CosyVoice 2: Scalable Streaming Speech Synthesis with Large Language Models

Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao
Yexin Yang, Changfeng Gao, Hui Wang, Fan Yu, Huadai Liu, Zhengyan Sheng
Yue Gu, Chong Deng, Wen Wang, Shiliang Zhang, Zhijie Yan, Jingren Zhou*

Alibaba Group, China

{neo.dzh,sly.zsl}@alibaba-inc.com

Abstract

In our previous work, we introduced CosyVoice, a multilingual speech synthesis model based on supervised discrete speech tokens. By employing progressive semantic decoding with two popular generative models, language models (LMs) and Flow Matching, CosyVoice demonstrated high prosody naturalness, content consistency, and speaker similarity in speech in-context learning. Recently, significant progress has been made in multi-modal large language models (LLMs), where the response latency and real-time factor of speech synthesis play a crucial role in the interactive experience. Therefore, in this report, we present an improved streaming speech synthesis model, CosyVoice 2, which incorporates comprehensive and systematic optimizations. Specifically, we introduce finite-scalar quantization to improve the codebook utilization of speech tokens. For the text-speech LM, we streamline the model architecture to allow direct use of a pre-trained LLM as the backbone. In addition, we develop a chunk-aware causal flow matching model to support various synthesis scenarios, enabling both streaming and non-streaming synthesis within a single model. By training on a large-scale multilingual dataset, CosyVoice 2 achieves human-parity naturalness, minimal response latency, and virtually lossless synthesis quality in the streaming mode. We invite readers to listen to the demos at https://funaudiollm.github.io/cosyvoice2.

1 Introduction

In recent years, neural text-to-speech (TTS) synthesis models have garnered significant attention for surpassing traditional concatenative and statistical parametric methods [1–7]. These models have achieved high fidelity and naturalness on pre-defined specific speakers. Recent studies show that zero-shot TTS models are able to synthesize speech for any speaker by imitating the timbre, prosody and style of a reference speech [8]. Beyond their in-context learning (ICL) capability, zero-shot TTS models benefit from large-scale training data, achieving synthesis quality and naturalness nearly indistinguishable from human speech.

Recent zero-shot TTS models can be broadly divided into three categories: codec language models, feature diffusion models and their hybrid systems. Codec language models utilize a speech codec model to extract discrete speech representation [9–11] and employ an autoregressive [8, 12–17] or masked [18] language model to predict the speech tokens, which are then synthesized to waveforms via codec vocoders [19, 20]. Continuous speech representations are also explored in [21]. Language model-based TTS can generate varied and prosody-consistent speech via autoregressive sampling.

*The code and pre-trained models are released at: https://github.com/FunAudioLLM/CosyVoice

Preprint. Work in progress.

Inspired by advances in image generation, denoising diffusion [22, 23] and flow matching models [24] have been introduced into non-autoregressive (NAR) speech synthesis. Early diffusion-based TTS models required duration prediction for each text (phone) to address the length disparity between text and speech features [25–28]. However, this rigid alignment can affect naturalness, resulting in flat prosody. To mitigate this issue, cross-attention and Diffusion Transformers (DiT) have been introduced into NAR TTS models [29, 30]. Recent research indicates simpler approaches for text-speech alignment in NAR TTS models, such as E2 TTS [31], F5-TTS [32] and Seed-TTS [33]. In these models, input text is padded with special tokens to match the total speech length which is either automatically predicted by the utterance duration prediction module or specified by the user in advance. Since NAR TTS models are not constrained by codec vocoders, they can achieve superior speech quality.

Hybrid systems combine the text-to-codec language model and codec-to-feature diffusion model [33–35]. The language model addresses the alignment between text and speech as well as the utterance duration prediction, while the codec-to-feature diffusion model synthesizes speech features (Mel spectrum) based on the generated codec and other conditions. By leveraging the strengths of both generative models, hybrid systems achieve high diversity, prosody consistency and speech quality.

Despite the success of recent zero-shot TTS models, they generally operate in non-streaming (offline) mode, which involves complete input text and requires synthesizing the entire utterance before returning the waveform. This results in high latency, negatively impacting user experience in applications like voice chat [36, 37]. To address this issue, streaming synthesis has been explored for language model-based zero-shot TTS models [38–41], but diffusion-based TTS models and hybrid systems lack well-established streaming solutions.

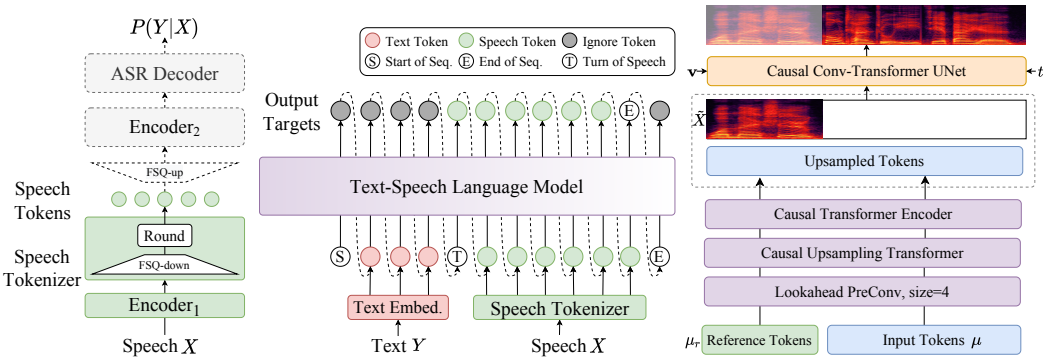
Building on the success of CosyVoice [34], we introduce CosyVoice 2, a streaming zero-shot TTS model with improved prosody naturalness, content consistency, and speaker similarity. Our contributions include:

- Unifying streaming and non-streaming synthesis in a single framework and proposing the unified text-speech language model and chunk-aware causal flow matching model, leading to lossless streaming synthesis compared to offline mode.
- Simplifying the LM architecture by removing the text encoder and speaker embedding, allowing pre-trained textual large language models (LLMs) to serve as the backbone, enhancing context understanding.
- Replacing vector quantization (VQ) in the speech tokenizer with finite scalar quantization (FSQ), improving codebook utilization and capturing more speech information.
- Upgrading the instructed TTS capacity to support more instructions, including emotion, accent, role style, and fine-grained control. In CosyVoice 2, the instruction and zero-shot capacity are integrated into a single model, enabling more versatile and vivid synthesis.

Through the above systemic modification and optimization, CosyVoice 2 achieves human-parity synthesis quality and is nearly lossless in streaming mode. The unified framework loosens deployment requirements, enabling a single model to support both streaming and non-streaming synthesis. The upgraded instructed TTS capacity provides a more powerful and easier approach for users to generate various speeches. In addition, the chunk-aware flow matching design can also be applied to NAR TTS models, which suggests the potential for streaming NAR models.

2 CosyVoice 2

CosyVoice 2 builds on the similar design philosophy of its predecessor [34] by separating the semantic and acoustic information of speech signals and modeling them independently. The speech generation process is redefined as a gradual semantic decoding procedure, where conditional information is progressively incorporated. Specifically, the text-speech language model (LM) focuses solely on semantic information, decoding high-level text tokens into supervised semantic speech tokens. In the Flow Matching model, acoustic details, such as timbre, are introduced through speaker embeddings and reference speech, converting speech tokens into the Mel spectrum for a given speaker. Finally, a pre-trained vocoder model reinstates the phases, transforming the Mel spectrum back into



(a) Supervised speech tokenizer (b) Text-Speech Language Model (c) Chunk-aware Flow Matching

Figure 1: An overview of CosyVoice 2. (a) demonstrates the supervised speech tokenizer, where dashed modules are only used at the training stage. (b) is a unified text-speech language model for streaming and non-streaming synthesis. Dashed lines indicate the autoregressive decoding at the inference stage. (c) illustrates the causal flow matching model conditioning on a speaker embedding $\mathbf{v}$, semantic tokens μ, masked speech features $\tilde{X}$ and intermediate state X_t at timestep t on the probabilistic density path.

the original audio signal. The following sections will introduce the details of CosyVoice 2 and the modifications for streaming synthesis from five respects: text tokenizer, supervised semantic speech tokenizer, unified text-speech LM for streaming/non-streaming synthesis and chunk-aware Flow Matching model. Figure 1 provides an overview of CosyVoice 2.

2.1 Text Tokenizer

CosyVoice 2 uses the raw text as input directly, which is tokenized using a BPE-based text tokenizer. This eliminates the need for a frontend model that obtains phonemes via the grapheme-to-phoneme (g2p) transformation. This approach not only simplifies the data preprocessing workflow but also enables the model to learn the pronunciations of words within various contexts in an end-to-end manner. Unlike the tokenizers commonly used in textual LLMs, CosyVoice 2 masks out the one-to-many tokens. This prevents the pronunciation of a token from becoming excessively long and reduces corner cases caused by data sparsity. Specifically, if a BPE token encodes more than one Chinese character, it will be masked out, and each character will be encoded separately during the tokenization process. Other languages, such as English, Japanese, and Korean, are not subject to special handling.

2.2 Supervised Semantic Speech Tokenizer

As shown in Figure 1 (a), we insert the finite scalar quantization (FSQ) module [42] into the encoder of SenseVoice-Large ASR model [43]. At the training stage, the input speech X goes through the $\mathrm{Encoder}_1$ to obtain the intermediate representations, where $\mathrm{Encoder}_1$ consists of six Transformer blocks with the rotary positional embedding [44]. Then, the intermediate representations are fed into the FSQ module for quantization, and the quantized representations are passed through the rest of SenseVoice-Large modules, including $\mathrm{Encoder}_2$ and ASR $\mathrm{Decoder}$, to predict the posterior probabilities of corresponding text tokens.

In the FSQ module, the intermediate representations H are firstly projected into a D-dimensional low-rank space, and the values of each dimension are quantized into $[-K, K]$ with the bounded round operation ROUND. Then, the quantized low-rank representations $\bar{H}$ are projected into the original dimension $\tilde{H}$ for the following modules:

$$\begin{aligned} \bar{H} &= \mathrm{ROUND}(\mathrm{Proj}_{down}(H)) \\ \hat{H} &= \mathrm{Proj}_{up}(\bar{H}) \end{aligned} \tag{1}$$

At the training stage, the straight-through estimation is used to approximate the gradients of FSQ module and $\mathrm{Encoder}_1$. The speech token μ_i can be obtained by calculating the index of quantized

low-rank representation $\bar{h}_i$ in the $(2K+1)$-ary system:

$$\mu_i = \sum_{j=0}^{D-1} \bar{h}_{i,j}(2K+1)^j \tag{2}$$

The $\mathrm{Encoder}_1$, low-rank projector of FSQ module, bounded round operation and index calculation form the speech tokenizer for CosyVoice 2. Our speech tokenizer works at a token rate of 25 Hz, i.e., 25 speech tokens per second.

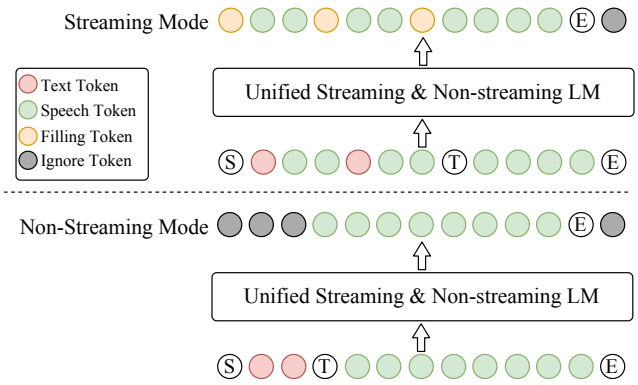


Figure 2: A diagram of the unified text-speech language model for streaming and non-streaming synthesis in CosyVoice 2.

2.3 Unified Text-Speech Language Model

In CosyVoice 2, the pre-trained textual LLM, Qwen2.5-0.5B [45], is used as the text-speech language model to generate the speech tokens autoregressively with the input text as a prompt. Similar to other LMs, the text-speech LM is also trained in a next-token-prediction scheme as shown in Figure 1 (b). Different from the previous CosyVoice, we remove the speaker embedding to avoid information leaking. More importantly, we find that such utterance-level vector contains not only speaker identify but also language and paralanguage information, which harms the prosody naturalness and cross-lingual capability of the text-speech LM. Besides, we also abandon the text encoder of the previous CosyVoice, since we find that the Qwen2.5-0.5B model is powerful enough to align the text and speech tokens, and the text encoder is no longer needed.

Benefiting from the simplicity of text-speech LM, we can build a unified model for both streaming and non-streaming synthesis. Here, "streaming mode" means the input text is received in a continuous flow rather than being known as a complete sentence in advance. In CosyVoice 2, the difference between streaming and non-streaming modes is only the way of sequence construction for LM:

- For the **Non-Streaming** mode, the "start of sequence" Ⓢ, all text tokens, "turn of speech" token Ⓣ, all speech tokens and the "end of sequence" Ⓔ are concatenated sequentially as shown in the bottom of Figure 2. Ignore token means that their losses are ignored while minimizing the cross-entropy objective function.
- For the **Streaming** mode, we mix up the text and speech tokens in a pre-defined ratio of N:M, i.e. every N text tokens are followed by M speech tokens seen in the top of Figure 2. If the next token is a text token, the model is expected to predict a filling token (rather than the text token), which indicates that the next N text tokens should be concatenated at the inference stage. Once the text tokens are ran out of, the "turn of speech" token Ⓣ and the remaining speech tokens are concatenated sequentially, forming the hybrid text-speech token sequence in the streaming mode.

By training the text-speech LM on the above two sequences simultaneously, we can perform streaming and non-streaming speech generation within a single unified model. In real-life scenarios, such as speaker fine-tuning (SFT) and in-context learning (ICL), the inference sequence differs as follows:

- **ICL, Non-Streaming:** In ICL, the LM requires prompt text and speech tokens from the reference audio to imitate the accent, prosody, emotion and style. In the non-streaming mode,

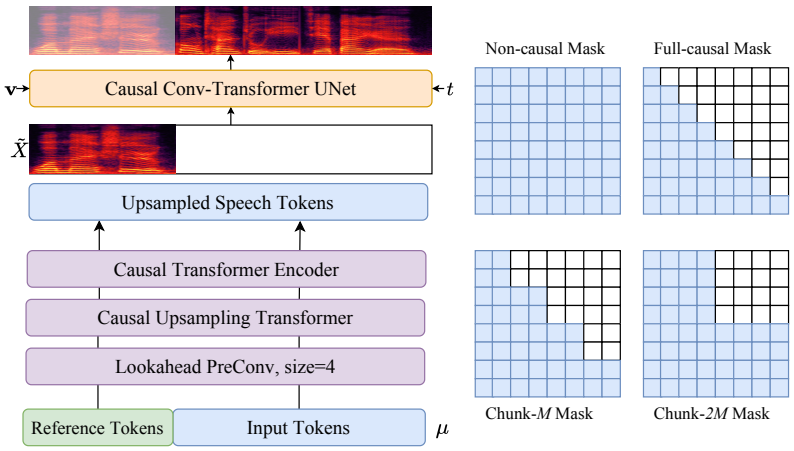


Figure 3: A diagram of the unified chunk-aware flow matching model for streaming and non-streaming synthesis in CosyVoice 2.

the prompt and to-synthesize text tokens are concatenated as the whole entity, and the prompt speech tokens are treated as the pre-generated results and are fixed: "Ⓢ, **prompt_text**, **text**, Ⓣ, **prompt_speech**". The autoregressive generation of LM is started from such sequence until the "End of sequence" token Ⓔ is detected.

- **ICL, Streaming:** In this scenario, we assume the to-generate text is already known and the speech tokens should be generated in a streaming manner. Similarly, we treat the prompt and to-generate text as a whole entity. Then, we mix it up with the prompt speech tokens on the ratio of N:M: "Ⓢ, **mixed_text_speech**, Ⓣ, **remaining_speech**". If the length of text is greater than that of prompt speech tokens, the LM will generate "filling token". In this situation, we manually pad N text tokens. If the text tokens run out of, the "Turn of speech" token Ⓣ will be added. In the streaming mode, we return generation results every M tokens until the Ⓔ is detected.
- **SFT, Non-Streaming:** In the SFT scenario, the LM is fine-tuned on a specific speaker, and the prompt text and speech are no longer needed. Thus, the initial sequence is very simple: "Ⓢ, **text**, Ⓣ". Starting from this, the text-speech LM can generate speech tokens autoregressively until Ⓣ.
- **SFT, Streaming:** In the streaming mode of SFT, we start the speech generation from the following sequence: "Ⓢ, **first_N_text**". Then, the LM will generate M speech tokens, and we manually pad the next N text tokens. We repeat the above process until all text tokens run out of, and then Ⓣ is added. Note that this mode can also be adopted by the speech-to-speech multi-modal large language models to obtain an extremely low latency.

2.4 Chunk-aware Flow Matching

In CosyVoice 2, we employ the Mel spectrogram as the acoustic feature with the frame rate of 50 Hz and the sampling rate of 24000. Due the frame-rate mismatch between speech tokens and Mel features, we up-sample the speech tokens with the ratio of two to match the frame rate of Mel spectrogram. Before the up-sampling operation, we add an additional look-ahead convolution layer to provide the future information for the following causal modules. The look-ahead layer is implemented by a right-padded 1-D convolution with the pad size of P and the kernel size of $P+1$. After these, several chunk-aware causal Transformer blocks are followed to align the representation space of speech tokens to match acoustic features.

Subsequently, our goal is to further decode the speech tokens into the Mel spectrogram specified by the speaker embedding and reference speech. To achieve this, we employ a conditional flow matching (CFM) model to sample the Mel spectrogram, given speech tokens, reference speech and speaker embedding as conditions. In the CFM model, the distribution of target Mel spectrogram is described by a probability density path from a prior distribution $p_0(X)$ and the data distribution $q(X)$. The probability density path can be defined by a time-dependent vector field. For sampling efficiency, we employ the optimal-transport (OT) flow to match the vector field ω_t, which is given

by an ordinary differential equation (ODE):

$$\omega_t(\phi_t^{OT}(X_0, X_1)|X_1) = X_1 - X_0 \tag{3}$$

$$\phi_t^{OT}(X_0, X_1) = (1-t)X_0 + tX_1 \tag{4}$$

$$X_0 \sim p_0(X) = \mathcal{N}(0, I) \tag{5}$$

$$X_1 \sim q(X) \tag{6}$$

A causal convolutional Transformer UNet is employed to learn the above ODE with the up-sampled token μ, masked Mel spectrogram $\tilde{X}_1$, speaker embedding[2] $\mathbf{v}$ and timestep t as the conditions:

$$\nu_t(\phi_t^{OT}(X_0, X_1)|\theta) = \mathrm{UNet}_\theta\left(\phi_t^{OT}(X_0, X_1), t; \mathbf{v}, \{\mu\}_{1:L}, \tilde{X}_1\right) \tag{7}$$

At the training stage, the masked Mel spectrogram is obtained by randomly masking out 70% to 100% of the final frames in X_1. As for the inference, it is provided by the Mel spectrogram extracted from the reference speech. By minimizing the L1 loss between the predicted and ground-truth ODE, we can optimize the UNet parameters θ as follows:

$$\theta = \arg\min_\theta \mathbb{E}_{p_0(X), q(X), t}\left|\omega_t(\phi_t^{OT}(X_0, X_1)) - \nu_t(\phi_t^{OT}(X_0, X_1)|\theta; \mu, \tilde{X}_1, \mathbf{v})\right|_1 \tag{8}$$

At the training stage, the timestep follows a uniform distribution $U[0, 1]$. However, during the inference, we employ the cosine scheduler to offer more steps for the initial generation stage:

$$t := 1 - \cos\left(\frac{1}{2}t\pi\right) \tag{9}$$

Besides, we also train the model on both conditional and non-conditional situations to enable the classifier-free guidance (CFG) [46–48] at the inference stage:

$$\tilde{\nu}_t(\phi_t^{OT}(X_0, X_1)|\theta; \Psi) = (1+\beta)\cdot\nu_t(\phi_t^{OT}(X_0, X_1)|\theta; \Psi) - \beta\cdot\nu_t(\phi_t^{OT}(X_0, X_1)|\theta) \tag{10}$$

where Ψ denotes the conditions $\{\mathbf{v}, \mu, \tilde{X}_1\}$. The CFG strength β and the number of flow estimation (NFE) are set to 0.7 and 10, respectively, according to the experimental results.

The current flow matching models always work on a offline mode, i.e., only all the speech tokens are generated, the Mel spectragram can be sampled, which is not friendly for the streaming synthesis. To overcome this issue, we treat the multi-step flow estimation as a stacked deeper neural network, which repeats the UNet ten times. Thus, by making the unfolded neural network causal, we can apply it on the streaming synthesis. We construct four masks to satisfy different application situations:

- **Non-causal Mask** is used for offline mode, which can achieve the best performance by attending all frames of conditions. Non-causal mask is suitable for the latency-insensitive situations.
- **Full-causal Mask** is designed for scenarios required extremely low latency, in which only the past frames can be attended.
- **Chunk-M Mask** is a trade off between latency and performance, which can leverage the information of the past and M future frames. This mask is more suitable for the first chunk of generation with low latency.
- **Chunk-$2M$ Mask** can achieve a approximate performance of offline mode by sacrificing more latency, which can be used for the cascade generation chunk for better performance.

For each training case in a mini-batch, we randomly sample a mask from the above four masks under the uniform distribution. In this manner, one flow matching model can be compatible to different scenarios, lowering the deployment complexity. Another advantage of this chunk-aware training is that the masks with more context sever as a teacher for the ones with less context, benefiting from the implicit self-distillation scheme.

[2]https://github.com/alibaba-damo-academy/3D-Speaker/tree/main/egs/3dspeaker/sv-cam++

2.5 Latency Analysis for Streaming Mode

The first-package latency is an important metric for streaming synthesis models, which significantly affects the user experience especially in LLM-based voice chat applications, such as GPT-4o [36]. In the context of TTS, the to-synthesize text is known in advance, and the latency comes from the aspects of speech token generation, Mel spectrogram reconstruction and waveform synthesis. Thus, the first-package latency L_{TTS} of CosyVoice 2 can be obtained as follows:

$$L_{TTS} = M \cdot d_{lm} + M \cdot d_{fm} + M \cdot d_{voc} \tag{11}$$

where d_{lm} denotes the computation time of LM to generate one speech token, d_{fm} represents the computation time of Flow Matching model to generate the frames of Mel spectrogram for one speech token and d_{voc} stands for the computation time of vocoder to synthesize waveforms corresponding to one speech token. In the context of LLM-based voice chat, the length of first-package-required text should also be considered, and the first-package latency L_{Chat} becomes as follows:

$$L_{Chat} \leq N \cdot d_{llm} + L_{TTS} \tag{12}$$

where d_{llm} represents the computation time of a LLM to generate one text token. Note that, since the multi-character tokens are masked out in CosyVoice 2's text tokenizer, the text tokens used by text LLMs always encode longer raw text than those of CosyVoice 2. Thus, the the first-package latency L_{Chat} must be lower than the summation of $N \cdot d_{llm}$ and L_{TTS}.

2.6 Instructed Generation

To enhance the controllability of CosyVoice 2, we integrated the instructed dataset into the base training set. We have collected 1500 hours of instructed training data, which includes both natural language instructions and fine-grained instructions, as outlined in Table 1. For natural language instructions, we prepend a natural language description and a special end token, "`<|endofprompt|>`" before the to-synthesize input text. These descriptions cover aspects such as emotion, speaking rate, role-playing, and dialects. For fine-grained instructions, we insert vocal bursts between text tokens, using markers like "`[laughter]`" and "`[breath]`". Additionally, we apply vocal feature tags to phrases; for instance, "`<strong>XXX</strong>`" indicates emphasis on certain words, while "`<laughter>XXX</laughter>`" signifies speaking with laughter.

Natural Language Instruction
Emotion: 高兴(Happy), 悲伤(Sad), 惊讶(Surprised), 愤怒(Angry), 恐惧(Fearful), 厌恶(Disgusted), 冷静(Calm), 严肃(Serious) **Speaking Rate:** 快速(Fast), 非常快速(Very Fast), 慢速(Slow), 非常慢速(Very Slow) **Dialect:** 粤语, 四川话, 上海话, 郑州话, 长沙话, 天津话 **Role-playing:** 神秘(Mysterious), 凶猛(Fierce), 好奇(Curious), 优雅(Elegant), 孤独(Lonely), 机器人(Robot), 小猪佩奇(Peppa), etc.
Fine-grained Instruction
Vocal Bursts: [laughter], [breath], etc. **Vocal Features:** <laughter></laughter>,
Examples
- 你能用高兴的情感说吗？< |endofprompt| >今天真是太开心了，马上要放假了！I'm so happy, Spring Festival is coming! - Please speaking very fast.< |endofprompt| >Today is a happy day, full of laughter and joy. - 请问你能模仿粤语的口音吗？< |endofprompt| >多保重，早休息。 - 尝试一下以机器人的角色和我交流。< |endofprompt| >接收知识光波！ - [laughter]有时候，看着小孩子们的天真行为[laughter]，我们总会会心一笑。 - She pursued her dreams with enthusiasm and grit.

Table 1: Examples of natural language instructions and fine-grained instructions.

2.7 Multi-Speaker Fine-tuning

Fine-tuning the pre-trained model on specific speakers (SFT) can further improve the generation quality and speaker similarity. In this report, we introduce the multi-speaker fine-tuning (mSFT), in

which the pretrained model is fine-tuned on multiply speakers simultaneously rather than a single speaker. This approach ensures comprehensive prosody and pronunciation coverage across multiple speakers and mitigates potential catastrophic forgetting from the pretrained models. To avoid timbre confusion between various speakers, we prepend speaker-prompt tags, "`Speaker A<|endofprompt|>`" to the input text for a specific speaker. If a training sample is not labeled to a speaker, a special tag, "`unknown<|endofprompt|>`", is utlized. The learning rate is set to 1e-5 during the whole multi-speaker fine-tuning process.

2.8 Reinforcement Learning for SFT

Reinforcement learning is a commonly used method in the training of large language models, which can make the LM output align with human preference. In CosyVoice 2, we employ speaker similarity (SS) and recognition word error rate (WER) from the ASR system as the reward function to improve speaker similarity and pronunciation accuracy in the fine-tuning stage. We use WER and SS to distinguish preferred sample x^w and rejected samples x^l and optimize the TTS system with direct preference optimization (DPO) [49] as follow:

$$L_{DPO}(\pi_\theta; \pi_{\text{ref}}) = -\log \sigma(\beta \log \frac{\pi_\theta(\mu^w|y)}{\pi_{\text{ref}}(\mu^w|y)} - \beta \log \frac{\pi_\theta(\mu^l|y)}{\pi_{\text{ref}}(\mu^l|y)}) \tag{13}$$

where μ^w and μ^l are the speech token extracted from the preferred and rejected samples x^w and x^l.

However, this method is time-consuming and computation-consuming as it should synthesis the audios through the TTS system repeatedly to obtain distinguishable preference and rejected samples. During training, four forward operations are needed for one training step. To simplify the process, we recover the LM predicted token $\mu_i \in \{0, 1, \ldots, (2K+1)^D - 1\}$ into quantized low-rank representations $\bar{H}$, and directly use the ASR backend of the speech tokenizer to re-predict the input text. Then the predicted log posterior can be regarded as the ASR reward function to optimize the text-speech language model. During training, the ASR backend parameters are frozen.

$$\bar{h}_{i,j} = \left\lfloor \frac{\mu_i}{(2K+1)^j} \right\rfloor \mod (2K+1) \tag{14}$$

$$\begin{aligned} \hat{H} &= \text{Proj}_{up}(\bar{H}) \\ L_{ASR} &= -\log P(Y|\hat{H}; \theta_{ASR}) \end{aligned} \tag{15}$$

where Y is the input text, and $\bar{H}$ are the recovered speech low-rank representations. As the sample operation of the $u_i \sim P(\mu_i|\mu_{1:i-1}, Y; \theta_{LM})$ still prevent us to optimize the model directly, we use the gumbel softmax sampling to make it differentiated and then optimize the θ_{LM} by the $\mathcal{L}_{ASR}$.

3 Experimental Settings

3.1 Training Data for Speech Tokenizer

A 200,000-hour dataset is used to train the speech tokenizer with normalized transcriptions as labels. Detailed data information is listed in Table 2. The training data comes from three different resources: open source ASR datasets, internal industrial datasets and TTS generation datasets. Although we only used Chinese and English data when training the speech tokenizer, as shown in Table 2, subsequent experiments revealed that the speech tokenizer had zero-shot capability for other languages. It can be also used for speech synthesis in languages such as Japanese and Korean.

Language	Duration (hours)
Chinese	110,884
English	99,918

Table 2: Details of training data for speech tokenizer.

3.2 Training Data for CosyVoice 2

CosyVoice 2 shares the same training data as its previous version [34]. We first collect the speech-only data with internal speech processing tools, including speech detection, signal-to-noise ratio (SNR) estimation, speaker diarization, and separation. Subsequently, the Paraformer [50] and SenseVoice [43] are employed to generate pseudo text labels for Chinese and other languages, respectively. We also employ an internal force-alignment model to filter out low-quality data and enhances the accuracy of punctuation. Data details are provided in Table 3.

Language	Duration (hours)
Chinese	130,000
English	30,000
Japanese	4,600
Korean	2,200

Table 3: Details of training data for CosyVoice 2.

3.3 Evaluation Settings

We evaluate our CosyVoice 2 on two test sets. The first one is constructed from the test-clean set of Librispeech corpus [51], denoting as *test-clean*. This test set is used to evaluate CosyVoice 2 on a limited English domain. The Whisper-large V3 is used as the ASR model to evaluate the content consistency. As for the speaker similarity (SS), we employ the ERes2Net model [52] to extract speaker embeddings of prompt and generated utterances, and their raw cosine similarity is treated as the speaker similarity. NMOS score [53] is used to evaluate the objective quality.

The second evaluation is conducted under the SEED settings [33], which is widely used to evaluate recent TTS models, covering various text domains and reference speeches. In this evaluation, about 2,000 Chinese and 1,000 English samples are selected from CommonVoice datasets, denoting as *test-zh* and *test-en*, respectively. In addition, about 400 hard test cases are also included to evaluate the robustness of TTS models on text repetition, tongue twister and other challenging synthesis cases, denoting as *test-hard* in this report. The Paraformer is employed to recognize the synthesis results of *test-zh* and *test-hard*, while the Whisper-large V3 is adopted for *test-en* to evaluate the content consistency.

3.4 Benchmark for Japanese and Korean

We prepare two test sets, denoted as *test-ja* and *test-ko*, for the evaluation on Japanese and Korean speech synthesis. The *test-ja* consists 1,000 samples extracted from the CommonVoice dataset, which are used to measure the model's performance on various metrics, such as WER, SS, MOS. Specifically, we randomly shuffle and pair the entire CommonVoice JA-test set as reference utterance and target utterance spoken. Considering the wide range of utterances' text lengths of JA-test set, we randomly selected 1,000 pairs of reference-target utterances from the length range from 8 to 32 characters as our final test set. For the *test-ko*, we selected 1,000 speech samples with a WER of less than 5% and no deletion or insertion errors, utilizing the Whisper-Large V3 [54] as the ASR model. These samples were used as reference utterances for the Korean speech synthesis. For the input text, we randomly selected 1,000 text samples from the remaining data. We have released the lists of prompt speeches, prompt transcriptions and input text from these two test sets are released to facilitate result reproduction. By providing this open-source data, we aim to establish a benchmark for evaluating Japanese and Korean TTS models. The Whisper-large V3 is used as the ASR model for Japanese and Korean evaluations.

4 Experimental Results

4.1 Evaluations on Speech Tokenizer

An ideal speech tokenizer is supposed to effectively utilizes the codebook, preserves information at a high fidelity, and demonstrates speaker independence. In this part we evaluate our supervised

Method	Codebook		ASR Error Rate (%)			
	Size	Util.	C.V. EN	C.V. CN	Fluers EN	Fluers CN
VQ	4,096	963 (23%)	18.26	11.56	7.65	5.03
FSQ	6,561	6,561 (100%)	10.67	7.29	6.58	4.43

Table 4: The comparision of VQ and FSQ inside Sensevoice-large encoder. C.V. stands for the CommonVoice benchmarks.

speech tokenizer from four aspects: 1) Codebook utilization rate; 2) ASR error rate within the entire encoder; 3) Token visualization of different speakers; 4) Speaker identification training. Table 4 shows the codebook utilization and ASR error rate. It turns out that the FSQ-based tokenizer fully utilizes the codebook and maintains more effective information from the aspect of ASR, indicating more semantic information maintained by FSQ.

We further analyze the characteristics of FSQ through the t-SNE visualization. As an upstream model for TTS tasks, the tokenizer should strive to minimize the entanglement of speaker identity information with the speech signal. We selected 100 speech samples from each of the three speakers in the VoxCeleb1 dataset and visualized the corresponding tokens. As illustrated in Figures 4(a) and (b), it is evident that before the quantization, Encoder$_1$'s outputs exhibit different distributions among different speakers. In contrast, the distributions of quantized representations are nearly indistinguishable. In addition, Figure 4(c) also shows that the tokenizer fully utilizes the codebook. Subsequently, the S3prl toolkit [55] is employed to further evaluate the speaker entanglement by performing speaker identification (SID) task. We use Sensevoice-large encoder with FSQ as an upstream feature extractor and train SID task with representations before or after the quantization. Figure 5 shows the accuracy curves during the training. The SID layer with quantized tokens does not converge, which proves the decoupling function of the tokenizer on speaker information.

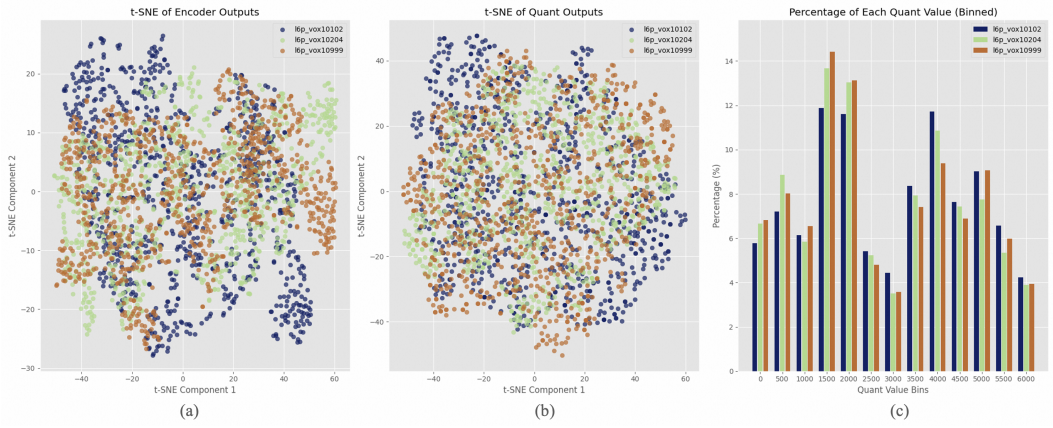


Figure 4: The t-SNE visualization of speech representations before (a) and after (b) the quantization for three different speakers in Voxceb1 dataset. (c) shows the codebook utilization in terms of the token percentage on the speakers (500 tokens each bin).

4.2 Comparison Results with Baselines

We first evaluated our CosyVoice 2 models on a limited English text domain and compared it with several open-source models, such as ChatTTS [56], GPT-SoVITs [57], OpenVoice [58], ParlerTTS [59], EmotiVoice [60], and its predecessor CosyVoice [34]. The objective results are presented in Table 5, including content consistency (WER), speech quality (NMOS) and speaker similarity (SS). From the table, we can see that CosyVoice 2 achieves state-of-the-art performance on the Librispeech test-clean set, surpassing all baseline models acros all evaluation metrics. Notably, CosyVoice 2 even demonstrates higher content consistency, speech quality, and speaker similarity than human utterances, indicating its human-parity synthesis quality.

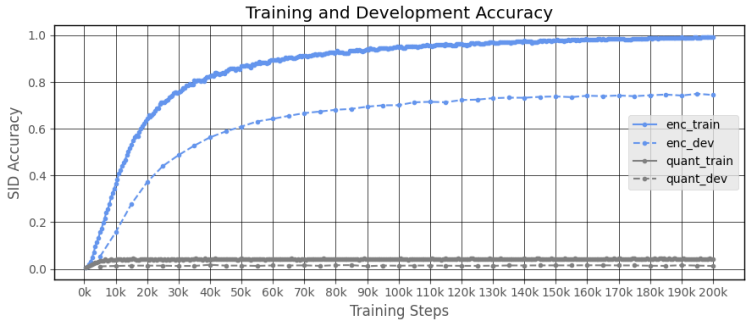


Figure 5: The convergence curves of SID training with tokens before or after quantization.

Model	WER (%)	NMOS	SS
Human	2.66	3.84	0.697
ChatTTS [56]	6.84	3.89	-
GPT-SoVITs [57]	5.13	3.93	0.405
OpenVoice [58]	3.47	3.87	0.299
ParlerTTS [59]	3.16	3.86	-
EmotiVoice [60]	3.14	3.93	-
CosyVoice [34]	2.89	3.93	0.743
CosyVoice 2	2.47	**3.96**	0.745
CosyVoice 2-S	**2.45**	3.90	**0.751**

Table 5: Content consistency (WER), speaker similarity (SS) and speech quality (NMOS) results on LibriSpeech test-clean subset of baselines and CosyVoice 2. Whisper-Large V3 is employed as the ASR model and punctuations are excluded before WER calculation.

We also evaluated CosyVoice 2 on the commonly-used test sets: SEED *test-zh*, *test-en* and *test-hard*, which include diverse input texts and reference speeches from various domains. The experimental results for CosyVoice 2 and the baseline models are presented in Table 6. On the *test-zh* set, CosyVoice 2 surpasses all open-sourced models in terms of CER and SS, falling short of the commercial model SEED-TTS by only a small margin. On the *test-en* set, CosyVoice 2 ranks fourth and third in terms of WER and SS, respectively. This may result from the imbalance in the volume of training data between Chinese and English. We plan to explore data scaling in future work to enhance content consistency in English. On the *test-hard* set, the offline CosyVoice 2 model

Model	*test-zh*		*test-en*		*test-hard*	
	CER (%)	SS	WER (%)	SS	WER (%)	SS
Human	1.26	0.760	2.14	0.730	-	-
Vocoder Resyn.	1.27	0.720	2.17	0.700	-	-
Seed-TTS† [33]	1.12	0.796	2.25	0.762	7.59	0.776
FireRedTTS [35]	1.51	0.630	3.82	0.460	17.45	0.639
MaskGCT [18]	2.27	0.774	2.62	0.774	10.27	0.748
E2 TTS (32 NFE)† [31]	1.97	0.73	2.19	0.710	-	-
F5-TTS (32 NFE) [32]	1.56	0.76	1.83	0.670	8.67	0.762
CosyVoice [34]	3.63	0.775	4.29	0.699	11.75	0.755
CosyVoice 2	1.45	0.806	2.57	0.736	6.83	0.776
CosyVoice 2-S	1.45	0.812	2.38	0.743	8.08	0.785

Table 6: Results of CosyVoice 2 and recent TTS models under the SEED evaluation settings. † denotes close-sourced models.

achieves state-of-the-art performance across all compared baseline, demonstrating its robustness in challenging synthesis scenarios. Compared with human-generated speeches, CosyVoice 2 shows comparable content consistency and superior speaker similarity. Considering the recognition errors can also stem from the ASR model, it is reasonable to conclude that CosyVoice 2 achieves a human-parity synthesis capability. We also evaluated the streaming mode, denoted as "CosyVoice 2-S" in Table 5 and 6. For both evaluation settings, the streaming mode's performance is nearly lossless in typical test cases. Only in challenging cases is there a slight degradation in content consistency, highlighting the strength of our unified streaming/non-streaming framework.

4.3 Modular Ablation Study

We conducted a modular ablation study on the text-speech language model to assess the impacts of our modifications, including LLM initialization, removing speaker embedding, and utilizing FSQ. Table 7 illustrates the step-by-step development of CosyVoice 2 from its predecessor. By replacing the randomly initialized language model with a pretrained LLM, we achieved relative improvements in content consistency of 18.46% and 15.40% on the *test-zh* and *test-hard* sets, respectively. Next, we removed the speaker embedding from the text-to-speech language model, which helps prevent information leakage and disturbances in in-context learning. This change resulted in a significant reduction in content errors while maintaining speaker similarity, indicating that content information is primarily modeled by the LM, and speaker information is mainly recovered by the flow matching model. Finally, by replacing VQ with FSQ, we achieved the CosyVoice 2 model, noting much higher content consistency and unchanged speaker similarity. By fully utilizing the codebook, FSQ captures more content information and context variation, leading to better alignment between text and speech tokens. Furthermore, we conducted a comparative experiment by incorporating pitch loss as a constraint during the training of the FSQ-based speech tokenizer. We found that this approach led to improved performance in downstream TTS tasks, as indicated in the last row of Table 7. In future versions of CosyVoice, we plan to carry out more detailed experiments and analyses.

Model	***test-zh***		***test-en***		***test-hard***	
	CER (%)	**SS**	**WER (%)**	**SS**	**WER (%)**	**SS**
CosyVoice	3.63	0.775	4.29	0.699	11.75	0.755
+ LLM init.	2.96	0.808	4.57	0.730	9.94	0.789
+ Drop Spk Emb.	2.56	0.804	3.81	0.740	9.66	0.778
+ FSQ (CosyVoice 2)	1.45	0.806	2.57	0.736	6.83	0.776
+ Pitch Loss	1.19	0.802	2.40	0.728	6.29	0.769

Table 7: Modular analysis on the modifications of text-speech language model.

We also conducted another modular analysis to evaluate the impact of streaming modules on the synthesis performance. Table 8 shows the results for content consistency and speaker similarity. We fount that the streaming LM has a minimal impact on typical cases from the *test-zh* and *test-en* sets, indicating the effectiveness of our unified training framework. The primary impact of the streaming LM is observed in challenging cases from the test-hard set, likely due to the loss of contextual information in streaming mode. Interestingly, the streaming flow matching model results in slightly higher speaker similarity compared to the offline mode. This may be due to the higher prompt-to-generation ratio of initial chunks in streaming mode, whereas the prompt-to-generation ratio in offline mode can be very low, with many padding tokens. The negative effect of the streaming flow matching model on content consistency is much less pronounced compared to streaming LMs, thanks to the semantic-acoustic decoupled modeling in CosyVoice 2.

4.4 Results on Japanese and Korean Benchmarks

In addition to Chinese and English, CosyVoice 2 also supports Japanese and Korean. We evaluated the content consistency, speaker similarity and speech quality on our constructed Japanese and Korean test sets. As shown in Table 9, CosyVoice 2 performs significantly better on Korean than on Japanese across all evaluation metrics. This discrepancy is primarily due to the overlap in the character set between Japanese and Chinese, which leads to Chinese pronunciations in Japanese contexts. In the future work, we plan to explore ways to enhance linguistic context for multilingual

Model	LM	FM	*test-zh*		*test-en*		*test-hard*	
			CER (%)	SS	WER (%)	SS	CER (%)	SS
M1	Offline	Offline	1.45	0.806	2.57	0.736	6.83	0.776
M2	Offline	Stream.	1.46	0.811	2.60	0.743	7.12	0.788
M3	Stream.	Offline	1.38	0.806	2.51	0.737	7.88	0.773
M4	Stream.	Stream.	1.45	0.812	2.38	0.743	8.08	0.785

Table 8: Modular analysis on the impact of streaming modules in CosyVoice 2. Chunk size is set to 15 for streaming modules.

synthesis. Since Korean does not have character overlap with other languages, its speech synthesis achieves much better performance. Another issue is data imbalance. We believe that increasing the volume of training data could further improve synthesis performance for both Japanese and Korean.

Model	*test-ja*			*test-ko*		
	CER (%)	SS	NMOS	CER (%)	SS	NMOS
CosyVoice 2	18.79	0.630	3.42	7.98	0.707	3.73
CosyVoice 2-S	21.41	0.629	3.35	9.06	0.714	3.60

Table 9: The content consistency (CER), speaker similarity (SS), and speech quality (NMOS) of CosyVoice 2 and its streaming counterpart on the Japanese *test-ja* and Korean *test-ko* test sets.

4.5 Results on Instructed Generation

To evaluate the performance of instructed generation, we have created a Chinese test set comprising 290 samples. This set includes 29 types of instructions, shown in Table 1, each with 10 different input texts. We utilize five audio prompts and speaker embeddings from five speakers (three female and two male) as conditions for the flow matching model. Our testing is conducted in offline mode. We objectively evaluate content consistency (CER), speaker similarity (SS), and speech quality (NMOS). Subjectively, we assess the accuracy and naturalness of instruction using the Mean Opinion Score for Instruction (MOS-I), which ranges from 1 to 5. Each sample is assessed by 10 native Chinese speakers, with scores assigned in increments of 0.5. The evaluation criteria focus on whether the speech adheres to all specified instructions, such as emotional expression, speech rate adjustment, dialect usage, and role-playing. Fine-grained controls, including the insertion of laughter, speaking with laughter, breath control, and emphasis, are evaluated for naturalness and accuracy. As illustrated in Table 10, CosyVoice 2 exhibits superior content consistency (CER), speaker similarity (SS), and accuracy and naturalness in instruction control (MOS-I), while maintaining comparable speech quality to CosyVoice-Instruct. When input instructions are removed from CosyVoice 2, there is a notable decline in MOS-I; however, improvements are observed in content consistency (CER), speaker similarity (SS), and speech quality (NMOS). This indicates that instruction controllability is difficult to implicitly emerge from content text.

Model	CER (%)	SS	NMOS	MOS-I
CosyVoice-Instruct [34]	1.72	0.797	3.94	3.14
CosyVoice 2	1.52	0.804	3.94	4.11
CosyVoice 2 w/o Instruction	0.97	0.817	4.02	2.54

Table 10: Evaluation results for content consistency (CER), speaker similarity (SS), speech quality (NMOS), and MOS-I (Instruction, assessing the accuracy and naturalness of instruction) on an in-house Chinese test set for CosyVoice-Instruct, CosyVoice 2, and CosyVoice 2 without instruction input. The Paraformer model is used as the ASR system, with punctuation marks excluded from the CER calculation. Dialect data is not included in the CER calculation because the Paraformer model cannot recognize Chinese dialect speech.

4.6 Results on Speaker Fine-tuned Models

During the fine-tuning phase, we employ unsupervised clustering on the speaker embeddings of the same speaker to ensure the stability of the speaker's timbre. We have demonstrated that a target speaker with as few as 400 audio recordings can achieve reasonably good speech synthesis performance, with only slight variations in objective metrics observed among different speakers, as shown in Figure 6. Our experiments indicate that most speakers can inherit the zero-shot TTS model's robust contextual understanding and perception, thereby naturally expressing various moods and emotions in response to the input text.

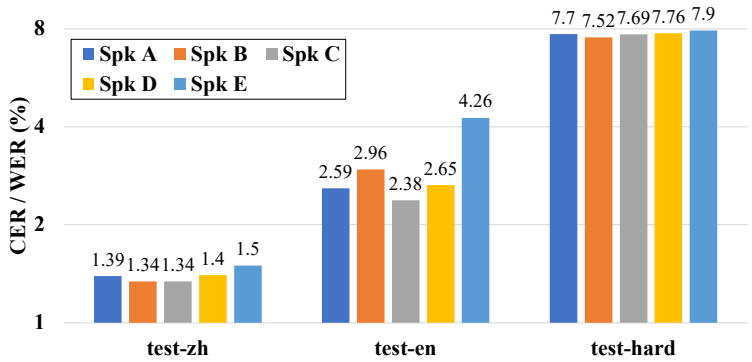


Figure 6: Results of CosyVoice 2 SFT Models under the SEED evaluation settings. CER is used for test-zh and test-hard, while WER is used for test-en.

4.7 LM Fine-tuning with Reinforcement Learning

Although the SFT can improve the performance on most speakers, the results of Spk E are still worse than the base model especially on English. Because it has a more complex voice and faster speech speed. Additionally, only Chinese recordings are available for Spk E. So we choose the Spk E to evaluate the effectiveness of reinforcement learning. During the SFT of text-speech LM, we apply the DPO to change the preference biasing of the LM by the ASR and SS rewards (DPO-ASR-SS). We also use the differentiable ASR rewards to optimize the LM parameters. After SFT, we evaluate the model with content consistency (WER), speaker similarity (SS) and speech quality (NMOS) on the test set of Spk E and further evaluated the WER on the SeedTTS test sets to explore whether the model can maintain robustness to out-of-domain or cross-lingual input text. Results are shown in Table 11.

Model	Inhome Target Speaker			SEED tests(%)		
	WER(%)	NMOS	SS	zh	en	hard
Ground Truth	6.00	3.87	0.697	1.26	2.14	-
CosyVoice 2	5.34	3.91	0.721	1.45	2.57	6.83
CosyVoice 2-SFT	7.15	3.96	0.795	1.50	4.26	7.90
+ L_{ASR}	6.79	3.96	0.795	1.29	3.53	7.30
+ L_{DPO}	6.83	3.96	0.792	1.43	4.02	8.31
+ L_{ASR} + L_{DPO}	6.64	3.97	0.796	1.25	3.17	6.66

Table 11: Content consistency (WER), speaker similarity (SS) and speech quality (NMOS) comparison for reinforcement learning models on Spk E.

Compared to the pre-trained base model, the SFT model shows higher speaker similarity and speech quality, however, the WER could be worse than the base model. We find that the audio synthesized by the base model always has a slower speed than the SFT and ground truth, which is more friendly to the ASR systems. For the target speaker dataset, both preference biasing and differentiable rewards can reduce the WER with little harmful effect on the other two metrics. But for the SEED test sets, the DPO based reinforcement only benefits the Chinese and English subset, while

the hard samples will be worse. The reason could be that the hard samples contain many repeated words or phrases, they could be regarded as rejected samples during DPO training. However, the differentiable ASR reward will not suffer this problem, as it can directly optimize the TTS system by the ASR posterior. This means that the differentiable ASR reward has a better generalization ability in the out-of-domain situations. Finally, we can combine them with each other for further improvements.

5 Conclusion

Building on the success of CosyVoice, this report presents CosyVoice 2, an improved streaming speech synthesis model that leverages large language models. By unifying streaming and non-streaming synthesis within a single framework, CosyVoice 2 achieves human-parity naturalness, minimal response latency, and virtually lossless synthesis quality in streaming mode. Key innovations include finite scalar quantization for full codebook utilization, a simplified text-to-speech language model architecture that incorporates pre-trained textual LLMs, and the development of a chunk-aware causal flow matching model to support diverse synthesis scenarios. Additionally, improvements in instructed TTS capacity allow for versatile and vivid speech generation with fine-grained control over emotion, accent, role style, and vocal bursts. Through systematic modifications and optimizations, CosyVoice 2 not only delivers superior synthesis quality but also loosens deployment requirements, making it suitable for both streaming and non-streaming applications. We believe that CosyVoice 2 represents a significant advancement in scalable, high-quality, and interactive text-to-speech synthesis.

6 Limitations

CosyVoice 2 has several limitations that need to be addressed. First, it supports only a limited number of languages. For languages with overlapping character sets, synthesis performance may degrade, presenting an open challenge for future research. Second, CosyVoice 2 cannot control acoustic characteristics, such as timbre, through textual instructions, which could be a fascinating area of exploration for role-playing applications. Additionally, CosyVoice does not perform well when tasked with singing.

References

[1] Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc V. Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. Tacotron: Towards end-to-end speech synthesis. In *INTERSPEECH*, pages 4006–4010. ISCA, 2017.

[2] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions. In *ICASSP*, pages 4779–4783. IEEE, 2018.

[3] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan Ömer Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep voice 3: 2000-speaker neural text-to-speech. *CoRR*, abs/1710.07654, 2017.

[4] Wei Ping, Kainan Peng, and Jitong Chen. Clarinet: Parallel wave generation in end-to-end text-to-speech. In *ICLR (Poster)*. OpenReview.net, 2019.

[5] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. In *NeurIPS*, pages 3165–3174, 2019.

[6] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. In *AAAI*, pages 6706–6713. AAAI Press, 2019.

[7] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *ICLR*. OpenReview.net, 2021.

[8] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. Neural codec language models are zero-shot text to speech synthesizers. *CoRR*, abs/2301.02111, 2023.

[9] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE ACM Trans. Audio Speech Lang. Process.*, 30:495–507, 2022.

[10] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Trans. Mach. Learn. Res.*, 2023, 2023.

[11] Zhihao Du, Shiliang Zhang, Kai Hu, and Siqi Zheng. Funcodec: A fundamental, reproducible and integrable open-source toolkit for neural speech codec. In *ICASSP*, pages 591–595. IEEE, 2024.

[12] Eugene Kharitonov, Damien Vincent, Zalán Borsos, Raphaël Marinier, Sertan Girgin, Olivier Pietquin, Matt Sharifi, Marco Tagliasacchi, and Neil Zeghidour. Speak, read and prompt: High-fidelity text-to-speech with minimal supervision. *Trans. Assoc. Comput. Linguistics*, 11:1703–1718, 2023.

[13] Yakun Song, Zhuo Chen, Xiaofei Wang, Ziyang Ma, and Xie Chen. ELLA-V: stable neural codec language modeling with alignment-guided sequence reordering. *CoRR*, abs/2401.07333, 2024.

[14] Chenpeng Du, Yiwei Guo, Hankun Wang, Yifan Yang, Zhikang Niu, Shuai Wang, Hui Zhang, Xie Chen, and Kai Yu. VALL-T: decoder-only generative transducer for robust and decoding-controllable text-to-speech. *CoRR*, abs/2401.14321, 2024.

[15] Detai Xin, Xu Tan, Kai Shen, Zeqian Ju, Dongchao Yang, Yuancheng Wang, Shinnosuke Takamichi, Hiroshi Saruwatari, Shujie Liu, Jinyu Li, and Sheng Zhao. RALL-E: robust codec language modeling with chain-of-thought prompting for text-to-speech synthesis. *CoRR*, abs/2404.03204, 2024.

[16] Sanyuan Chen, Shujie Liu, Long Zhou, Yanqing Liu, Xu Tan, Jinyu Li, Sheng Zhao, Yao Qian, and Furu Wei. VALL-E 2: Neural codec language models are human parity zero-shot text to speech synthesizers. *CoRR*, abs/2406.05370, 2024.

[17] Bing Han, Long Zhou, Shujie Liu, Sanyuan Chen, Lingwei Meng, Yanming Qian, Yanqing Liu, Sheng Zhao, Jinyu Li, and Furu Wei. VALL-E R: robust and efficient zero-shot text-to-speech synthesis via monotonic alignment. *CoRR*, abs/2406.07855, 2024.

[18] Yuancheng Wang, Haoyue Zhan, Liwei Liu, Ruihong Zeng, Haotian Guo, Jiachen Zheng, Qiang Zhang, Shunsi Zhang, and Zhizheng Wu. Maskgct: Zero-shot text-to-speech with masked generative codec transformer. *CoRR*, abs/2409.00750, 2024.

[19] Takuma Okamoto, Haruki Yamashita, Yamato Ohtani, Tomoki Toda, and Hisashi Kawai. Wavenext: Convnext-based fast neural vocoder without ISTFT layer. In *ASRU*, pages 1–8. IEEE, 2023.

[20] Hubert Siuzdak. Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis. In *ICLR*. OpenReview.net, 2024.

[21] Lingwei Meng, Long Zhou, Shujie Liu, Sanyuan Chen, Bing Han, Shujie Hu, Yanqing Liu, Jinyu Li, Sheng Zhao, Xixin Wu, Helen Meng, and Furu Wei. Autoregressive speech synthesis without vector quantization. *CoRR*, abs/2407.08551, 2024.

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[23] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*. OpenReview.net, 2021.

[24] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*. OpenReview.net, 2023.

[25] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-guided multilingual universal speech generation at scale. In *NeurIPS*, 2023.

[26] Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Eric Liu, Yichong Leng, Kaitao Song, Siliang Tang, Zhizheng Wu, Tao Qin, Xiangyang Li, Wei Ye, Shikun

Zhang, Jiang Bian, Lei He, Jinyu Li, and Sheng Zhao. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. In *ICML*. OpenReview.net, 2024.

[27] Yiwei Guo, Chenpeng Du, Ziyang Ma, Xie Chen, and Kai Yu. Voiceflow: Efficient text-to-speech with rectified flow matching. In *ICASSP*, pages 11121–11125. IEEE, 2024.

[28] Shivam Mehta, Ruibo Tu, Jonas Beskow, Éva Székely, and Gustav Eje Henter. Matcha-tts: A fast TTS architecture with conditional flow matching. In *ICASSP*, pages 11341–11345. IEEE, 2024.

[29] Yuan Gao, Nobuyuki Morioka, Yu Zhang, and Nanxin Chen. E3 TTS: easy end-to-end diffusion-based text to speech. In *ASRU*, pages 1–8. IEEE, 2023.

[30] Keon Lee, Dong Won Kim, Jaehyeon Kim, and Jaewoong Cho. Ditto-tts: Efficient and scalable zero-shot text-to-speech with diffusion transformer. *CoRR*, abs/2406.11427, 2024.

[31] Sefik Emre Eskimez, Xiaofei Wang, Manthan Thakker, Canrun Li, Chung-Hsien Tsai, Zhen Xiao, Hemin Yang, Zirun Zhu, Min Tang, Xu Tan, Yanqing Liu, Sheng Zhao, and Naoyuki Kanda. E2 TTS: embarrassingly easy fully non-autoregressive zero-shot TTS. *CoRR*, abs/2406.18009, 2024.

[32] 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. *CoRR*, abs/2410.06885, 2024.

[33] Philip Anastassiou, Jiawei Chen, Jitong Chen, Yuanzhe Chen, Zhuo Chen, Ziyi Chen, Jian Cong, Lelai Deng, Chuang Ding, Lu Gao, Mingqing Gong, Peisong Huang, Qingqing Huang, Zhiying Huang, Yuanyuan Huo, Dongya Jia, Chumin Li, Feiya Li, Hui Li, Jiaxin Li, Xiaoyang Li, Xingxing Li, Lin Liu, Shouda Liu, Sichao Liu, Xudong Liu, Yuchen Liu, Zhengxi Liu, Lu Lu, Junjie Pan, Xin Wang, Yuping Wang, Yuxuan Wang, Zhen Wei, Jian Wu, Chao Yao, Yifeng Yang, Yuanhao Yi, Junteng Zhang, Qidi Zhang, Shuo Zhang, Wenjie Zhang, Yang Zhang, Zilin Zhao, Dejian Zhong, and Xiaobin Zhuang. Seed-tts: A family of high-quality versatile speech generation models. *CoRR*, abs/2406.02430, 2024.

[34] Zhihao Du, Qian Chen, Shiliang Zhang, Kai Hu, Heng Lu, Yexin Yang, Hangrui Hu, Siqi Zheng, Yue Gu, Ziyang Ma, Zhifu Gao, and Zhijie Yan. Cosyvoice: A scalable multilingual zero-shot text-to-speech synthesizer based on supervised semantic tokens. *CoRR*, abs/2407.05407, 2024.

[35] Haohan Guo, Kun Liu, Feiyu Shen, Yi-Chen Wu, Feng-Long Xie, Kun Xie, and Kaituo Xu. Fireredtts: A foundation text-to-speech framework for industry-level generative speech applications. *CoRR*, abs/2409.03283, 2024.

[36] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[37] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *EMNLP (Findings)*, pages 15757–15773. Association for Computational Linguistics, 2023.

[38] Trung Dang, David Aponte, Dung N. Tran, and Kazuhito Koishida. Livespeech: Low-latency zero-shot text-to-speech via autoregressive modeling of audio discrete codes. *CoRR*, abs/2406.02897, 2024.

[39] Trung Dang, David Aponte, Dung N. Tran, Tianyi Chen, and Kazuhito Koishida. Zero-shot text-to-speech from continuous text streams. *CoRR*, abs/2410.00767, 2024.

[40] Mateusz Lajszczak, Guillermo Cámbara, Yang Li, Fatih Beyhan, Arent van Korlaar, Fan Yang, Arnaud Joly, Álvaro Martín-Cortinas, Ammar Abbas, Adam Michalski, Alexis Moinet, Sri Karlapati, Ewa Muszynska, Haohan Guo, Bartosz Putrycz, Soledad López Gambino, Kayeon Yoo, Elena Sokolova, and Thomas Drugman. BASE TTS: lessons from building a billion-parameter text-to-speech model on 100k hours of data. *CoRR*, abs/2402.08093, 2024.

[41] Avihu Dekel, Slava Shechtman, Raul Fernandez, David Haws, Zvi Kons, and Ron Hoory. Speak while you think: Streaming speech synthesis during text generation. In *ICASSP*, pages 11931–11935. IEEE, 2024.

[42] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *ICLR*. OpenReview.net, 2024.

[43] Tongyi Speech Team. Funaudiollm: Voice understanding and generation foundation models for natural interaction
between humans and llms. *arxiv*, 2024.

[44] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[45] Qwen Team. Qwen2.5: A party of foundation models, September 2024.

[46] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[47] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[48] 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. *Advances in neural information processing systems*, 36, 2024.

[49] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

[50] 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*, pages 2063–2067. ISCA, 2022.

[51] Chenpeng Du, Yiwei Guo, Feiyu Shen, Zhijun Liu, Zheng Liang, Xie Chen, Shuai Wang, Hui Zhang, and Kai Yu. Unicats: A unified context-aware text-to-speech framework with contextual vq-diffusion and vocoding. In *AAAI*, pages 17924–17932. AAAI Press, 2024.

[52] Yafeng Chen, Siqi Zheng, Hui Wang, Luyao Cheng, Qian Chen, and Jiajun Qi. An enhanced res2net with local and global feature fusion for speaker verification. In *Interspeech*. ISCA, 2023.

[53] Chandan K. A. Reddy, Vishak Gopal, and Ross Cutler. Dnsmos P.835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors. In *ICASSP*, pages 886–890. IEEE, 2022.

[54] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR, 2023.

[55] Shu-wen Yang, Heng-Jui Chang, Zili Huang, Andy T Liu, Cheng-I Lai, Haibin Wu, Jiatong Shi, Xuankai Chang, Hsiang-Sheng Tsai, Wen-Chin Huang, et al. A large-scale evaluation of speech foundation models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.

[56] 2noise. Chattts. `https://github.com/2noise/ChatTTS`, 2024.

[57] RVC-Boss. Gpt-sovits. `https://github.com/RVC-Boss/GPT-SoVITS`, 2024.

[58] Zengyi Qin, Wenliang Zhao, Xumin Yu, and Xin Sun. Openvoice: Versatile instant voice cloning. *CoRR*, abs/2312.01479, 2023.

[59] Daniel Lyth and Simon King. Natural language guidance of high-fidelity text-to-speech with synthetic annotations. *CoRR*, abs/2402.01912, 2024.

[60] Netease Youdao. Emotivoice. `https://github.com/netease-youdao/EmotiVoice`, 2024.