

dots.tts Technical Report

dots.tts Team

Code: <https://github.com/rednote-hilab/dots.tts>
Model: <https://huggingface.co/collections/rednote-hilab/dotstts>
Demo: <https://rednote-hilab.github.io/dots.tts-demo>

Abstract

We present **dots.tts**, a 2B-parameter continuous autoregressive text-to-speech (TTS) foundation model that models speech in a continuous latent space. Compared with existing continuous autoregressive models, our key innovations are threefold. First, we train an Audio-VAE with multiple objectives to build a semantically structured and prediction-friendly continuous speech space. Second, we use full-history conditioning in the flow-matching head to preserve long-range consistency and reduce drift during generation. Third, we apply reward-free self-corrective post-training to the flow-matching head to further improve robustness and acoustic quality. After being trained on a large-scale multilingual corpus, **dots.tts** achieves the best average performance on Seed-TTS-Eval, with WERs of 0.94%/1.30%/6.60% and SIM scores of 81.0/77.1/79.5 on the zh/en/zh-hard test sets, respectively. Across other benchmarks, **dots.tts** also consistently demonstrates open-source state-of-the-art performance, exhibiting strong generation stability, voice cloning ability, and emotional expressiveness. For efficient inference, we further apply CFG-aware MeanFlow distillation, enabling low-latency speech generation with first-packet latencies of 85/54 ms in output streaming and dual-streaming modes, respectively. To facilitate reproducible research and practical deployment, we release the training and inference code, together with the pretrained, post-trained, and MeanFlow-distilled checkpoints, under the Apache 2.0 license.

1 Introduction

Text-to-speech (TTS) systems have largely solved intelligibility on standard read-speech benchmarks. What users expect from a modern system is broader: expressive and controllable output, real-time synthesis, and coverage of neutral reading, emotional dialogue, paralinguistic events, singing, and general audio. Current systems pursue this goal along three roughly distinct technical routes, and each route has its own unresolved problem.

The first route is non-autoregressive (NAR) generation on top of a flow-matching (Lipman et al., 2023) or Diffusion infilling. Voicebox (Le et al., 2023), F5-TTS (Chen et al., 2025a), OmniVoice (Zhu et al., 2026), and LongCat-AudioDiT (Xin et al., 2026) fall into this group. They generate an utterance in a single parallel pass, and few-step distillation (Liu et al., 2023; Geng et al., 2025) can push inference latency very low. NAR is a good fit for offline data synthesis, dubbing, and asset production, whereas autoregressive (AR) generation is more natural for interactive use. We focus on the AR side in this report.

Following the success of large language models, AR generation over discrete tokens has become the mainstream production paradigm. The CosyVoice family (Du et al., 2025), Qwen3-TTS (Qwen Team, 2026), Llasa (Ye et al., 2025a), IndexTTS (Deng et al., 2025; Zhou et al., 2026), and Seed-TTS (Seed Team, ByteDance, 2024a) all follow some variant of this recipe. Reducing speech to a discrete vocabulary lets the system reuse most of the textual large language model (LLM) stack: stable next-token prediction

training, mature SFT and preference-optimization tooling, well-studied scaling behavior, and efficient inference kernels. Hence its current dominance in deployment. The limits of discrete AR come from the tokenizer, it caps what the language model (LM) can express. This bottleneck explains why discrete-token systems struggle to cover speech, emotional paralinguistics, singing, and ambient sound within a single distribution.

A recent line of work removes this bottleneck by modeling speech as continuous-representation AR generation. These methods differ mainly in their per-step distribution parameterization. KALL-E (Xia et al., 2026) predicts a Flow-VAE next-frame latent with a Kullback-Leibler divergence (KL) objective and no diffusion component. DiTAR (Jia et al., 2025), VibeVoice (Peng et al., 2025) and VoxCPM (Zhou et al., 2025) pair an LM backbone with a per-patch diffusion or flow-matching head, whereas ARDiT (Liu et al., 2024) folds both roles into a decoder-only diffusion transformer. Any2Speech (Song et al., 2026) argues more broadly for native-agentic generation over continuous tokenizers. Avoiding discrete quantization raises the perceptual ceiling, permits a single distribution over speech, paralinguistics, singing, and general audio, and preserves the streaming, prompt-conditioned behavior that makes LM-based systems easy to deploy.

The main issue which has so far kept this paradigm short of production maturity is *long-range error accumulation*. With discrete tokens, the codec snaps an imperfect sample back to a valid acoustic configuration before it reaches the waveform. Continuous latents have no such quantization buffer: every small prediction error is reconstructed faithfully by the decoder and then fed back as conditioning for the next AR step.

dots.tts is our attempt to close the gap: an end-to-end AR TTS model over continuous latents. The system combines three practices. First, we build the system on a high-fidelity semantic AudioVAE, following the HoliTok-style VAE recipe (Li et al., 2026). After reconstruction-oriented training, we add multitask downstream objectives and a WavLM (Chen et al., 2022) representation-alignment loss, making the latent space both semantic and learnable for the AR backbone. Second, inspired by ARDiT’s autoregressive diffusion formulation, we decompose continuous generation into three specialized modules: a semantic encoder for content-aligned representation, an LLM for long-range text-to-content modeling, and a full-context AR flow-matching head for local acoustic rendering. This separation keeps semantic reasoning and acoustic rendering from competing inside a single module, reducing error accumulation during long rollouts. Third, for post-training, we adapt the reward-free self-correction idea of SOAR (Qin et al., 2026) to the AR flow-matching head, exposing the acoustic DiT to its own off-trajectory inference errors without requiring a reward model or external teacher. Together, these choices improve long-range stability while preserving the fidelity and expressiveness enabled by continuous-latent generation.

Trained on 1.5M hours of speech, **dots.tts** achieves state-of-the-art stability and zero-shot voice-cloning quality on Seed-TTS-Eval (Seed Team, ByteDance, 2024b), leads average speaker similarity on the 24-language MiniMax multilingual test set (MiniMax Team, 2025), and is competitive on EmergentTTS-Eval (Manku et al., 2025) and CV3-Eval (Du et al., 2025), which together cover stability, speaker similarity, naturalness, prosody, paralinguistics, and multilingual coverage.

For real-time and conversational use without quality loss, the full model is designed from the outset to be causal at the latent-patch level. This causal structure lets text-side planning and audio-latent emission proceed incrementally, making 1T1A interleaved dual-stream inference possible. We further apply MeanFlow distillation to compress the flow-matching ODE to as few as 2 to 4 function evaluations, yielding low first-packet latency on the same backbone: 85 ms at RTF 0.231 in plain mode and 54 ms at RTF 0.245 in interleaved streaming mode. The full efficiency profile, including voice-cloning scenarios and an audio-prompt cache, is reported in Section 3.4. Alongside the model, we document a full training recipe for continuous AR TTS, covering pretraining, our Self-corrective alignment stage for the flow-matching head, and the MeanFlow distillation above.

In summary, our contributions are:

- We present **dots.tts**, a 2B-parameter fully continuous end-to-end AR TTS system that removes discrete acoustic tokens while achieving state-of-the-art stability and zero-shot voice-cloning quality on Seed-TTS-Eval.
- We alleviate the instability of continuous AR generation with three complementary designs: a semantic causal AudioVAE, an ARDiT-inspired decomposition into semantic planning and acoustic rendering, and reward-free Self-corrective alignment for the flow-matching head.
- We improve inference efficiency with CFG-aware MeanFlow distillation and fully causal 1T1A interleaved streaming, enabling low first-packet latency (85 ms at RTF 0.231 plain, 54 ms at RTF 0.245 interleaved; Section 3.4) suitable for real-time deployment.

2 Model

2.1 Overview

dots.tts is a fully continuous, end-to-end autoregressive TTS system built from two decoupled networks: an audio variational autoencoder (AudioVAE) that defines the continuous representation, and an autoregressive backbone that predicts that representation one patch at a time. The AudioVAE encodes 48 kHz mono speech into a 128-dimensional latent stream at 25 Hz ($1920\times$ temporal downsampling) and decodes it back via a BigVGAN-style decoder (Lee et al., 2023); once trained, it is frozen and serves as both the generation target and the input representation for the backbone.

The backbone has three components (Figure 1): a semantic encoder, an LLM, and an autoregressive flow-matching head. The LLM handles the semantic side of generation and the flow-matching head handles the acoustic side. The LLM, initialized from a pretrained text LLM, consumes BPE text tokens together with a 6.25 Hz audio-semantic embedding stream and emits one hidden state per audio step. The text is placed as a prefix in plain TTS mode, or interleaved with audio in a 1T1A layout for low-latency streaming (Section 2.3). The autoregressive flow-matching head (AR-FM) conditions on that hidden state and generates the next four-frame patch of the 25 Hz VAE latent. The semantic encoder, re-used from the AudioVAE training pipeline, projects each newly generated patch back into a single 6.25 Hz embedding that feeds the LLM at the next step. The LLM sees only this semantic summary, not the raw VAE latent. We found this necessary to keep continuous-AR rollouts stable.

2.2 AudioVAE

The AudioVAE follows and adapts the training recipe and model architecture of HoliTok, with a causal variant of the BigVGAN-v2 decoder on the synthesis side. The encoder is a fully causal convolutional stack of strided residual blocks with downsample strides [2, 2, 2, 4, 6, 10], ending in a posterior projection that produces per-frame mean and log-variance. Following KALL-E, we add flow regularization on top of the standard KL prior to shape a smooth, low-noise latent space. All convolutions on both sides are causal, keeping the VAE compatible with strict streaming synthesis.

We train the AudioVAE in two stages. The first stage targets reconstruction quality alone: following the BigVGAN-v2 recipe, we combine multi-period plus multi-scale sub-band CQT adversarial loss, multi-scale mel-spectral reconstruction loss, and feature-matching loss, together with the KL + flow regularization above.

The second stage targets *learnability*. Such a heavily compressed latent is reconstructive but retains so much acoustic variation that a downstream LLM struggles to use it as a generation target. To make the space semantically structured without sacrificing reconstruction, we keep the first-stage losses and add

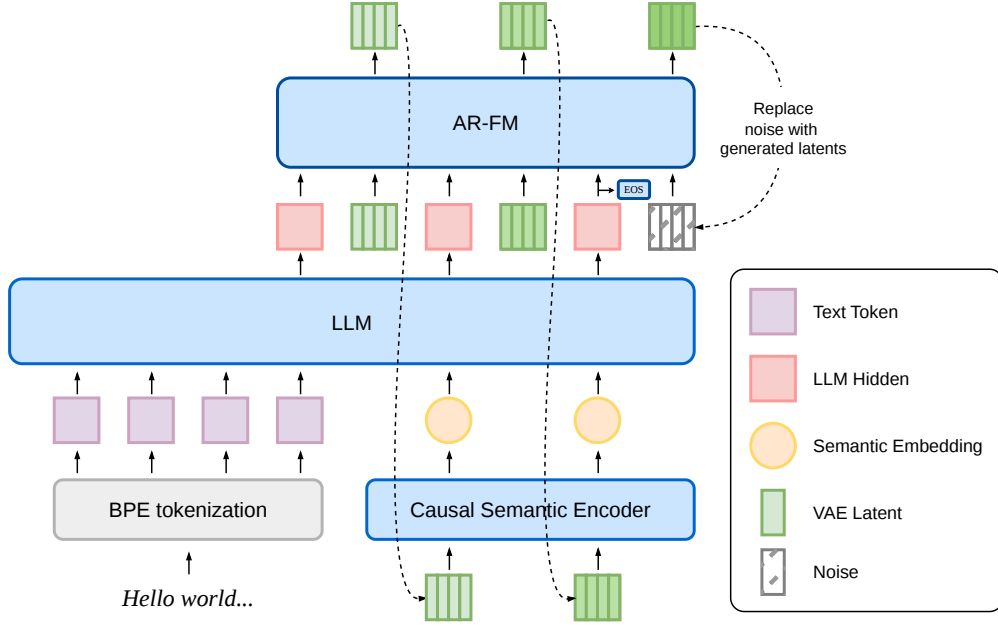


Figure 1: Overview of the `dots.tts` backbone. BPE text tokens and 6.25 Hz audio-semantic embeddings share a single LLM stream; each LLM hidden state conditions the AR-FM head to generate the next four-frame VAE-latent patch, which is fed back through the semantic encoder at the next step. The AudioVAE (Section 2.2) is trained separately and frozen here.

two supervisions on the latent: (i) a frame-level alignment loss against a frozen WavLM teacher, and (ii) a multitask downstream block consisting of a small encoder followed by a small LLM head, jointly trained on a mixture of ASR, emotion, and speaker classification objectives. The downstream LLM is discarded afterwards; we retain only the encoder, which the backbone reuses as its semantic frontend (Section 2.4). This second stage improves the downstream diffusion learnability of the latent while preserving reconstruction quality; see Table 1 for the full reconstruction comparison against discrete codecs and other continuous representations.

2.3 LLM backbone

We initialize the LLM from Qwen2.5-1.5B Base (Qwen Team, 2024) and feed it text directly as BPE instead of phonemes. On the audio side it runs at 6.25 Hz: the semantic encoder (Section 2.4) compresses each 25 Hz VAE latent patch into one audio-semantic embedding for the next LLM step, and the AR-FM head (Section 2.5) consumes the LLM hidden state to emit the next four-frame VAE patch.

Training supports two sequence layouts: a *plain mode*, in which the full text is placed as a prefix before the audio span and used for standard TTS, and a 1T1A interleaved mode for low-latency dual streaming, detailed in Section 2.3.1. Only the sequence layout differs between the two, and the per-step interface between LLM, semantic encoder, and AR-FM head is identical. Text positions carry no loss. The LLM is optimized end-to-end through the flow-matching gradient that propagates back from the AR-FM head.

Starting from a text LLM gives us better prosody, better text normalization, and the ability to follow natural-language style prompts. The cost is data: a BPE-driven backbone needs substantially more speech-text pairs than a phoneme one. We scale the training corpus accordingly.

2.3.1 Interleaved sequence for dual streaming mode

In 1T1A interleaved mode, a single BPE text token alternates with one 6.25 Hz audio step until a text-end marker, after which audio continues alone:

$$\underbrace{T A T A \dots T A}_{\text{interleaved span}} \langle \text{EOT} \rangle \underbrace{A A \dots A}_{\text{audio-only tail}},$$

where T is a BPE text token, A is a 6.25 Hz audio position, and $\langle \text{EOT} \rangle$ marks the end of the text stream. The 1:1 interleaving lets an upstream conversational LLM drive synthesis at its own text-emission rate: each text token it emits is consumed at the next audio step, so speech can start within a single text token of generation and continue token by token from there. The backbone then serves directly as the audio output of a streaming dialogue LLM, without buffering a full utterance before speaking. The $\langle \text{EOT} \rangle$ marker covers the common case where the dialogue model finishes its text before the corresponding speech has been fully rendered: audio continues from the already-received text context until the stop head fires.

This mode is intended for real-time dialogue systems, where text and audio are produced incrementally and synthesis should begin before a full utterance is available. Plain mode remains the best choice when the full text has already been prepared before synthesis.

2.4 Semantic encoder

The semantic encoder is the same module that supervises Stage 2 of the AudioVAE training, transplanted into the backbone with its pretrained weights. It converts each newly generated 25 Hz VAE-latent patch into a single 6.25 Hz embedding for the LLM, stripping out the high-variance acoustic detail along the way.

Architecturally, a strided causal-convolution projector first halves the input frame rate, followed by a 24-layer causal Transformer with hidden dim 1024 and FFN dim 4096; the encoder output is then grouped in pairs along time and linearly projected to the LLM embedding dimension, yielding an end-to-end $4\times$ downsampling from 25 Hz latent frames to 6.25 Hz LLM tokens. Strict causality at every step lets the same module be unrolled one patch at a time during streaming inference.

As the encoder is already pretrained to expose semantically structured features, it maps a latent patch onto a representation aligned with the LLM’s text semantic space. The LLM can then ignore acoustic detail during the AR rollout and condition on a compact summary of the history.

2.5 Autoregressive flow-matching head

2.5.1 Per-step latent generation

Following DiTAR and ARDiT, we use a Diffusion Transformer (DiT) (Peebles and Xie, 2023) as the velocity-field predictor, instantiated with 18 layers, hidden dim 1024, and FFN dim 4096, with RoPE on every layer, RMSNorm with QK-norm, and adaLN-zero modulation driven by the diffusion timestep and a speaker-embedding side input. The training target is the rectified-flow vector field of Liu et al. (2023), regressed against the straight-line velocity between Gaussian noise and the clean latent patch. At inference we solve the ODE with a small number of Euler steps.

Each per-step VAE patch is a block of 4 latent frames at 25 Hz (matching the 4:1 frame-rate ratio between the VAE latent and the 6.25 Hz LLM stream); we write $P_n = (\ell_n^1, \dots, \ell_n^4)$ for the clean patch and $Z_n = (z_n^1, \dots, z_n^4)$ for its noisy counterpart. At every audio position, the head consumes a *flow-matching context* assembled from three streams projected into a common hidden space:

- the LLM hidden state H_n at that position (one token),

- the clean patches $P_{<n}$ of all earlier audio positions (four tokens each), and
- the noisy patch Z_n under generation (four tokens).

These are interleaved into a single sequence

$$[H_0, P_0, H_1, P_1, \dots, H_{n-1}, P_{n-1}, H_n, Z_n], \quad (1)$$

where, with the convention above, each H_n contributes one token and each P_n / Z_n contributes a four-token block. The top-right inset of Figure 1 illustrates this layout together with the substitution step that replaces Z_n with the integrated P_n in the next-step history. A speaker x-vector extracted by a frozen CAM++ encoder (Wang et al., 2023) is added as a global adaLN-zero condition, and the LLM hidden stream and the speaker stream are each dropped independently with probability $p_{\text{drop}} = 0.5$ during training to enable classifier-free guidance (Ho and Salimans, 2021) over text content and timbre at inference. CFG is then applied jointly across the two conditions: the conditional branch keeps both streams, the unconditional branch drops both, and the velocity actually fed to the ODE solver is the standard linear extrapolation of the two. At inference, the AR loop runs as follows: starting from $[H_0, Z_0]$ the integrator returns P_0 ; we substitute P_0 for Z_0 , append the next pair $[H_1, Z_1]$, integrate, and repeat. Each newly produced P_n is also fed back through the semantic encoder to update the LLM’s audio-semantic input for the next forward step.

2.5.2 Block-causal training attention

We train the AR-FM head in parallel across all N patches of an utterance with a single forward pass, while reproducing exactly the per-step autoregressive context each patch would see at inference. The mechanism is a block-causal attention mask over a concatenated cause/generation sequence whose two halves share positional indices.

The training sequence is built from two halves of equal length. The *cause* part

$$C = [H_0, P_0, H_1, P_1, \dots, H_N, P_N]$$

holds the clean LLM-conditioned history, and the *generation* part

$$Z = [H_0, Z_0, H_1, Z_1, \dots, H_N, Z_N]$$

holds the noisy patches under denoising, each at an independently sampled flow-matching time. Position indices are reset between the two halves: tokens within C are numbered $0, 1, \dots, |C| - 1$, and Z restarts from 0 with the same per-block layout, so each (H_n, Z_n) in Z inherits the position range of the corresponding (H_n, P_n) in C . The RoPE phases between Z_n and $P_{<n}$ therefore match those of a per-step inference forward.

The block-causal mask partitions the $(C + Z) \times (C + Z)$ attention matrix into four sub-blocks (Figure 2): $C \rightarrow C$ is standard causal, mirroring the LLM-side ordering of the prefix; $C \rightarrow Z$ is fully masked, so the cause stream’s hidden states are independent of the noisy targets and equal to their inference-time values; $Z \rightarrow C$ is prefix-causal, letting each generation block Z_n attend to exactly $H_{\leq n}$ and $P_{<n}$, the autoregressive context the patch sees at inference; and $Z \rightarrow Z$ is block-diagonal, so different patches denoise independently under their own sampled times. Combined with the shared positional indices, this layout makes the parallel training forward numerically identical to a per-step inference rollout while training all N heads in one pass.

Because the AR-FM prefix already carries every LLM hidden state, the AR-FM head is on its own a complete text-conditioned speech generator, in the same sense as ARDiT: it could in principle synthesize

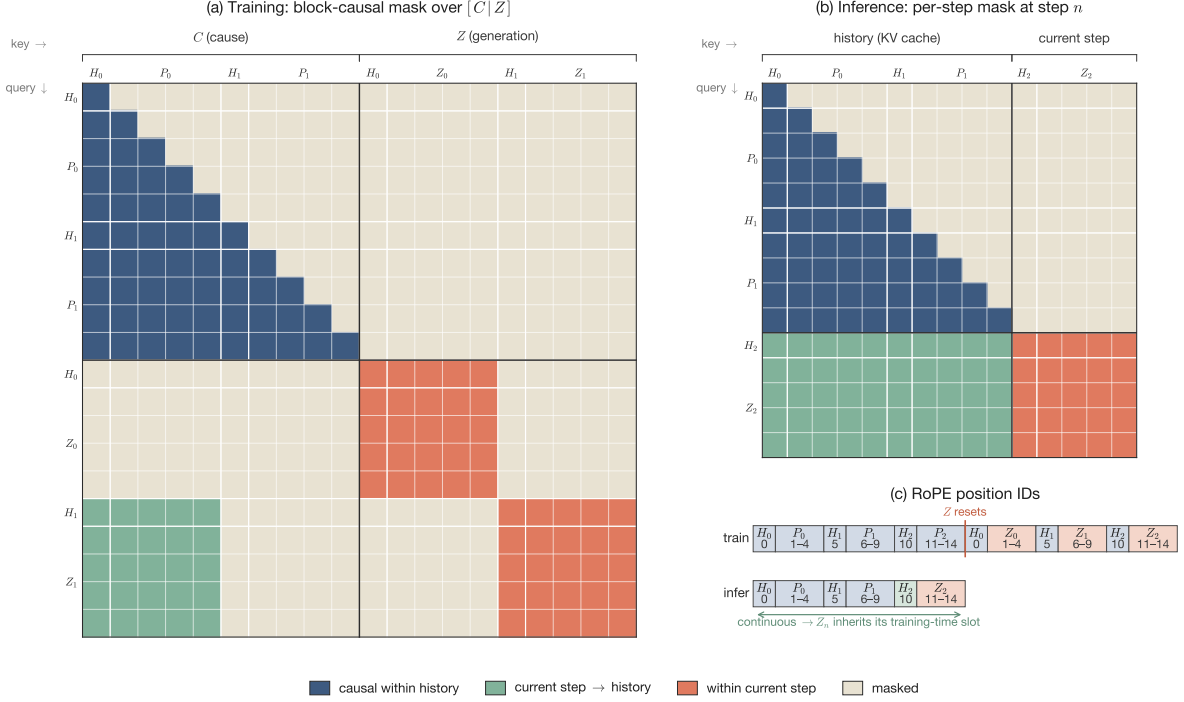


Figure 2: Attention masks and RoPE position IDs of the AR flow-matching head (H blocks of size 1, P/Z blocks of size $L = 4$). **(a)** Block-causal training mask over $[C | Z]$. **(b)** Per-step inference mask at step n , with the KV-cached history on the left and the newly appended (H_n, Z_n) on the right. **(c)** Position IDs: the train row is the C and Z segments concatenated, with the red marker at the boundary where Z 's positions reset; the inference row places Z_n at the same positions Z_n occupied in training.

audio with no further acoustic feedback from the LLM. We observe that this pushes the LLM toward encoding semantic rather than acoustic information.

2.6 Training objectives

2.6.1 AudioVAE objectives

The AudioVAE has its own two-stage objective, optimized ahead of the backbone and frozen for every backbone stage below. Per training clip,

$$\mathcal{L}_{\text{vae}} = \underbrace{\mathcal{L}_{\text{mel}} + \mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{fm}} + \beta_{\text{kl}} \mathcal{L}_{\text{kl}}}_{\text{Stage 1}} + \underbrace{\lambda_{\text{wavlm}} \mathcal{L}_{\text{wavlm}} + \lambda_{\text{sup}} \mathcal{L}_{\text{sup}}}_{\text{added in Stage 2}}, \quad (2)$$

where the Stage 1 group is a BigVGAN-v2-style reconstruction stack ($\mathcal{L}_{\text{mel}}$ multi-scale mel, $\mathcal{L}_{\text{adv}}$ multi-period plus sub-band CQT adversarial, $\mathcal{L}_{\text{fm}}$ their feature-matching counterpart) regularized by a KL + flow prior on the latent, and the Stage 2 group adds a frame-level cosine alignment loss $\mathcal{L}_{\text{wavlm}}$ against a frozen WavLM teacher together with a supervision loss $\mathcal{L}_{\text{sup}}$ summed over ASR, emotion, and speaker heads routed through a downstream encoder-LLM block, whose encoder is the module retained as the semantic frontend (Section 2.4). All backbone losses below treat the AudioVAE as a fixed encoder/decoder pair.

2.6.2 Pretraining objectives

During backbone pretraining, two objectives drive the full model.

Flow-matching loss. A per-patch mean-squared error between the predicted velocity and the analytic conditional vector field on the (sampled) VAE latent. Following the rectified-flow formulation, with P_n denoting the clean four-frame VAE-latent patch at audio position n and the per-step flow-matching context defined in Section 2.5, we optimize

$$\mathcal{L}_{\text{fm}} = \mathbb{E}_{n, t \sim \mathcal{U}(0,1), \epsilon \sim \mathcal{N}(0,I), P_n} \|v_\theta(Z_n^t, t, H_{\leq n}, P_{<n}, s) - (P_n - \epsilon)\|^2, \quad (3)$$

where $\epsilon \sim \mathcal{N}(0, I)$ is sampled from a standard Gaussian prior, P_n is the ground-truth patch of four 25 Hz VAE latent frames, $Z_n^t = (1-t)\epsilon + tP_n$ is the linear interpolation between the noise and the clean patch, v_θ is the velocity field predicted by the DiT-based AR-FM head, H_n is the LLM hidden state at audio position n that serves as the per-step conditioning signal, $P_{<n}$ are the previously generated clean patches forming the flow-matching prefix (cf. Equation (1)), and s is the global speaker x-vector. This is the *only* loss that backpropagates through the LLM and the semantic encoder: text positions carry no loss weight, audio-span positions are routed to the flow-matching head, and the LLM is therefore optimized end-to-end through the gradient flowing back from the AR-FM head.

Stop loss. A dedicated EOS prediction head, a two-layer MLP attached on top of a *detached* copy of the LLM hidden state at every audio position, is trained with a balanced binary cross-entropy that gives the (single) positive position the same total mass as all earlier negatives. Let N be the number of audio positions in the utterance, with position $N-1$ being the EOS target, and let

$$p_n = \sigma(\text{MLP}(\text{sg}(H_n)))$$

denote the predicted stop probability at position n , where $\text{sg}(\cdot)$ is the stop-gradient operator and σ is the sigmoid. The stop loss is then

$$\mathcal{L}_{\text{eos}} = -\frac{1}{2} \log p_{N-1} - \frac{1}{2(N-1)} \sum_{n=0}^{N-2} \log(1 - p_n), \quad (4)$$

so that the single positive at position $N-1$ contributes the same total mass as the $N-1$ earlier negatives combined. Detaching the hidden state keeps the stop predictor from interfering with the AR rollout dynamics.

The total backbone pretraining loss is therefore

$$\mathcal{L}_{\text{pre}} = \mathcal{L}_{\text{fm}} + \mathcal{L}_{\text{eos}}, \quad (5)$$

with the two terms weighted equally. The AudioVAE and the speaker encoder remain frozen throughout backbone training. Only the LLM, the semantic encoder, the AR-FM head, the stop head, and the small input/output projections that connect these modules receive gradients.

2.6.3 Post-training objectives

After pretraining, we post-train only the DiT acoustic generator inside the AR-FM head. The AudioVAE, speaker encoder, semantic encoder, and LLM are frozen and provide fixed acoustic, speaker, and semantic context. Let v_θ denote the trainable velocity predictor in the self-corrective stage, and let v_ϕ denote the MeanFlow student in the distillation stage. Superscript (b) indexes a minibatch element, where $b = 1, \dots, B$, and B is the minibatch size. Unless otherwise specified, squared norms are computed over the latent patch dimensions.

For notational simplicity, we use a compact noise-to-data notation in this section, consistent with the

flow-matching formulation in pretraining. For a sampled VAE latent patch P_i , we write

$$x_0 \equiv \epsilon \sim \mathcal{N}(0, I), \quad x_1 \equiv P_i, \quad x_t \equiv Z_i^t = (1 - t)x_0 + tx_1,$$

and denote all conditioning information by c . Here, x_0 is Gaussian noise, x_1 is the clean VAE latent patch, x_t is the interpolated patch at flow time $t \in [0, 1]$, and c contains the fixed acoustic, speaker, and semantic context provided by the frozen modules. The index i denotes the sampled latent patch and is independent of the minibatch index b .

For any velocity predictor f and conditioning c , classifier-free guidance (CFG) is written as

$$f^{\text{cfg}}(x_t, t, c; \gamma) = f(x_t, t, c) + \gamma (f(x_t, t, c) - f(x_t, t, \emptyset)), \quad (6)$$

where $\emptyset$ denotes the dropped-conditioning branch and $\gamma \geq 0$ is the CFG scale. When $f = v_\theta$, we write v_θ^{cfg} for the corresponding CFG-guided predictor. This convention is used by both post-training stages.

Self-corrective alignment. Following SOAR, the first stage uses a reward-free self-correction objective for the pretrained DiT. We use $\omega(t) \geq 0$ to denote the time-dependent regression weight for flow matching, which balances the contribution of training samples at different noise levels. The same weighting function is used for both on-trajectory and auxiliary off-trajectory losses.

For each minibatch element b , we sample a flow time $\tau^{(b)} \sim p_\tau$, where p_τ is the training-time distribution over $[0, 1]$. For any time $s \in [0, 1]$, we write

$$x_s^{(b)} = (1 - s)x_0^{(b)} + sx_1^{(b)}.$$

We also introduce $h_{\text{soar}} > 0$ as the self-correction rollout step size in normalized flow time. Since our time convention moves from noise at $t = 0$ to data at $t = 1$, a rollout from time $\tau^{(b)}$ advances to

$$\tau_+^{(b)} = \min(\tau^{(b)} + h_{\text{soar}}, 1).$$

The on-trajectory term is the usual flow-matching regression:

$$\ell_{\text{on}}^{(b)} = \omega(\tau^{(b)}) \left\| v_\theta(x_{\tau^{(b)}}^{(b)}, \tau^{(b)}, c^{(b)}) - (x_1^{(b)} - x_0^{(b)}) \right\|_2^2. \quad (7)$$

To expose the model to its own inference-time errors, we perform a single detached Euler rollout from $x_{\tau^{(b)}}^{(b)}$ using the current CFG-guided predictor:

$$\hat{x}_{\tau_+}^{(b)} = \text{sg} \left[x_{\tau^{(b)}}^{(b)} + \left(\tau_+^{(b)} - \tau^{(b)} \right) v_\theta^{\text{cfg}} \left(x_{\tau^{(b)}}^{(b)}, \tau^{(b)}, c^{(b)}; \gamma_{\text{soar}} \right) \right], \quad (8)$$

where $\gamma_{\text{soar}} \geq 0$ is the CFG scale used during self-corrective rollout. The stop-gradient operator $\text{sg}[\cdot]$ prevents gradients from backpropagating through the rollout trajectory.

We then re-noise this off-trajectory state toward the original noise endpoint. Let K_{aux} be the number of auxiliary samples drawn for each minibatch element. For $k = 1, \dots, K_{\text{aux}}$,

$$\begin{aligned} \alpha^{(b,k)} &\sim \mathcal{U}(0, 1), \\ \tau_{\text{aux}}^{(b,k)} &= (1 - \alpha^{(b,k)})\tau_+^{(b)}, \\ x_{\text{aux}}^{(b,k)} &= \text{sg} \left[(1 - \alpha^{(b,k)})\hat{x}_{\tau_+}^{(b)} + \alpha^{(b,k)}x_0^{(b)} \right]. \end{aligned} \quad (9)$$

The auxiliary target is the endpoint-consistent velocity that transports $x_{\text{aux}}^{(b,k)}$ to the clean endpoint $x_1^{(b)}$ at $t = 1$:

$$u_{\text{aux}}^{(b,k)} = \text{sg} \left(\frac{x_1^{(b)} - x_{\text{aux}}^{(b,k)}}{1 - \tau_{\text{aux}}^{(b,k)} + \varepsilon_{\text{aux}}} \right), \quad (10)$$

where $\varepsilon_{\text{aux}} > 0$ is a small numerical constant used to avoid division by zero near the data endpoint.

The corresponding auxiliary loss is

$$\ell_{\text{aux}}^{(b,k)} = \omega \left(\tau_{\text{aux}}^{(b,k)} \right) \left\| v_{\theta} \left(x_{\text{aux}}^{(b,k)}, \tau_{\text{aux}}^{(b,k)}, c^{(b)} \right) - u_{\text{aux}}^{(b,k)} \right\|_2^2. \quad (11)$$

Let

$$\mathcal{A} \subseteq \{1, \dots, B\} \times \{1, \dots, K_{\text{aux}}\}$$

denote the set of retained auxiliary samples, and let $M_{\mathcal{A}} = |\mathcal{A}|$. If no auxiliary samples are filtered, then $M_{\mathcal{A}} = BK_{\text{aux}}$. With auxiliary-loss weight $\lambda_{\text{aux}} \geq 0$, the self-corrective alignment loss is the count-normalized combination

$$\mathcal{L}_{\text{soar}} = \frac{\sum_{b=1}^B \ell_{\text{on}}^{(b)} + \lambda_{\text{aux}} \sum_{(b,k) \in \mathcal{A}} \ell_{\text{aux}}^{(b,k)}}{B + \lambda_{\text{aux}} M_{\mathcal{A}}}. \quad (12)$$

This is the flow-matching-native post-training stage detailed in Section 3.2.3; our self-corrective alignment applies reward-free self-correction directly to the acoustic DiT rather than to the full TTS stack.

CFG-aware MeanFlow distillation. The second stage freezes the self-corrected DiT as a teacher. We denote this frozen teacher by v_{θ_T} , and train a student DiT v_{ϕ} to predict the mean velocity over a time interval $[t_a, t_b]$. The interval endpoints are sampled from a distribution p_{mf} over pairs satisfying

$$0 \leq t_a < t_b \leq 1, \quad \Delta t = t_b - t_a.$$

For a training sample (x_0, x_1, c) , the student input at the start of the interval is

$$x_{t_a} = (1 - t_a)x_0 + t_ax_1.$$

Starting from

$$x_{t_a}^{\text{T,cfg}} = x_{t_a},$$

we generate a teacher trajectory over $[t_a, t_b]$ using the frozen teacher with CFG scale γ_{mf} :

$$\frac{dx_t^{\text{T,cfg}}}{dt} = v_{\theta_T}^{\text{cfg}} \left(x_t^{\text{T,cfg}}, t, c; \gamma_{\text{mf}} \right), \quad t \in [t_a, t_b].$$

The teacher mean-velocity target is then approximated by the finite difference

$$\bar{v}_{t_a \rightarrow t_b}^{\text{T,cfg}} \approx \frac{x_{t_b}^{\text{T,cfg}} - x_{t_a}^{\text{T,cfg}}}{t_b - t_a}, \quad (13)$$

where the approximation sign reflects the numerical integration used to obtain $x_{t_b}^{\text{T,cfg}}$.

The student uses the same DiT backbone as the teacher, adds a duration embedder for Δt , and predicts the interval mean velocity with a single conditional forward pass:

$$v_{\phi}(x_{t_a}, t_a, \Delta t, c).$$

It is trained with

$$\begin{aligned}\mathcal{L}_{\text{mv}} &= \mathbb{E}_{(x_0, x_1, c), (t_a, t_b)} [w_{\text{mv}} \ell_{\text{mv}}], \\ \ell_{\text{mv}} &= \left\| v_{\phi}(x_{t_a}, t_a, \Delta t, c) - \bar{v}_{t_a \rightarrow t_b}^{\text{T, cfg}} \right\|_2^2.\end{aligned}\tag{14}$$

We use the adaptive per-sample weight

$$w_{\text{mv}} = (\text{sg}(\ell_{\text{mv}}) + \varepsilon_{\text{mv}})^{-1/2},\tag{15}$$

where $\varepsilon_{\text{mv}} > 0$ is a small numerical constant. The stop-gradient in w_{mv} prevents gradients from flowing through the adaptive weighting term.

Because CFG is fused into the teacher target, the student directly matches the teacher’s CFG-guided mean-velocity prediction with a single conditional forward pass at inference, avoiding the separate conditional and unconditional evaluations required by standard CFG while preserving the corrected teacher behavior.

3 Experiments

This section summarizes the `dots.tts` training recipe and evaluates four axes: foundational quality, multilingual coverage, cross-lingual voice cloning, and expressiveness. We use Seed-TTS-Eval, the MiniMax-Speech multilingual test set, CV3-Eval, and EmergentTTS-Eval.

3.1 Data

In total, the backbone is trained on 1.5M hours of audio across speech, captioned speech, and a small fraction of general audio, drawn from three sources: an in-house Chinese/English speech pool, a curated mixture of open-source TTS and ASR corpora, and a small caption-paired set, described in turn below.

In-house data. The bulk of our training audio comes from an internal Chinese and English speech corpus. A unified preprocessing stack applies vocal enhancement, source separation, speaker-aware diarization, and language-routed ASR: Whisper-Large-v3 (Radford et al., 2023) for English and most other languages, and Paraformer (Gao et al., 2022) for Mandarin Chinese. We then filter clips with cross-ASR consistency, effective-bandwidth estimation, UTMOS, and intra-clip x-vector variance. After filtering, the set contains approximately 1.2M hours of cleaned, transcribed, and speaker-organized speech for backbone training.

Open-source corpora. Alongside the in-house pool, we incorporate a curated mixture of open-source TTS and ASR corpora that meet our quality bar after re-running the same scoring stack. The mixture includes Emilia, LibriTTS-R, HiFi-TTS, HiFi-TTS-2, WenetSpeech4TTS, AISHELL-3, Magicdata, MLS, MSR-86K, IndicVoices-R, EuroSpeech, WaxalNLP-TTS, and FLEURS, totalling roughly 300K hours and contributing the bulk of our non-CJK language coverage.

Caption-style data. To bootstrap natural-language style control and general-audio generation, we also curate a small caption-paired set. It combines a sample of AutoACD (Sun et al., 2024) with natural-language captions and an in-house subset of open-source corpora augmented with Gemini-generated descriptions of speaker traits, emotion, delivery, and acoustic environment. This caption-paired set totals approximately 7K hours.

3.2 Training

The full training pipeline proceeds in three stages: pretraining on the mixture of Section 3.1, flow-matching-native post-training, and MeanFlow distillation for low-NFE inference. Detailed recipes for each stage are deferred to the following subsections.

3.2.1 AudioVAE

The AudioVAE is trained on 48 kHz audio drawn from the in-house and open-source pools of Section 3.1, augmented with a small general-audio share so the latent covers non-speech sounds as well. Optimization uses AdamW with $\beta = (0.8, 0.99)$, $\epsilon = 10^{-6}$, and an exponential learning-rate decay from 10^{-4} to 10^{-6} . Stage 1 runs for 500K steps on 9.6-second cropped segments. Stage 2 runs for a further 200K steps and matches the latent against the 23rd-layer hidden representation of WavLM as the frame-level teacher. The resulting AudioVAE is frozen for the rest of this section.

3.2.2 Pretraining

Pretraining proceeds in three stages (modality alignment, general training, and annealing) that progressively widen the trainable parameter set, the training pool, and the data difficulty. Throughout all three stages we use the AdamW optimizer under a single Warmup-Stable-Decay (WSD) schedule: each stage opens with its own linear warmup from 2×10^{-6} over the first 1% of that stage’s steps to the shared peak learning rate of 2×10^{-4} , and the decay phase is deferred to the final annealing stage, where the learning rate is linearly decayed from 2×10^{-4} to 3×10^{-5} .

Stage 1: modality alignment. The first stage opens a usable channel between the audio representation and the LLM’s semantic space. We freeze the LLM backbone and update only the semantic encoder and the AR-FM head on an internal GPU cluster with a global batch size of approximately 0.5 hours of audio, training for 100K optimization steps. We restrict the data to Emilia in this stage: pilot runs on the full pretraining mixture were severely unstable, and we attribute this to the cost of routing gradients through encoders that have not yet aligned with a frozen LLM. We also observe that, despite the LLM being held fixed, the backbone already produces intelligible speech end-to-end. Pronunciation is frequently incorrect and the resulting audio scores around 42% WER on Seed-TTS-Eval, but the alignment channel is unambiguously open before any LLM parameter is touched.

Stage 2: general training. Once the modality channel is established, we unfreeze every module and train end-to-end on the full data mixture of Section 3.1 without any explicit reweighting between sources. This stage runs with a global batch size of approximately 8 hours of audio for 700K steps, corresponding to four epochs over the corpus. Its role is to build the text-token-to-audio mapping at scale and to absorb the bulk of the cross-lingual, multi-domain coverage of the training pool.

Stage 3: annealing. The final stage anneals on a higher-quality subset obtained by re-filtering the general training pool with a tighter threshold on the in-house quality score described in Section 3.1. This stage runs for 100K steps, approximately one epoch over the filtered subset, during which the WSD schedule enters its decay phase and the learning rate is linearly annealed from the peak 2×10^{-4} down to 3×10^{-5} . All evaluation numbers reported under the **dots.tts** (*Pretrain*) rows in the remainder of this section are produced with the checkpoint at the end of this stage.

3.2.3 Self-corrective alignment

The self-corrective alignment stage updates only the DiT acoustic generator in the AR-FM head. The LLM, semantic encoder, AudioVAE, and speaker encoder are kept frozen. We use the self-corrective alignment objective in Equation (12) on the same multilingual / multi-dialect training pool as pretraining, filtered to high-quality utterances. Training runs with a global batch containing 4 hours of audio for 50K optimization steps with a 5K-step linear warmup to a peak learning rate of 3×10^{-5} , followed by cosine decay to 2×10^{-6} . The auxiliary correction weight is $\lambda_{\text{aux}} = 1.0$. The one-step rollout uses CFG scale $\gamma_{\text{soar}} = 1.2$, and each main sample generates $N = 6$ auxiliary correction states. Integration times follow a logit-normal sampling distribution under the noise-to-data convention of Section 2.6.3.

This stage is reward-free: it does not use a reward model, human preference model, or external acoustic teacher. Instead, the current DiT constructs its own detached off-trajectory states with Equations (8) and (9), and the supervised correction target in Equation (10) teaches the same DiT to steer those states back toward the clean latent endpoint. In practice this directly addresses the multi-step ODE mismatch between pretraining and inference, where small velocity errors accumulate across AR patches. Throughout this paper, **dots.tts (SOAR)** denotes the checkpoint obtained after this self-corrective post-training stage.

3.2.4 MeanFlow distillation

After self-corrective alignment, we freeze the corrected DiT as the teacher and initialize a student DiT from it. The student keeps the same backbone and adds only the interval-duration embedder used by Equation (14); all compatible parameters are copied from the teacher, while the duration embedder is initialized separately. Teacher trajectories are constructed with a 16-step Euler solver, and CFG is applied when forming the teacher target so that the student learns the CFG-guided mean-velocity prediction directly.

The MeanFlow (MF) stage uses the same setup and 8-hour global audio batch, and is trained for 50K steps with a 5K-step warmup and cosine decay to 2×10^{-6} . We use a larger peak learning rate of 1×10^{-4} because the student learns an interval-conditioned mean-velocity objective rather than the original instantaneous velocity. Intervals are sampled from a log-normal distribution with mean -0.4 and standard deviation 1.0 in the same noise-to-data time parameterization. Following the MeanFlow recipe, anchor samples are mixed with probability 0.5 for optimization stability. At inference, the student updates intervals as $x_{t_b} = x_{t_a} + \Delta t v_\phi(x_{t_a}, t_a, \Delta t, c)$, with $\Delta t = t_b - t_a$, and needs only one conditional forward pass per step because CFG has already been fused into the distillation target. The resulting MeanFlow-distilled checkpoint is denoted as **dots.tts (MF)** and is used for all evaluations labeled **dots.tts (MF)** in the remainder of this section.

3.3 Evaluation

3.3.1 Setup

Metrics. Throughout this section, content fidelity is measured by ASR-based word/character error rate (WER for English and other non-Chinese languages, CER reported as WER for Chinese for table compatibility), and speaker similarity (SIM) is the cosine similarity between WavLM-SV embeddings of the generated speech and the corresponding reference prompt. We follow the de facto ASR convention used by recent reports: Whisper-Large-v3 for English and multilingual benchmarks, Paraformer for Mandarin Chinese, and the per-benchmark default ASR for the remaining languages on the MiniMax multilingual set. All numbers reported under the **dots.tts (Pretrain)** row are produced with the pretrained checkpoint from Section 3.2.2; the MeanFlow-distilled student is reported in the inference-efficiency section (Section 3.4).

Table 1: Speech reconstruction performance on *LibriSpeech test-other*. FPS is the temporal frame rate of the underlying representation. **Bold** marks the best non-oracle entry per column. “–” indicates a metric not reported by the corresponding source.

Model	Sample Rate	FPS	PESQ $\uparrow$		STOI $\uparrow$	UTMOS $\uparrow$	SIM $\uparrow$	WER(%) $\downarrow$
			NB	WB				
Ground Truth	–	–	4.55	4.64	1.000	3.50	1.000	4.59
<i>Discrete tokens</i>								
XY-Tokenizer	16 kHz	100	2.80	2.23	0.89	3.46	0.82	6.19
WavTokenizer	16 kHz	75	2.40	1.96	0.87	3.22	0.68	13.35
X-codec2	16 kHz	50	2.83	2.26	0.90	3.64	0.81	6.85
SAC	16 kHz	62.5	2.92	2.39	0.90	3.84	0.85	5.77
<i>Continuous representations</i>								
SemanticVAE	16 kHz	40	3.99	3.80	0.969	3.76	0.963	4.15
MingTok-Audio	16 kHz	50	4.23	4.12	0.981	3.75	0.950	4.27
VibeVoice	24 kHz	7.5	2.85	–	0.823	3.72	–	–
dots.tts VAE	48 kHz	25	4.09	3.95	0.973	3.75	0.969	4.14

Inference configuration. dots.tts-Pretrain and dots.tts-SOAR use 10 Euler solver steps with CFG, i.e., each step performs one conditional and one unconditional DiT forward pass for an effective NFE of 20. In contrast, dots.tts-MF is evaluated at $\text{NFE} \in \{2, 3, 4\}$ with no CFG: the guidance has already been fused into the MeanFlow distillation target, so each step uses a single conditional forward pass. The CFG scale is $\gamma = 1.2$ for CFG in dots.tts-Pretrain and dots.tts-SOAR, and for the teacher target used to train dots.tts-MF (Equation (6)). All runs use float32 and have *torch.compile* disabled.

Baselines. We compare against a broad set of recent zero-shot TTS systems, spanning both open-source releases and commercial products: CosyVoice 2 (Du et al., 2024), CosyVoice 3 (Du et al., 2025), Qwen3-TTS (Qwen Team, 2026), IndexTTS 2 (Zhou et al., 2026), FireRedTTS-2 (Xie et al., 2025), SeedTTS (Seed Team, ByteDance, 2024a), MiniMax-Speech (MiniMax Team, 2025), Fish-Audio S2 (Liao et al., 2026), F5-TTS (Chen et al., 2025a), MegaTTS3 (Jiang et al., 2025), DiTAR (Jia et al., 2025), VibeVoice (Peng et al., 2025), and VoxCPM 2 (VoxCPM Team, 2026). All baseline numbers reported in this section are taken either from the original papers or from the official default inference configuration of the corresponding open-source release; we do not re-tune any baseline.

3.3.2 Reconstruction Quality of the AudioVAE

Before turning to the end-to-end TTS benchmarks, we first measure the AudioVAE’s reconstruction performance. Table 1 reports waveform-reconstruction metrics on *LibriSpeech test-other* against representative discrete neural codecs (XY-Tokenizer (Gong et al., 2025), WavTokenizer (Ji et al., 2024), X-codec2 (Ye et al., 2025b), SAC (Chen et al., 2025b)) and continuous representations (SemanticVAE (Niu et al., 2025), MingTok-Audio (Yan et al., 2025), and the continuous tokenizer of VibeVoice (Peng et al., 2025)). All baseline numbers are taken from the corresponding original publications.

The discrete tokenizers in the upper block trail the continuous representations on PESQ, STOI, SIM, and WER. Within the continuous group, the dots.tts VAE has the highest input bandwidth and second-lowest frame rate in the table, with PESQ, STOI, UTMOS, SIM, and WER all in the top band of the group. The only other low-frame-rate continuous entry, VibeVoice at 7.5 Hz, drops to PESQ-NB 2.85 and STOI 0.823.

At WER 4.14 and SIM 0.969, the latent itself adds almost nothing to end-to-end error, so reconstruction is not a downstream bottleneck. Stage 2 (Section 2.2) further adds WavLM-aligned and multitask super-

Table 2: Zero-shot results on Seed-TTS-Eval. **Bold** marks the best per column.

Model	Params	test-en		test-zh		test-zh-hard		Average	
		WER(%)↓	SIM↑	WER↓	SIM↑	WER↓	SIM↑	WER↓	SIM↑
CosyVoice 3	1.5B	2.22	72.0	1.12	78.1	5.83	75.8	3.06	75.3
F5-TTS	0.3B	2.00	67.0	1.53	76.0	8.67	71.3	4.10	71.4
FireRedTTS 2	1.5B	1.95	66.5	1.14	73.6	8.98	70.3	4.02	70.1
IndexTTS 2	1.5B	2.23	70.6	1.03	76.5	7.12	75.5	3.46	74.2
MegaTTS 3	0.5B	2.79	77.1	1.52	79.0	—	—	—	—
MiniMax-Speech	—	1.65	69.2	0.83	78.3	—	—	—	—
Qwen3-TTS	1.7B	1.23	71.7	1.22	77.0	6.76	74.8	3.07	74.5
Seed-TTS	—	2.25	76.2	1.12	79.6	7.59	77.6	3.65	77.8
DiTAR	0.6B	1.69	73.5	1.02	75.3	—	—	—	—
VibeVoice	1.5B	3.04	68.9	1.16	74.4	—	—	—	—
VoxCPM 2	2B	1.84	75.3	0.97	79.5	8.13	75.3	3.65	76.7
dots.tts (Pretrain)	2B	1.34	76.8	0.96	80.5	6.46	79.2	2.92	78.8
dots.tts (SOAR)	2B	1.30	77.1	0.94	81.0	6.60	79.5	2.95	79.2
– MF, NFE=4	2B	1.29	76.2	0.94	80.0	6.60	78.5	2.94	78.2
– MF, NFE=3	2B	1.41	75.9	1.02	79.9	7.19	78.6	3.21	78.1
– MF, NFE=2	2B	1.51	75.2	1.04	79.1	7.74	76.7	3.43	77.0

vision that makes the latent learnable as an LLM generation target. Both contribute to the downstream TTS results in the remainder of this section.

3.3.3 Seed-TTS-Eval

Seed-TTS-Eval is a widely used zero-shot voice-cloning benchmark and the primary point of comparison in this report. We evaluate on all three subsets (test-en, test-zh, test-zh-hard) under the standard zero-shot protocol: the model is conditioned on an approximately 3-second reference prompt unseen during training, generates the target transcript, and is scored with the benchmark’s reference ASR and WavLM-SV similarity.

Table 2 reports the results. **dots.tts** takes the best average on both metrics. On SIM, the SOAR row leads at 79.2, 1.4 above the next-best baseline Seed-TTS (77.8) and 2.5 above VoxCPM 2 (76.7). On WER, the Pretrain (2.92%), SOAR (2.95%), and MF NFE = 4 (2.94%) rows all sit below every reported baseline, with CosyVoice 3 next at 3.06%.

The post-training stage of Section 3.2.3 corrects the multi-step ODE mismatch between pretraining and inference. The Pretrain → SOAR rows in the table reflect this correction with broadly improved numbers. MeanFlow distillation at NFE = 4 keeps WER within 0.01 of SOAR on every column at a cost of about one point of SIM. We also report MF NFE = 2 and NFE = 3 in the table for comparison, with NFE = 4 as the best operating point of the three.

3.3.4 MiniMax-multilingual Test Set

The MiniMax-Speech multilingual test set covers 24 languages with 100 utterances per language and two MCV reference speakers per language, and is widely used as a multilingual zero-shot benchmark. We evaluate the full 24-language set with the benchmark’s per-language ASR.

Table 3 reports per-language WER and SIM. **dots.tts** (SOAR) leads the average SIM at 83.9, 1.6 above the next-best baseline VoxCPM 2 (82.3), and a **dots.tts** variant takes the per-language SIM lead outright on 19 of 24 languages and ties on 2 more.

The picture on WER is mixed. **dots.tts** is competitive on most languages, but the average is pulled up

Table 3: Per-language WER / SIM on the MiniMax-Speech multilingual test set. **Bold** marks the best per metric per row. The Average row is computed only over the languages where each system reports a number. *Cantonese WER reflects an ASR-faithfulness floor common to all systems in the table; the SIM column remains comparable.

Language	MiniMax	ElevenLabs	Fish-Audio S2	VoxCPM 2	dots.tts (Pre.)	dots.tts (SOAR)	dots.tts (MF ₄)
	WER(%) / SIM	WER / SIM	WER / SIM	WER / SIM	WER / SIM	WER / SIM	WER / SIM
Arabic	1.67 / 73.6	1.67 / 70.6	3.50 / 75.0	13.05 / 79.1	37.91 / 77.5	36.19 / 79.1	39.65 / 77.6
Cantonese*	34.11 / 77.8	51.51 / 67.0	30.67 / 80.5	38.58 / 83.5	37.91 / 84.7	42.32 / 85.0	37.82 / 84.0
Chinese	2.25 / 78.0	16.03 / 67.7	0.73 / 81.6	1.14 / 82.5	1.08 / 82.3	0.77 / 82.5	1.01 / 81.8
Czech	3.88 / 79.6	2.11 / 68.5	2.84 / 79.8	24.13 / 78.3	5.05 / 83.8	4.25 / 84.2	5.67 / 83.9
Dutch	1.14 / 73.8	0.80 / 68.0	0.99 / 73.0	0.91 / 80.8	1.20 / 81.4	1.39 / 82.2	1.30 / 82.1
English	2.16 / 75.6	2.34 / 61.3	1.62 / 79.7	2.29 / 85.4	1.06 / 86.9	1.03 / 87.5	1.09 / 86.9
Finnish	4.67 / 83.5	2.96 / 75.9	3.33 / 81.9	2.63 / 89.0	3.44 / 88.0	4.08 / 88.3	3.61 / 88.3
French	4.10 / 62.8	5.22 / 53.5	3.05 / 69.8	4.53 / 73.5	3.82 / 78.2	3.56 / 78.6	3.26 / 78.5
German	1.91 / 73.3	0.57 / 61.4	0.55 / 76.7	0.68 / 80.3	1.03 / 79.5	1.70 / 80.6	0.91 / 79.5
Greek	2.02 / 82.6	0.99 / 73.3	5.74 / 79.5	2.84 / 86.0	2.97 / 87.6	3.00 / 87.6	3.19 / 87.3
Hindi	6.96 / 81.8	5.83 / 73.0	14.64 / 82.1	19.70 / 85.6	14.32 / 84.5	14.24 / 84.7	14.75 / 84.8
Indonesian	1.24 / 72.9	1.06 / 66.0	1.46 / 76.3	1.08 / 80.0	2.71 / 80.8	2.96 / 80.8	3.91 / 81.2
Italian	1.54 / 69.9	1.74 / 57.9	1.27 / 74.7	1.56 / 78.0	3.16 / 84.5	3.12 / 84.7	2.16 / 84.3
Japanese	3.52 / 77.6	10.65 / 73.8	2.76 / 79.6	4.63 / 82.8	7.16 / 83.1	5.28 / 83.7	5.17 / 83.1
Korean	1.75 / 77.6	1.87 / 70.0	1.18 / 81.7	1.96 / 83.3	5.30 / 84.3	5.66 / 83.6	3.93 / 84.9
Polish	1.42 / 80.2	0.77 / 72.9	1.26 / 81.9	1.14 / 88.4	2.72 / 87.3	3.59 / 87.8	3.42 / 87.5
Portuguese	1.88 / 80.5	1.33 / 71.1	1.14 / 78.1	1.94 / 83.7	1.64 / 83.1	2.00 / 84.3	2.40 / 83.1
Romanian	2.88 / 80.9	1.35 / 69.9	10.74 / 73.3	21.58 / 79.7	3.36 / 86.2	3.87 / 87.1	3.38 / 86.1
Russian	4.28 / 76.1	3.88 / 67.6	2.40 / 79.0	3.63 / 81.1	3.64 / 83.0	4.28 / 83.2	4.42 / 83.2
Spanish	1.03 / 76.2	1.08 / 61.5	0.91 / 77.6	1.44 / 83.1	0.96 / 83.9	1.27 / 84.0	0.80 / 84.0
Thai	2.70 / 80.0	73.94 / 58.8	4.23 / 78.6	2.96 / 84.0	7.45 / 83.8	7.86 / 83.9	8.03 / 84.2
Turkish	1.52 / 77.9	0.70 / 59.6	0.87 / 83.5	0.82 / 87.1	5.45 / 87.4	4.96 / 87.3	6.20 / 86.8
Ukrainian	1.08 / 73.0	1.00 / 64.7	2.30 / 74.7	6.32 / 79.8	1.61 / 80.5	1.27 / 81.2	1.66 / 80.0
Vietnamese	0.88 / 74.3	73.42 / 36.9	7.41 / 74.0	3.31 / 80.6	3.85 / 80.7	3.89 / 81.6	5.43 / 80.5
Average	2.8 / 76.6	7.5 / 65.5	3.7 / 78.0	5.7 / 82.3	6.6 / 83.5	6.8 / 83.9	6.8 / 83.5

Table 4: Results on CV3-Eval. W = WER, S = SIM.

Model	Monolingual W(%)↓				en→zh		zh→en	
	zh	en	hard-zh	hard-en	W↓	S↑	W↓	S↑
CosyVoice 2	4.08	6.32	12.58	11.96	13.50	63.3	6.47	64.3
CosyVoice 3 (1.5B)	3.91	4.99	9.77	10.55	8.01	66.9	4.32	66.4
Fish-Audio S2	2.65	2.43	9.10	4.40	—	—	—	—
VoxCPM 2	3.65	5.00	8.55	8.48	—	—	—	—
dots.tts (Pretrain)	3.51	5.24	9.69	5.99	10.88	74.6	4.97	71.9
dots.tts (SOAR)	3.71	4.50	9.22	4.49	10.75	75.0	5.66	72.8
dots.tts (MF, NFE=4)	3.95	4.05	9.10	4.37	10.73	73.8	5.24	70.9

by a few low-resource outliers. Since SIM stays in the normal band on these languages, the WER gap points to insufficient BPE token coverage (Section 2.3). Means to close the low-resource WER gap are discussed in Section 5.

MeanFlow distillation (MF₄) matches SOAR at the average level (6.8% / 83.5 vs 6.8% / 83.9), with occasional per-language wins and about 0.4 SIM cost on the average row.

3.3.5 CV3-Eval

CV3-Eval, released with CosyVoice 3, complements the previous two benchmarks with a hard-subset Chinese/English split and, importantly, a *cross-lingual* voice-cloning subset that uses a reference speaker from one language to generate text in another. Cross-lingual cloning is one of the hardest tests of timbre disentanglement.

Table 5: Selected rows from EmergentTTS-Eval, sorted by overall. ★ marks closed-source / commercial systems. **Bold** marks the best per column, underline the best open-source per column. Win-rate is judged head-to-head against gpt-4o-mini-tts by Gemini-2.5-Pro-0506.

Model	Voice	WER(%)↓	Overall↑	Emo.↑	Paraling.↑	Foreign↑	C. Pron.↑	Quest.↑	Syntax↑
Gemini-2.5-Flash-TTS★	Zephyr	10.39	70.7%	95.9%	91.3%	58.5%	55.7%	63.0%	57.9%
Gemini-2.5-Pro-TTS★	Zephyr	11.79	69.3%	86.9%	82.3%	58.2%	64.8%	61.3%	61.8%
gpt-4o-audio-preview★	Ballad	11.87	65.2%	88.8%	82.1%	60.2%	40.4%	57.0%	59.5%
gpt-4o-mini-tts★	Alloy	10.76	56.3%	59.2%	58.8%	57.3%	52.4%	52.7%	57.1%
<i>BASELINE: gpt-4o-mini-tts</i>	Alloy	10.61	50.0%	—	—	—	—	—	—
dots.tts (Pretrain)	basic_ref_en	10.86	<u>49.2%</u>	<u>72.7%</u>	54.7%	<u>39.5%</u>	18.0%	48.4%	58.4%
dots.tts (MF4)	basic_ref_en	11.75	47.9%	59.8%	55.2%	36.3%	16.7%	50.5%	64.8%
dots.tts (SOAR)	basic_ref_en	<u>10.45</u>	47.6%	63.9%	52.7%	39.4%	16.4%	47.0%	65.7%
Qwen3-TTS	basic_ref_en	17.32	42.8%	39.8%	50.7%	25.4%	<u>30.0%</u>	48.9%	60.4%
HumeAI★	—	12.85	42.7%	61.6%	36.9%	34.6%	34.3%	43.2%	44.6%
Qwen3-TTS	Ryan	19.65	42.3%	60.5%	<u>62.7%</u>	17.1%	9.8%	<u>56.4%</u>	43.0%
VoxCPM2	basic_ref_en	11.84	41.1%	42.3%	44.1%	33.3%	18.6%	53.4%	52.3%
MiniMax/speech-02-hd★	EN-narr	10.02	36.6%	40.9%	34.3%	34.3%	16.3%	47.3%	43.9%
11Labs Multilingual v2★	Brian	11.19	33.9%	30.4%	45.5%	35.5%	14.5%	39.5%	35.5%
F5-TTS	basic_ref_en	16.47	15.3%	26.8%	21.6%	1.8%	1.4%	14.8%	23.8%

Table 4 reports the CV3-Eval numbers. **dots.tts** (MF₄) takes the hard-en lead at 4.37%, while FishAudio S2 and VoxCPM 2 lead the other three monolingual subsets, with **dots.tts** competitive but not leading. The Pretrain → SOAR drop on hard-en (5.99 → 4.49) is the largest SOAR-stage gain we observe across this section’s benchmarks, and MF₄ inherits it through distillation.

On the cross-lingual subset, **dots.tts** (SOAR) leads SIM in both directions at 75.0 (en→zh) and 72.8 (zh→en), 6 to 8 absolute points above CosyVoice 3 (66.9 / 66.4). Cross-lingual WER trails CosyVoice 3 (zh→en 5.66% vs 4.32%, en→zh 10.75% vs 8.01%).

3.3.6 EmergentTTS-Eval

EmergentTTS-Eval uses a Gemini-2.5-Pro-0506 audio judge to compare systems head-to-head against a fixed gpt-4o-mini-tts reference, across six expressiveness-oriented scenarios: Emotions, Foreign Words, Paralinguistics, Complex Pronunciation, Questions, and Syntactic Complexity. We report both ASR-based WER and the judge’s preference win-rate (higher is better) on the canonical zero-shot configuration of **dots.tts**. For open-source systems we use *basic_ref_en*¹ as the common zero-shot voice-clone prompt, with all runs in continuation voice-cloning mode.

Table 5 reports the results. **dots.tts** (Pretrain) leads the open-source field on overall win-rate at 49.2%, with SOAR and MF₄ close behind (47.6%, 47.9%) and all three variants near the gpt-4o-mini-tts baseline (50% by construction). SOAR also has the lowest WER among open-source systems at 10.45%.

On Syntactic Complexity, SOAR reaches 65.7%, the top score across both open- and closed-source systems (next Gemini-2.5-Pro 61.8%). The Pretrain → SOAR uplift on this column is +7.3 points, the largest single-column gain we observe from the SOAR stage (Section 3.2.3). The same shift drops Emotions (72.7% → 63.9%) and Paralinguistics (54.7% → 52.7%), suggesting SOAR tightens text faithfulness at the cost of expressiveness. On Emotions, Pretrain leads the open-source field at 72.7%. Closed-source systems on this column (87–96%) use carefully tuned built-in voices, while **dots.tts** runs zero-shot voice cloning from a basic reference clip.

On the remaining columns **dots.tts** is mid-field. The weakest are Complex Pronunciation (16–18%) and Foreign Words (36–40%), both stressing rare or out-of-distribution lexical items.

¹https://github.com/SWivid/F5-TTS/blob/main/src/f5_tts/infer/examples/basic/basic_ref_en.wav

3.4 Efficiency

We report inference-time efficiency on the Seed-TTS-Eval test set under vllm-omni (Yin et al., 2026). The LLM runs on vLLM with continuous batching and paged-KV attention, while the AR-FM head and the semantic encoder are JIT-compiled with *torch.compile*. *Time to first-packet latency* (TTFP, ms) is the wall-clock latency from request arrival to the first emitted audio packet, and *RTF* is the ratio of generation wall-clock time to synthesized-audio duration.

All numbers are measured on a single NVIDIA H800 GPU using the MeanFlow-distilled student (Section 3.2.4), with the AR-FM head running at NFE = 4 per audio patch. In text-only synthesis, **dots.tts** runs comfortably in real time, reaching RTF 0.231 in plain mode and RTF 0.245 in 1T1A interleaved mode. Interleaving reduces first-packet latency from 85.4 ms to 54.4 ms by consuming the upstream LLM token stream as it is decoded, so audio generation can start once the large language model begins emitting its response.

4 Conclusion

We presented **dots.tts**, a 2B-parameter fully continuous, end-to-end autoregressive TTS system that targets the two open problems we identified for the continuous-AR paradigm: long-range error accumulation during AR rollouts, and an immature post-training stack relative to the discrete-token cascade. The backbone is decomposed into a semantic encoder $\rightarrow$ LLM $\rightarrow$ autoregressive flow-matching head, with the audio-side input to the LLM restricted to a 6.25 Hz semantic summary of each newly generated patch, not the raw VAE latent. This decoupling keeps the LLM operating on a compact semantic view of the history, important for long-rollout stability. On the synthesis side, a high-fidelity continuous AudioVAE (trained with a WavLM-alignment loss and a multitask downstream block so that the high-rate continuous latent remains learnable for the downstream LLM) preserves the timbral and paralinguistic detail that low-bitrate discrete codecs typically flatten. We complement the pretraining recipe with Self-corrective alignment, a reward-free, flow-matching-native post-training stage that teaches the acoustic DiT to recover from its own inference-time errors, and with CFG-aware MeanFlow distillation, which enables few-step generation with a single conditional model evaluation per step.

Trained on 1.5M hours of speech, **dots.tts** attains state-of-the-art average WER (2.92) and SIM (79.2) on Seed-TTS-Eval, the highest average speaker similarity (83.9) on the 24-language MiniMax multilingual benchmark, and leading hard-subset and cross-lingual results on CV3-Eval. On EmergentTTS-Eval, it takes the top Syntactic Complexity score in the table (65.7%, ahead of every closed-source system) and is the strongest open-source system on Emotions (72.7%). Combined with CFG-aware MeanFlow distillation and a 1-text-1-audio interleaved streaming layout, the same backbone reaches 54 ms TTFB at RTF 0.245 on a single H800, making it deployable for real-time and conversational use cases. We release the full training and inference code, together with the pretrained, self-corrective-aligned, and MeanFlow-distilled checkpoints, under the Apache 2.0 license as a reproducible reference stack for continuous-AR TTS.

5 Limitations

Several limitations remain in the current system. Feeding the LLM raw BPE rather than phonemes inherits its text capabilities at the cost of higher data appetite, which on the script-divergent and under-represented languages of Section 3.3.4 (Arabic, Hindi, Turkish, Vietnamese) drives the low-resource WER gap visible there, and on the Foreign Words and Complex Pronunciation scenarios of Section 3.3.6 hits the same coverage limit on loanwords, technical terms, and proper names. Expanding the multilingual pretraining mix on those languages, adding a small phoneme-side auxiliary input, and inserting

a language-balanced post-training stage are direct levers to address this. The released system is also evaluated only under canonical zero-shot conditions, without explicit style or instruction control. An instruction-tuned variant built on the caption-paired data of Section 3.1 is a natural next step. Although the AudioVAE is in principle modality-agnostic, the backbone is trained on a speech-heavy mixture, so singing and unified speech-and-sound generation are not covered in this release. Finally, high-fidelity zero-shot voice cloning carries well-known misuse risks. The released checkpoints are intended for research and authorized deployment, and we encourage downstream users to combine them with consent-aware reference-audio policies, robust synthetic-speech detection, and content watermarking.

Contributors

`dots.tts` is jointly developed by dots, Xiaohongshu Inc.¹ and the X-LANCE Lab² at the School of Computer Science, Shanghai Jiao Tong University. Both teams contributed to the model design, the training recipe, and the evaluation, and the checkpoints, code, and this report are released jointly.

Authors Shi Lian¹, Changtao Li¹, Bohan Li², Hankun Wang², Da Zheng¹, Junfeng Tian¹, Yufeng Ma¹, Colin Zhang¹, Kai Yu².

References

- Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.
- Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, et al. Voicebox: Text-guided multilingual universal speech generation at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- Yushen Chen, Zhikang Niu, Ziyang Ma, Keqi Deng, Chunhui Wang, Jian Zhao, Kai Yu, and Xie Chen. F5-tts: A fairytaler that fakes fluent and faithful speech with flow matching. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL): Long Papers*, pages 6255–6271, 2025a.
- Han Zhu, Lingxuan Ye, Wei Kang, Zengwei Yao, Liyong Guo, Fangjun Kuang, Zhifeng Han, Weiji Zhuang, Long Lin, and Daniel Povey. Omnivoice: Towards omnilingual zero-shot text-to-speech with diffusion language models. *arXiv preprint arXiv:2604.00688*, 2026.
- Detai Xin, Shujie Hu, Chengzuo Yang, Chen Huang, Guoqiao Yu, Guanglu Wan, and Xunliang Cai. Longcat-audiodit: High-fidelity diffusion text-to-speech in the waveform latent space. *arXiv preprint arXiv:2603.29339*, 2026.
- Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.
- Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Oral.
- Zhihao Du, Changfeng Gao, Yuxuan Wang, Fan Yu, Tianyu Zhao, Hao Wang, Xiang Lv, Hui Wang, Chongjia Ni, Xian Shi, et al. Cosyvoice 3: Towards in-the-wild speech generation via scaling-up and post-training. *arXiv preprint arXiv:2505.17589*, 2025.
- Qwen Team. Qwen3-tts technical report. *arXiv preprint arXiv:2601.15621*, 2026.

- Zhen Ye, Xinfu Zhu, Chi-Min Chan, Xinsheng Wang, Xu Tan, Jiahe Lei, Yi Peng, Haohe Liu, Yizhu Jin, Zheqi Dai, et al. Llasa: Scaling train-time and inference-time compute for llama-based speech synthesis. *arXiv preprint arXiv:2502.04128*, 2025a.
- Wei Deng, Siyi Zhou, Jingchen Shu, Jinchao Wang, and Lu Wang. Indextts: An industrial-level controllable and efficient zero-shot text-to-speech system. *arXiv preprint arXiv:2502.05512*, 2025.
- Siyi Zhou, Yiquan Zhou, Yi He, Xun Zhou, Jinchao Wang, Wei Deng, and Jingchen Shu. Indextts2: A breakthrough in emotionally expressive and duration-controlled auto-regressive zero-shot tts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.
- Seed Team, ByteDance. Seed-tts: A family of high-quality versatile speech generation models. *arXiv preprint arXiv:2406.02430*, 2024a.
- Kangxiang Xia, Xinfu Zhu, Jixun Yao, Wenjie Tian, Wenhao Li, and Lei Xie. Kall-e: Autoregressive speech synthesis with next-distribution prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.
- Dongya Jia, Zhuo Chen, Jiawei Chen, Chenpeng Du, Jian Wu, Jian Cong, Xiaobin Zhuang, Chumin Li, Zhen Wei, Yuping Wang, et al. Ditar: Diffusion transformer autoregressive modeling for speech generation. In *International Conference on Machine Learning (ICML)*, 2025.
- Zhiliang Peng, Jianwei Yu, Wenhui Wang, Yaoyao Chang, Yutao Sun, Li Dong, Yi Zhu, Weijiang Xu, Hangbo Bao, Zehua Wang, et al. Vibevoice technical report. *arXiv preprint arXiv:2508.19205*, 2025.
- Yixuan Zhou, Guoyang Zeng, Xin Liu, Xiang Li, Renjie Yu, Ziyang Wang, Runchuan Ye, Weiyue Sun, Jiancheng Gui, Kehan Li, et al. Voxcpm: Tokenizer-free tts for context-aware speech generation and true-to-life voice cloning. *arXiv preprint arXiv:2509.24650*, 2025.
- Zhijun Liu, Shuai Wang, Sho Inoue, Qibing Bai, and Haizhou Li. Autoregressive diffusion transformer for text-to-speech synthesis. *arXiv preprint arXiv:2406.05551*, 2024.
- Xingchen Song, Di Wu, Dinghao Zhou, Pengyu Cheng, Hongwu Ding, Yunchao He, Jie Wang, Shengfan Shen, Sixiang Lv, Lichun Fan, et al. Any2speech: Borderless long speech synthesis. *arXiv preprint arXiv:2603.19798*, 2026.
- Bohan Li, Shi Lian, Hankun Wang, Yiwei Guo, Yu Xi, Zhihan Li, Da Zheng, Colin Zhang, and Kai Yu. Holitok: A continuous holistic tokenization with robust dual capabilities of speech generation and understanding. *arXiv preprint arXiv:2605.29948*, 2026.
- Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.
- You Qin, Linqing Wang, Hao Fei, Roger Zimmermann, Liefeng Bo, Qinglin Lu, and Chunyu Wang. Soar: Self-correction for optimal alignment and refinement in diffusion models. *arXiv preprint arXiv:2604.12617*, 2026.
- Seed Team, ByteDance. Seed-tts-eval benchmark. Introduced in Seed-TTS, arXiv:2406.02430, 2024b.
- MiniMax Team. Minimax-speech: Intrinsic zero-shot text-to-speech with a learnable speaker encoder. *arXiv preprint arXiv:2505.07916*, 2025.
- Ruskin Raj Manku, Yuzhi Tang, Xingjian Shi, Mu Li, and Alex Smola. Emergenttts-eval: Evaluating tts models on complex prosodic, expressiveness, and linguistic challenges using model-as-a-judge. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2025.

- Sang-gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon. Bigvgan: A universal neural vocoder with large-scale training. In *International Conference on Learning Representations (ICLR)*, 2023.
- Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.
- William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
- Hui Wang, Siqi Zheng, Yafeng Chen, Luyao Cheng, and Qian Chen. Cam++: A fast and efficient network for speaker verification using context-aware masking. In *Interspeech*, 2023.
- Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021.
- Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning (ICML)*, 2023.
- Zhifu Gao, Shiliang Zhang, Ian McLoughlin, and Zhijie Yan. Paraformer: Fast and accurate parallel transformer for non-autoregressive end-to-end speech recognition. In *Interspeech*, 2022.
- Luoyi Sun, Xuenan Xu, Mengyue Wu, and Weidi Xie. Auto-acd: A large-scale dataset for audio-language representation learning. In *Proceedings of the 32nd ACM International Conference on Multimedia (MM)*, 2024.
- Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, et al. Cosyvoice 2: Scalable streaming speech synthesis with large language models. *arXiv preprint arXiv:2412.10117*, 2024.
- Kun Xie, Feiyu Shen, Junjie Li, Fenglong Xie, Xu Tang, and Yao Hu. Fireredtts-2: Towards long conversational speech generation for podcast and chatbot. *arXiv preprint arXiv:2509.02020*, 2025.
- Shijia Liao, Yuxuan Wang, Songting Liu, Yifan Cheng, Ruoyi Zhang, Tianyu Li, Shidong Li, Yisheng Zheng, Xingwei Liu, Qingzheng Wang, et al. Fish audio s2 technical report. *arXiv preprint arXiv:2603.08823*, 2026.
- Ziyue Jiang, Yi Ren, Ruiqi Li, Shengpeng Ji, Boyang Zhang, Zhenhui Ye, Chen Zhang, Jionghao Bai, Xiaoda Yang, Jialong Zuo, et al. Megatts 3: Sparse alignment enhanced latent diffusion transformer for zero-shot speech synthesis. *arXiv preprint arXiv:2502.18924*, 2025.
- VoxCPM Team. Voxcpm2: Tokenizer-free tts for multilingual speech generation, creative voice design, and true-to-life cloning. *GitHub*, 2026.
- Yitian Gong, Luozhijie Jin, Ruifan Deng, Dong Zhang, Xin Zhang, Qinyuan Cheng, Zhaoye Fei, Shimin Li, and Xipeng Qiu. Xy-tokenizer: Mitigating the semantic-acoustic conflict in low-bitrate speech codecs. *arXiv preprint arXiv:2506.23325*, 2025.
- Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, et al. Wavtokenizer: An efficient acoustic discrete codec tokenizer for audio language modeling. *arXiv preprint arXiv:2408.16532*, 2024.
- Zhen Ye, Xinfu Zhu, Chi-Min Chan, Xinsheng Wang, Xu Tan, Jiahe Lei, Yi Peng, Haohe Liu, Yizhu Jin, Zheqi Dai, et al. Llasa: Scaling train-time and inference-time compute for llama-based speech synthesis. *arXiv preprint arXiv:2502.04128*, 2025b.
- Wenxi Chen, Xinsheng Wang, Ruiqi Yan, Yushen Chen, Zhikang Niu, Ziyang Ma, Xiquan Li, Yuzhe

- Liang, Hanlin Wen, Shunshun Yin, et al. Sac: Neural speech codec with semantic-acoustic dual-stream quantization. *arXiv preprint arXiv:2510.16841*, 2025b.
- Zhikang Niu, Shujie Hu, Jeongsoo Choi, Yushen Chen, Peining Chen, Pengcheng Zhu, Yunting Yang, Bowen Zhang, Jian Zhao, Chunhui Wang, et al. Semantic-vae: Semantic-alignment latent representation for better speech synthesis. *arXiv preprint arXiv:2509.22167*, 2025.
- Canxiang Yan, Chunxiang Jin, Dawei Huang, Haibing Yu, Han Peng, Hui Zhan, Jie Gao, Jing Peng, Jingdong Chen, Jun Zhou, et al. Ming-uniaudio: Speech llm for joint understanding, generation and editing with unified representation. *arXiv preprint arXiv:2511.05516*, 2025.
- Peiqi Yin, Jiangyun Zhu, Han Gao, Chenguang Zheng, Yongxiang Huang, Taichang Zhou, Ruirui Yang, Weizhi Liu, Weiqing Chen, Canlin Guo, et al. vllm-omni: Fully disaggregated serving for any-to-any multimodal models. *arXiv preprint arXiv:2602.02204*, 2026.