, it is the first openly available duplex S2S model with training and inference code to foster reproducibility.

Index Terms: duplex, speech-to-speech, conversation, barge-in

1. Introduction

Large language models (LLMs) [1–4] have made significant strides in natural language processing, sparking interest in multimodal models that extend beyond text. Speech, as a natural interface for human-computer interaction, is a key part of this trend. Recent studies suggest adapting LLMs to process speech prompts for various speech-to-text (STT) tasks [2, 4–9].

While traditional systems often respond with text, speech outputs are more intuitive for human-computer interaction. Cascaded spoken dialogue systems, like AudioGPT [10], use text as an intermediate representation, involving sequential modules such as ASR, LLM, and TTS. However, these systems face drawbacks like high latency, lack of interactive behaviors, and loss of paralinguistics. To address these issues, research has shifted towards end-to-end speech-to-speech (S2S) modeling.

Previous S2S models focus on half-duplex, turn-based interactions. For instance, SpeechGPT [11], initialized from LLaMA, undergoes sequential fine-tuning on speech-only data and multimodal instruction sets to handle spoken question-answer (QA) tasks. Similarly, USDM [12] extends Mistral's pretraining with interleaved speech-text data for enhanced multimodal understanding. GLM-4-voice [13] efficiently tokenizes speech using one codebook and large-scale speech-text pretraining for downstream tasks like ASR, TTS, and SQA.

Several pioneering or concurrent full-duplex S2S models have been recently proposed [14–17]. However, these systems face increased complexity in model, data, and computation, which hinders their widespread research and adoption. The introduction of additional submodules for turn-taking between user and agent increases system complexity and reduces the end-to-end nature of the models. Moreover, the extensive speech-text pretraining required on top of the LLM backbone is resource-intensive and limits scalability to any LLMs. Finally, using codecs to model user and agent interactions simultaneously necessitates a delicate balance between speech perception and generation, presenting another significant challenge.

To tackle the above problems, we propose a novel duplex S2S system with the following contributions: 1) A novel duplex S2S architecture featuring continuous user inputs and codec agent outputs with channel fusion that directly models simultaneous text and speech of both the user and agent. 2) We demonstrate several key advantages over existing duplex models: The use of a pretrained encoder as input enables the first duplex S2S model **without speech pretraining** requirement; As the agent and user are modeled by the codec and the pretrained encoder separately, this facilitates **codec fine-tuning** toward better agent voices. 3) We propose a set of systematic metrics to evaluate conversational behaviors such as turn-taking and barge-in. Finally, it is the first **open duplex S2S model** with both training and inference code pubicly available to foster reproducibility.

2. Related Work

Interest in full-duplex S2S models has grown in the past year. Key challenges here include handling simultaneous user and agent streams and enabling turn-taking. Systems like [16, 18, 19] model single-channel interactions but use external signals, such as stopping commands [19] or submodules [16], to decide when to respond. Models like SyncLLM [20] and OmniFlatten [17] achieve full-duplex conversation by employing time chunking methods, embedding time information into LLMs for synchronization. This interleaving processing allows the model to handle user inputs like barge-in with low latency.

Our duplex S2S model is trained without speech-text pretraining, unlike [14]. In multi-turn conversation, we align text and speech at the turn level, which simplifies data preparation compared to word-level alignment. Compared to [15, 18, 19], our model predicts text and speech simultaneously without requiring an explicit TTS component. Our speech codec model uses parallel codebooks (see details in Sec. 3.2) and enables speech generation with minimal latency. Our design further enables codec fine-tuning for improved agent voices while halving the required bitrate of previous works (0.6 kbps).

3. Model Architecture

To achieve duplex behavior, our S2S model takes two input streams simultaneously: user speech stream, and agent speech and text stream. As shown in Fig. 1, the user speech is first

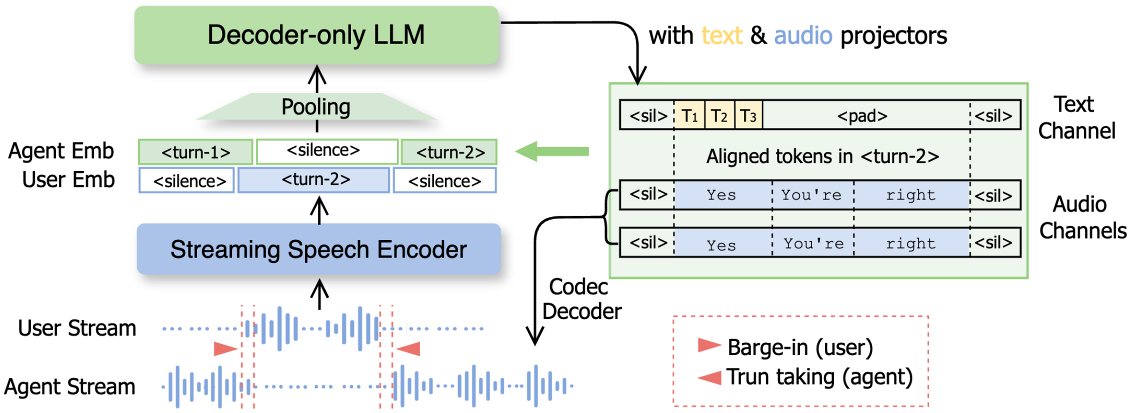

Figure 1: *The proposed duplex S2S model* ***without*** *requiring speech-text pretraining. Our model includes a streaming speech encoder, a personalized codec, and an LLM. The model is trained to predict both text and audio channels in parallel with turn-level alignments.*

encoded to generate continuous embeddings by the speech encoder using an 80-ms frame rate. We use a 100M streaming speech encoder from a CTC model [21]. We initialize the backbone LLM using the TinyLlama-1.1B-chat model [22]. A modality adapter is used between the speech encoder and the text LLM. To obtain the agent embeddings in training, we use a codec model [23] to generate 12.5 Hz speech codes for the agent speech. LLM vocabulary is extended to include extra tokens from speech codec with zero initialization. The two inputs are time-aligned and summed as the input to the text LLM (similar to [24]). Both our speech encoder and text LLM are causal and thus streaming. In training, we fine-tune both the speech encoder and the backbone LLM. Text and speech loss are weighted differently in training (see Sec. 5.1). Our model is trained by multi-channel next token prediction similar to [1].

3.1. Simultaneous Agent Text and Speech Prediction

As shown in Fig. 1, we encode speech using 4 codebooks at a rate of 12.5 frames per second [23], and text targets are tokenized into a separate channel. We align the text and speech tokens at the turn level based on their start time. We prepend separate `<BOS>` tokens for text and speech at the beginning of the turn and append `<EOS>` at the end of the turn. The gap between text and speech tokens are padded by text pad ID. We also tried word-level alignment between text and speech as in [14] and did not find improvement. Empirically, we find that the model tends to learn agent text first. Therefore, we introduce a small delay (i.e., one token) to the speech channels for improved speech quality without introducing significant latency.

3.2. Personalization-friendly Speech Tokenization

We employ a partially causal neural audio codec to transform raw speech signals into streaming tokenized representations. Given an audio signal $\mathbf{a}$, the codec generates a two-dimensional acoustic matrix, $\mathbf{C}_{T\times N} = \mathit{CodecModel}(\mathbf{a})$, where T denotes the downsampled sequence length, and N represents the number of codebooks per timestep. Each element in $\mathbf{C}_{T\times N}$ is an m-bit discrete code. We adopt the state-of-the-art NanoCodec [23], which achieves reasonable-quality audio compression at 0.6 kbps with a frame rate of 12.5 frames per second, employing $N = 4$ independent codebooks. The codec leverages Finite Scalar Quantization (FSQ) [25], ensuring independence among codebooks. This independence removes the need for additional models or delay mechanisms, allowing all N codebooks to be predicted in parallel at each timestep, thereby enabling fully parallel modeling with low latency.

Our duplex design allows us to personalize the pretrained codec for agent voices to further enhance audio quality. This is enabled by modeling the agent and user separately with the speech codec and a pretrained causal speech encoder. In the experimental section, we will evaluate the benefits of speech and reasoning quality resulting from codec personalization.

Table 1: *Synthetic training data with multi-turn and barge-in.*

Task	Dataset	#Hours	Speech	Multi-turn	Barge-in
Spoken QA	ASR-QA	20k	Mix	Augment	×
	MS MARCO	0.2k	TTS	Augment	×
	Alpaca	0.2k	TTS	Augment	×
	Internal SFT	3k	TTS	Real	✓
Conv-ersation	UltraChat	3k	TTS	Augment	✓
	Topic	0.3k	TTS	Augment	✓

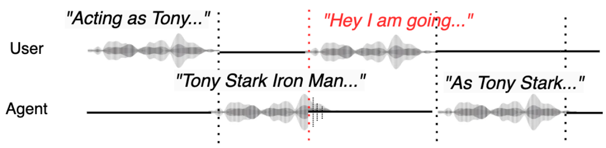

Figure 2: *Duplex training data format. Our duplex data consists of separate user and agent streams including turn taking and barge-in behavior. Here, the user barges in at the second turn.*

4. Duplex Data for Training

Table 1 summarizes our training data which can be categorized into spoken QA and multi-turn conversations.

4.1. Single-turn synthetic and real spoken QA

Our most basic training data structure consists of a single-turn spoken QA between the user and agent. We use a multi-speaker TTS model [26] to synthesize the context, questions and answers from *MS MARCO* [27] and *Alpaca* [28]. To mitigate overfitting to synthetic data, we follow [29] to create additional synthetic QA pairs using the Mixtral-8x22B LLM from English ASR-labeled data (8k public [1] and 12k in-house). This data is then synthesized using the same TTS, denoted as *ASR-QA*.

[1] A subset from the NeMo ASR set in [21].

The resultant user speech contains both TTS and real data. An evaluation set used in Sec. 5.2 is created from the public data portion. We use a fixed speaker to generate agent speech and randomly select speakers for user speech.

We create duplex training data from the aforementioned user-agent QA pairs. First, we split a pair of utterances into two streams, corresponding to the user and agent portions separately, and then insert silence into the agent stream when the user speaks, and vice versa. This gives us two streams of speech (shown as the first turn in Fig.2). This duplex structure enables the model to listen and speak simultaneously at any time. To prevent the agent from barging in unexpectedly, we insert a 0.64s silence between user and agent before the agent speaks.

4.2. Augment with Multi-turn and Barge-in

In order for the model to learn the ability for multi-turn conversation, we also create duplex data that includes two or more turns of conversation between the user and agent (e.g., Fig. 2). First, we synthesize 3k hours of duplex data from a text-based multi-turn *Internal SFT* dataset to form multi-turn spoken QA. To ensure a more conversational flow, we limit each turn of the text SFT data, which is typically very long, to under 25 seconds. Second, we augment the single-turn data from Sec. 4.1 by randomly concatenating two QA pairs from the same dataset. The multi-turn data topics focus on role-playing, daily topics, scientific topics, etc. Moreover, when creating multi-turn data, we allow the user to barge in by cutting off the agent speech. After the cutoff, we keep a small duration (0.64 s) of agent speech to account for barge-in latency, and pad the rest of the agent turn with silence. As we show in later results, this straightforward approach enables the model to learn barge-in behavior.

4.3. Conversational data

To enhance the model's conversation ability on daily topics, we create *Topic* and *UltraChat* datasets (totaling 3.3k hours as shown in Table 1). For both datasets, we first generate 4-turn text-based conversations and then synthesize them using a TTS model [26]. For *Topic*, we randomly choose a topic between user and agent and prompt the Meta-Llama-3.1-70B-Instruct model [4] to generate a conversation. The topics are randomly chosen from the everyday-conversation dataset [30], which covers 63 everyday and science topics. To generate concise replies for efficient training, we restrict the words of each turn to be 30 words in the prompt. The generated conversations are then synthesized into speech and prepared to the duplex data format. For *UltraChat*, we randomly sample a chat conversation from the *UltraChat* dataset [31] to use as contextual information in the prompt to generate a 4-turn conversation similar to *Topic*.

5. Experiment Details

5.1. Training Details

We implement the model with PyTorch using the NeMo Toolkit [32], and the model is trained on 32 A100 (80G) GPUs with a batch duration of 1000 sec per GPU. The speech encoder is initialized from a 100M streaming pretrained encoder with 80ms right context [21], and the LLM is initialized from the 1.1B TinyLlama [22]. We use a 32k SentencePiece tokenizer for text, and a personalized 0.6 kbps NanoCodec [23] for speech by default. Ablations for personalization are presented in Sec. 6.3. The speech codes have 4 channels, with a vocabulary size of 4,037 for each channel. Text and speech channel training loss are weighted by 3 and 1 respectively. We use FusedAdam, and an inverse Square Root Annealing learning rate (LR) schedule for optimization. The LR schedule starts with an initial learning rate of 3e-4 with a warm-up of 2500 steps. Gradient clipping is applied at the threshold of 1.0 to stabilize training.

5.2. Evaluation Data and Metrics

Our evaluation data consists of: 1) *multi-turn* conversations: *UltraChat*, *Roleplay* (part of *Internal SFT*), and *Topic*, and 2) *spoken QA reasoning*: *ASR-QA* and *Alpaca*. We select one shard for each dataset in Sec. 4, which is unseen during training, for this evaluation. To evaluate model performance on a more challenging scenario where the user frequently interrupts the agent, we create an evaluation set called *Impatient*. When creating *Impatient*, we halve the silence time between the current and the next user turn (from the original duration in the *ASR-QA* set) to increase the chance of the agent being interrupted by the user. By doing this, the interruption cases for our model and Moshi (more details in Sec. 6.1) in the *Impatient* dataset are as high as 95.4% and 96.7%, respectively.

In terms of evaluation metrics, we evaluate the reasoning ability of our model using GPT scores generated by `gpt-4o-mini-2024-07-18` ranging from 0 to 10 based on the hypotheses and references of all the agent turns. The reasoning quality is evaluated using the aforementioned *multi-turn* and *spoken QA reasoning* datasets. The hypotheses of agent turns are produced by transcribing the generated speech using the ASR model `nvidia/parakeet-tdt_ctc-110m`.

We evaluate turn-taking ability and speech generation quality using the *UltraChat* and *Impatient* datasets. We use two types of metrics to measure the turn-taking ability: barge-in performance and 1st response latency (see Table 2). For barge-in performance, we introduce the following metrics: 1) Barge-in latency: The time delay between the user's speech onset and the agent stopping its response; 2) Success rate: The percentage of cases where the agent successfully stops speaking within 1.5 seconds after the user interruption; and 3) False alarm rate: The frequency at which the agent incorrectly barges in while the user speaks. Additionally, if the user stops speaking within 0.1s, the event is not counted as a false alarm, as we found that Moshi tends to proactively respond. The 1st response latency is defined as the time taken by the agent to respond to the 1st user turn. To evaluate the speech quality, we compute the UTMOS [33] using the generated agent speech after removing silence.

6. Results and Comparison

6.1. Conversation and Speech Generation Quality

We first evaluate the turn-taking and speech generation quality of our model in Table 2. Compared to Moshi, our model has significantly higher barge-in success rate (94.5% v.s. 55.1%), the same false alarm rates, and lower barge-in latency (0.69s v.s. 0.81s). We observe that, in multi-turn conversations, Moshi often initiates dialogue more proactively, leading to user barge-in failures for both *UltraChat* and *Impatient*.

We cannot directly compare our 1st response latency with Moshi's as Moshi almost always responds before the user finishes talking and thus does not fit for this metric. We also note that our 1st response latency is affected by our data generation, as we always add a 0.64-second silence after the user turns to ensure no unexpected agent barge-in. Further reducing this delay is our future work. Lastly, we report UTMOS and our model generates better quality speech than Moshi by up to 0.4.

Table 2: *Comparison of turn-taking and speech generation quality.*

Dataset	Model	Barge-in Performance Success ↑	False Alarms ↓	Latency (s) ↓	1st Response Latency (s) ↓	UT MOS ↑
UltraChat	Ours	**83.0%**	0.0%	**0.52**	0.72	**4.3**
	Moshi	56.0%	0.0%	0.63	n/a	3.9
Impatient	Ours	**94.5%**	0.0%	**0.69**	0.92	**4.0**
	Moshi	55.1%	0.0%	0.81	n/a	3.8

Table 3: *Reasoning quality of multi-turn conversation and spoken QA. GT+LLM denotes an optimal cascaded system which feeds every ground-truth user turn to the LLM.*

GPT Score	Multiturn Conversation UltraChat	Roleplay	Topic	Spoken QA ASR-QA	Alpaca
Ours	**3.5**	**4.6**	**6.1**	**7.8**	**2.9**
Moshi	3.4	1.7	2.8	1.9	1.7
GT+LLM	6.4	4.9	5.5	5.8	5.0

Table 4: *Evaluation of audio reconstruction and the resultant S2S quality across different codecs.*

Codec	Bitrate kbps	Audio Reconstruction MOS↑	CER↓	SECS↑	S2S ASR-BLEU↑
Mimi[14]	1.1	4.16	3.00	0.65	n/a
Nano[23]	1.2	4.67	1.44	0.77	18.1
Nano[23]	0.6	4.54	3.55	0.57	16.2
+personalized	0.6	**4.75**	**1.36**	**0.94**	**18.7**

6.2. Reasoning Quality

In Table 3, we compare the reasoning ability of our model to Moshi [14] and an *optimal cascaded system* formed by feeding every ground-truth user turn text to LLM (i.e., GT+LLM in Table 3). The backbone of our model, TinyLlama, is used as the LLM. We report the aforementioned GPT scores on two types of test sets: *multi-turn conversation* and *spoken QA*. Compared to Moshi, our model shows better scores on all datasets despite the fact that our model uses much less data and smaller backbone. Compared to the optimal cascaded system, our model shows competitive results, better on two and worse on three sets. The slightly worse performance of end-to-end versus cascaded is not new and has been shown by other research [2, 11, 14, 29]. Future works include i) a more fair comparison with full pipeline (VAD, streaming ASR and TTS, LLM), and ii) improving the reasoning of duplex S2S models.

6.3. Speech Codec Personalization

We personalize the codec to our agent voice by fine-tuning the codec on 21k ground-truth utterances from the target speaker. The model is evaluated on 228 test samples that are not seen during training. Perceptual quality is assessed using estimated Mean Opinion Scores (MOS) with Torchaudio-Squim [34]. Intelligibility is measured by computing the Character Error Rate (CER), comparing transcriptions from the Massively Multilingual Speech (MMS) model [35] for both ground-truth and reconstructed audio. Speaker similarity is evaluated using the Speaker Encoder Cosine Similarity (SECS) [36], computed with the state-of-the-art ECAPA2 speaker encoder [37].

Table 4 presents the evaluation results for the 1.1 kbps Mimi Codec [14], 1.2 kbps, and 0.6 kbps versions of NanoCodec [23], and the proposed personalized version of 0.6 kbps NanoCodec. Personalization significantly enhances the performance of the 0.6 kbps NanoCodec. Notably, despite operating at nearly half the bitrate, our personalized codec outperforms both Mimi and NanoCodec at 1.2 kbps across all audio reconstruction metrics on the target speaker.

As an ablation study, we further train our duplex S2S models with different codecs (last three rows in Table 4). For simplicity, we report ASR-BLEU, which is calculated based on the reference agent texts and ASR transcripts of generated agent speech. Results on *ASR-QA* in Table 4 indicate that personalization enhances duplex modeling as well, leading to improved perceptual quality and higher BLEU scores.

6.4. Listening Duplex Conversation Examples

We include representative listening examples in an anonymous demo page[2]. Specifically, the following capabilities of our duplex S2S model on *unseen* data are highlighted:

Robustness with frequent interruption. In the example of Fig. 3 and the webpage, the user interrupts the agent three times in 15 seconds, and leaves limited time for the agent to respond. Despite these challenges, the agent still demonstrates robust conversational behavior in handling frequent barge-in.

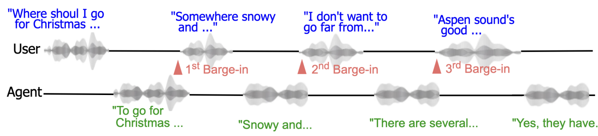

Figure 3: *Multi-turn conversation with frequent barge-in.*

Unseen reasoning problem. Beyond leveraging learned knowledge to generate responses, the agent also demonstrates the ability to utilize contextual information, effectively summarizing the main topic of each conversation in Fig. 4 and webpage that was unseen during training.

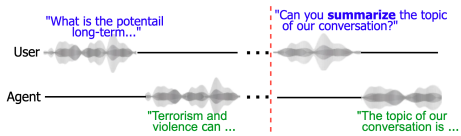

Figure 4: *Spoken QA example on an unseen topic.*

7. Conclusion

We introduced a novel duplex S2S architecture that models simultaneous user and agent streams without requiring speech pretraining. Our data-efficient approach maintains end-to-end modeling of conversation reasoning and behaviors. Experimental results show competitive performance in reasoning, barge-in, and turn-taking. Our open-sourced training and inference code will also be a valuable resource for future research.

[2]https://anonymous598e.github.io/INTERSPEECH2025-DEMO/

8. References

[1] T. B. Brown, "Language models are few-shot learners," *arXiv preprint ArXiv:2005.14165*, 2020.

[2] G. Team, P. Georgiev, V. I. Lei, R. Burnell, L. Bai, A. Gulati, G. Tanzer, D. Vincent, Z. Pan, S. Wang *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024.

[3] J. Achiam, S. Adler, S. Agarwal *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[4] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[5] Y. Fathullah, C. Wu, E. Lakomkin *et al.*, "Prompting large language models with speech recognition abilities," in *ICASSP*. IEEE, 2024, pp. 13 351–13 355.

[6] Y. Chu, J. Xu, X. Zhou, Q. Yang, S. Zhang, Z. Yan, C. Zhou, and J. Zhou, "Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models," *arXiv preprint arXiv:2311.07919*, 2023.

[7] Z. Chen, H. Huang, A. Andrusenko *et al.*, "Salm: Speech-augmented language model with in-context learning for speech recognition and translation," in *ICASSP*. IEEE, 2024, pp. 13 521–13 525.

[8] Z. Kong, A. Goel, R. Badlani *et al.*, "Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities," *arXiv preprint arXiv:2402.01831*, 2024.

[9] K. Hu, Z. Chen, C.-H. H. Yang, P. Żelasko, O. Hrinchuk, V. Lavrukhin, J. Balam, and B. Ginsburg, "Chain-of-thought prompting for speech translation," *arXiv preprint arXiv:2409.11538*, 2024.

[10] R. Huang, M. Li, D. Yang, J. Shi, X. Chang, Z. Ye, Y. Wu, Z. Hong, J. Huang, J. Liu *et al.*, "Audiogpt: Understanding and generating speech, music, sound, and talking head," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 21, 2024, pp. 23 802–23 804.

[11] D. Zhang, S. Li, X. Zhang, J. Zhan, P. Wang, Y. Zhou, and X. Qiu, "Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities," *arXiv preprint arXiv:2305.11000*, 2023.

[12] H. Kim, S. Seo, K. Jeong, O. Kwon, J. Kim, J. Lee, E. Song, M. Oh, S. Yoon, and K. M. Yoo, "Unified speech-text pretraining for spoken dialog modeling," *arXiv preprint arXiv:2402.05706*, 2024.

[13] A. Zeng, Z. Du, M. Liu, K. Wang, S. Jiang, L. Zhao, Y. Dong, and J. Tang, "Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot," *arXiv preprint arXiv:2412.02612*, 2024.

[14] A. Défossez, L. Mazaré, M. Orsini, A. Royer, P. Pérez, H. Jégou, E. Grave, and N. Zeghidour, "Moshi: a speech-text foundation model for real-time dialogue," *arXiv preprint arXiv:2410.00037*, 2024.

[15] W. Yu, S. Wang, X. Yang, X. Chen, X. Tian, J. Zhang, G. Sun, L. Lu, Y. Wang, and C. Zhang, "Salmonn-omni: A codec-free llm for full-duplex speech understanding and generation," *arXiv preprint arXiv:2411.18138*, 2024.

[16] Q. Chen, Y. Chen, Y. Chen, M. Chen, Y. Chen, C. Deng, Z. Du, R. Gao, C. Gao, Z. Gao *et al.*, "Minmo: A multimodal large language model for seamless voice interaction," *arXiv preprint arXiv:2501.06282*, 2025.

[17] Q. Zhang, L. Cheng, C. Deng, Q. Chen, W. Wang, S. Zheng, J. Liu, H. Yu, C. Tan, Z. Du *et al.*, "Omniflatten: An end-to-end gpt model for seamless voice conversation," *arXiv preprint arXiv:2410.17799*, 2024.

[18] X. Wang, Y. Li, C. Fu, Y. Shen, L. Xie, K. Li, X. Sun, and L. Ma, "Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen llm," *arXiv preprint arXiv:2411.00774*, 2024.

[19] Z. Xie and C. Wu, "Mini-omni2: Towards open-source gpt-4o with vision, speech and duplex capabilities," *arXiv preprint arXiv:2410.11190*, 2024.

[20] B. Veluri, B. N. Peloquin, B. Yu, H. Gong, and S. Gollakota, "Beyond turn-based interfaces: Synchronous llms as full-duplex dialogue agents," *arXiv preprint arXiv:2409.15594*, 2024.

[21] NVIDIA, "STT En FastConformer Hybrid Transducer-CTC Large Streaming 80ms," 2023, version 1.20.0, Released June 22, 2023. [Online]. Available: https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_fastconformer_hybrid_large_streaming_80ms

[22] P. Zhang, G. Zeng, T. Wang, and W. Lu, "Tinyllama: An open-source small language model," 2024.

[23] Anonymous, "Nanocodec: Towards high-quality ultra fast speech llm inference," *Preprint (Under Review)*, 2025.

[24] Z. Ma, Y. Song, C. Du, J. Cong, Z. Chen, Y. Wang, Y. Wang, and X. Chen, "Language model can listen while speaking," *arXiv preprint arXiv:2408.02622*, 2024.

[25] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen, "Finite scalar quantization: Vq-vae made simple," *arXiv preprint arXiv:2309.15505*, 2023.

[26] S. Hussain, P. Neekhara, X. Yang, E. Casanova, S. Ghosh, M. T. Desta, R. Fejgin, R. Valle, and J. Li, "Koel-tts: Enhancing llm based speech generation with preference alignment and classifier free guidance," *arXiv preprint arXiv:2502.05236*, 2025.

[27] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen *et al.*, "A human generated machine reading comprehension dataset," *arXiv preprint arXiv:1611.09268*, 2018.

[28] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[29] V. Noroozi, Z. Chen, S. Majumdar, S. Huang, J. Balam, and B. Ginsburg, "Instruction data generation and unsupervised adaptation for speech language models," *arXiv preprint arXiv:2406.12946*, 2024.

[30] H. Face, "Everyday conversations for llms," https://huggingface.co/datasets/HuggingFaceTB/everyday-conversations-llama3.1-2k, 2024.

[31] N. Ding, Y. Chen, B. Xu, Y. Qin, Z. Zheng, S. Hu, Z. Liu, M. Sun, and B. Zhou, "Enhancing chat language models by scaling high-quality instructional conversations," *arXiv preprint arXiv:2305.14233*, 2023.

[32] O. Kuchaiev, J. Li, H. Nguyen *et al.*, "Nemo: a toolkit for building ai applications using neural modules," *arXiv preprint arXiv:1909.09577*, 2019.

[33] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," *arXiv preprint arXiv:2204.02152*, 2022.

[34] A. Kumar, K. Tan, Z. Ni, P. Manocha, X. Zhang, E. Henderson, and B. Xu, "Torchaudio-squim: Reference-less speech quality and intelligibility measures in torchaudio," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[35] V. Pratap, A. Tjandra, B. Shi, P. Tomasello, A. Babu, S. Kundu, A. Elkahky, Z. Ni, A. Vyas, M. Fazel-Zarandi *et al.*, "Scaling speech technology to 1,000+ languages," *Journal of Machine Learning Research*, vol. 25, no. 97, pp. 1–52, 2024.

[36] E. Casanova, J. Weber, C. D. Shulby, A. C. Junior, E. Gölge, and M. A. Ponti, "Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone," in *International Conference on Machine Learning*. PMLR, 2022, pp. 2709–2720.

[37] J. Thienpondt and K. Demuynck, "Ecapa2: A hybrid neural network architecture and training strategy for robust speaker embeddings," *arXiv preprint arXiv:2401.08342*, 2024.