SyncSpeech: Low-Latency and Efficient Dual-Stream Text-to-Speech based on Temporal Masked Transformer

Anonymous ACL submission

Abstract

This paper presents a dual-stream text-to-speech (TTS) model, SyncSpeech, capable of receiving streaming text input from upstream models while simultaneously generating streaming speech, facilitating seamless interaction with large language models. SyncSpeech has the following advantages: Low latency, as it begins generating streaming speech upon receiving the second text token; High efficiency, as it decodes all speech tokens corresponding to the each arrived text token in one step. To achieve this, we propose a temporal masked transformer as the backbone of SyncSpeech, combined with token-level duration prediction to predict speech tokens and the duration for the next step. Additionally, we design a two-stage training strategy to improve training efficiency and the quality of generated speech. We evaluated the SyncSpeech on both English and Mandarin datasets. Compared to the recent dual-stream TTS models, SyncSpeech significantly reduces the first packet delay of speech tokens and accelerates the real-time factor. Moreover, with the same data scale, SyncSpeech achieves performance comparable to that of traditional autoregressive-based TTS models in terms of both speech quality and robustness. Speech samples are available at https://SyncSpeech.github.io/.

1 Introduction

In recent years, with advancements in generative models and the expansion of training datasets, text-to-speech (TTS) models (Wang et al., 2023; Le et al., 2023; Ju et al., 2024) have made breakthrough progress in naturalness and quality, gradually approaching the level of real recordings. However, low-latency and efficient dual-stream TTS, which involves processing streaming text inputs while simultaneously generating speech in real time, remains a challenging problem (Dang et al., 2024). These models are ideal for integration with upstream tasks, such as large language models (LLMs) (OpenAI, 2023) and streaming translation models (Barrault et al., 2023), which can generate text in a streaming manner. Addressing these challenges can improve live human-computer interaction, paving the way for various applications, such as speech-to-speech translation and personal voice assistants.

Recently, inspired by advances in image generation, denoising diffusion (Ho et al., 2020; Song et al., 2021), flow matching (Lipman et al., 2023), and masked generative models (Chang et al., 2022) have been introduced into non-autoregressive (NAR) TTS (Anastassiou et al., 2024; Chen et al., 2024b; Kim et al., 2023; Wang et al., 2024), demonstrating impressive performance in offline inference. During this process, these offline TTS models first add noise or apply masking guided by the predicted duration. Subsequently, context from the entire sentence is leveraged to perform temporally-unordered denoising or mask prediction for speech generation. However, this temporally-unordered process hinders their application to streaming speech generation[1].

When it comes to streaming speech generation, autoregressive (AR) TTS models (Wang et al., 2023; Song et al., 2024) hold a distinct advantage because of their ability to deliver outputs in a temporally-ordered manner. However, compared to recently proposed NAR TTS models, AR TTS models have a distinct disadvantage in terms of generation efficiency (Li et al., 2024). Specifically, the autoregressive steps are tied to the frame rate of speech tokens, resulting in slower inference speeds. While advancements like VALL-E 2 (Chen et al., 2024a) have boosted generation efficiency through group code modeling, the challenge remains that the manually set group size is typically small, sug-

[1]Here, "temporally" refers to the physical time of audio samples, not the iteration step $t \in [0, 1]$ of the above NAR TTS models.

gesting room for further improvements. In addition, most current AR TTS models (Dekel et al., 2024) cannot handle stream text input and they only begin streaming speech generation after receiving the complete text, ignoring the latency caused by the streaming text input. The most closely related works to SyncSpeech are CosyVoice2 (Du et al., 2024b) and IST-LM (Yang et al., 2024b), both of which employ interleaved speech-text modeling to accommodate dual-stream scenarios. However, their autoregressive process generates only one speech token per step, leading to low efficiency.

To seamlessly integrate with upstream LLMs and facilitate dual-stream speech synthesis, this paper introduces **SyncSpeech**, designed to keep the generation of streaming speech in synchronization with the incoming streaming text. SyncSpeech has the following advantages: 1) **low latency**, which means it begins generating speech in a streaming manner as soon as the second text token is received, and 2) **high efficiency**, which means for each arriving text token, only one decoding step is required to generate all the corresponding speech tokens.

SyncSpeech is based on the proposed **T**emporal **M**asked generative **T**ransformer (TMT). During inference, SyncSpeech adopts the Byte Pair Encoding (BPE) token-level duration prediction, which can access the previously generated speech tokens and performs top-k sampling. Subsequently, mask padding and greedy sampling are carried out based on the duration prediction from the previous step.

Moreover, sequence input is meticulously constructed to incorporate duration prediction and mask prediction into a single decoding step. During the training process, we adopt a two-stage training strategy to improve training efficiency and model performance. First, high-efficiency masked pre-training is employed to establish a rough alignment between text and speech tokens within the sequence, followed by fine-tuning the pre-trained model to align with the inference process.

Our experimental results demonstrate that, in terms of generation efficiency, SyncSpeech operates at 6.4 times the speed of the current dual-stream TTS model for English and at 8.5 times the speed for Mandarin. When integrated with LLMs, SyncSpeech achieves latency reductions of 3.2 and 3.8 times, respectively, compared to the current dual-stream TTS model for both languages. Moreover, with the same scale of training data, SyncSpeech performs comparably to traditional AR models in terms of the quality of generated English speech. For Mandarin, SyncSpeech demonstrates superior quality and robustness compared to current dual-stream TTS models. This showcases the potential of SyncSpeech as a foundational model to integrate with upstream LLMs.

2 Related Work

2.1 Text-to-Speech

Text-to-Speech, the transformation of text into audible signals understandable by humans, is pivotal for human-computer interaction. TTS systems can be mainly divided into AR-based and NAR-based categories. For AR-based systems, VALL-E (Wang et al., 2023) predicts the first layer of acoustic tokens extracted by EnCodec (Défossez et al., 2023) using an AR codec language model, while a NAR model is used to predict the remaining layers. CosyVoice (Du et al., 2024a) employs an AR model to predict supervised semantic representations and combines flow matching to predict acoustic representations. AR-based TTS models, with their in-context learning capability, can generate natural, prosody-diverse speech in a streaming manner. However, AR-based TTS models exhibit shortcomings in generation efficiency. Besides the previously mentioned VALL-E 2 (Chen et al., 2024a), MEDUSA (Li et al., 2024) and VALL-E R (Han et al., 2024) introduce speculative decoding (Leviathan et al., 2023) and a codec-merging method, respectively, to accelerate autoregressive generation. Nonetheless, the efficiency gains achieved by these approaches remain limited, unable to perform synchronized decoding steps with text tokens.

For NAR-based TTS models, most previous approaches require speech duration prediction conditioned on the input text, followed by upsampling the text representations to match the acoustic feature length before feeding them into the generation model. Following FastSpeech (Ren et al., 2021), VoiceBox (Le et al., 2023) and NaturalSpeech 2 (Shen et al., 2024) predict phone-level durations using a regression-based approach. NaturalSpeech 3 (Ju et al., 2024) adopts a discrete diffusion model, combining classification loss and duration prompts for duration prediction, which outperforms text-dependent regression-based duration prediction in terms of speech robustness and quality. However, NaturalSpeech 3 requires an additional duration prediction model, which complicates the pipeline, whereas SyncSpeech integrates duration

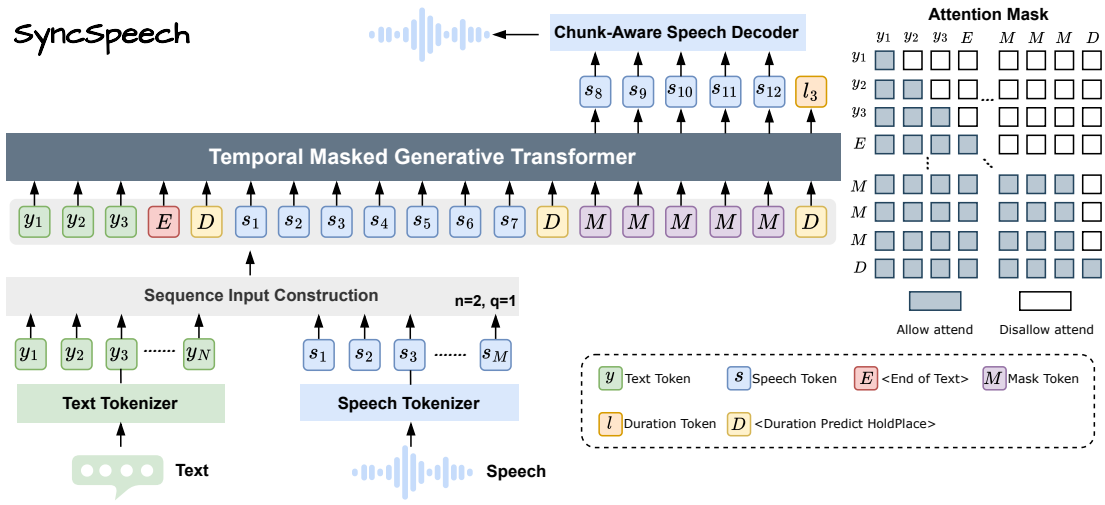

Figure 1: An overview of the proposed SyncSpeech, comprising a text tokenizer, a speech tokenizer, a temporal masked generative transformer and a chunk-aware speech decoder. The figure shows that, with the random number $n = 2$ and text look-ahead value $q = 1$, it estimates all speech tokens (from s_8 to s_{12}) corresponding to the text token y_2 and the duration (l_3) of the next text token y_3 in one decoding step.

and speech token predictions into a unified framework. The NAR TTS model most relevant to SyncSpeech is MaskGCT (Wang et al., 2024), which predicts the total duration of the speech and then performs temporally-unordered multi-step mask prediction. Unlike MaskGCT, SyncSpeech employs temporally-ordered mask prediction and BPE token-level duration prediction to achieve speech generation in a dual-stream scenario.

2.2 Speech Large Language Models

Speech Large Language Models (SLLMs) empower LLMs to interact with users through speech, responding to user's instruction with low latency (Ji et al., 2024a). A basic approach (Huang et al., 2024) to achieve this speech interaction involves a cascade of automatic speech recognition (ASR), LLM and TTS models, where the ASR transcribes the users' speech instruction into text, and the TTS model converts the LLM's textual response into speech. However, most current AR TTS models cannot process streaming text input, resulting in significant latency in the aforementioned cascaded systems. In contrast, some end-to-end speech-language models have been proposed that can generate speech tokens directly, thereby achieving extremely low response latency. LLaMA-Omni (Fang et al., 2024) aligns the hidden states of LLMs with discrete HuBERT (Hsu et al., 2021) representations using CTC loss, but the generated speech exhibits less natural prosody. Mini-Omni (Xie and Wu, 2024) employs a parallel decoder approach to generate text and speech tokens simultaneously. However, due to the significantly longer length of speech tokens compared to text tokens, its generation efficiency remains low. The proposed SyncSpeech can process streaming text input and generates speech in synchronization, with the potential to unite with LLMs to become end-to-end SLLMs.

3 Method

A dual-stream TTS model simultaneously processes streaming text input and generates speech in a streaming manner. Upon receiving newly generated text tokens $\boldsymbol{y}_{\text{arr}}$ from the upstream LLMs, the objective of the dual-streaming TTS it to estimate $p(\boldsymbol{x}_{\text{arr}}|\boldsymbol{y}_{\text{arr}}, \boldsymbol{x}_{\text{pre}}, \boldsymbol{y}_{\text{pre}})$. In this context, $\boldsymbol{x}_{\text{arr}}$ represents the speech waveform segment corresponding to $\boldsymbol{y}_{\text{arr}}$, while $\boldsymbol{y}_{\text{pre}}$ and $\boldsymbol{x}_{\text{pre}}$ denote the preceding text tokens and its corresponding speech waveform, respectively.

SyncSpeech is a two-stage TTS system, consisting of the text-to-token and token-to-speech stages. The estimation of $p(\boldsymbol{x}_{\text{arr}}|\boldsymbol{y}_{\text{arr}}, \boldsymbol{x}_{\text{pre}}, \boldsymbol{y}_{\text{pre}})$ is decomposed into a text-to-token model $p(\boldsymbol{s}_{\text{arr}}|\boldsymbol{y}_{\text{arr}}, \boldsymbol{x}_{\text{pre}}, \boldsymbol{y}_{\text{pre}})$ and a token-to-speech model $p(\boldsymbol{x}_{\text{arr}}|\boldsymbol{s}_{\text{arr}})$, where $\boldsymbol{s}_{\text{arr}}$ is the speech tokens corresponding to the speech waveform segment $\boldsymbol{x}_{\text{arr}}$. Specifically, the proposed TMT is adopted as the backbone of text-to-token model. Then, an off-the-shelf chunk-aware speech decoder (Du et al., 2024b) is adopted as the token-to-speech model.

The proposed TMT module is based on a llama-style Transformer (Touvron et al., 2023). We have specifically designed a novel attention mask to accommodate temporally-ordered mask generation. Below, I will detail the two-stage training strategy of the TMT-based text-to-token model and its attention mask, cover the details of the other modules, and describe the inference process.

3.1 Training

Given a dataset of transcribed speech $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$, where $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ denote an audio sample and its transcript, respectively, the transcript $\tilde{\boldsymbol{y}}$ is tokenized into a BPE token sequence $\boldsymbol{y} = [y_1, y_2, y_3, ..., y_L]$, where L is the number of BPE tokens. An off-the-shelf speech tokenizer is used to encode the speech sample $\tilde{\boldsymbol{x}}$ into T frame discrete speech tokens $\boldsymbol{s} = [s_1, s_2, s_3, ..., s_T]$. We further define duration tokens $\boldsymbol{a} = [a_1, a_2, a_3, ..., a_L]$ as the positions indicating the end time of each corresponding BPE token within the speech token sequence, with $a_L = T$. For a pair of $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$, $\boldsymbol{a}$ can be obtained through an open-source alignment tool.

As shown in Figure 1, to maintain consistency with the inference process (see Section 3.4), the sequence input is then constructed as follows. We select a random number $n \in [1, L]$, which indicates that when receiving streaming text input, SyncSpeech needs to generate the speech tokens corresponding to the n-th BPE token at this moment. To avoid unnatural pauses, SyncSpeech allows look ahead q text tokens, obtaining a truncated text token sequence $\boldsymbol{y}' = [y_1, y_2, y_3, ..., y_{L'}]$, where $L' = min(L, n + q)$. Based on the duration tokens $\boldsymbol{a}$, the truncated speech token sequence $\boldsymbol{s}_{1:a_n} = [s_1, s_2, ..., s_{a_n}]$ is obtained. Then, we define the masked speech token sequence $\boldsymbol{s}'$ and and corresponding binary mask $\boldsymbol{m}$ as follows,

$$\boldsymbol{s}' = \boldsymbol{s}_{1:a_n} \odot \boldsymbol{m}, \tag{1}$$

$$\boldsymbol{m} = [m_i]_{i=1}^{a_n}, \boldsymbol{m}_{1:a_{n-1}} = 0, \boldsymbol{m}_{a_{n-1}:a_n} = 1. \tag{2}$$

That is all speech tokens corresponding to $\boldsymbol{x}_n$ are replaced with the specific mask token, while the rest remain unchanged. Then, the truncated text token sequence $\boldsymbol{y}'$, along with the masked speech token sequence $\boldsymbol{s}'$ and duration tokens $\boldsymbol{a}$, are used to construct the input sequence as follows,

$$\boldsymbol{f} = [\boldsymbol{y}', E, D, \boldsymbol{s}'_{1:a_1}, ..., D, \boldsymbol{s}'_{a_{n-1}:a_n}, D], \tag{3}$$

where E is end-of-text token, D is a placeholder for duration prediction. Based on the duration tokens $\boldsymbol{a}$, D is used to separate the masked speech token sequence corresponding to different BPE tokens. In practice, E is inserted only when $n = L$.

The sequence $\boldsymbol{f}$ is used as input for the TMT with the mask prediction and duration prediction as training objectives. Specifically, the sequence $\boldsymbol{f}$ is fed into the TMT forward to obtain the hidden states, which then pass through two different linear layers to predict the speech tokens corresponding to text token y_n and the duration of the **next text token** y_{n+1}. This enables us to integrate duration prediction and mask prediction into a single decoding step during inference, except for the first text token duration prediction (Details are provided in Section 3.4). We minimize the following negative log-likelihood function for masked generative training and duration training,

$$\mathcal{L}^1_{\text{mask}} = -\log p\left(\boldsymbol{s}_{a_{n-1}:a_n} \mid \boldsymbol{f}; \theta\right), \tag{4}$$

$$\mathcal{L}^1_{\text{duration}} = -\log p\left(l_{n+1} \mid \boldsymbol{f}; \theta\right), \tag{5}$$

where θ represents the neural network parameters of TMT, $l_{n+1} = a_{n+1} - a_n$ and $a_0 = 0$. In this way, we simulate the scenario of receiving streaming text input during the training process and are able to generate speech in sync.

We design a corresponding attention mask, as shown in Figure 1. Specifically, a causal mask is used for the truncated text sequence $\boldsymbol{y}'$ and duration placeholder parts. For the masked speech token sequence $\boldsymbol{s}'$, a dynamic chunk attention mask is applied based on the duration tokens $\boldsymbol{a}$, enabling it to attend all historical tokens, as well as all speech tokens and mask tokens corresponding to their own text BPE tokens.

3.2 Pretraining

While the aforementioned method aligns with the prediction process, it suffers from low training efficiency. This training inefficiency arises because, during each training step, only the gradients of speech tokens $\boldsymbol{s}_{a_{n-1}:a_n}$ and durations for y_{n+1} are backpropagated. To further improve the training efficiency, we first perform masked pre-training on the TMT.

Given speech tokens $\boldsymbol{s}$ of a speech sample, we obtain the masked speech tokens $\hat{\boldsymbol{s}} = \boldsymbol{s} \odot \hat{\boldsymbol{m}}$, where $\hat{\boldsymbol{m}} = [\hat{m}_i]_{i=1}^{a_L}$ is a binary mask of speech tokens. We design the masking rules primarily from two perspectives, high masking probability and consistency with the prediction process as much as possible. Specifically, the binary mask $\hat{\boldsymbol{m}}_{\text{bpe}}$ of

text tokens is constructed first, where the first value is distributed according to a Bernoulli distribution ($p = 0.5$) and the subsequent adjacent values cannot be the same. Based on the duration tokens $\boldsymbol{a}$, the text token mask $\hat{\boldsymbol{m}}_{\text{bpe}}$ is converted into the corresponding speech token mask $\hat{\boldsymbol{m}}$. Then, we build the following sequence as the input for TMT,

$$\hat{\boldsymbol{f}} = [\boldsymbol{y}, E, D, \hat{\boldsymbol{s}}_{1:a_1}, ..., D, \hat{\boldsymbol{s}}_{a_{L-1}:a_L}], \quad (6)$$

and the TMT is optimized to minimize the negative log-likelihood for masked generative training and duration training as follows,

$$\mathcal{L}^2_{\text{mask}} = -\sum_{j \in J} \log p\left(\boldsymbol{s}_{a_{j-1}:a_j} \mid \hat{\boldsymbol{f}}_{\leq a_j}; \theta\right), \quad (7)$$

$$\mathcal{L}^2_{\text{duration}} = -\sum_{j \in J} \log p\left(l_j \mid \hat{\boldsymbol{f}}_{\leq a_{j-1}}; \theta\right), \quad (8)$$

where J denote the sequence where each element satisfies the condition $\hat{m}_j = 1$, $\hat{\boldsymbol{f}}_{\leq a_j} = [\boldsymbol{y}, E, D, \hat{\boldsymbol{s}}_{1:a_1}, ..., D, \hat{\boldsymbol{s}}_{a_{j-1}:a_j}]$ and $l_j = a_j - a_{j-1}$. Additionally, the attention mask mentioned above is also utilized for pretrain.

In summary, an efficient masked pretraining is initially performed with a high masking probability to facilitate the alignment between text and speech tokens. Subsequently, we fine-tune the pretrained model using a training strategy consistent with the prediction process. This approach enhances the efficiency of the training process, and the masked pretraining also contributes to the robustness of the generated speech.

3.3 Other Modules

In this subsection, we introduce the other modules in SyncSpeech besides TMT. 1) Text BPE tokenizer: To facilitate interaction with upstream LLMs, we utilize the Qwen tokenizer (Yang et al., 2024a) directly. 2) Speech tokenizer: the open-source supervised speech semantic (S3) tokenizer (Du et al., 2024b) is selected, which operates at 25 Hz. The S3 tokenizer is developed by integrating finite scalar quantization (FSQ) (Mentzer et al., 2024) into the intermediate representations of an ASR model trained on large-scale data, and then fine-tuning it for the ASR task. 3) The off-the-shelf speech decoder (Du et al., 2024b) is based on the conditional flow matching (CFM) decoder and HiFi-GAN vocoder (Kong et al., 2020). The CFM decoder employs a chunk-aware training strategy, enabling the streaming generation of Mel-spectrograms from the chunk-size input speech tokens. These Mel-spectrograms are then converted into speech using the vocoder, which operates in parallel with a fully convolutional network.

Algorithm 1: Inference in Python Style

Input: Streaming text input $\boldsymbol{y}$
Output: Streaming speech output $\boldsymbol{o}$

```
sequence input f = [] ;
speech tokens s = [];
if length(y)> q then
    for Index, y in enumerate(y) do
        f = build_seq(y, f);
        # Follow Equation3 ;
        if Index== 0 then
            dur = TMT(f) ;
            f = pad_seq(f, dur) ;
            pre_dur = dur
        s_cur, dur = TMT(f) ;
        s.append(s_cur) ;
        f[pre_dur-1:-1] = s ;
        f = pad_seq(f, dur) ;
        pre_dur = dur;
        if length(s)≥ chunk then
            o = decoder(s);
            # output new generated speech ;
            s.update();
```

3.4 Inference

During the inference process, SyncSpeech processes text in a streaming manner and synchronously generates speech, with the general algorithm flow shown in Algorithm 1. Specifically, when the number of input text BPE tokens $\boldsymbol{y}$ exceeds the look-ahead number q, the input sequence $\boldsymbol{f} = [\boldsymbol{y}, D]$ is built, which is fed into TMT to predict the duration of speech tokens corresponding to y_1. Then, based on the predicted duration, we perform sequence padding by inserting the mask tokens and a duration prediction placeholder. Subsequently, the sequence is fed back into TMT for synchronous mask prediction of y_1 and the duration prediction of y_2, followed by the input sequence $\boldsymbol{s}$ update and padding. For subsequent BPE token input, the above prediction step, update step, and padding step are repeated to generate speech tokens in a streaming manner. In the process described above, once the number of generated speech tokens surpasses the chunk size of the off-the-shelf speech decoder, these tokens and the speaker prompt can be utilized to stream speech output.

Additionally, existing speech tokens can be accessed during duration prediction and speech token generation, which allows SyncSpeech to control the prosody of the generated speech with in-context learning. Specifically, given a speech prompt, we construct the prompt sequence according to Equation 3, serving as the generated sequence for prosody control. Figure 2 in the Appendix shows detailed inference visualizations.

4 Experiments

4.1 Experimental Settings

Datasets We trained SyncSpeech on datasets in both English and Mandarin, including the 585-hour LibriTTS (Zen et al., 2019) dataset and 600 hours of internal Mandarin datasets. The internal Mandarin dataset was further expanded to approximately 2000 hours, employing techniques such as speed alteration and pitch shifting. The Montreal Forced Aligner (MFA) (McAuliffe et al., 2017) aligned transcripts according to its phone set, after which the alignment was transformed into text BPE-level format. We evaluated SyncSpeech using three benchmarks: (1) LibriSpeech *text-clean* (Panayotov et al., 2015), a standard English TTS evaluation set; (2) SeedTTS *test-zh* (Anastassiou et al., 2024), with 2,000 samples from the out-of-domain Mandarin DiDiSpeech dataset (Guo et al., 2021); and (3) SeedTTS *test-hard*, containing approximately 400 difficult cases to evaluate TTS model robustness with repeated text, tongue twisters, and other complex synthesis scenarios.

Settings We set the number of text tokens to look ahead $q = 1$. The chunk size of speech decoder is 15. TMT has 16 layers, 16 attention heads, 1024-dimensional embeddings, and 2048-dimensional feed-forward layers. SyncSpeech was trained on 4 NVIDIA A800 80G GPUs. The pre-training stage lasts for 70K steps, and the second stage lasts for 20K steps.

Baseline Models This paper focuses on low-latency and efficient TTS in dual-stream scenarios. Under the same data scale, we reproduced the following baseline models for comparison: CosyVoice (Du et al., 2024a) and recently proposed CosyVoice2 (Du et al., 2024b). CosyVoice requires complete text input before speech generation. CosyVoice2 uses interleaved text-speech modeling to process streaming text input and simultaneously generate streaming speech. We trained CosyVoice, CosyVoice2, and SyncSpeech using the same speech tokenizer and text tokenizer, and employed the same open-source streaming speech decoder. We utilized the official code[2] to reproduce the model and adopted a Llama-style Transformer, matching the size of SyncSpeech, as the backbone of the text-to-speech model. Additionally, we compared the open-sourced TTS model MaskGCT (Wang et al., 2024), F5-TTS (Chen et al., 2024b), and VALL-E (Wang et al., 2023), which were trained on large-scale data. More details about baseline models can be found in the Appendix A.

Evaluation Metrics For the three benchmarks, we evaluated speech quality, latency, and efficiency. For speech robustness, we chose Whisper-V3 and Paraformer as the ASR models for English and Mandarin, respectively, to transcribe the generated speech. Then, we calculated the WER compared to the original transcriptions to evaluate the spech robustness. We adopted the ERes2Net-based (Chen et al., 2023) speaker verification model[3] to evaluate speaker similarity (SS). We selected 100 sentences from each system and invited 10 native listeners to conduct a subjective MOS evaluation for speech naturalness (MOS-N), scoring from 1 to 5. In terms of latency and efficiency, we compared the performance of various models on a single A800 GPU. Due to the off-the-shelf speech decoder, we evaluate the latency and efficiency of the text-to-token stage across all models, except for F5-TTS. We calculated the time required for the number of speech tokens to reach the chunk size of the speech decoder as First-packet latency (FPL). There are two scenarios: one assumes the text is already available (FPL-A), while the other involves receiving output from the upstream LLM model (FPL-L), accounting for the time required for text generation. For the real-time factor (RTF), we measure the ratio of the total duration of generated speech to the total time taken by the model. More details about FPL and RTF can be found in the Appendix B.

4.2 Main Results

The evaluation results for SyncSpeech and the baseline models are presented in Table 1.

Speech Robustness We found that SyncSpeech exhibits different performance compared to the baselines across the three benchmarks. Specifically,

[2]https://github.com/FunAudioLLM/CosyVoice
[3]https://github.com/modelscope/3D-Speaker

Model	#Scenario	#Data(hrs)	WER(%) ↓	SS(%) ↑	FPL-A(s)↓	FPL-L(s) ↓	RTF(%) ↓	MOS-N ↑
LibriSpeech *test-clean*								
Ground Truth	-	-	2.12	69.67	-	-	-	$4.62_{\pm 0.12}$
F5-TTS*	Offline	100K Multi.	**2.51**	**73.10**	1.27	1.98	0.23	-
MASK-GCT*	Offline	100K Multi.	2.77	70.81	2.15	2.55	0.37	-
VALL-E*	Output Stream	60K EN	5.90	59.71	0.75	1.47	1.41	-
CosyVoice	Output Stream	585 EN	3.47	63.52	0.22	0.94	0.45	$4.39_{\pm 0.12}$
CosyVoice2	Dual-Stream	585 EN	3.00	63.48	0.22	0.35	0.45	$\mathbf{4.48}_{\pm 0.13}$
SyncSpeech	Dual-Stream	585 EN	3.07	63.47	**0.06**	**0.11**	**0.07**	$\mathbf{4.48}_{\pm 0.14}$
Seed *test-zh*								
Ground Truth	-	-	1.26	75.15	-	-	-	$4.68_{\pm 0.10}$
CosyVoice	Output Stream	2K ZH	3.03	61.51	0.22	0.62	0.43	$4.34_{\pm 0.14}$
CosyVoice2	Dual-Stream	2K ZH	3.31	61.89	0.22	0.35	0.43	$4.37_{\pm 0.13}$
SyncSpeech	Dual-Stream	2K ZH	**2.38**	**62.14**	**0.04**	**0.09**	**0.05**	$\mathbf{4.45}_{\pm 0.11}$
Seed *test-hard*								
CosyVoice	Output Stream	2K ZH	26.26	66.71	0.22	1.22	0.44	$3.84_{\pm 0.15}$
CosyVoice2	Dual-Stream	2K ZH	21.61	67.13	0.22	0.35	0.44	$3.86_{\pm 0.14}$
SyncSpeech	Dual-Stream	2K ZH	**17.21**	**67.21**	**0.05**	**0.10**	**0.08**	$3.86_{\pm 0.11}$

Table 1: The evaluation results of SyncSpeech and baseline models across the three benchmarks. * indicates the model trained on the large-scale dataset. Underline indicates the best performance in terms of WER and SS with the 585 hours training scale. #Data refers to the used training dataset in hours.

on the LibriSpeech *test-clean* benchmark, the performance of SyncSpeech was very close to that of CosyVoice2 based on the WER metric, with only a minor difference of 0.07%. SyncSpeech achieved a lower WER score on the Seed *test-zh* set compared to CosyVoice and CosyVoice2, with improvements of 0.65% and 0.93%, respectively. A key difference between the English and Mandarin datasets is the higher compression rate of the LLM tokenizer for Mandarin. In English, one word typically equals one token, while in Mandarin, a common phrase often corresponds to a single token. This means that, compared to the baseline model, SyncSpeech is better suited to the high compression rate tokenizer of the upstream large model. Furthermore, on the Seed *test-hard* set, the robustness advantage of SyncSpeech was even more pronounced, with the improvements 9.05% and 4.40%, respectively. In handling complex text, the explicit duration modeling in SyncSpeech helped the model learn the alignment between text and speech.

Speaker Similarity Due to the same speech decoder and the excellent voice disentanglement capability of the speech tokens, SyncSpeech, CosyVoice, and CosyVoice2 exhibited similar performance in terms of speaker similarity.

Speech Naturalness The MOS-N scores for SyncSpeech and CosyVoice2 were quite similar on the LibriSpeech *text-clean*, indicating that the naturalness of the generated speech was generally comparable. On the Seed *test-zh* benchmark, SyncSpeech outperformed CosyVoice2 by 0.08. In the Seed *test-hard* benchmark, high WER and uncommon text led to unnatural prosody and generally low MOS-N scores in the generated speech.

Latency SyncSpeech has made a breakthrough in terms of latency, as shown in Table 1. Specifically, on the LibriSpeech *test-clean* benchmark, SyncSpeech was approximately 4 times faster than traditional AR models and over 20 times faster than the SOTA offline models in terms of FPL-A. On the Seed *test-zh* benchmark, SyncSpeech achieved speed improvements of over 5 times and 30 times, respectively. When receiving streaming text from the upstream large model (FPL-L), SyncSpeech can begin generating speech with just two text tokens. In contrast, CosyVoice2 requires five tokens, while CosyVoice and other baseline models need the entire text input. This highlights the distinct advantage of SyncSpeech in practical applications.

Efficiency In terms of RTF, SyncSpeech is about 6.4 times faster on the LibriSpeech *test-clean* benchmark and about 8.6 times faster on the Seed *test-zh* benchmark compared to previous AR models. On the Seed *test-hard* set, due to the increased number of text tokens caused by the uncommon text, the efficiency of SyncSpeech is slightly reduced. Theoretically, the time complexity of AR models is $O(T)$, while the time complexity of SyncSpeech is $O(L)$, where T represents the number of speech tokens and L denotes the number of text tokens, thereby significantly improving efficiency.

Sampling Strategy	WER(%)↓	UTMOSv2↑
Duration Prediction		
Ground Truth	2.59	3.45
Greedy Search	2.50	3.44
Top-k 3	**2.44**	**3.46**
Top-k 5	2.93	3.44
Top-k 10	2.76	3.41
Speech Token Prediction		
Greedy Search	**2.44**	**3.46**
Top-k 3	3.82	3.43
Top-k 5	4.23	3.43

Table 2: Performance across various Top-k thresholds for duration prediction and speech token prediction on the LibriTTS validation set.

5 Analysis

Sampling Strategy In the LibriSpeech validation set, we provided the ground-truth durations and applied greedy search along with different Top-k thresholds for duration prediction, as shown in Table 2. We found that, in terms of speech robustness, both Top-k 3 and greedy search outperformed the use of ground-truth durations in terms of the WER metric. This is because the model struggled to effectively generalize to anomalies in the ground-truth durations. We employed UTMOSv2[4] as a surrogate objective metric of MOS-N. In terms of speech naturalness, the results of Top-k 3 sampling are slightly better than those with the given ground-truth durations. Additionally, we applied different Top-k thresholds for speech token prediction. SyncSpeech exhibited superior performance during greedy search, which is different from the previous AR TTS models or offline models. This is because the speech tokens obtained through single-step decoding have the temporal dependency, which cannot be compensated by subsequent generation.

Number of Look-ahead Tokens We evaluated how varying the number of tokens to look ahead affects speech robustness and speech naturalness on two validation sets, with the results presented in Table 3. We discovered that the optimal number of look-ahead text tokens varies across different languages in terms of WER performance. This is influenced by the difference in the compression rate of text tokens and the contextual dependency in different languages. In terms of speech naturalness, when the look-ahead number q is greater than 2, the generated speech exhibits slightly more natural pauses and speed, but it results in increased latency.

[4]https://github.com/sarulab-speech/UTMOS22

	LH Num.	WER(%)↓	FPL-L(s)↓	UTMOS-v2↑
EN	q=1	**2.44**	**0.11**	3.46
	q=2	2.87	0.13	3.41
	q=3	2.52	0.16	**3.48**
	q=4	2.52	0.19	**3.48**
ZH	q=1	2.51	**0.09**	-
	q=2	2.49	0.12	-
	q=3	**2.41**	0.14	-
	q=4	**2.41**	0.17	-

Table 3: Performance with different numbers of look-ahead text tokens across two validation sets.

	English	Mandarin
SyncSpeech	**2.44**	**2.41**
w/o pretrain	3.61	3.47
w/o designed Mask	8.19	7.97

Table 4: WER (%) results of the ablation study across the two validation sets.

Ablation Study We conducted an ablation study on the pre-training strategy by directly training the randomly initialized model in a manner consistent with the prediction process. The WER results on the two validation sets are shown in Table 4. We found that pre-training significantly improved the speech robustness of the model, improving the WER metric by 1.17% and 1.06% on the two languages, respectively. This indicated that masked pre-training not only improved training efficiency but also enhanced the robustness of the synthesized speech. Additionally, a standard causal attention mask was applied to replace the designed attention mask, as shown in Table 4. If the mask token sequence of the same text token cannot attend to each other during inference, the robustness of the generated speech significantly decreased. This further demonstrated the effectiveness of the designed attention mask.

6 Conclusion

This paper presents SyncSpeech, a dual-stream speech generation model built on a temporal masked transformer. SyncSpeech can efficiently generate low-latency streaming speech from the real-time text input, maintaining the high quality and robustness of the generated speech. We conducted comprehensive performance evaluations and analysis experiments in both English and Mandarin, demonstrating its capability as a foundational model for integration with upstream LLMs. In the future, SyncSpeech will be trained on larger datasets to further improve its performance.

7 Limitations

In this section, we will analyze the limitations of SyncSpeech and discuss potential future work. SyncSpeech requires token-level alignment information, which is challenging to achieve for sentences with mixed languages, and preprocessing becomes time-consuming on large-scale datasets. In the future, we will explore semi-supervised duration prediction, which only requires the duration of a complete sentence without strict token-level alignment information, and integrate SyncSpeech into SLLM as a speech generation module. In addition, since the off-and-shelf streaming speech decoder relies on flow matching, it limits the off-the-shelf RTF and the FPL. Moreover,‘ current single-codebook acoustic tokens, such as WavTokenizer (Ji et al., 2024b), do not support streaming decoding. In the future, we will investigate efficient and low-latency streaming speech decoders.

References

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. 2024. Seed-tts: A family of high-quality versatile speech generation models. *CoRR*, abs/2406.02430.

Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, John Hoffman, Min-Jae Hwang, Hirofumi Inaguma, Christopher Klaiber, Ilia Kulikov, Pengwei Li, Daniel Licht, Jean Maillard, Ruslan Mavlyutov, Alice Rakotoarison, Kaushik Ram Sadagopan, Abinesh Ramakrishnan, Tuan Tran, Guillaume Wenzek, Yilin Yang, Ethan Ye, Ivan Evtimov, Pierre Fernandez, Cynthia Gao, Prangthip Hansanti, Elahe Kalbassi, Amanda Kallet, Artyom Kozhevnikov, Gabriel Mejia Gonzalez, Robin San Roman, Christophe Touret, Corinne Wong, Carleigh Wood, Bokai Yu, Pierre Andrews, Can Balioglu, Peng-Jen Chen, Marta R. Costa-jussà, Maha Elbayad, Hongyu Gong, Francisco Guzmán, Kevin Heffernan, Somya Jain, Justine Kao, Ann Lee, Xutai Ma, Alexandre Mourachko, Benjamin N. Peloquin, Juan Pino, Sravya Popuri, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, Anna Y. Sun, Paden Tomasello, Changhan Wang, Jeff Wang, Skyler Wang, and Mary Williamson. 2023. Seamless: Multilingual expressive and streaming speech translation. *CoRR*, abs/2312.05187.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. 2022. Maskgit: Masked generative image transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11305–11315. IEEE.

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

Yafeng Chen, Siqi Zheng, Hui Wang, Luyao Cheng, Qian Chen, and Jiajun Qi. 2023. An enhanced res2net with local and global feature fusion for speaker verification. In *24th Annual Conference of the International Speech Communication Association, Interspeech 2023, Dublin, Ireland, August 20-24, 2023*, pages 2228–2232. ISCA.

Yushen Chen, Zhikang Niu, Ziyang Ma, Keqi Deng, Chunhui Wang, Jian Zhao, Kai Yu, and Xie Chen. 2024b. F5-TTS: A fairytaler that fakes fluent and faithful speech with flow matching. *CoRR*, abs/2410.06885.

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

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

Avihu Dekel, Slava Shechtman, Raul Fernandez, David Haws, Zvi Kons, and Ron Hoory. 2024. Speak while you think: Streaming speech synthesis during text generation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2024, Seoul, Republic of Korea, April 14-19, 2024*, pages 11931–11935. IEEE.

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. 2024a. Cosyvoice: A scalable multilingual zero-shot text-to-speech synthesizer based on supervised semantic tokens. *CoRR*, abs/2407.05407.

Zhihao Du, Yuxuan Wang, Qian Chen, Xian Shi, Xiang Lv, Tianyu Zhao, Zhifu Gao, Yexin Yang, Changfeng Gao, Hui Wang, et al. 2024b. Cosyvoice 2: Scalable streaming speech synthesis with large language models. *arXiv preprint arXiv:2412.10117*.

Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. 2024. Llama-omni: Seamless speech interaction with large language models. *CoRR*, abs/2409.06666.

Tingwei Guo, Cheng Wen, Dongwei Jiang, Ne Luo, Ruixiong Zhang, Shuaijiang Zhao, Wubo Li, Cheng Gong, Wei Zou, Kun Han, and Xiangang Li. 2021. Didispeech: A large scale mandarin speech corpus. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 6968–6972. IEEE.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460.

Rongjie Huang, Mingze Li, Dongchao Yang, Jiatong Shi, Xuankai Chang, Zhenhui Ye, Yuning Wu, Zhiqing Hong, Jiawei Huang, Jinglin Liu, Yi Ren, Yuexian Zou, Zhou Zhao, and Shinji Watanabe. 2024. Audiogpt: Understanding and generating speech, music, sound, and talking head. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 23802–23804. AAAI Press.

Shengpeng Ji, Yifu Chen, Minghui Fang, Jialong Zuo, Jingyu Lu, Hanting Wang, Ziyue Jiang, Long Zhou, Shujie Liu, Xize Cheng, Xiaoda Yang, Zehan Wang, Qian Yang, Jian Li, Yidi Jiang, Jingzhen He, Yunfei Chu, Jin Xu, and Zhou Zhao. 2024a. Wavchat: A survey of spoken dialogue models. *CoRR*, abs/2411.13577.

Shengpeng Ji, Ziyue Jiang, Xize Cheng, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Ruiqi Li, Ziang Zhang, Xiaoda Yang, Rongjie Huang, Yidi Jiang, Qian Chen, Siqi Zheng, Wen Wang, and Zhou Zhao. 2024b. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. *CoRR*, abs/2408.16532.

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. 2024. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Sungwon Kim, Kevin J. Shih, Rohan Badlani, João Felipe Santos, Evelina Bakhturina, Mikyas Desta, Rafael Valle, Sungroh Yoon, and Bryan Catanzaro. 2023. P-flow: A fast and data-efficient zero-shot TTS through speech prompting. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. 2023. Voicebox: Text-guided multilingual universal speech generation at scale. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR.

Bohan Li, Hankun Wang, Situo Zhang, Yiwei Guo, and Kai Yu. 2024. Fast and high-quality auto-regressive speech synthesis via speculative decoding. *CoRR*, abs/2410.21951.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. 2023. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *18th Annual Conference of the International Speech Communication Association, Interspeech 2017, Stockholm, Sweden, August 20-24, 2017*, pages 498–502. ISCA.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. 2024. Finite scalar quantization: VQ-VAE made simple. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pages 5206–5210. IEEE.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2021. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Kai Shen, Zeqian Ju, Xu Tan, Eric Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian. 2024. Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

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. 2023. Neural codec language models are zero-shot text to speech synthesizers. *CoRR*, abs/2301.02111.

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

Zhifei Xie and Changqiao Wu. 2024. Mini-omni: Language models can hear, talk while thinking in streaming. *CoRR*, abs/2408.16725.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024a. Qwen2 technical report. *CoRR*, abs/2407.10671.

Yifan Yang, Ziyang Ma, Shujie Liu, Jinyu Li, Hui Wang, Lingwei Meng, Haiyang Sun, Yuzhe Liang, Ruiyang Xu, Yuxuan Hu, et al. 2024b. Interleaved speech-text language models are simple streaming text to speech synthesizers. *arXiv preprint arXiv:2412.16102*.

Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. 2019. Libritts: A corpus derived from librispeech for text-to-speech. In *20th Annual Conference of the International Speech Communication Association, Interspeech 2019, Graz, Austria, September 15-19, 2019*, pages 1526–1530. ISCA.

A Details of Baselines

CosyVoice A two-stage large-scale TTS system. The first stage is an autoregressive model similar to VALL-E (Wang et al., 2023), and the second stage is a diffusion model. We use the official code and the 25Hz version of the pre-trained checkpoint[5].

CosyVoice2 Compared to CosyVoice, improvements have been made in the following three areas: 1) The quantizer speech tokenizer has been upgraded to FSQ, further improve the performance

[5]https://www.modelscope.cn/iic/CosyVoice-300M-25Hz.git

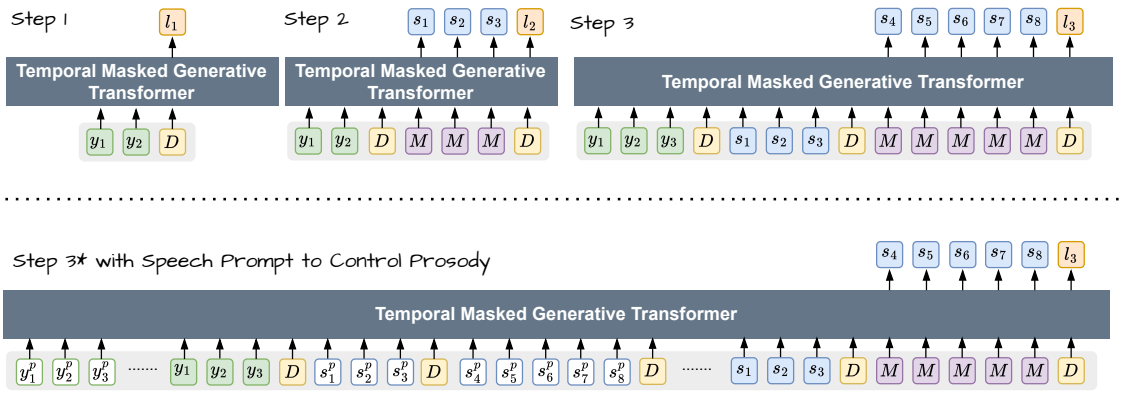

Figure 2: Illustrations of the inference process in two scenarios.The upper part represents the scenario without using speech prompts to control prosody, where in the first step, the duration of the first character needs to be predicted separately; in the subsequent decoding steps, both the current speech token and the duration of the next text token are predicted simultaneously. The lower part shows the illustration of using speech prompts to control prosody, where y^p and s^p denote the text tokens and speech tokens of the speech prompt, respectively.

of the quantization encoder. 2) Interleaved text-speech modeling is employed, allowing for streaming text input. 3) A chunk-aware speech decoder is used for streaming speech generation. We use the official code and the 25Hz version of the pre-trained checkpoint[6].

VALL-E A large-scale TTS system employs both an autoregressive and an auxiliary non-autoregressive model to predict discrete tokens derived from the Encodec (Défossez et al., 2023). We used an open-source checkpoint for inference. As there is currently no open-source streaming speech decoder for Encodec, we assumed 15 frames when calculating the FPL metric for a fair comparison.

MaskGCT (Wang et al., 2024) This is a large-scale, two-stage trained model. In the first stage, the model utilizes text to predict semantic tokens extracted from a speech self-supervised learning (SSL) model. In the second stage, it predicts acoustic tokens based on these semantic tokens. During training, MaskGCT learns to predict masked semantic or acoustic tokens given specific conditions and prompts. During inference, MaskGCT generates speech through multi-step temporally non-sequential masked prediction. Here, we use the official code and pre-trained checkpoint[7].

F5-TTS (Chen et al., 2024b) a fully non-autoregressive text-to-speech system based on flow matching with Diffusion Transformer (DiT). The text input is simply padded with filler tokens to the same length as input speech, and then the denoising is performed for speech generation. F5-TTS does not utilize speech tokens and directly maps text to acoustic features. Here, we use the official code and pre-trained checkpoint[8].

B Details of Latency and Efficiency Evaluation Metrics

The first-package latency (FPL) and real-time factor (RTF) are two import metrics for streaming TTS models. We define d_{LLM} as the average time required by the upstream LLM to generate one text token and d_{TTS} as the the time for the corresponding AR TTS models to forward one step and for the NAR TTS models to perform one sampling. The FPL-L of baseline models and SyncSpeech are as follows,

$$L^{\text{CosyVoice}}_{\text{FPL-L}} = L \cdot d_{\text{LLM}} + 15 \cdot d_{\text{TTS}}, \quad (9)$$

$$L^{\text{VALL-E}}_{\text{FPL-L}} = L \cdot d_{\text{LLM}} + 15 \cdot d_{\text{TTS}}, \quad (10)$$

$$L^{\text{CosyVoice2}}_{\text{FPL-L}} = 5 \cdot d_{\text{LLM}} + 15 \cdot d_{\text{TTS}}, \quad (11)$$

$$L^{\text{MaskGCT}}_{\text{FPL-L}} = L \cdot d_{\text{LLM}} + b \cdot d_{\text{TTS}}, \quad (12)$$

$$L^{\text{F5-TTS}}_{\text{FPL-L}} = L \cdot d_{\text{LLM}} + b \cdot d_{\text{TTS}}, \quad (13)$$

$$L^{\text{SyncSpeech}}_{\text{FPL-L}} = (k+1) \cdot d_{\text{LLM}} + c \cdot d_{\text{TTS}}, \quad (14)$$

where b represents the number of sampling iterations for the NAR model, and c denotes the number of BPE text tokens when the generated speech tokens surpass the decoder's chunk size, typically ranging from 1 to 3. Here, we assume the upstream LLM model is Qwen-7B, and when running on a

[6] https://github.com/FunAudioLLM/CosyVoice
[7] https://github.com/openmmlab/Amphion
[8] https://github.com/SWivid/F5-TTS

Modulation Factor	0.8	0.9	1.0	1.1	1.2	1.3
LibriSpeech	14.3	4.20	3.07	**2.85**	3.22	4.31
SeedTTS test-zh	12.1	3.38	2.38	**2.15**	2.53	3.48

Table 5: Performance comparison with different modulation factors for duration control in terms of WER.

single NVIDIA A800 GPU, we obtain an average token generation time $d_{LLM} = 25ms$. When the first term in FPL-L is omitted, it becomes FPL-A. It is important to note that when calculating above metrics, we did not apply any engineering optimizations, such as KV cache.

We also conducted a brief theoretical analysis of RTF for SyncSpeech. The RTF for SyncSpeech is calculated as follows,

$$L_{RTF} = \frac{(L+1) \cdot d_{\text{TTS}}}{T \cdot F}, \tag{15}$$

where L and T represent the number of BPE tokens and speech tokens, respectively F refers to the frame length of the speech tokens. The time complexity for SyncSpeech to generate an entire sentence can be simplified to $O(L)$, whereas the time complexity for concurrent approaches, such as CosyVoice2 and IST-LM, is $O(T)$. As a result, SyncSpeech can significantly expedite speech generation.

C Duration Control

Since we have implemented duration prediction and control, we can multiply the predicted durations by a modulation factor to adjust speech rate. The results, shown in Table 5, indicate that the robustness of synthesized speech is optimal when the modulation factor is 1.1. However, when the modulation factor is too small or too large, the WER of the synthesized speech by SyncSpeech increases significantly. This is because when we multiply the predicted duration of each text token by a fixed modulation factor of less than 1, SyncSpeech's contextual learning capability causes the subsequent tokens to be spoken increasingly faster, leading to a surge in WER. When the modulation factor is set to 0.8, the average total duration of the synthesized speech is 0.68 times that when the modulation factor is 1. Therefore, more reasonable duration control requires two inference processes: the duration obtained from the first inference is multiplied by a modulation factor during the second inference to control the speech rate.

D Other Strategies for Sequence Construction

We also experimented with other sequence construction strategies. (1) One approach is to separate duration prediction and speech tokens prediction into two steps. This method reduces efficiency by half but achieves better speech robustness, with a WER of around 2.75 on the LibriSpeech *test-clean* dataset. (2) We also tried removing the duration placeholder and using the last speech token of the previous text token to predict the number of speech tokens corresponding to the current text token. However, we found that this sequence construction made the corresponding pre-training less effective than it is now. (3) We also attempted a method similar to ELLA-V (Song et al., 2024), where the corresponding text token is placed before each placeholder. However, we found that this sequence generated speech that was unnatural, with a noticeable disconnection between words.