

D2: High-Capability Full-Duplex Spoken Language Models from Half-Duplex Backbones

Anonymous ACL submission

Abstract

Full-duplex spoken language models (FD-SLMs) support simultaneous listening and speaking, but often lag behind half-duplex spoken language models (HD-SLMs) in general capabilities. We propose **Dialogue-to-Duplex (D2)**, a framework that converts pretrained HD-SLMs into high-capability FD-SLMs at fine temporal granularity. D2 adapts the original audio encoder to streaming through decoder-free representation distillation and learns full-duplex interaction through supervised fine-tuning. We systematically study the effects of chunk size on full-duplex interaction, general capabilities, and inference computation. We apply D2 to two distinct HD-SLM architectures, Qwen3-Omni and LLaMA-Omni 2. At a 12.5 Hz update rate, Qwen3-Omni-D2 achieves 99.2% turn-taking success with 521-ms latency, 99.5% interruption success, and a state-of-the-art VoiceBench score of 70.1.

1 Introduction

People continuously listen while speaking, interrupt each other, and respond without explicit turn boundaries (Sacks et al., 1974; Levinson and Torreira, 2015). Modeling these dynamics remains a central challenge for spoken dialogue systems (Skantze, 2021). Full-duplex spoken language models (FD-SLMs), including Moshi (Défossez et al., 2024) and MiniCPM-o 4.5 (Cui et al., 2026), can perceive user audio while generating speech, supporting interruptions, overlapping speech, and low-latency turn taking.

Despite these interaction capabilities, native FD-SLMs often lag behind strong half-duplex spoken language models (HD-SLMs) in audio understanding, knowledge, reasoning, and instruction following (Chen et al., 2026). HD-SLMs such as Qwen3-Omni (Xu et al., 2025) and Kimi-Audio (Ding et al.,

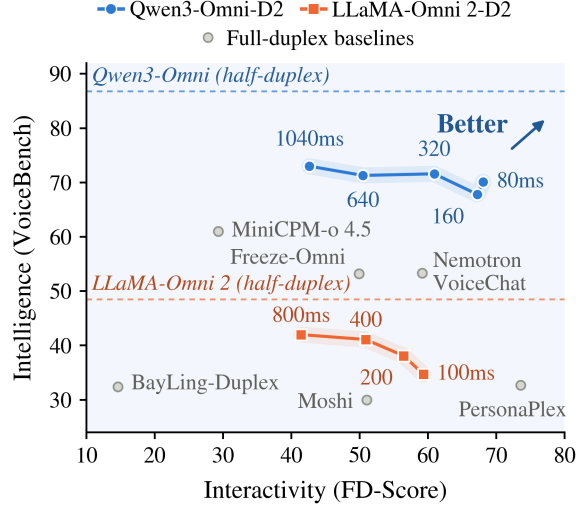


Figure 1: Capability and interactivity of full-duplex models. FD-Score is defined in Appendix F.

2025) provide stronger general capabilities. Converting these models to full-duplex operation offers a promising path toward high-capability FD-SLMs.

Previous work has explored converting HD-SLMs into FD-SLMs. BayLing-Duplex (Fang et al., 2026) converts GLM-4-Voice (Zeng et al., 2024) to full-duplex operation, but its interactivity is limited by audio encoder blocks of up to 4 s and an 800-ms chunk duration. Moreover, increasing the update rate can lead to unstable generation and substantial performance degradation (Cui et al., 2026).

We propose **Dialogue-to-Duplex (D2)**, a framework for converting pretrained HD-SLMs into high-capability, fine-grained FD-SLMs (Figure 2). D2 addresses the encoder streaming bottleneck through representation distillation and avoids the sharp performance degradation at higher update rates observed in prior work, maintaining strong general capabilities even at 12.5 Hz (80-ms chunks). We also systematically study the effects of chunk size

on full-duplex interaction, general capabilities, and inference computation.

Applied to Qwen3-Omni, D2 achieves a state-of-the-art VoiceBench (Chen et al., 2026) score of 70.1 at 12.5 Hz, outperforming the strongest evaluated full-duplex baseline by 9.1 points, while achieving 99.2% turn-taking TOR and 99.5% interruption TOR on Full-Duplex-Bench (Lin et al., 2025).

In summary, our contributions are as follows:

High-capability HD-to-FD conversion. We convert pretrained HD-SLMs into FD-SLMs, achieving state-of-the-art VoiceBench performance among evaluated FD-SLMs and strong full-duplex interaction, while mitigating performance degradation at high update rates.

Decoder-free streaming adaptation. We reduce encoder latency through representation distillation without requiring the original training decoder or transcription labels.

Chunk-size trade-offs. We systematically study the effects of chunk size on full-duplex interaction, general capabilities, and inference computation.

2 Related Work

2.1 Half-Duplex Spoken Language Models

HD-SLMs combine speech understanding and generation with a language-model backbone. Earlier systems include SpeechGPT and AudioPaLM (Zhang et al., 2023; Rubenstein et al., 2023), while VITA and Mini-Omni target low-latency or streaming speech interaction (Fu et al., 2024; Xie and Wu, 2024). Recent models such as GLM-4-Voice, LLaMA-Omni 2, Kimi-Audio, Qwen3-Omni, and Qwen3.5-Omni provide capable pretrained backbones for conversion to full-duplex operation (Zeng et al., 2024; Fang et al., 2025b; Ding et al., 2025; Xu et al., 2025; Qwen, 2026).

2.2 Full-Duplex Spoken Language Models

Systems such as LSLM, Freeze-Omni, Moshi, and PersonaPlex support simultaneous listening and speaking, including interruption handling and overlapping speech (Ma et al., 2025; Wang et al., 2024a; Défossez et al., 2024; Roy et al., 2026). MiniCPM-o 4.5 combines full-duplex interaction with stronger general capabilities, but uses a 1-s chunk duration and degrades substantially when this duration is reduced to 0.1 s (Cui et al., 2026). This motivates conversion methods that retain strong capabilities at smaller chunk durations.

2.3 Conversion Strategy

OmniFlatten converts a pretrained text LLM into a spoken-dialogue model through speech-modality alignment followed by full-duplex dialogue training (Zhang et al., 2025). D2 starts from an HD-SLM that already possesses speech understanding and generation capabilities.

BayLing-Duplex (Fang et al., 2026) similarly converts an HD-SLM, GLM-4-Voice, to full-duplex operation. It retains a speech tokenizer requiring input blocks of up to 4 s and uses a large 800-ms chunk duration, resulting in latencies of several seconds. It also uses an additional DPO stage to stabilize turn-taking (Rafailov et al., 2023). D2 adapts the original audio encoder to low-latency streaming through distillation and learns full-duplex interaction through supervised fine-tuning, supporting chunk durations as short as 80–100 ms.

3 Method

3.1 Half-Duplex vs. Full-Duplex

We formulate a spoken dialogue as a sequence of interaction units, $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^N$, where X_i denotes the user audio observed in the i -th interaction unit and Y_i denotes the corresponding model output, including speech and optional intermediate text tokens. Let $H_i = (X_{<i}, Y_{<i})$ denote the preceding dialogue context. Given the user observations, the model factorizes the likelihood of its outputs as

$$p(Y_{1:N} | X_{1:N}) = \prod_{i=1}^N p(Y_i | X_i, H_i). \quad (1)$$

Half-duplex and full-duplex models differ primarily in how the interaction units are segmented. For a half-duplex model (HD-SLM), each X_i corresponds to a complete user turn, yielding

$$p_{\text{HD}}(Y | X) = \prod_{i=1}^{N_{\text{turn}}} p(Y_i | X_i, H_i). \quad (2)$$

A full-duplex model (FD-SLM), in contrast, operates at a finer temporal granularity. Let Δ denote the encoder resolution, i.e., the duration of one audio encoder output frame, and let $k \in \mathbb{N}_{>0}$ denote the chunk size, defined as the number of encoder frames processed per chunk. We partition the conversation into chunks of duration $k\Delta$, yielding

$$p_{\text{FD}}(Y | X; k) = \prod_{i=1}^{N_k} p(Y_i | X_i, H_i), \quad (3)$$

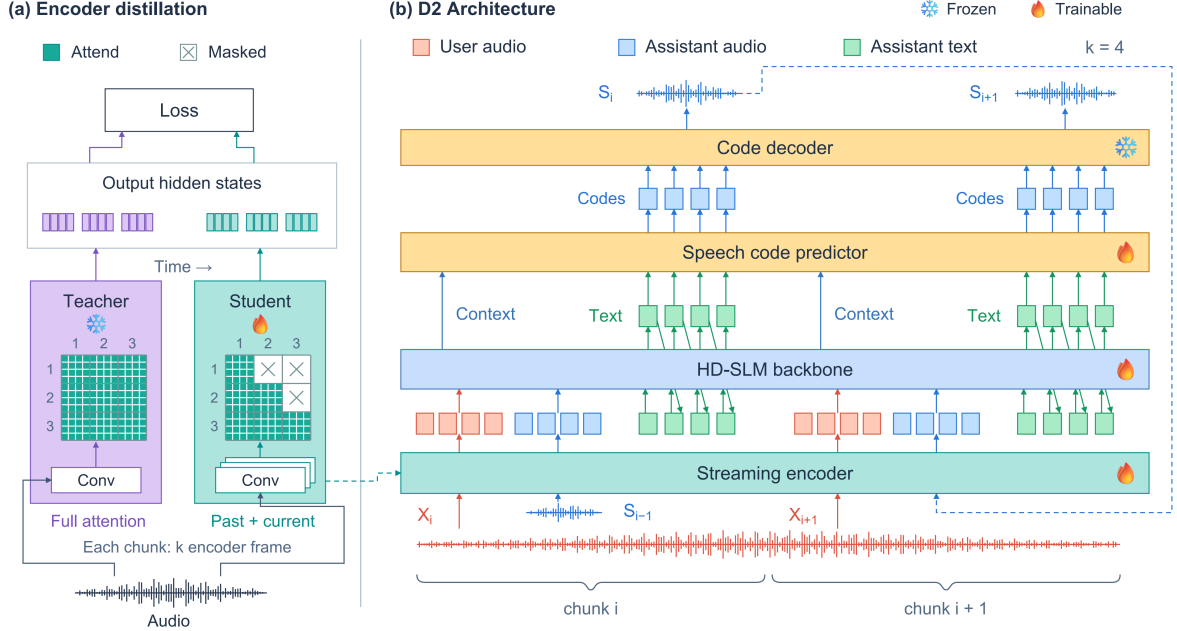


Figure 2: **Overview of Dialogue-to-Duplex (D2).** (a) A frozen offline encoder supervises a streaming student with bidirectional attention within each chunk and causal attention across chunks. (b) At chunk i , D2 uses current user audio X_i and preceding model audio S_{i-1} to autoregressively generate S_i . Here, $k = 4$.

where each X_i contains k encoder frames and N_k is the number of chunks. For example, Moshi (Défossez et al., 2024) uses $k = 1$ and $\Delta = 80$ ms, while MiniCPM-o 4.5 (Cui et al., 2026) uses $k = 10$ and $\Delta = 100$ ms. During each chunk, the model may generate speech or remain silent.

3.2 Dialogue-to-Duplex (D2)

The formulations above show that HD and FD models share a similar generation objective; they differ in the temporal granularity of the interaction sequence. An HD-SLM operates over variable-length, turn-defined units, whereas an FD-SLM operates over smaller chunks of duration $k\Delta$.

This suggests viewing full-duplex interaction as ordinary dialogue conducted at a finer temporal granularity. As k decreases, the model observes and responds to the user more frequently, enabling increasingly fine-grained real-time interaction.

D2 converts pretrained HD-SLMs into FD-SLMs. Finer interaction granularity introduces two challenges. First, the audio encoder must operate with limited future context. Second, the model must learn to make conversational decisions at a finer temporal scale. D2 addresses these challenges through **streaming encoder distillation** followed by **D2 supervised fine-tuning**.

3.3 Stage I: Encoder Distillation

Audio encoders in pretrained HD-SLMs are typically non-streaming or blockwise streaming and rely on future acoustic context, making them unsuitable for low-latency full-duplex inference. Chunk-wise attention and memory-based streaming encoders have been widely studied for low-latency speech recognition (Shi et al., 2021; Yao et al., 2021). However, directly replacing the encoder with a streaming encoder may introduce a representation mismatch with the pretrained language model.

We instead convert the original encoder into a streaming student and distill it from the unchanged offline encoder (Hinton et al., 2015; Chang et al., 2022). For each newly available chunk of duration $k\Delta$, the student recomputes the convolutional frontend using only the audio available up to the current chunk, while bidirectional self-attention in the Transformer is replaced with block-causal attention. Attention remains bidirectional within each chunk, but is causal across chunks. Given the same input audio, the original encoder serves as

the teacher and provides frame-level target representations. We minimize:

$$\mathcal{L}_{\text{distill}} = \frac{1}{TD} \sum_{t=1}^T \|\mathbf{h}_t^{\text{stu}} - \mathbf{h}_t^{\text{tea}}\|_1 - \frac{1}{T} \sum_{t=1}^T \log \sigma(\cos(\mathbf{h}_t^{\text{stu}}, \mathbf{h}_t^{\text{tea}})). \quad (4)$$

Here, T and D denote the sequence length and hidden dimension, respectively.

3.4 Full-Duplex Inference

D2 performs full-duplex inference in chunks of duration $k\Delta$. During each chunk, the model processes the newly accumulated user audio together with the self-generated audio from the preceding chunk, prefills their representations into the model context, and autoregressively generates the next k output frames. User audio is captured continuously, including while the model is speaking. Algorithm 1 summarizes the inference procedure.

Algorithm 1 D2 Full-Duplex Inference

Require: Streaming audio encoder E , dialogue model M , chunk duration $k\Delta$

- 1: Initialize model context H
 - 2: $S_0 \leftarrow \text{Silence}(k\Delta)$
 - 3: **for** each chunk $i = 1, 2, \dots$ **do**
 - 4: Observe user audio X_i of duration $k\Delta$
 - 5: $U_i \leftarrow E(X_i)$ $\triangleright k$ user embeddings
 - 6: $A_i \leftarrow E(S_{i-1})$ $\triangleright k$ self embeddings
 - 7: $H \leftarrow \text{Prefill}_M(H, [U_i; A_i])$
 - 8: $Y_i \leftarrow \emptyset$
 - 9: **for** $j = 1, \dots, k$ **do**
 - 10: $(y_{i,j}, H) \leftarrow \text{GenerateFrame}_M(H)$
 - 11: $Y_i \leftarrow Y_i \parallel y_{i,j}$
 - 12: **end for**
 - 13: $S_i \leftarrow \text{Render}(Y_i)$
 - 14: Emit S_i
 - 15: **end for**
-

Here, $y_{i,j}$ denotes an output frame containing speech together with its associated intermediate text. The preceding output S_{i-1} provides delayed self-audio feedback.

Increasing k results in coarser interaction granularity and weaker full-duplex responsiveness. Under our formulation, an HD-SLM lies at the coarse-grained extreme of this spectrum, where interaction boundaries are determined by complete conversational turns rather than short chunks.

3.5 Stage II: D2 SFT

We train the model to follow the inference procedure in Algorithm 1 by converting ordinary dialogues into temporally aligned sequences of fixed-duration chunks. User and assistant audio are placed on a shared timeline that preserves silence and overlapping speech, and the timeline is partitioned into chunks of duration $k\Delta$. For each chunk i , X_i contains the newly observed user audio, S_{i-1} contains the assistant audio from the preceding chunk, and Y_i contains the target k output frames. Explicit control tokens indicate response initiation and interruption, while padding tokens occupy positions without lexical output.

Starting from the pretrained HD-SLM equipped with the streaming encoder from Stage I, we fine-tune the model with teacher forcing using the same temporal ordering as during inference. Let $O_i = [E(X_i); E(S_{i-1})]$ denote the audio observations prefilled at chunk i . We minimize

$$\mathcal{L}_{\text{D2}} = -\mathbb{E}_{\mathcal{D}} \left[\sum_{i=1}^{N_k} \sum_{j=1}^k w_{i,j} \cdot \log p_{\theta}(y_{i,j} \mid O_i, H_i, Y_{i,<j}) \right]. \quad (5)$$

where $w_{i,j}$ controls output weighting and masking. We upweight sparse conversational control outputs, such as response-initiation and interruption events, while downweighting padding outputs. This trains the HD-SLM to follow the same fine-grained inference procedure used by D2.

4 Experimental Setup

We describe the HD-SLM backbones, datasets, training procedure, and evaluation benchmarks used for D2.

4.1 Half-Duplex Models

We apply D2 to Qwen3-Omni and LLaMA-Omni 2 (Xu et al., 2025; Fang et al., 2025b), whose streaming speech decoders support low-latency output. The temporal resolution of their audio representations determines Δ used in D2.

Qwen3-Omni. Qwen3-Omni-30B-A3B uses a Mixture-of-Experts Thinker–Talker architecture with an Audio Transformer (AuT) encoder, a vision encoder, and a streaming codec decoder. D2 uses only its audio input pathway. The AuT convolutional frontend downsamples the input to 12.5 Hz ($\Delta = 80$ ms). Its Transformer attends

bidirectionally within audio blocks; Stage I adapts this encoder to streaming. The Thinker generates text, while the Talker produces 16-codebook speech frames conditioned on text embeddings and Thinker hidden states. Code2Wav incrementally converts these frames into waveforms.

LLaMA-Omni 2. We use the bilingual 7B variant, built on Qwen2.5. Its Whisper-large-v3 encoder feeds a speech adapter that concatenates five consecutive encoder frames and projects them into the language model’s embedding space. The resulting representations have a rate of 10 Hz ($\Delta = 100$ ms). A separate autoregressive TTS model generates speech conditioned on text-token embeddings and language-model hidden states, supporting streaming speech output.

4.2 Datasets

Stage I: Encoder distillation. We use the LibriSpeech train-clean-100 and train-clean-360 splits (Panayotov et al., 2015) as speech data, together with the audio tracks from AVQA (Yang et al., 2022) as general audio data. AVQA provides non-speech events and environmental sounds, such as rainfall and traffic.

Stage II: D2 training. Figure 3 illustrates how we construct full-duplex training sequences from the turn-based conversations in InstructS2S-200K (Fang et al., 2025a) and SODA (Kim et al., 2023). We also include AVQA to preserve general audio understanding. We resynthesize model responses using Kokoro TTS (Hexgrad, 2025) to provide a consistent output voice.

To construct each training sequence, we concatenate complete user–model turn pairs in their original order until the sequence spans at least 720 s of audio. At each transition to the next user turn, we sample one of three transition types: interruption with 50% probability, direct concatenation with 30%, or a silent gap with 20%. An INTERRUPT target placed after user-speech onset teaches the model to stop its current response, while a RESPONSE target placed after user-speech offset indicates that a new response should begin. An audio EOS target terminates model speech either at its natural end or upon interruption.

Additive environmental noise and channel distortions augment half of the constructed InstructS2S-200K and SODA sequences. AVQA audio remains unaugmented.

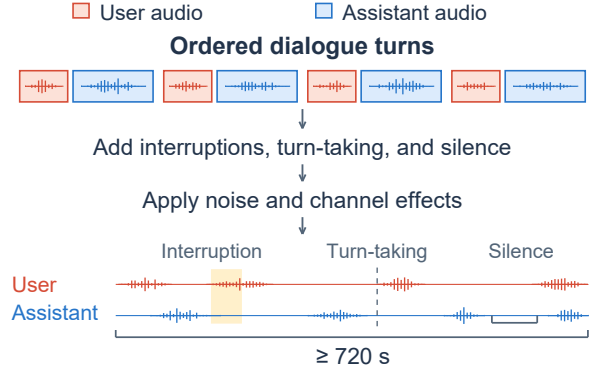


Figure 3: Full-duplex data construction. Ordered turns become sequences of at least 720 s with interruptions, turn-taking, and silence.

4.3 Implementation Details

Stage I: Encoder distillation. We train each streaming audio encoder for 30,000 updates with a batch size of four, corresponding to approximately 338 hours of LibriSpeech and 66.4 hours of AVQA. We use constant learning rates of 5×10^{-5} for the convolutional frontend and 2×10^{-6} for the fully fine-tuned attention layers. The remaining encoder components use LoRA (Hu et al., 2021) with a learning rate of 1×10^{-4} .

Stage II: D2 training. We train each D2 model for 10,000 updates with a batch size of four, corresponding to approximately 380 hours of InstructS2S-200K, 200 hours of SODA, and 100 hours of AVQA. We use Transformer-XL-style training (Dai et al., 2019) with 90-s training segments and a detached 90-s KV cache carried over from the preceding segment.

Text cross-entropy weights are 1 for lexical tokens, 0.05 for PAD, and 20 for RESPONSE and INTERRUPT. Ordinary speech-code targets have weight 1, while audio EOS has weight 20 for Qwen3-Omni and 3 for LLaMA-Omni 2. Trainable modules, LoRA configurations, learning rates, schedules, and hardware are detailed in appendix C.

4.4 Benchmarks

Full-Duplex-Bench. Full-Duplex-Bench evaluates interactive full-duplex behavior (Lin et al., 2025). We focus on three dimensions. *Turn taking* measures whether the model begins responding after the user finishes speaking. *Interruption handling* measures whether the model yields when interrupted and whether it can generate an appropriate response after the user’s turn. *Pause handling*

Model	Chunk duration (ms)	Quality (1–5) ↑	Turn taking (%) ↑	TT latency (s) ↓	Interrupt (%) ↑	Int. latency (s) ↓	Pause (%) ↑
Qwen3-Omni-D2	80	4.56	99.2	0.521	99.5	0.403	21.3
	160	4.67	100	0.539	99.5	0.432	19.0
	320	4.61	100	0.710	99.5	0.571	25.5
	640	4.62	99.2	1.01	99.5	0.928	55.1
	1040	4.84	100	1.38	99.5	1.30	61.6
LLaMA-Omni 2-D2	100	3.33	100	0.746	99.5	0.621	74.5
	200	3.62	99.2	0.825	100	0.710	78.2
	400	3.69	100	1.02	99.0	0.894	69.4
	800	4.03	98.3	1.45	99.0	1.32	70.8
BayLing-Duplex [†]	–	3.55	85.7	5.57	100	5.19	72.2
Freeze-Omni	–	3.82	34.5	0.288	98.5	0.349	69.4
MiniCPM-o 4.5	–	4.45	75.6	1.92	94.5	1.89	91.2
PersonaPlex	–	4.43	99.2	0.346	95.0	0.294	31.9
Moshi	–	3.40	100	0.615	89.5	1.23	0.0
Nemotron VoiceChat	–	3.97	91.6	0.584	99.0	0.638	60.2

Table 1: Full-Duplex-Bench results. Chunk duration is $k\Delta$ for D2 models; – marks non-D2 models. Quality is post-interruption response quality rated by GPT-4o. Turn taking/Interrupt are take-over rates (TOR); Pause is 100 minus Synthetic Pause Handling TOR (%). TT/Int. mean turn-taking/interruption.

measures whether the model avoids falsely taking the turn during short pauses within a user’s utterance.

VoiceBench. VoiceBench (Chen et al., 2026) evaluates general spoken-language capabilities through nine sub-benchmarks spanning question answering, reasoning, instruction following, and safety. AlpacaEval (Li et al., 2023), CommonEval (Chen et al., 2026; Ardila et al., 2020), and WildVoice (Chen et al., 2026) evaluate open-ended question answering. SD-QA (Faisal et al., 2021) evaluates reference-based question answering, while OpenBookQA (OBQA) (Mihaylov et al., 2018) and MMSU (Chen et al., 2026; Wang et al., 2024b) evaluate multiple-choice question answering. BBH (Suzgun et al., 2023) evaluates reasoning, IFEval (Zhou et al., 2023) evaluates instruction following, and AdvBench (Zou et al., 2023) evaluates safety. We evaluate the first 200 samples from each sub-benchmark to assess capability retention relative to the original HD-SLMs and compare D2 with other FD-SLMs.

5 Results

5.1 Full-Duplex-Bench

We compare D2 with six full-duplex baselines in Table 1 (Fang et al., 2026; Wang et al., 2024a; Cui et al., 2026; Roy et al., 2026; Défossez et al., 2024;

[†]BayLing-Duplex results were obtained using the authors’ official implementation under our evaluation protocol.

NVIDIA, 2026). Both Qwen3-Omni and LLaMA-Omni 2 achieve strong full-duplex performance after D2 conversion. Turn-taking and interruption take-over rates (TOR) remain close to 100% across all chunk sizes. Qwen3-Omni-D2 also achieves the highest post-interruption response quality among the evaluated models, with GPT scores of 4.56–4.84.

At the smallest chunk size, Qwen3-Omni-D2 achieves turn-taking and interruption latencies of 521 ms and 403 ms, respectively, while LLaMA-Omni 2-D2 achieves 746 ms and 621 ms. Larger chunk sizes increase latency but generally improve post-interruption response quality. Qwen3-Omni-D2’s pause-handling success increases from 21.3% at 80 ms to 61.6% at 1040 ms, showing the trade-off between responsiveness and tolerance of user pauses.

5.2 VoiceBench

We compare D2 with its original backbones and seven spoken-language baselines in Table 2 (Xu et al., 2025; Fang et al., 2025b; Défossez et al., 2024; Roy et al., 2026; Fang et al., 2026; Wang et al., 2024a; Cui et al., 2026; NVIDIA, 2026; Ding et al., 2025). Figure 4 shows the results across chunk durations alongside the original half-duplex backbones.

Qwen3-Omni-D2 achieves the highest overall VoiceBench score among the evaluated full-duplex models at every tested chunk duration. Even at

Model	Chunk duration (ms)	Alpaca Eval	Common Eval	Wild Voice	SD-QA	MMSU	OBQA	BBH	IF Eval	Adv Bench	Overall
Qwen3-Omni-D2	80	4.30	3.47	3.58	64.5	64.5	81.5	64.5	37.2	91.5	70.1
	160	4.33	3.44	3.63	68.0	60.0	71.5	59.0	29.6	94.0	67.8
	320	4.40	3.70	3.76	67.0	60.0	86.5	62.0	36.9	95.0	71.6
	640	4.38	3.96	3.78	57.0	63.0	85.5	60.0	38.0	95.5	71.3
	1040	4.53	3.67	3.94	66.5	63.5	86.0	60.0	39.4	98.5	73.0
Qwen3-Omni	HD	4.84	4.52	4.69	68.5	73.0	91.0	87.5	80.7	99.5	86.8
LLaMA-Omni 2-D2	100	3.38	1.94	2.13	9.0	27.5	36.0	48.5	12.5	29.5	34.7
	200	3.59	2.31	2.28	15.0	30.5	37.0	50.5	11.7	35.0	38.1
	400	3.52	2.48	2.35	18.5	26.5	43.0	51.5	13.5	49.5	41.1
	800	3.59	2.86	2.64	20.5	32.0	40.0	47.5	15.8	40.5	42.0
LLaMA-Omni 2	HD	3.94	3.50	3.25	29.0	25.0	44.5	48.5	14.6	61.5	48.5
Moshi	–	2.04	1.53	1.49	17.5	22.5	21.0	47.0	13.2	47.5	30.0
PersonaPlex	–	3.14	2.30	2.31	21.0	27.5	24.0	49.5	12.5	5.0	32.7
BayLing-Duplex	–	3.24	2.23	2.13	6.0	30.5	28.0	49.5	14.2	11.0	32.4
Freeze-Omni	–	3.74	3.47	3.20	45.0	26.0	33.5	47.0	21.5	98.0	53.2
MiniCPM-o 4.5	–	3.57	3.28	3.02	51.0	46.0	72.5	57.5	37.9	86.5	61.0
Nemotron VoiceChat	–	3.48	2.79	2.83	28.5	36.5	60.0	52.5	20.3	100	53.3
Kimi-Audio	HD	4.15	3.06	3.88	55.0	57.0	86.5	73.0	57.9	99.5	72.3

Table 2: VoiceBench scores. Chunk duration is $k\Delta$ for D2 models; HD denotes a half-duplex model and – denotes another full-duplex model. AlpacaEval, CommonEval, and WildVoice use a 1–5 scale. The remaining sub-benchmarks report percentages; Overall is the aggregate score on a 0–100 scale.

80 ms, it scores 70.1, exceeding the strongest baseline, MiniCPM-o 4.5, by about 9.1 points. At 1040 ms, it reaches 73.0, comparable to the half-duplex model Kimi-Audio’s 72.3. These results demonstrate that D2 brings strong general capabilities to full-duplex operation while supporting fine-grained interaction.

5.3 Computation Analysis

We measure the full inference computation time of Qwen3-Omni-D2 on a single RTX PRO 6000 GPU (96 GB), using BF16 and a batch size of one. Table 3 shows that both mean and p95 computation times remain below the chunk duration at all tested chunk sizes, supporting real-time inference under this setting. Larger chunk sizes yield lower real-time factors by amortizing the audio encoder’s computation over larger input chunks. Measurement details are provided in Appendix A.

6 Ablation

6.1 Stage-I Encoder Distillation

Before Stage II, we adapt the audio encoder to streaming through distillation. We examine whether Stage I improves the converted model and whether supervised encoder training can serve as an alternative. Distillation requires only the pre-trained encoder, allowing D2 conversion even when

Chunk duration (ms)	Mean (ms)	p95 (ms)	RTF ↓
80	56.9	57.2	0.711
160	110	110	0.688
320	216	217	0.676
640	429	431	0.670
1,040	682	684	0.656

Table 3: Full inference computation time of Qwen3-Omni-D2 with a fixed 2,048-slot Thinker KV cache per layer. RTF is computation time divided by chunk duration.

its training decoder is not publicly available. When the decoder is available, Stage I can instead use the original speech-recognition objective.

We evaluate both approaches and a baseline without Stage I using LLaMA-Omni 2-D2 with a 100-ms chunk duration. Table 4 shows that omitting Stage I lowers VoiceBench from 34.7 to 30.9 and post-interruption quality from 3.33 to 2.75. Stage I therefore improves response quality. Supervised fine-tuning achieves comparable results, with a slightly higher VoiceBench score but lower post-interruption quality than distillation.

6.2 Control Signal

In Stage II, we upweight control signal tokens and downweight PAD tokens in the text cross-entropy loss to improve turn control and reduce

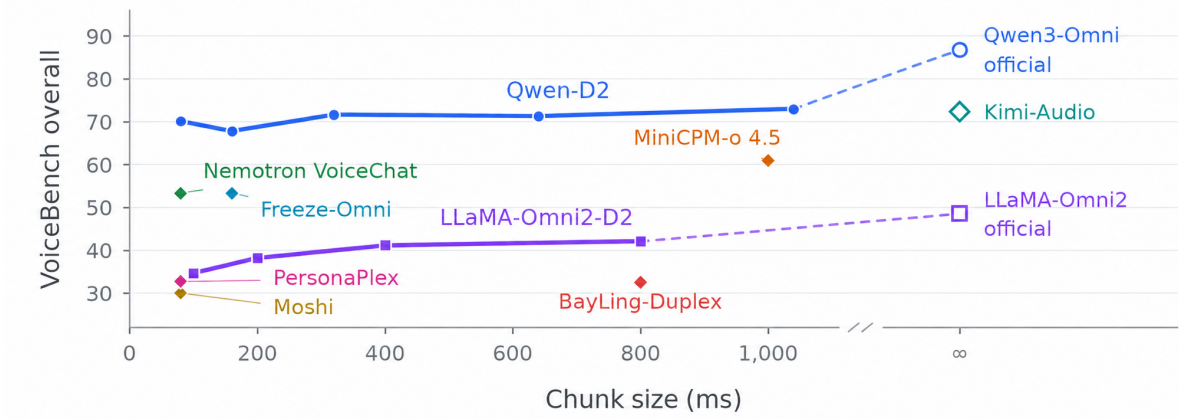


Figure 4: VoiceBench versus chunk duration. Filled circles/squares and solid lines show D2 variants; open markers show original half-duplex models. Dashed connections compare D2 with its backbones. The axis break and ∞ denote turn-based operation without fixed-duration chunks.

Metric	Distill.	Supervised	No Stage I
Pause (%) $\uparrow$	74.5	61.6	62.0
Int. TOR (%) $\uparrow$	99.5	100	99.0
TT TOR (%) $\uparrow$	100	100	100
Quality (1–5) $\uparrow$	3.33	3.09	2.75
Int. latency (ms) $\downarrow$	621	590	617
TT latency (ms) $\downarrow$	746	676	686
VoiceBench $\uparrow$	34.7	35.4	30.9

Table 4: Stage-I encoder training ablation for LLaMA-Omni 2-D2 with a 100-ms chunk duration. Distill. denotes distillation; No Stage I omits encoder adaptation.

CE weight		Turn taking	Interruption	
PAD	Control	TOR $\uparrow$	TOR $\uparrow$	Score $\uparrow$
0.05	20	100	99.0	3.67
1	20	95.8	98.0	3.44
0.05	1	95.8	98.5	3.55
1	1	62.2	90.5	3.45

Table 5: Control-signal weighting ablation on Full-Duplex-Bench. Lexical-token weight is 1 throughout. TOR is reported in percentages. Scores cover qualifying interruption responses.

under-emission. We examine these choices using LLaMA-Omni 2-D2 with a 100-ms chunk duration. We fine-tune the model for 1,000 additional updates under each weight configuration, and evaluate all four models on Full-Duplex-Bench.

Table 5 shows that removing either weighting adjustment lowers turn-taking TOR from 100% to 95.8%. Removing both reduces turn-taking and interruption TOR to 62.2% and 90.5%, respectively.

The original weighting also gives the highest post-interruption score. Both adjustments help maintain response initiation and turn control.

7 Conclusion

We proposed **Dialogue-to-Duplex (D2)**, a framework for converting HD-SLMs into FD-SLMs through streaming encoder distillation and full-duplex supervised fine-tuning, using several hundred hours of training data per stage. D2 enables low-latency streaming and strong full-duplex interaction while retaining substantial general capabilities of the original HD-SLM. Across two different HD-SLM architectures, D2 achieves strong results on full-duplex interaction benchmarks and state-of-the-art VoiceBench performance among evaluated full-duplex models, narrowing the capability gap between FD-SLMs and HD-SLMs.

Limitations

First, the speech synthesis module contributes a large portion of the end-to-end latency in D2. Reducing speech-generation latency often leads to degraded speech quality or unstable generation, which restricts the choice of HD-SLM backbones to models with sufficiently low-latency speech synthesis. Second, we construct full-duplex training data from turn-based conversations because high-quality publicly available full-duplex dialogue data is currently limited. Finally, pause handling remains limited, particularly for Qwen3-Omni-D2 at small chunk durations.

Ethical Considerations

Full-duplex spoken language models raise privacy and consent concerns because they continuously process incoming audio, including while generating speech. Their natural, low-latency interactions may also facilitate impersonation, social engineering, and deceptive conversations. Furthermore, D2 may inherit biases and harmful behaviors from pretrained models. Responsible deployment should therefore consider informed user consent, transparency regarding AI-generated speech, privacy-preserving audio processing, and safeguards against misuse.

References

- Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *Proceedings of the twelfth language resources and evaluation conference*, pages 4218–4222.
- Heng-Jui Chang, Shu-wen Yang, and Hung-yi Lee. 2022. Distilhubert: Speech representation learning by layer-wise distillation of hidden-unit bert. In *ICASSP 2022-2022 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 7087–7091. IEEE.
- Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T Tan, and Haizhou Li. 2026. Voicebench: Benchmarking llm-based voice assistants. *Transactions of the Association for Computational Linguistics*, 14:378–398.
- Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, et al. 2026. Minicpm-o 4.5: Towards real-time full-duplex omni-modal interaction. *arXiv preprint arXiv:2604.27393*.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 2978–2988.
- Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. 2024. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*.
- Ding Ding, Zeqian Ju, Yichong Leng, Songxiang Liu, Tong Liu, Zeyu Shang, Kai Shen, Wei Song, Xu Tan, Heyi Tang, et al. 2025. Kimi-audio technical report. *arXiv preprint arXiv:2504.18425*.
- Fahim Faisal, Sharlina Keshava, Md Mahfuz Ibn Alam, and Antonios Anastasopoulos. 2021. Sd-qa: Spoken dialectal question answering for the real world. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3296–3315.
- Qingkai Fang, Shoutao Guo, and Yang Feng. 2026. Bayling-duplex: Native full-duplex speech dialogue with a single autoregressive llm. *arXiv preprint arXiv:2606.14528*.
- Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. 2025a. Llama-omni: Seamless speech interaction with large language models. In *International Conference on Learning Representations*, volume 2025, pages 57607–57624.
- Qingkai Fang, Yan Zhou, Shoutao Guo, Shaolei Zhang, and Yang Feng. 2025b. Llama-omni 2: Llm-based real-time spoken chatbot with autoregressive streaming speech synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18617–18629.
- Chaoyou Fu, Haojia Lin, Zuwei Long, Yunhang Shen, Yuhang Dai, Meng Zhao, Yi-Fan Zhang, Shaoqi Dong, Yangze Li, Xiong Wang, et al. 2024. Vita: Towards open-source interactive omni multimodal llm. *arXiv preprint arXiv:2408.05211*.
- Hexgrad. 2025. [Kokoro-82m](#).
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- 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. *arXiv preprint arXiv:2106.09685*.
- Hyunwoo Kim, Jack Hessel, Liwei Jiang, Peter West, Ximing Lu, Youngjae Yu, Pei Zhou, Ronan Bras, Malihe Alikhani, Gunhee Kim, et al. 2023. Soda: Million-scale dialogue distillation with social commonsense contextualization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12930–12949.
- Stephen C Levinson and Francisco Torreira. 2015. Timing in turn-taking and its implications for processing models of language. *Frontiers in psychology*, 6:731.
- Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. AlpacaEval: An automatic evaluator of instruction-following models.
- Guan-Ting Lin, Jiachen Lian, Tingle Li, Qirui Wang, Gopala Anumanchipalli, Alexander H Liu, and Hung-yi Lee. 2025. Full-duplex-bench: A benchmark to evaluate full-duplex spoken dialogue models on turn-taking capabilities. In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE.

- Ziyang Ma, Yakun Song, Chenpeng Du, Jian Cong, Zhuo Chen, Yuping Wang, Yuxuan Wang, and Xie Chen. 2025. Language model can listen while speaking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24831–24839.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391.
- NVIDIA. 2026. NVIDIA NemotronLabs VoiceChat 11B. <https://huggingface.co/nvidia/NVIDIA-NemotronLabs-VoiceChat-11B>. Hugging Face model card, version 1.0.
- Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.
- Qwen. 2026. Qwen3.5-Omni technical report. *arXiv preprint arXiv:2604.15804*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.
- Rajarshi Roy, Jonathan Raiman, Sang-gil Lee, Teodor-Dumitru Ene, Robert Kirby, Sungwon Kim, Jaehyeon Kim, and Bryan Catanzaro. 2026. Personaplex: Voice and role control for full duplex conversational speech models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16137–16141. IEEE.
- Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. 2023. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*.
- Harvey Sacks, Emanuel A Schegloff, and Gail Jefferson. 1974. A simplest systematics for the organization of turn-taking for conversation. *language*, 50(4):696–735.
- Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. 2022. Utmos: Utokyo-sarulab system for voicemos challenge 2022. *arXiv preprint arXiv:2204.02152*.
- Yangyang Shi, Yongqiang Wang, Chunyang Wu, Ching-Feng Yeh, Julian Chan, Frank Zhang, Duc Le, and Mike Seltzer. 2021. Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6783–6787. IEEE.
- Gabriel Skantze. 2021. Turn-taking in conversational systems and human-robot interaction: a review. *Computer Speech & Language*, 67:101178.
- Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051.
- Xiong Wang, Yangze Li, Chaoyou Fu, Yunhang Shen, Lei Xie, Ke Li, Xing Sun, and Long Ma. 2024a. Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen llm. *arXiv preprint arXiv:2411.00774*.
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, et al. 2024b. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.
- Zhifei Xie and Changqiao Wu. 2024. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*.
- Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfu Zhu, et al. 2025. Qwen3-omni technical report. *arXiv preprint arXiv:2509.17765*.
- Pinci Yang, Xin Wang, Xuguang Duan, Hong Chen, Runze Hou, Cong Jin, and Wenwu Zhu. 2022. Avqa: A dataset for audio-visual question answering on videos. In *Proceedings of the 30th ACM international conference on multimedia*, pages 3480–3491.
- Zhuoyuan Yao, Di Wu, Xiong Wang, Binbin Zhang, Fan Yu, Chao Yang, Zhendong Peng, Xiaoyu Chen, Lei Xie, and Xin Lei. 2021. Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit. *arXiv preprint arXiv:2102.01547*.
- Aohan Zeng, Zhengxiao Du, Mingdao Liu, Kedong Wang, Shengmin Jiang, Lei Zhao, Yuxiao Dong, and Jie Tang. 2024. Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot. *arXiv preprint arXiv:2412.02612*.
- Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15757–15773.

- 713 Qinglin Zhang, Luyao Cheng, Chong Deng, Qian Chen,
714 Wen Wang, Siqi Zheng, Jiaqing Liu, Hai Yu, Chao-
715 Hong Tan, Zhihao Du, et al. 2025. Omniflatten: An
716 end-to-end gpt model for seamless voice conversa-
717 tion. In *Proceedings of the 63rd Annual Meeting of*
718 *the Association for Computational Linguistics (Vol-*
719 *ume 1: Long Papers)*, pages 14570–14580.
- 720 Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Sid-
721 dhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou,
722 and Le Hou. 2023. Instruction-following evalu-
723 ation for large language models. *arXiv preprint*
724 *arXiv:2311.07911*.
- 725 Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr,
726 J Zico Kolter, and Matt Fredrikson. 2023. Univer-
727 sal and transferable adversarial attacks on aligned
728 language models. *arXiv preprint arXiv:2307.15043*.

A Real-Time Computation

Hardware and inputs. We use one NVIDIA RTX PRO 6000 Blackwell Server Edition GPU (96 GB), one CPU core, and a batch size of one. Neural weights and activations use BF16; the CPU log-Mel frontend uses FP32. The benchmark uses synthetic PCM inputs. We use PyTorch 2.11.0+cu128, Transformers 5.13.0, SDPA attention, batched_mm experts, and CUDA graphs.

Timed operations. Timing runs from CPU input PCM to CPU output PCM. It includes waveform-to-Mel conversion, both streaming encoders for user and delayed self audio, Thinker prefill and decoding, text-token selection, Talker, all 15 residual-code predictions per speech frame, and incremental Code2Wav decoding. Data transfers, codec sampling, and PCM conversion are included. Each chunk generates k text tokens and k speech-code frames, with codec EOS disabled to measure continuous speech generation. Initialization, graph capture, cache reset, audio acquisition, transport, and playback are outside the timed region.

Measurement protocol. We evaluate $k \in \{1, 2, 4, 8, 13\}$. The CUDA graphs preallocate 2,048 Thinker KV-cache slots per layer for every k ; unwritten slots are masked. Each measurement starts with zero conversational history and one Talker speaker-prefix token. We report the mean and 95th percentile. RTF is $t/(k\Delta)$, where t is computation time and $\Delta = 80$ ms.

B Implementation Pseudocode

B.1 Stage-I Streaming Encoder

Qwen3-Omni. Each chunk supplies $8k$ new Mel frames at 100 Hz. We run the convolutional frontend once on the latest 104 Mel frames, then keep the last k of its 13 output positions. Each Transformer layer attends to these k positions and at most $104 - k$ cached positions. Attention is bidirectional within the current chunk; future chunks are absent. User and self audio have separate states but share encoder weights. Mel history and per-layer KV caches are initially empty. Algorithm 2 gives one streaming step.

For Stage-I training, we batch the convolution windows and apply the same attention pattern to the resulting sequence. For query position q , let $b = k\lfloor q/k \rfloor$. Key position p is visible exactly when $\max(0, b - (104 - k)) \leq p < b + k$. We match the

student output to the frozen original encoder using the Stage-I loss. Gradients pass through all valid positions in the training sequence; the streaming cache in Algorithm 2 describes inference.

LLaMA-Omni 2. Whisper receives $10k$ new Mel frames and produces $5k$ encoder rows per chunk. Its convolution uses the last two Mel frames plus the current input. Five consecutive Transformer rows are concatenated and projected to one 3,584-dimensional embedding. The implementation retains convolution outputs and recomputes block-causal attention within a window of $R = 5k\lfloor 1500/(5k) \rfloor$ rows; it does not use the Qwen encoder’s 104-token KV cache. The initial two-frame Mel tail is zero. Algorithm 3 gives the streaming step.

B.2 Qwen3-Omni-D2 Training and Inference

Let T be the number of 80-ms encoder frames in a training segment, with T divisible by k . User and self embeddings have shape $[B, T, 2048]$, text targets $[B, T]$, and codec targets $[B, T, 16]$. For each chunk, we serialize

$$[U_{i,1:k}; A_{i,1:k}; e(y_{i-1,k}); e(y_{i,1:k-1})].$$

Here, A_i contains the model audio from the preceding chunk. The resulting Thinker input has shape $[B, 3T, 2048]$ and uses ordinary causal attention in this order. Learned stream embeddings distinguish user and self audio. The first text input is PAD; later inputs use the preceding target or generated token, including across chunk boundaries. A task-specific system prompt is prefixed once per conversation.

Alignment and control. Text and codec targets share the 80-ms clock. RESPONSE is placed eight native frames after the last audible user frame. The first lexical token and model audio start one frame later; lexical tokens then occupy consecutive frames, followed by PAD. INTERRUPT is placed eight native frames after user onset. These offsets are independent of k . Training self audio includes the measured 371-sample playback shift at 16 kHz, followed by the one-chunk feedback delay. The first self-audio chunk uses a learned silence embedding. RESPONSE and INTERRUPT reuse $\langle |audio_start| \rangle$ and $\langle |audio_end| \rangle$; PAD is $\langle |duplex_pad| \rangle$.

Training. Algorithm 4 omits padding masks and distributed reductions. pack serializes each chunk in the order above; text_slots selects its last k

Algorithm 2: Stage-I streaming encoder for Qwen3-Omni.

```

def encoder_step(new_mel, state, k):
    # new_mel: [B, 128, 8*k]
    w = cat([state.mel, new_mel], dim=-1)
    w = left_pad(w[..., -104:], length=104)
    x = w[:, None] # [B,1,128,104]
    for conv in (conv1, conv2, conv3):
        x = gelu(conv(x))
    x = conv_out(flatten_channels(x))
    h = x[:, -k:] + position[13 - k : 13]
    # h: [B,k,1280]; Q/K/V split into heads
    for layer, block in enumerate(blocks):
        q, key, val = block.qkv(block.norm1(h))
        key = cat([state.K[layer], key], dim=-2)
        val = cat([state.V[layer], val], dim=-2)
        a = sdpa(q, key, val, is_causal=False)
        h = h + block.out(merge_heads(a))
        h = h + block.mlp(block.norm2(h))
        state.K[layer] = key[..., -(104 - k) :, :]
        state.V[layer] = val[..., -(104 - k) :, :]
    state.mel = w
    return proj2(gelu(proj1(norm(h)))), state
    # output: [B,k,2048]

```

Algorithm 3: Stage-I streaming encoder for LLaMA-Omni 2.

```

def whisper_step(mel, state, k):
    # mel: [B,128,10*k]; tail: [B,128,2]
    w = cat([state.tail, mel], dim=-1)
    x = gelu(conv2(gelu(conv1(w))))
    x = x.transpose(1, 2)[:, -5 * k :]
    if state.rows.shape[1] == R:
        state.rows = empty_rows()
    state.rows = cat([state.rows, x], dim=1)
    # mask[q,p] = (p // (5*k) <= q // (5*k))
    h = transformer(state.rows + positions(), mask=block_causal(width=5 * k))
    h = norm(h[:, -5 * k :]).reshape(B, k, 6400)
    state.tail = mel[..., -2:]
    return linear2(redu(linear1(h))), state

```

positions. Encoder states are independent for the two audio streams. The task prompt and Talker speaker prefix are initialized once in training and inference.

The text loss uses weights 1 for lexical tokens, 0.05 for PAD, and 20 for RESPONSE and INTERRUPT. Codec targets have weight 1, with weight 20 for codebook-0 EOS. BOS has no codec loss; residual codebooks also exclude EOS. The two losses are normalized separately. We carry detached encoder state, the last k self embeddings, the last text and codec targets, and about 90 s of Thinker/Talker

KV memory into the next training segment. State is cleared at a conversation boundary.

Inference. Algorithm 5 shows the decoding loop. Each chunk uses the current user audio and the model audio from the preceding chunk. The model generates and emits speech one frame at a time, then stores the output as self-audio input for the next chunk.

In `speech_step`, an open Talker conditions on the current text embedding and preceding codec frame; a closed Talker uses the Thinker’s layer-24

Algorithm 4: Qwen-D2 training on one conversation, with teacher forcing.

```

for segment in training_segments:
    U, enc_u = E(segment.user, enc_u)
    A, enc_a = E(segment.self, enc_a)
    A = delay(A, k, previous_self)
    y, c = segment.text, segment.codec
    z = shift_right(y, previous_text)
    x = pack(U + user_row, A + self_row, embed(z))
    h, h24, kv = Thinker(x, kv, causal=True)
    h, h24 = text_slots(h), text_slots(h24)
    logits = text_head(h) # [B,T,V_text]
    opened = inclusive_BOS_EOS_mask(c[...], 0])
    c_in = shift_right(c, previous_codec)
    c_in[y == RESPONSE] = BOS_frame
    ctx = where(opened, text_proj(embed(y)), hidden_proj(h24))
    v = ctx + where(opened, codec_embed(c_in), 0)
    g, talk_kv = Talker(v, talk_kv, causal=True)
    p0 = codec0_head(g) # [B,T,V_codec]
    pr = MTP_teacher_force(g, c[...], :15])
    # pr: [B,T,15,V_codec]
    loss = text_CE(logits, y)
    loss = loss + codec_CE(p0, pr, c)
    loss.backward()
    clip_grad_norm_(trainable, 1.0)
    optimizer.step()
    optimizer.zero_grad()
    save_detached_segment_state()

```

hidden state. RESPONSE opens Talker and supplies codec BOS. During speech, Talker predicts codebook 0 and MTP predicts the remaining 15 codebooks autoregressively. INTERRUPT closes an open Talker with EOS; a sampled codec EOS also ends speech. PAD ends lexical generation but does not itself close Talker. BOS, EOS, and closed frames produce silence. Talker KV and Code2Wav state advance on every native frame, including silence, and rendered audio is fed back only in the next complete chunk.

C Training and Reproducibility Details

Optimization. Tables 6 and 7 summarize the trainable modules. Full denotes dense fine-tuning of the listed module. All LoRA adapters use scaling $\alpha = 2r$ and zero dropout. Both stages use AdamW with gradient clipping at 1.0. Computation uses BF16 with FP32 trainable parameters and optimizer states. Frozen Qwen Thinker/Talker routed experts use weight-only INT8 storage during Stage II and are dequantized for computation.

Sampling and segmentation. Stage I alternates four LibriSpeech batches with one AVQA batch.

Module (both backbones)	Training	LR
Encoder frontend	Full	5×10^{-5}
Encoder attention	Full	2×10^{-6}
Encoder MLP/projector	LoRA, $r = 32$	10^{-4}
Original teacher	Frozen	—
Updates		30,000
Global batch / accumulation		4 / 1
Weight decay		0.01
Learning-rate schedule		Constant

Table 6: Stage-I training. Each chunk size has its own distilled encoder. Hardware and logged time are in Table 8.

Stage II cycles through two InstructS2S cohorts, one SODA cohort, and sixteen AVQA cohorts. Each cohort contains four examples. Long conversations contribute several training segments, while short AVQA examples usually contribute one. Each training segment produces one optimizer update; gradients do not accumulate across segments. Four-GPU runs use one example per GPU, and two-GPU runs use two examples per GPU. Training and augmentation use seeds 1337 and 13371, respectively.

Algorithm 5: Qwen-D2 inference with delayed self-audio feedback.

```

previous_text = PAD
feedback = None
kv = prefill_system_prompt()
state = init_speech_state()
while stream_open:
    user = wait_for_audio(1280 * k) # 16 kHz
    U, enc_u = E(user, enc_u)
    if feedback is None:
        A = learned_self_silence(k)
    else:
        A, enc_a = E(feedback, enc_a)
    x = cat([U + user_row, A + self_row, embed(previous_text)], dim=1)
    # One prefill of 2k+1 positions
    h, h24, kv = Thinker(x, kv, causal=True)
    output = []
    for j in range(k):
        if j > 0: # k-1 forwards
            h, h24, kv = Thinker(embed(y), kv)
        logits = text_head(h[:, -1])
        y = sample_allowed(logits, state)
        pcm, state = speech_step(y, h24[:, -1], state)
        output.append(pcm) # 80 ms, 24 kHz
        emit(pcm)
    previous_text = y
    feedback = resample(cat(output), 24000, 16000)

```

Backbone	Module	Training	Peak LR
Both	Thinker attention	LoRA, $r = 128$	5×10^{-5}
Both	Encoder frontend	Full	2×10^{-6}
Both	Encoder attention/MLP/projector	LoRA, $r = 32$	10^{-5}
Both	New control and stream embeddings	Full	10^{-4}
Qwen3-Omni	Initial self-silence embedding	Full	10^{-4}
Qwen3-Omni	Talker attention/shared expert, context projections, MTP	LoRA, $r = 128$	5×10^{-5}
Qwen3-Omni	Codec embeddings	Full	5×10^{-5}
Qwen3-Omni	Codec output heads	Full	10^{-4}
Qwen3-Omni	Small pretrained interfaces (e.g., Talker norms)	Full	2×10^{-5}
LLaMA-Omni 2	Native TTS attention and MLP	LoRA, $r = 128$	5×10^{-5}
Both	Updates / global batch / accumulation	10,000 / 4 / 1	
Both	Weight decay / warmup	0 / 200 updates	
Both	Training segment / detached memory	90 s / 90 s, rounded down to whole chunks	

Table 7: Stage-II training. Unlisted pretrained parameters are frozen, including Qwen routed experts and routers, Code2Wav, and the LLaMA-Omni 2 TTS input bridge, tokenizer, flow model, and vocoder.

Initialization and schedules. Stage II starts from the Stage-I encoder trained at the same k . Encoder MLP adapters are retained; fresh attention adapters are added over its now-frozen dense attention. Qwen learning rates decay after warmup to their floors at update 2,000 and remain there. Thinker/Talker/ context/MTP LoRA uses an absolute floor of 10^{-5} ; other groups use 10% of peak. LLaMA-Omni 2 uses cosine decay to 10% of peak

at update 10,000. Its audio EOS loss weight is 3; other text and ordinary audio weights follow the main text.

D Dataset Construction

Source data. Table 9 lists the prepared source inventory. Stage I uses only audio; LibriSpeech clips are filtered to 1–20 s and selected in split and row order. For Stage II, we synthesize both

Backbone	k	Chunk duration (ms)	Stage-I GPUs	Stage-I time (h)	Stage-II GPUs	Stage-II time (h)
Qwen3-Omni	1	80	4× RTX	1.73	4× RTX	12.5
	2	160	2× L40S	1.81	4× RTX	13.2
	4	320	2× L40S	1.58	2× H200	13.0
	8	640	2× L40S	2.04	4× RTX	11.9
	13	1040	4× RTX	1.21	4× RTX	10.8
LLaMA-Omni 2	1	100	4× RTX	1.50	4× RTX	12.4
	2	200	4× RTX	1.49	4× RTX	11.1
	4	400	4× RTX	2.09	4× RTX	12.5
	8	800	4× RTX	2.17	4× RTX	11.2

Table 8: Hardware and approximate logged training time. RTX denotes NVIDIA RTX PRO 6000 Blackwell Server Edition; L40S and H200 denote NVIDIA L40S and NVIDIA H200. Times are elapsed hours, not GPU-hours, summed within resumed log segments. They include replayed updates and data/checkpoint overhead, but exclude queue time and initialization between segments.

Source	Unit	Count
LibriSpeech	Speech clip	35,245
AVQA	Audio-QA pair	35,245
InstructS2S-200K	Conversation	149,998
SODA	Conversation	41,776

Table 9: Prepared source examples before conversation concatenation and repeated sampling. Stage I uses only the audio from AVQA.

conversation roles with Kokoro. Each conversation uses one of 904 user voice recipes and the fixed af_heart model voice. We remove exact-zero samples only at utterance edges and reject conversations containing invalid or all-zero turns. SODA’s two alternating speakers become user and model; an unmatched final user turn is dropped.

Full-duplex sequences. We preserve turn order and append complete conversations until sequence duration reaches at least 720 s. Initial silence and inserted gaps are sampled uniformly from 2–30 s on the frame grid. At each model-to-user transition, we sample interruption, direct concatenation, or silence with probabilities 0.5, 0.3, and 0.2. An interruption starts the next user at a sampled interior frame of the preceding model utterance. Model speech continues until the INTERRUPT target. Qwen additionally shortens eligible user turns with probability 0.1, sampling an audible cutoff in the latter half; this augmentation is disabled for LLaMA-Omni 2.

Labels and augmentation. We locate speech boundaries on clean waveforms using two consecutive 10-ms windows above −60 dBFS RMS. Control and speech targets are fixed before acoustic augmentation. Half of the complete user tracks receive one or two colored-noise components at 8–30 dB SNR, optional HVAC noise, and sampled gain, filtering, limiting, clipping, or bit-depth distortion. Augmentation also covers silence gaps and varies by epoch. Self audio and speech targets stay clean.

AVQA. The input concatenates the original audio clip with a spoken question and its answer choices. The model says the correct letter followed by the complete choice text. We keep AVQA audio unaugmented. Source scheduling and training segmentation follow Appendix C.

E Speech-Quality Evaluation

Protocol. Table 10 reports speech quality using UTMOS (Saeki et al., 2022). The evaluation uses the first user turn from 100 InstructS2S rows, excluded from our local D2 training inventory. All four models receive the same Kokoro-generated user waveforms and generate their responses without teacher forcing. D2 checkpoints are taken at update 10,000, with 60 s of silence appended after each prompt for the response. The original backbones use their released speech-generation paths, with a 512-token text limit and a 60-s audio limit. Qwen uses the Chelsie voice in both settings. The original backbones use BF16 inference.

Scoring. We resample each response to 16 kHz and remove only outer silence using a 20-ms RMS window, a −50 dBFS threshold, and 100-ms margins. Internal pauses and streaming boundaries

Model	Chunk size k (duration)	UTMOS $\uparrow$
Qwen3-Omni-D2	1 (80 ms)	4.54 ± 0.03
Qwen3-Omni	HD	4.46 ± 0.07
LLaMA-Omni 2-D2	1 (100 ms)	4.16 ± 0.20
LLaMA-Omni 2	HD	4.43 ± 0.09

Table 10: Speech quality at chunk size $k = 1$ for both D2 backbones. UTMOS is reported as mean $\pm$ sample standard deviation on shared prompts. HD denotes the original half-duplex backbone.

remain in the scored waveform. We apply no additional gain normalization. UTMOS runs in FP32 and assigns one score to each response crop. The reported mean weights each scorable response equally; the standard deviation is across responses. Silent outputs and crops shorter than 0.5 s receive no score. Truncated outputs remain included.

Results. On the 99 mutually scorable Qwen prompts, D2 and its backbone score 4.54 and 4.46, respectively. All 100 LLaMA-Omni 2 prompts are scorable; D2 and its backbone score 4.16 and 4.43.

F FD-Score Definition

We define FD-Score in figure 1 as an equally weighted average of latency-adjusted turn-taking and interruption take-over rates (TORs):

$$\text{FD-Score} = \frac{1}{2} \left(\frac{R_{\text{TT}}}{1 + L_{\text{TT}}/\tau} + \frac{R_{\text{Int}}}{1 + L_{\text{Int}}/\tau} \right), \quad (6)$$

where R_{TT} and R_{Int} are the turn-taking and interruption TORs, expressed as percentage values on a 0–100 scale. L_{TT} and L_{Int} are their corresponding measured latencies, and $\tau = 1$ s is a fixed reference duration. Thus, when latencies are expressed in seconds, the denominators are numerically $1 + L_{\text{TT}}$ and $1 + L_{\text{Int}}$.

Each term rewards successful interaction while penalizing latency. For nonnegative latencies, FD-Score lies between 0 and 100; higher scores indicate better performance under this definition. We compute the two terms separately, average them, and round only the final score.

For example, Qwen3-Omni-D2 with 80-ms chunks has $R_{\text{TT}} = 99.2$, $L_{\text{TT}} = 0.521$ s, $R_{\text{Int}} = 99.5$, and $L_{\text{Int}} = 0.403$ s, yielding

$$\text{FD-Score} = \frac{1}{2} \left(\frac{99.2}{1.521} + \frac{99.5}{1.403} \right) \approx 68.07. \quad (7)$$