CM-TTS: Enhancing Real Time Text-to-Speech Synthesis Efficiency through Weighted Samplers and Consistency Models

Anonymous NAACL submission

Abstract

Neural Text-to-Speech (TTS) systems find broad applications in voice assistants, e-learning, and audiobook creation. The pursuit of advanced models, like Diffusion Models (DMs), holds promise for achieving high-fidelity, real-time speech synthesis. Yet, the efficiency of multi-step sampling in Diffusion Models presents challenges. Efforts have been made to integrate GANs with DMs, speeding up inference by approximating denoising distributions, but this introduces issues with model convergence due to adversarial training. To overcome this, we introduce CM-TTS, a novel architecture grounded in consistency models (CM). Drawing inspiration from continuous-time diffusion models, CM-TTS achieves top-quality speech synthesis in fewer steps without adversarial training or pre-trained model dependencies. We further design weighted samplers to incorporate different sampling positions into model training with dynamic probabilities, ensuring unbiased learning throughout the entire training process. We present a real-time mel-spectrogram generation consistency model, validated through comprehensive evaluations. Experimental results underscore CM-TTS's superiority over existing single-step speech synthesis systems, representing a significant advancement in the field[1].

1 Introduction

The advanced Neural Text-to-Speech (TTS) system (Mehrish et al., 2023; Shen et al., 2018; Ren et al., 2020; Liu et al., 2022b) stands out for its exceptional naturalness and efficiency, proving versatile in human-computer interaction and content generation scenarios like real-time voice broadcasting and speech content creation. Comprising three integral modules, the system involves a text encoder collaborating with a speech feature predictor, followed by an acoustic model transforming conditional features into speech features, and a vocoder converting synthesized features into audible speech. This intricate process ensures efficient synthesis of human-like speech.

From a formulation perspective, TTS architecture align with autoregressive (AR) (Oord et al., 2016; Amodei et al., 2016; Wang et al., 2017; Shen et al., 2018) and non-autoregressive (NAR) (Ren et al., 2019; Ren et al., 2020) models. AR frameworks, using RNN models with attention mechanisms, generate spectrograms sequentially, ensuring stable synthesis but suffering from accumulated prediction errors and slower inference speeds. Conversely, NAR models, often based on transformer architecture (Vaswani et al., 2017), employ parallel feed-forward networks for simultaneous mel-spectrogram generation, reducing computational complexity and enabling real-time applications. Various generative models, including Generative Adversarial Networks (GANs) (Kumar et al., 2019; Kong et al., 2020; Kumar et al., 2019; Donahue et al., 2020), Flow (Kim et al., 2018, 2020; Shih et al., 2021; Valle et al., 2020)-based models, and hybrid approaches like Flow with GAN (Cong et al., 2021), contribute to high-fidelity, real-time speech synthesis.

Diffusion Models (DMs) are advanced generative models, excelling in image generation (Ho et al., 2020; Kumar et al., 2019; Song et al., 2020; Rombach et al., 2021), molecular design (You et al., 2018; Gómez-Bombarelli et al., 2016; Thomas et al., 2023), and speech synthesis (Kim et al., 2022a,b; Popov et al., 2021). Employing a forward diffusion process with noise addition and a parameterized reverse iterative denoising process, DMs efficiently capture high-dimensional data distributions. Despite their exceptional performance, the efficiency of their multi-step iterative sampling is hindered by Markov chain limitations. To address these challenges, Ye et al. (2023) propose a TTS

[1] Code and generated samples are available at: `https://anonymous.4open.science/r/CM-TTS-code-25D7/`

architecture based on consistency models (Song et al., 2023). This architecture achieves high audio quality through a single diffusion step, applying a consistency constraint to distill a model from a well-designed diffusion-based teacher model. However, a drawback is the method's reliance on distillation from a teacher model, introducing complexity to synthesis pipeline. Importantly, the proposed TTS architecture is trained on the single-speaker LJSpeech dataset (Ito and Johnson, 2017), limiting its suitability for multi-speaker speech generation. This constraint should be considered in applications where broader speaker diversity is essential.

The integration of GANs into DMs for TTS synthesis (Liu et al., 2022b) has proven effective in minimizing the number of sampling steps during the speech synthesis process. However, this improvement comes at the cost of hindered model convergence due to the additional training required for the discriminator. Some approaches enhance synthesis performance with fewer inference steps by incorporating a shallow mechanism (Liu et al., 2022b). Nonetheless, the introduction of an additional pre-trained model adds complexity to the overall architecture.

We present a novel TTS architecture, CM-TTS, addressing current limitations without relying on a teacher model for distillation. Drawing inspiration from continuous-time diffusion and consistency models, our approach frames speech synthesis as a generative consistency procedure, achieving superior quality in a single step. CM-TTS eliminates the need for adversarial training Liu et al., 2022b or auxiliary pre-trained models (Ye et al., 2023). We enhance model training efficacy with weighted samplers, mitigating sampling biases. CM-TTS maintains traditional diffusion-based TTS benefits and introduces a few-step iterative generation, balancing synthesis efficiency and quality. Experimental results confirm CM-TTS outperforms other single-step speech synthesis systems in quality and efficiency, presenting a significant advancement in TTS architecture. Our key contributions can be summarized as follows:

- We present a consistency model-based architecture for generating a mel-spectrogram designed to meet the demands of real-time speech synthesis with its efficient few-step iterative generation process.
- Moreover, CM-TTS can also synthesize speech in a single step, eliminating the need for adversarial training and pre-trained model dependencies.
- We enhance the model training process by introducing weighted samplers, which adjust weights associated with different sampling points. This refinement mitigates biases introduced during model training due to the inherent randomness of the sampling process.
- Qualitative and quantitative experiments covering 12 metrics demonstrate the effectiveness and efficiency of our model in both fully supervised and zero-shot settings.

2 Related Work

Non-Autoregressive Generative Models Non-autoregressive generative models (NAR) excel in swiftly generating output, making them ideal for real-time applications. Their efficiency, derived from parallelized output generation and lack of dependence on previous results, finds applications in diverse domains like image generation and speech synthesis. GAN networks have been applied in non-autoregressive speech synthesis. Donahue et al. (2020) employ adversarial training and a differentiable alignment scheme for end-to-end speech synthesis. Additionally, Kim et al. (2021) integrate adversarial training into Variational Autoencoders (VAE)(Kingma et al., 2019), enhancing expressive power in speech generation. However, GANs face training instability due to non-overlapping distributions between input and generated data. To address this, CM-TTS incorporates Diffusion Model principles for improved model training and mel-spectrogram generation.

Diffusion Models (DMs) DMs provide robust frameworks for learning complex high-dimensional data distributions through continuous-time diffusion processes. After surpassing GAN (Dhariwal and Nichol, 2021) in image synthesis, DMs have shown promise in speech synthesis. Jeong et al. (2021) utilize a denoising diffusion framework for efficient speech synthesis, transforming noise signals into mel-spectrograms. While DMs excel in data distribution modeling, they may require numerous network function evaluations (NFEs) during sampling. Combining diffusion modeling with traditional generative models enhances efficiency. Diff-GAN (Liu et al., 2022b) adopts an adversarially trained model for expressive denoising distribution approximation. Yang et al. (2023) use

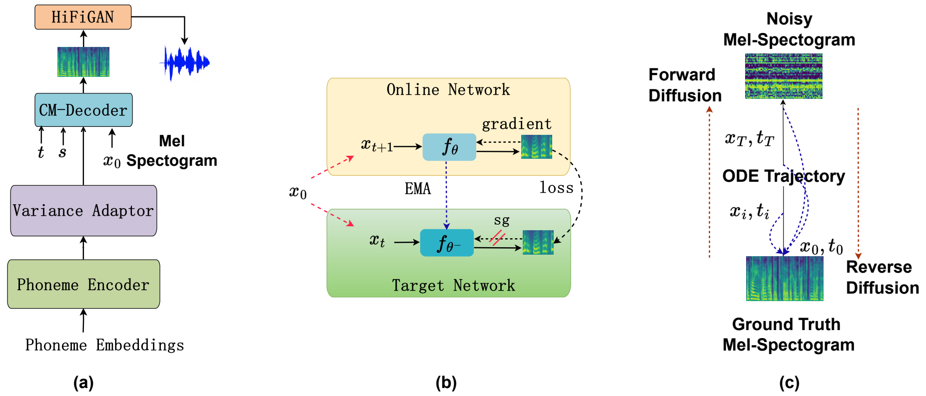

Figure 1: (a) CM-TTS architecture. (b) Decoder training scheme, where f_θ is parameterized to satisfy consistency constrain disucssed in Eq. 4. (c) ODE trajectory during training.

VQ-VAE (Van Den Oord et al., 2017) to transfer text features to mel-spectrograms, reducing diffusion model computational complexity.

3 Background: Consistency Models

The diffusion model is distinguished by a sequential application of Gaussian noise to a target dataset, followed by a subsequent reverse denoising process (Ho et al., 2020). This iterative methodology is designed to generate samples from an initially noisy state, effectively capturing the intrinsic structure of the data. Consider the sequence of noisy data $\{x\}_{t\in[0,T]}$, where $p_0(\mathbf{x}) \equiv p_{\text{data}}(\mathbf{x})$, $p_T(\mathbf{x})$ approximates a Gaussian distribution, and T represents the time constant. The diffusion process can be mathematically expressed as a stochastic process using following stochastic differential equation (SDE).

$$\mathbf{x}_t = \boldsymbol{\mu}(\mathbf{x}_t, t)\mathrm{d}_t + \sigma(t)\mathrm{d}\mathbf{w}_t \quad (1)$$

where $t \in [0, T]$, is the index for forward diffusion time steps. Here, $\boldsymbol{\mu}(.,.)$ and $g(.)$ correspond to the drift and diffusion coefficients, and $\{w_t\}_{t\in[0,T]}$ denotes the standard Brownian motion.

A fundamental characteristic of the SDE lies in its inherent possession of a well-defined reverse process, manifested in the form of a probability flow ODE (Song et al., 2020; Karras et al., 2022). Consequently, the trajectories sampled at time t follow a distribution governed by $p_t(\mathbf{x})$:

$$\mathrm{d}\mathbf{x}_t = \left[\mu(\mathbf{x}_t, t) - \frac{1}{2}\sigma(t)^2 \nabla log\, p_t(\mathbf{x}_t)\right] \mathrm{d}t \quad (2)$$

$\nabla \log p_t(\mathbf{x}_t)$ represents the score function, a key element in score-based generative models (Song et al., 2020). The forward step induces a shift in the sample away from the data distribution, dependent on the noise level. Conversely, a backward step guides the sample closer to the expected data distribution. The probability flow ODE (referenced as Eq. 2) for sample generation utilizes the score function $\nabla \log p_t(\mathbf{x}_t)$. Obtaining the score function involves minimizing the denoising error $||f(x_t, t) - x||^2$ (Karras et al., 2022), where $f(x_t, t)$ is the denoiser function refining the sample x_t at step t.

$$\nabla \log p_t(\mathbf{x}_t) = \frac{(f(x_t, t) - x_t)}{\sigma(t)^2} \quad (3)$$

Probability flow ODEs sampling follows a two-step approach: first, samples are drawn from a noise distribution, and then, a denoising process is applied using a numerical ODE solver, like Euler or Heun (Song et al., 2020, 2023). However, the sampling process from the ODE solver requires a substantial number of iterations, leading to the drawback of slow inference speed. To further accelerate the sampling (Song et al., 2023) proposed consistency property for the diffusion model with the following condition for any time step t and t':

$$\begin{aligned} f(x_t, t) &= f(x_{t'}, t') \\ f(x_t, t) &= x_0 \end{aligned} \quad (4)$$

Given the aforementioned condition, one-step sampling $f(x_T, T)$ becomes viable, as each point along the sampling trajectory of the ODE is directly associated with the origin $p_0(x)$. For a more in-depth discussion, refer to Song et al. (2023). The consistency model is categorized into two types: isolated training or distillation from a pre-trained diffusion-based teacher model. The distillation-based approach relies on the teacher model, adding intricacy to the construction pipeline of the speech synthesis system. In this work, we opt for isolated training of the consistency model.

4 CM-TTS

Diffusion models, known for their high-quality outputs, often struggle with real-time demands in TTS systems due to slow sampling. Existing attempts, like Diff-GAN (Liu et al., 2022b), often rely on additional adversarial training or pre-trained models for efficiency and accuracy. In this section, we discuss the architecture of CM-TTS.

4.1 Model Overview

As shown in Figure 1, the CM-TTS consists of four key components: 1) Phoneme encoder for processing text; 2) Variance adaptor predicting pitch, duration, and energy features; 3) the CM-Decoder for mel-spectrograms generation; and 4) Vocoder, using HiFi-GAN (Kong et al., 2020), to convert mel-spectrograms into time-domain waveforms.

4.2 Phoneme Encoder and Variance Adaptor

The phoneme encoder, incorporating multiple Transformer blocks (Ren et al., 2019, 2020), adapts the feed-forward network into a convolutional network to effectively capture local dependencies within the phoneme sequence. The variance adaptor aligns with FastSpeech2's design, including pitch, energy, and duration prediction modules, each following a consistent model structure with several convolutional blocks. To facilitate training, ground-truth duration, energy, and pitch serve as learning targets, computed using Mean Squared Error (MSE) loss ($\mathcal{L}_{duration}$, $\mathcal{L}_{pitch}$, and $\mathcal{L}_{energy}$). In the training phase, the ground-truth duration expands the hidden sequence from the phoneme encoder to yield a frame-level hidden sequence, followed by the integration of ground-truth pitch information. During inference, the corresponding predicted duration and pitch values are utilized.

4.3 Consistency Models

To establish the divisions within the time horizon $[\epsilon, T_{max}]$, the interval is segmented into $N-1$ sub-intervals, delineated by boundaries $t_1 = \epsilon < t_2 < \ldots < t_N = T_{max}$. As recommended by Karras et al. (2022) to mitigate numerical instability, a small positive value is set for ϵ. Similar to Karras et al. (2022), in this work we use $T_{max} = 80$ and $\epsilon = 0.002$. The mel-spectrogram is denoted as $\mathbf{x}$, where $\mathbf{x}_0$ signifies the initial mel-spectrogram devoid of any added noise.

The fundamental concept introduced in Song et al. (2023) to formulate the consistency model f_θ involves learning a consistency function from data by enforcing the self-consistency property defined in Eq. 4. In order to ensure $f_\theta(x_0, \epsilon) = \mathbf{x_0}$, the consistency model f_θ is parameterized as follows:

$$f_\theta(\mathbf{x}, t) = c_{skip}(t)\mathbf{x} + c_{out}(t)F_\theta(\mathbf{x}, t) \quad (5)$$

Here, c_{skip} and c_{out} are differentiable functions with $c_{skip}(\epsilon) = 1$ and $c_{out}(\epsilon) = 0$, respectively. The term $F_\theta(\mathbf{x}, t)$ represents a neural network. To enforce the self-consistency property, a target model θ^- is concurrently maintained with the online network θ. The weight of the target network θ^- is updated using the exponential moving average (EMA) of parameters θ intended for learning Grill et al., 2020, specifically,

$$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu\boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta}). \quad (6)$$

The consistency loss $\mathcal{L}_{CT}^N(\boldsymbol{\theta}, \boldsymbol{\theta}^-)$ is defined as:

$$\sum_{n \geq 1} \mathbb{E}[\lambda(t_n) d(\boldsymbol{f_\theta}(\mathbf{x}_0 + t_{n+1}\mathbf{z}), \boldsymbol{f_{\theta^-}}(\mathbf{x}_0 + t_n\mathbf{z}))] \quad (7)$$

Here, $d(.,.)$ denotes a chosen metric function for measuring the distance between two samples, such as the squared l_2 distance $d(x, y) = ||x-y||_2^2$. The values $\mathbf{x}_{t+1}$ and $\mathbf{x}_t$ are obtained by sampling two points along the trajectory of the probability flow ODE using a forward diffusion process, starting with mel-spectrograms of the training data $\mathbf{x}_0 \sim \mathcal{D}(dataset)$:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{x}_0 + t_{n+1}\mathbf{z} \\ \mathbf{x}_t &= \mathbf{x}_0 + t_n\mathbf{z} \end{aligned} \quad (8)$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and step t_n is obtained as follows:

$$t_n = \left[T_{\max}{}^{\frac{1}{p}} + \frac{n-1}{N-1}\left(\epsilon^{\frac{1}{p}} - T_{\max}{}^{\frac{1}{p}}\right) \right]^p \quad (9)$$

where N denotes the sub-intervals, n is sampled from the interval $[1, N-1]$ using different weighted sampling strategies (Section 4.3.2), and value of $p = 7$ following Karras et al. (2022).

Similar to DiffGAN-TTS (Liu et al., 2022b), the architecture of $F_\theta(\mathbf{x}, t)$ in CM-TTS embraces a non-causal WaveNet structure (Oord et al., 2016). The difference lies in their approach to sampling t. In CM-TTS, two decoders, denoted as f_θ and f_θ^-, with identical architectures serve as the online and target networks, respectively. The diffusion process in CM-TTS is characterized by Eq. 8, whereas DiffGAN-TTS employs the creation of a parameter-free T-step Markov chain (Liu et al., 2022b).

4.3.1 Training and Loss

Following the training procedure established in Grill et al. (2020), we designate the two decoders shown in Figure 1 as the online $\boldsymbol{f_\theta}$ and target $\boldsymbol{f_{\theta^-}}$. Leveraging the states $\mathbf{x}_{t+1}$ and $\mathbf{x}_t$, we derive corresponding mel predictions, expressed as $\boldsymbol{f_\theta}(\mathbf{x}_0 + t_{n+1}\mathbf{z})$ and $\boldsymbol{f_{\theta^-}}(\mathbf{x}_0 + t_n\mathbf{z})$, through the online and target networks, respectively. The online component undergoes gradient updates via the computation of MSE loss between these prediction pairs. Simultaneously, the gradients of the target network are updated through EMA, as discussed in section 4.3.

During training, the online and target networks engage in an iterative interplay, facilitating mutual learning and crucially contributing to model stability. The mel reconstruction loss $\mathcal{L}_{mel}$ is determined by computing the Mean Absolute Error (MAE) between the ground truth and the generated mel-spectrogram. Finally, $\mathcal{L}_{recon}$ can be expressed as follows:

$$\begin{aligned}\mathcal{L}_{recon} =& \mathcal{L}_{mel}(\mathbf{x_0}, \mathbf{x_0}') + \lambda_d \mathcal{L}_{duration}(\mathbf{d}, \hat{\mathbf{d}}) + \\ & \lambda_p \mathcal{L}_{pitch}(\mathbf{p}, \hat{\mathbf{p}}) + \lambda_e \mathcal{L}_{energy}(\mathbf{e}, \hat{\mathbf{e}})\end{aligned} \tag{10}$$

Here, $\mathbf{d}$, $\mathbf{p}$, and $\mathbf{e}$ denote the ground truth duration, pitch, and energy, respectively, while $\hat{\mathbf{d}}$, $\hat{\mathbf{p}}$, and $\hat{\mathbf{e}}$ represent the predicted values. The weights assigned to each loss component are denoted by λ_d, λ_p, and λ_e. For this study, we maintain uniform loss weights set at 0.1. The optimization objective for training the CM-TTS involves minimizing the following composite loss function.

$$\mathcal{L}_{CM-TTS} = \mathcal{L}^N_{CT}(\boldsymbol{\theta}, \boldsymbol{\theta}^-) + \mathcal{L}_{recon} \tag{11}$$

During single-step generation in inference, a single forward pass through f_θ is undertaken. Conversely, multi-step generation is achievable by alternating denoising and noise injection steps, enhancing the quality, as depicted in Figure 2.

4.3.2 Weighted Sampler

The training procedure relies on sampling the time step t_n as defined in Eq. 9. Consequently, to investigate the impact of sampling various positions (t_n) along the ODE trajectory, we employ three distinct

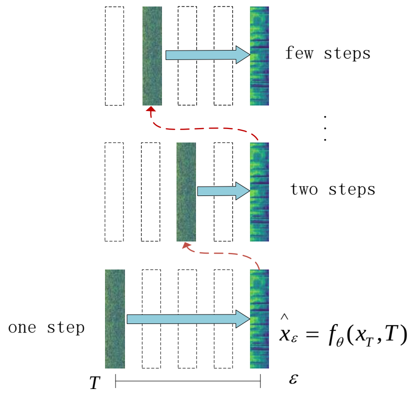

Figure 2: Single-step and multi-step inference utilizing the CM-TTS. For multi-step generation, process of alternating denoising and noise injection steps is executed iteratively until the desired number of steps is achieved.

weighted sampling strategies. Each strategy governs the probabilities associated with selecting the step t_n throughout the training, thereby allowing for an in-depth examination of the effects arising from different sampling positions.

In the forward diffusion process during training, the variable n denotes the index of a sampling point, where $n \in [1, N-1]$, and is used in Eq. 9 for computing t_n. We introduce c_n as the weight assigned to the current index n by the sampler, s_n the probability of selecting index n is given by $s_n = \frac{c_n}{\sum_{i=1}^{N-1} c_n}$. The three sampler designs are outlined as follows:

Uniform Sampler This sampler serves as a baseline for validating other methods, where each point is chosen with equal probability ($c_n = 1$).

Linear Sampler The sampling weight varies linearly with the position of the sampling point, defined as $c_n = \alpha \cdot n$, with $\alpha = 1$ in all experiments.

Loss-Based Second-Moment (LSM) Sampler Following Nichol and Dhariwal, 2021, we use the LSM sampler to assign weights to sampling points. The formulation is given by $c_n = (1 - \phi)\frac{\sum_{j=1}^{H} L(t,j)}{\sum_{i=1}^{N-1}\sum_{j=1}^{H} L(i,j)} + \phi$. Here, L $\in \mathbb{R}^{N-1 \times H}$ represents a matrix recording historical losses for all sampling points, and H denotes the number of historical losses stored for each point (set to 10 in our experiments). The small quantity ϕ serves as a balancing factor, adjusting c_n. This design modulates the probability of current sampling based on historical losses, thereby prioritizing points with greater significance for model training.

Model	FFE↓	S.Cos↑	mfccFID↓	melFID↓	mfccRecall↑	MCD↓	SSIM↑	mfccCOS↑	F0↓	RTF↓	WER↓	MOS↑
Reference	-	-	-	1.46e-11	0.6428	-	-	-	-	-	0.0300	-
Reference (voc.)	0.1427	0.9424	31.98	3.48	0.5644	4.57	0.8132	0.8457	89.21		0.0412	4.3576(±0.1206)
FastSpeech2(900K)	0.3562	0.8318	548.50	43.79	0.3530	8.23	0.3441	0.5867	151.59	**0.02**	0.0649	-
VITS	0.3509	0.8154	428.91	15.40	**0.5141**	6.96	0.4411	0.7418	117.99	0.23	**0.0451**	-
DiffSpeech	**0.3343**	0.7400	76.01	11.55	0.5096	7.25	0.3421	0.6445	119.98	9.19	0.5708	2.3851(±0.2603)
DiffGAN-TTS(T=1)	0.3489	0.8284	97.65	20.01	0.3560	5.98	0.4589	0.7537	118.47	**0.02**	0.0809	3.6744(±0.1391)
DiffGAN-TTS(T=2)	0.3411	0.8333	38.64	7.79	0.3974	5.94	0.4610	0.7581	**117.19**	0.03	0.0827	-
DiffGAN-TTS(T=4)	0.3465	0.8358	**37.11**	**6.58**	0.3662	5.94	0.4614	0.7571	120.10	0.04	0.0751	-
CM-TTS(T=1)	0.3387	0.8396	39.17	7.58	0.3946	5.91	0.4772	0.7599	119.29	**0.02**	0.0688	**3.8166(±0.1174)**
CM-TTS(T=2)	0.3383	**0.8401**	38.79	7.34	0.3972	**5.90**	0.4780	0.7598	120.01	0.03	0.0680	-
CM-TTS(T=4)	0.3385	0.8399	38.78	7.34	0.3976	**5.90**	**0.4783**	**0.7599**	119.23	0.07	0.0696	-

Table 1: Objective and subject evaluation: Comparison with baselines on VCTK dataset.

5 Experiments

5.1 Data and Preprocessing

Our experiments are based on CSTR VCTK (Veaux et al., 2013), LJSpeech (Ito and Johnson, 2017), and LibriSpeech (Panayotov et al., 2015) datasets. CSTR VCTK Corpus includes speech data from 110 English speakers, while LJSpeech features $13,100$ short audio clips, totaling around 24 hours. For zero-shot experiments, the LibriTTS corpus is used for model training. All samples are resampled to $22,050$ Hz. The test set consists of 512 randomly selected speech samples, and we assess the model's performance with various objective and subjective metrics. In pre-processing, mel-spectrograms has 80 frequency bins, generated with a window size of 25 ms and a frameshift of 10 ms. Ground truth pitch, duration, and energy are computed using the PyWorld toolkit[2].

5.2 Baseline Models

Reference and Reference (Voc.) Reference denotes the ground truth. The process of obtaining the Reference (voc.) involves transforming the original reference speech into mel-spectrograms, followed by the subsequent reconstruction of speech using HiFi-GAN (Kong et al., 2020)

Fastspeech2 NAR transformer architecture (Ren et al., 2019), generating speech in parallel for faster inference. Utilizing mel-spectrogram prediction, duration prediction, and variance modeling

VITS The VITS model (Kim et al., 2021) combines variational inference, normalizing flows, and adversarial training. It introduces a stochastic duration predictor to synthesize diverse rhythms, capturing natural variability in speech.

DiffSpeech & DiffGAN-TTS DiffSpeech (Liu et al., 2022a) and DiffGAN-TTS (Liu et al., 2022b) are diffusion-based TTS architecture. Both architectures focus on addressing real-time speech synthesis in TTS systems, which diffusion models often struggle with due to slow sampling. DiffGAN-TTS addresses the challenge by incorporating additional adversarial training.

5.3 Model Configuration

The transformer encoder and the variance adaptor of the CM-TTS adopt identical network structures and hyper-parameters as those in FastSpeech2. The former is composed of 4 feed-forward transformer (FFT) blocks, where the kernel size and filter size are set to 256, 2, 9, and 1024, respectively. The latter continues to consist of a duration predictor, a pitch predictor, and an energy predictor. The CM-Decoder adopts a structure similar to WaveNet, employing 1D convolution to process the noisy mel spectrogram, followed by activation through the ReLU. Speaker-IDs are activated through WaveNet residual blocks and transformed into embedding vectors. The diffusion step t is encoded using sinusoidal tpositional encoding as in Song et al. (2023). The mel decoder comprises 4 FFT blocks. The number of parameters in our model is 28.6 million.

5.4 Training and Inference

We conduct all experiments using a single NVIDIA Tesla V100 GPU with 32 GB. The average runtime of training under VCTK, LJSpeech, and LibriSpeech is 34.2 hours, 42.8 hours, and 45.6 hours, respectively. The training employs the multi-speaker dataset VCTK, and speaker embeddings, computed using Li et al. (2017), have a dimension

[2] https://github.com/JeremyCCHsu/Python-Wrapper-for-World-Vocoder

Simplers	FFE↓	S.Cos↑	mfccFID↓	melFID↓	mfccRecall↑	MCD↓	SSIM↑	mfccCOS↑	F0↓	WER↓
Uniform	**0.3351**	0.8333	56.31	10.08	0.4015	5.98	0.4396	0.7456	118.87	0.0872
Linear(↗)	0.3367	0.8356	63.11	11.35	**0.4297**	6.03	0.4549	0.7485	**118.74**	0.0822
Linear(↘)	0.3403	0.8315	54.58	11.05	0.4102	6.02	0.4694	0.7454	120.32	0.0861
LSM	0.3387	**0.8396**	**39.17**	**7.58**	0.3946	**5.91**	**0.4772**	**0.7599**	119.29	**0.0688**

Table 2: Performance under different sampler.

Loss	FFE↓	S.Cos↑	mfccFID↓	melFID↓	mfccRecall↑	MCD↓	SSIM↑	mfccCOS↑	F0↓	WER↓
l_1	0.3387	**0.8396**	39.17	7.5772	0.3946	5.9093	**0.4772**	0.7599	119.29	**0.0688**
$l_1^{w/o\ padding}$	0.3374	0.8379	43.28	10.16	0.3961	**5.7815**	0.4593	**0.7606**	**117.45**	0.0741
l_2	0.3368	0.8320	**38.73**	**8.49**	**0.4062**	5.8836	0.4505	0.7573	120.05	0.0751
$l_2^{w/o\ padding}$	**0.3366**	0.8294	48.09	12.14	0.3841	5.8355	0.4613	0.7585	118.52	0.0756

Table 3: Effect on performance due to padding under different loss. l_1 and l_2 represent the loss with padding, whereas $l_1^{w/o\ padding}$ and $l_2^{w/o\ padding}$ represent loss calculation without considering padding.

of 512. In our experiments, we randomly select 512 samples for testing, utilizing the remaining for training. The batch size during training is 32. We train all the models for $300K$ steps. Following the same learning rate schedule in DiffGAN-TTS, we use an exponential learning rate decay with rate 0.999 for training and the initial learning rate is $10e^{-4}$. In addition, Song et al. (2023) find that periodically adjusting sub-interval N and decay constant μ in Eq 6 during training, following schedule functions $N(k)$ and $\mu(k)$ based on training steps k, improves performance. In this paper, we adopts the same strategy as outlined in Song et al. (2023).

5.5 Evaluation Metrics

Objective Metrics In our rigorous evaluation of speech synthesis, we leverage a diverse array of objective metrics to holistically appraise the synthesized output's quality and efficiency. This multifaceted set of metrics encompasses the F0 Frame Error (FFE) for evaluating fundamental frequency tracking, Speaker Cosine Similarity (SCS) to gauge the similarity of speaker embeddings, and Fréchet Inception Distance (FID) based on Mel-Frequency Cepstral Coefficients (mfccFID) for a comprehensive assessment of spectrogram divergence. Furthermore, we incorporate metrics such as mfccRecall, MCD24, SSIM, mfccCOS, Word Error Rate (WER), and F0 to provide nuanced insights into various dimensions of synthesis performance. Detailed descriptions in given in Appendix D.

Subjective Metrics The Mean Opinion Score (MOS), as introduced in Chu and Peng (2006), serves as a pivotal metric for evaluating the perceived quality of the synthesized audio. In our evaluation, we involve presenting a carefully curated test set to 10 listeners and soliciting their subjective opinions. Participants are then tasked with rating the quality of the synthesized audio on a scale ranging from 1 to 5.

6 Results and Discussion

Comparison with Baselines The outcomes of our experiments, comparing the proposed model against various baseline models, are presented in Table 1. Notably, our model (CM-TTS) demonstrates a significant performance advantage over Fastspeech2, VITS, and DIffSpeech in objective evaluations. The results also affirm the efficacy of CM-TTS when pitted against DiffGAN-TTS; the proposed TTS architecture outperforms DiffGAN-TSS across the majority of metrics. Particularly noteworthy is CM-TTS's superior performance in single-step generation ($T = 1$), where it outperforms DiffGAN-TSS across all objective metrics, with only a minimal gap observed in f_0. Furthermore, when evaluating speaker similarity (S.Cos), CM-TTS achieves the highest S.Cos score of 0.8401, underscoring its effectiveness in multi-speaker speech generation.

We conduct a subjective evaluation to compare the naturalness and quality of synthesized speech against a reference sample. The MOS scores from the listening test, showcased in Table 1, reveal CM-TTS achieving an impressive MOS of 3.816. This marks a substantial advancement over DiffSpeech and a significant outperformance of DiffGAN-TTS in overall performance.

Few-Step Speech Generation In evaluating single-step synthesis performance, we can observe

Model	FFE↓	S.Cos↑	mfccFID↓	melFID↓	mfccRecall↑	MCD↓	SSIM↑	mfccCOS↑	F0-RMSE↓	WER↓
DiffGAN-TTS(T=1)	0.4134	0.6874	283.77	44.47	0.1901	9.00	0.2712	0.5351	135.79	0.0488
DiffGAN-TTS(T=2)	**0.4107**	0.6908	254.84	36.44	0.1950	9.05	0.2764	0.5356	133.96	**0.0465**
DiffGAN-TTS(T=4)	0.4112	0.6915	256.75	36.50	0.2023	**9.05**	0.2709	0.5343	135.56	0.0501
CM-TTS(T=1)	0.4219	0.7108	157.91	26.75	0.2072	9.16	0.2829	0.5548	131.27	0.0536
CM-TTS(T=2)	0.4225	0.7107	155.91	**26.34**	**0.2135**	9.16	0.2836	**0.5557**	**131.13**	0.0536
CM-TTS(T=4)	0.4226	**0.7110**	**155.56**	26.36	0.2089	9.18	**0.2845**	0.5553	132.04	0.0530

Table 4: The zero-shot performance of CM-TTS and DiffGAN-TTS on VCTK for synthesis steps 1, 2, and 4.

Prosody	Model	Mean↓	Std↓	Skew↓	Kurt↓
Pitch	DiffGAN-TTS(T=1)	12.95	22.19	**3.33**	**15.75**
	CM-TTS(T=1)	**12.36**	**21.53**	3.40	16.37
Duration	DiffGAN-TTS(T=1)	1.47	0.56	1.52	4.84
	CM-TTS(T=1)	**1.36**	**0.54**	**1.43**	**4.83**

Table 5: The prosody similarity between synthesized and reference speech of pitch and duration.

from Table 1 CM-TTS that consistently surpasses DiffGAN-TTS across all metrics, with a marginal difference observed in the F0-RMSE. When extending to a multi-step synthesis scenario ($T = 4$), CM-TTS outperforms DiffGAN-TTS in all metrics, except for melFID (7.34 compared to 6.58). These findings emphasize that, beyond its impressive single-step synthesis capabilities, our proposed method demonstrates robust synthesis proficiency in scenarios involving multiple iterative steps.

Length robustness during training Incorporating padding in the model's loss calculation is common, especially for variable-length sequences in training. The goal is to guide the model in capturing meaningful representations from both genuine input data and padded segments. TTS models face challenges in handling diverse input texts during training. To assess the model's resilience and investigate the impact of padding, we conduct experiments comparing the inclusion or exclusion of the padding portion in the loss calculation ($\mathcal{L}_{mel}$). Results in Table 3 demonstrate that including the padding portion improves the overall performance of the model. We experiment with both l_1-norm and l_2-norm while computing $\mathcal{L}_{mel}$ in Eq. 10.

Ablation to Weighted Sampler In this subsection, we conduct experiments to explore the impact of different sampling methods, as discussed in Section 4.3.2, on the performance of the CM-TTS. The results presented in Table 2 reveal a significant enhancement in the CM-TTS's performance across various metrics when the LSM sampler is employed. Notably, S.Cos exhibits an improvement to 0.8396, indicating enhanced speaker similarity with the use of the LSM sampler. Furthermore, as illustrated in the Figure 4, we observe there is no significant impact on the convergence of CM-TTS when utilizing a different sampler.

Generalization To Unseen Speakers To assess how well CM-TTS performs with speakers it hasn't seen before, we train the model on the LibriTTS (Zen et al., 2019)(train-clean-100) dataset, which mainly contains longer input texts. To test its zero-shot performance, we randomly selected 512 speech samples from VCTK and LJSpeech datasets. In Table 4, we compare DiffGAN and CM-TTS on VCTK for different generation steps ($T = 1, 2, \&4$). Additionally, we use an alignment tool to get phoneme-level duration and pitch and compute the prosody similarity between the synthesized and the reference speech. The results are displayed in Table 5. Interestingly, in multi-speaker scenarios, CM-TTS consistently outperforms the baseline DiffGAN-TTS. However, in single-speaker scenarios (see Table 7), DiffGAN-TTS outperforms CM-TTS. For more details on zero-shot performance on LJSpeech, please refer to the Appendix A.

Conclusion

In this work, we introduced CM-TTS, a novel architecture focused on real-time speech synthesis. CM-TTS leverages consistency models, steering away from the complexities associated with adversarial training and pre-trained model dependencies. Through comprehensive evaluations, our results underscore the effectiveness of CM-TTS over established single-step speech synthesis architectures. This marks a significant improvement in promising avenues for applications ranging from voice assistant systems to e-learning platforms and audiobook generation. The future work entails advancing training through the utilization of diverse datasets, thereby enhancing the CM-TTS to generalize better across previously unseen speakers.

Limitations

In terms of the model, the presented CM-TTS framework primarily optimizes and enhances the training mechanism, aiming to facilitate comparative experiments. However, the inherent structure of the network, including aspects like the number of layers or residual modules, hasn't been extensively explored for this paper. Future endeavors could delve into lightweight studies focusing on the network itself, potentially enhancing the overall performance of CM-TTS.

Regarding the task, the experiments conducted in this paper exclusively center around TTS tasks, without extending to other related tasks such as sound generation. Future work could encompass experimental validation across a broader spectrum of tasks, providing a more comprehensive assessment.

Ethics Statement

Given the ability of CM-TTS to synthesize speech while preserving the speaker's identity, potential risks of misuse, such as deceiving voice recognition systems or impersonating specific individuals, may arise. In our experiments, we operate under the assumption that users willingly agree to be the designated speaker for speech synthesis. In the event of the model's application to unknown speakers in real-world scenarios, it is imperative to establish a protocol ensuring explicit consent from speakers for the utilization of their voices. Additionally, implementing a synthetic speech detection model is recommended to mitigate the potential for misuse.

References

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR.

Min Chu and Hu Peng. 2006. Objective measure for estimating mean opinion score of synthesized speech. US Patent 7,024,362.

Wei Chu and Abeer Alwan. Reducing f0 frame error of f0 tracking algorithms under noisy conditions with an unvoiced/voiced classification frontend. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3969–3972.

Jian Cong, Shan Yang, Lei Xie, and Dan Su. 2021. Glow-wavegan: Learning speech representations from gan-based variational auto-encoder for high fidelity flow-based speech synthesis. *arXiv preprint arXiv:2106.10831*.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794.

Jeff Donahue, Sander Dieleman, Mikołaj Bińkowski, Erich Elsen, and Karen Simonyan. 2020. End-to-end adversarial text-to-speech. *arXiv preprint arXiv:2006.03575*.

Rafael Gómez-Bombarelli, David Duvenaud, José Miguel Hernández-Lobato, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. 2016. Automatic chemical design using a data-driven continuous representation of molecules. *arXiv preprint 732 arXiv:1610.02415*.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Keith Ito and Linda Johnson. 2017. The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`.

Myeonghun Jeong, Hyeongju Kim, Sung Jun Cheon, Byoung Jin Choi, and Nam Soo Kim. 2021. Diff-tts: A denoising diffusion model for text-to-speech. *arXiv preprint arXiv:2104.01409*.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577.

Heeseung Kim, Sungwon Kim, and Sungroh Yoon. 2022a. Guided-TTS: A diffusion model for text-to-speech via classifier guidance. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11119–11133. PMLR.

Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. 2020. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *Advances in Neural Information Processing Systems*, 33:8067–8077.

Jaehyeon Kim, Jungil Kong, and Juhee Son. 2021. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR.

Sungwon Kim, Heeseung Kim, and Sungroh Yoon. 2022b. Guided-tts 2: A diffusion model for high-quality adaptive text-to-speech with untranscribed data. *arXiv preprint arXiv:2205.15370*.

Sungwon Kim, Sang-gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. 2018. Flowavenet: A generative flow for raw audio. *arXiv preprint arXiv:1811.02155*.

Diederik P Kingma, Max Welling, et al. 2019. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33:17022–17033.

Kundan Kumar, Rithesh Kumar, Thibault De Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre De Brebisson, Yoshua Bengio, and Aaron C Courville. 2019. Melgan: Generative adversarial networks for conditional waveform synthesis. *Advances in neural information processing systems*, 32.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2019. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32.

Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. 2017. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*.

Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen, and Zhou Zhao. 2022a. Diffsinger: Singing voice synthesis via shallow diffusion mechanism. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 11020–11028.

Songxiang Liu, Dan Su, and Dong Yu. 2022b. Diffgan-tts: High-fidelity and efficient text-to-speech with denoising diffusion gans. *arXiv preprint arXiv:2201.11972*.

Ambuj Mehrish, Navonil Majumder, Rishabh Bharadwaj, Rada Mihalcea, and Soujanya Poria. 2023. A review of deep learning techniques for speech processing. *Information Fusion*, page 101869.

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

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE.

Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2018. Deep voice 3: 2000-speaker neural text-to-speech. *proc. ICLR*, pages 214–217.

Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. 2021. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2020. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. *Advances in neural information processing systems*, 32.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-resolution image synthesis with latent diffusion models.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4779–4783. IEEE.

Kevin J Shih, Rafael Valle, Rohan Badlani, Adrian Lancucki, Wei Ping, and Bryan Catanzaro. 2021. Rad-tts: Parallel flow-based tts with robust alignment learning and diverse synthesis. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency models.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Morgan Thomas, Andreas Bender, and Chris de Graaf. 2023. Integrating structure-based approaches in generative molecular design. *Current Opinion in Structural Biology*, 79:102559.

Rafael Valle, Kevin Shih, Ryan Prenger, and Bryan Catanzaro. 2020. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. *arXiv preprint arXiv:2005.05957*.

Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Christophe Veaux, Junichi Yamagishi, and Simon King. 2013. The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. In *2013 International Conference Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–4.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. 2017. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*.

Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. 2023. Diffsound: Discrete diffusion model for text-to-sound generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Zhen Ye, Wei Xue, Xu Tan, Jie Chen, Qifeng Liu, and Yike Guo. 2023. Comospeech: One-step speech and singing voice synthesis via consistency model. *arXiv preprint arXiv:2305.06908*.

Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay S. Pande, and Jure Leskovec. 2018. Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6412–6422.

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. *arXiv preprint arXiv:1904.02882*.

A Experiments on LJSpeech

Our CM-TTS model, trained for 300K steps on the LJSpeech single speaker dataset, exhibits impressive performance in 1, 2, and 4-step synthesis, detailed in Table 6. Compared to DiffGAN-TTS, CM-TTS achieves optimal scores (S.Cos: 0.9010, melFID: 2.97) across varied training and synthesis scenarios, highlighting its effectiveness in single-speaker scenarios.

In a detailed performance comparison between CM-TTS and DiffGAN-TTS, we analyze the convergence of these models across various training steps, as illustrated in Figure 3. Initially, both models exhibit relatively consistent convergence. However, as the training steps increase, CM-TTS demonstrates significantly better convergence, indicating superior fitting performance when compared to DiffGAN-TTS.

Model	FFE$\downarrow$	S.Cos$\uparrow$	mfccFID$\downarrow$	melFID$\downarrow$	mfccRecall$\uparrow$	MCD$\downarrow$	SSIM$\uparrow$	mfccCOS$\uparrow$	F0$\downarrow$	RTF$\downarrow$	WER$\downarrow$
Reference	-	-	-	4.49e-11	0.7013	-	-	-	-	-	0.0808
Reference (voc.)	0.0891	0.9861	0.8323	0.11	0.6768	3.1995	0.9310	0.9589	67.61	-	0.0712
FastSpeech2	0.4974	0.8989	**21.91**	2.86	0.2986	6.5300	**0.6689**	0.7921	143.82	-	0.0823
DiffSpeech	0.4885	0.8742	27.45	4.38	0.2775	7.0267	0.5562	0.7332	132.59	-	0.1171
CoMoSpeech	0.4900	0.8666	369.96	17.81	0.2865	7.7416	0.5660	0.7275	144.23	-	0.0823
VITS	0.4820	0.8811	264.89	17.82	0.3192	7.0700	0.6248	0.7776	**123.24**	-	0.0847
DiffGAN-TTS(T=1)	0.4872	0.8959	27.22	3.70	0.2527	**6.0798**	0.6530	0.7991	136.80	-	0.0697
DiffGAN-TTS(T=2)	**0.4818**	0.8995	25.03	3.09	**0.2463**	6.1205	0.6547	**0.7995**	133.71	-	0.0749
DiffGAN-TTS(T=4)	0.4856	0.8969	23.48	3.15	0.2590	6.0856	0.6539	0.7991	136.50	-	**0.0693**
CM-TTS(T=1)	0.4860	0.9009	24.52	2.97	0.2586	6.0978	0.6558	0.7989	135.58	-	0.0727
CM-TTS(T=2)	0.4861	0.9010	24.70	2.97	0.2597	6.0978	0.6553	0.7990	136.02	-	0.0725
CM-TTS(T=4)	04861	**0.9010**	24.72	**2.97**	0.2591	6.0965	0.6553	0.7989	136.26	-	0.0725

Table 6: Objective evaluation: Comparison with baselines on LJSpeech dataset.

B Zero-shot Performance on LJSpeech

We trained CM-TTS on the LibriTTS' train-clean-100 dataset and evaluated LJSpeech's zero-shot performance. The results are presented in Table 8 and Table 7. It is evident that CM-TTS consistently outperforms in most metrics.

LJSpeech	Pitch				Duration			
	Mean$\downarrow$	Std$\downarrow$	Skew$\downarrow$	Kurt$\downarrow$	Mean$\downarrow$	Std$\downarrow$	Skew$\downarrow$	Kurt$\downarrow$
DiffGAN-TTS(T=1)	20.56	32.11	**3.45**	**18.34**	**0.93**	**0.65**	**0.75**	4.39
CM-TTS(1)	**18.34**	**29.99**	3.73	21.35	1.08	0.92	1.70	**4.38**

Table 7: The prosody similarity between synthesized and prompt speech in terms of the difference in mean (Mean), standard variation (Std), skewness (Skew), and kurtosis (Kurt) of pitch and duration on LJSpeech. **Best** numbers are highlighted in each column.

Model	FFE$\downarrow$	S.Cos$\uparrow$	mfccFID$\downarrow$	melFID$\downarrow$	mfccRecall$\uparrow$	MCD$\downarrow$	SSIM$\uparrow$	mfccCOS$\uparrow$	F0-RMSE$\downarrow$	WER$\downarrow$
DiffGAN-TTS(T=1)	0.5164	0.7278	162.90	21.83	0.2523	8.3634	**0.4491**	0.6513	170.26	0.1118
DiffGAN-TTS(T=2)	0.5151	**0.7339**	93.96	13.50	0.2772	8.2702	0.4479	0.6561	164.80	0.1146
DiffGAN-TTS(T=4)	0.5153	0.7315	95.08	13.38	0.2859	**8.2692**	0.4447	0.6547	161.62	**0.1094**
CM-TTS(T=1)	**0.4934**	0.7271	**86.90**	10.84	**0.4013**	8.6616	0.4433	0.6540	148.04	0.1194
CM-TTS(T=2)	0.5060	0.7290	105.34	9.12	0.3082	8.5547	0.4458	0.6587	148.83	0.1190
CM-TTS(T=4)	0.5081	0.7301	102.35	**8.91**	0.2876	8.6102	0.4392	**0.6596**	**147.38**	0.1264

Table 8: The zero-shot performance of CM-TTS and DiffGAN-TTS on LJSpeech. T equal to 1, 2 & 4 represents steps for synthesis. **Best** numbers are highlighted in each column.

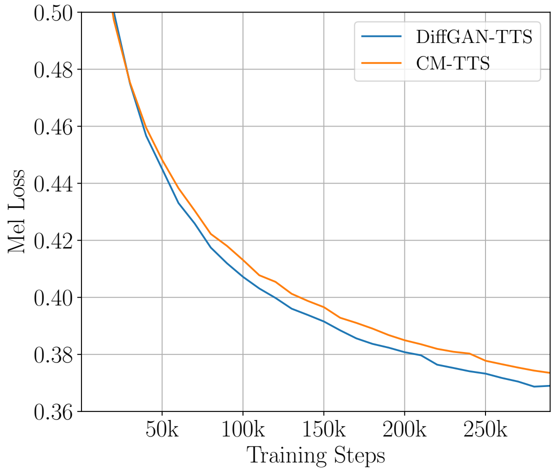

Figure 3: An Illustration of the Convergence of Loss Across DiffGAN-TTS and CM-TTS.

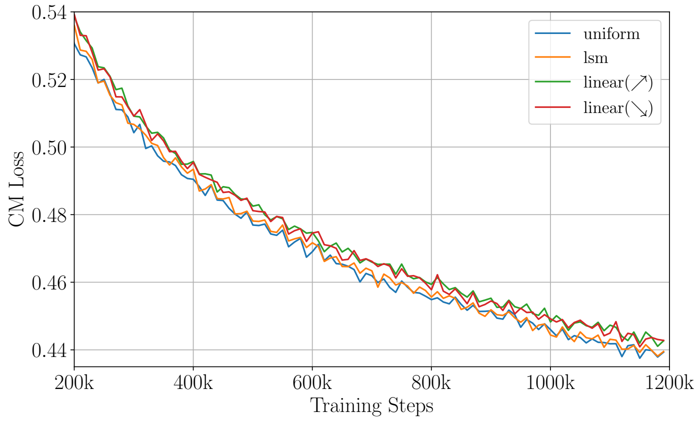

Figure 4: Convergence of loss across different Samplers.

C 50 Particularly Hard Sentences

To evaluate the robustness of CM-TTS, we follow the practice in (Ren et al., 2020; Ping et al., 2018) and generate 50 sentences which are particularly hard for the TTS system. Subjectively assessing the results, we observed that, aside from occasional inaccuracies in pronouncing individual words, the synthesis quality across the majority of examples is notably clear. This observation strongly supports the claim that CM-TTS exhibits considerable robustness in handling a wide range of linguistic complexities. The specific textual representations for all the sentences are provided below for reference.

01. a
02. b
03. c
04. H
05. I
06. J
07. K
08. L
09. 22222222 hello 22222222
10. S D S D Pass zero - zero Fail - zero to zero - zero - zero Cancelled - fifty nine to three - two - sixty four Total - fifty nine to three - two -

11. S D S D Pass - zero - zero - zero - zero Fail - zero - zero - zero - zero Cancelled - four hundred and sixteen - seventy six -
12. zero - one - one - two Cancelled - zero - zero - zero - zero Total - two hundred and eighty six - nineteen - seven -
13. forty one to five three hundred and eleven Fail - one - one to zero two Cancelled - zero - zero to zero zero Total -
14. zero zero one , MS03 - zero twenty five , MS03 - zero thirty two , MS03 - zero thirty nine ,
15. 1b204928 zero zero zero zero zero zero zero zero zero zero zero zero zero zero one seven ole32
16. zero zero zero zero zero zero zero zero two seven nine eight F three forty zero zero zero zero zero six four two eight zero one eight
17. c five eight zero three three nine a zero bf eight FALSE zero zero zero bba3add2 - c229 - 4cdb -
18. Calendaring agent failed with error code 0x80070005 while saving appointment .
19. Exit process - break ld - Load module - output ud - Unload module - ignore ser - System error - ignore ibp - Initial breakpoint -
20. Common DB connectors include the DB - nine , DB - fifteen , DB - nineteen , DB - twenty five , DB - thirty seven , and DB - fifty connectors .
21. To deliver interfaces that are significantly better suited to create and process RFC eight twenty one , RFC eight twenty two , RFC nine seventy seven , and MIME content .
22. int1 , int2 , int3 , int4 , int5 , int6 , int7 , int8 , int9 ,
23. seven _ ctl00 ctl04 ctl01 ctl00 ctl00
24. Http0XX , Http1XX , Http2XX , Http3XX ,
25. config file must contain A , B , C , D , E , F , and G .
26. mondo - debug mondo - ship motif - debug motif - ship sts - debug sts - ship Comparing local files to checkpoint files ...
27. Rusbvts . dll Dsaccessbvts . dll Exchmembvt . dll Draino . dll Im trying to deploy a new topology , and I keep getting this error .
28. You can call me directly at four two five seven zero three seven three four four or my cell four two five four four four seven four seven four or send me a meeting request with all the appropriate information .
29. Failed zero point zero zero percent < one zero zero one zero zero zero zero Internal . Exchange . ContentFilter . BVT ContentFilter . BVT_ log . xml Error ! Filename not specified .
30. C colon backslash o one two f c p a r t y backslash d e v one two backslash oasys backslash legacy backslash web backslash HELP
31. src backslash mapi backslash t n e f d e c dot c dot o l d backslash backslash m o z a r t f one backslash e x five
32. copy backslash backslash j o h n f a n four backslash scratch backslash M i c r o s o f t dot S h a r e P o i n t dot
33. Take a look at h t t p colon slash slash w w w dot granite dot a b dot c a slash access slash email dot
34. backslash bin backslash premium backslash forms backslash r e g i o n a l o p t i o n s dot a s p x dot c s Raj , DJ ,
35. Anuraag backslash backslash r a d u r five backslash d e b u g dot one eight zero nine underscore P R two h dot s t s contains
36. p l a t f o r m right bracket backslash left bracket f l a v o r right bracket backslash s e t u p dot e x e
37. backslash x eight six backslash Ship backslash zero backslash A d d r e s s B o o k dot C o n t a c t s A d d r e s
38. Mine is here backslash backslash g a b e h a l l hyphen m o t h r a backslash S v r underscore O f f i c e s v r
39. h t t p colon slash slash teams slash sites slash T A G slash default dot aspx As always , any feedback , comments ,
40. two thousand and five h t t p colon slash slash news dot com dot com slash i slash n e slash f d slash two zero zero three slash f d
41. backslash i n t e r n a l dot e x c h a n g e dot m a n a g e m e n t dot s y s t e m m a n a g e

42. I think Rich's post highlights that we could have been more strategic about how the sum total of XBOX three hundred and sixtys were distributed .
43. 64X64 , 8K , one hundred and eighty four ASSEMBLY , DIGITAL VIDEO DISK DRIVE , INTERNAL , 8X ,
44. So we are back to Extended MAPI and C++ because . Extended MAPI does not have a dual interface VB or VB .Net can read .
45. Thanks , Borge Trongmo Hi gurus , Could you help us E2K ASP guys with the following issue ?
46. Thanks J RGR Are you using the LDDM driver for this system or the in the build XDDM driver ?
47. Btw , you might remember me from our discussion about OWA automation and OWA readiness day a year ago .
48. empidtool . exe creates HKEY_ CURRENT_ USER Software Microsoft Office Common QMPerSNum in the registry , queries AD , and the populate the registry with MS employment ID if available else an error code is logged .
49. Thursday, via a joint press release and Microsoft AI Blog, we will announce Microsoft's continued partnership with Shell leveraging cloud, AI, and collaboration technology to drive industry innovation and transformation.
50. Actress Fan Bingbing attends the screening of 'Ash Is Purest White (Jiang Hu Er Nv)' during the 71st annual Cannes Film Festival

D Metrics

We employ 12 metrics to assess the quality and efficiency of speech synthesis. This includes 11 objective metrics and one subjective metric. The following provides a detailed analysis of the calculation methods and objectivity for all the metrics involved in the experiments.

- **FFE (Fundamental Frequency Frame Error):**
  - FFE, or F0 Frame Error (Chu and Alwan), combines Gross Pitch Error (GPE) and Voicing Decision Error (VDE) to objectively evaluate fundamental frequency (F0) tracking methods.
  - The Fundamental Frequency Frame Error (FFE) quantifies errors during the estimation of the fundamental frequency using the formula:

$$FFE = \frac{1}{N}\sum_{i=1}^{N}|F_{0i,\text{estimated}} - F_{0i,\text{actual}}|$$

    where N is the total number of frames, $F_{0i,\text{estimated}}$ is the estimated fundamental frequency of the i-th frame, and $F_{0i,\text{actual}}$ is the actual fundamental frequency of the i-th frame.

- **S.Cos (Speaker Cosine Similarity):**
  - S.Cos, or Speaker Cosine Similarity, measures the degree of similarity between speaker embeddings corresponding to synthesized speech and ground truth.
  - The Cosine Similarity is calculated as:

$$\text{Cosine Similarity}(\mathbf{P}, \mathbf{A}) = \frac{\mathbf{P}\cdot\mathbf{A}}{\|\mathbf{P}\|\|\mathbf{A}\|}$$

    where $\mathbf{P}\cdot\mathbf{A}$ is the dot product between speaker embeddings, and $\|\mathbf{P}\|\|\mathbf{A}\|$ is their Euclidean norm.

- **mfccFID (Fréchet Inception Distance based on MFCC):**
  - mfccFID calculates the Fréchet Inception Distance (FID) between MFCC features extracted from predicted and actual speech, measuring similarity between their distributions.

- The FID formula is given by:

$$FID = \|\mu_p - \mu_a\|^2 + \text{Tr}(\Sigma_p + \Sigma_a - 2(\Sigma_p \Sigma_a)^{1/2})$$

where μ_p and μ_a are mean vectors, and $\Sigma_p + \Sigma_a$ is the covariance matrix.

- **melFID (Fréchet Inception Distance based on Mel Spectrogram):**
  - melFID directly calculates FID between Mel spectrograms of predicted and actual frames.
- **mfccRecall:**
  - As outlined in Kynkäänniemi et al. (2019), we denote the feature vectors of real and generated mel spectrograms as ϕ_r and ϕ_g, respectively. In our approach, we utilized the MFCC features of the speeches, representing the sets of feature vectors as Φ_r and Φ_g. We ensured an equal number of samples were drawn from each distribution. Recall is computed by querying, for each real image, whether the image falls within the estimated manifold of generated images.
  - The formula is:

$$recall(\Phi_r, \Phi_g) = \frac{1}{|\Phi_r|} \sum_{\phi_r \in \Phi_r} f(\phi_r, \Phi_g)$$

$f(\phi, \Phi_g)$ provides a way to determine whether it could be reproduced by the generator.

- **MCD (Mel Cepstral Distortion):**
  - MCD measures the difference between two acoustic signals in the domain of Mel Cepstral Coefficients (MFCC).
  - The formula is:

$$MCD = \frac{1}{T} \sum_{t=1}^{T} d(c(p), c(a))$$

where T is the total number of frames, and $c(p)$ and $c(a)$ are the MFCC vectors of real and synthesized speech.

- **SSIM (Structural Similarity Index):**
  - SSIM measures the similarity between two spectrograms using luminance, contrast, and structure information.
  - The SSIM formula is given by:

$$\text{SSIM}(p, a) = \frac{(2\mu_p \mu_a + c_1)(2\sigma_{pa} + c_2)}{(\mu_p^2 + \mu_a^2 + c_1)(\sigma_p^2 + \sigma_a^2 + c_2)}$$

where p and a are the spectrograms, and μ_p, μ_a, σ_p^2, σ_a^2, σ_{pa}, c_1, and c_2 are constants.

- **mfccCOS (MFCC Cosine Similarity):**
  - mfccCOS measures the similarity between MFCC features of real and predicted speech using the same calculation method as S.Cos.
- **F0-RMSE (F0 Root Mean Squared Error):**
  - F0-RMSE is a metric measuring the difference between two pitch sequences (fundamental frequency).
  - The RMSE formula is:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (f_{0,i} - \hat{f}_{0,i})^2}$$

where N is the total number of frames, $f_{0,i}$ is the fundamental frequency of the i-th frame in the real pitch sequence, and $\hat{f}_{0,i}$ is the fundamental frequency of the i-th frame in the predicted pitch sequence.

- **RTF (Real-time Factor):**
  - RTF represents the time (in seconds) required for the system to synthesize one second of waveform.
- **MOS (Mean Opinion Score):**
  - MOS is an objective evaluation metric obtained through subjective experiments, assessing the quality of speech synthesis.
  - The MOS formula is:
  $$\text{MOS} = \frac{1}{N}\sum_{i=1}^{N} a_i$$
  where N is the number of participants, and a_i is the score provided by the i-th participant.
- **WER (Word Error Rate):**
  - WER measures the disparity between the transcribed text of the model's predicted speech and the actual speech. The calculation of WER includes three types of errors : Insertions, Deletions, and Substitutions.
  - The WER formula is:
  $$\text{WER} = \frac{S + D + I}{N} \times 100$$
  where S is the number of substitution errors, D is the number of deletion errors, I is the number of insertion errors and N is is the total number of words in the transcribed text.

E Metric

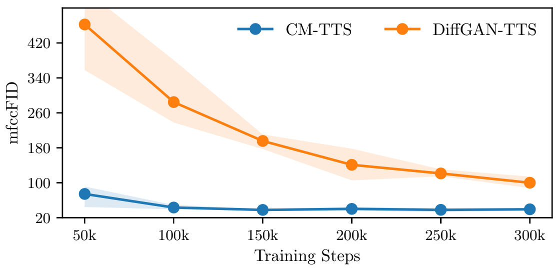

Figure 5: The trend of DiffGAN-TTS and CM-TTS on the mfcc-FID metric during training on VCTK.

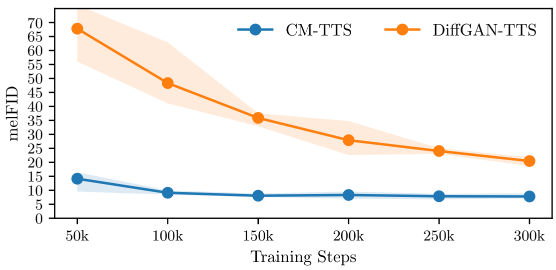

Figure 6: The trend of DiffGAN-TTS and CM-TTS on the mel-FID metric during training on VCTK.

As depicted in Figure 5 and Figure 6, the trend in metric changes highlights that CM-TTS displays faster convergence and a more stable model performance.

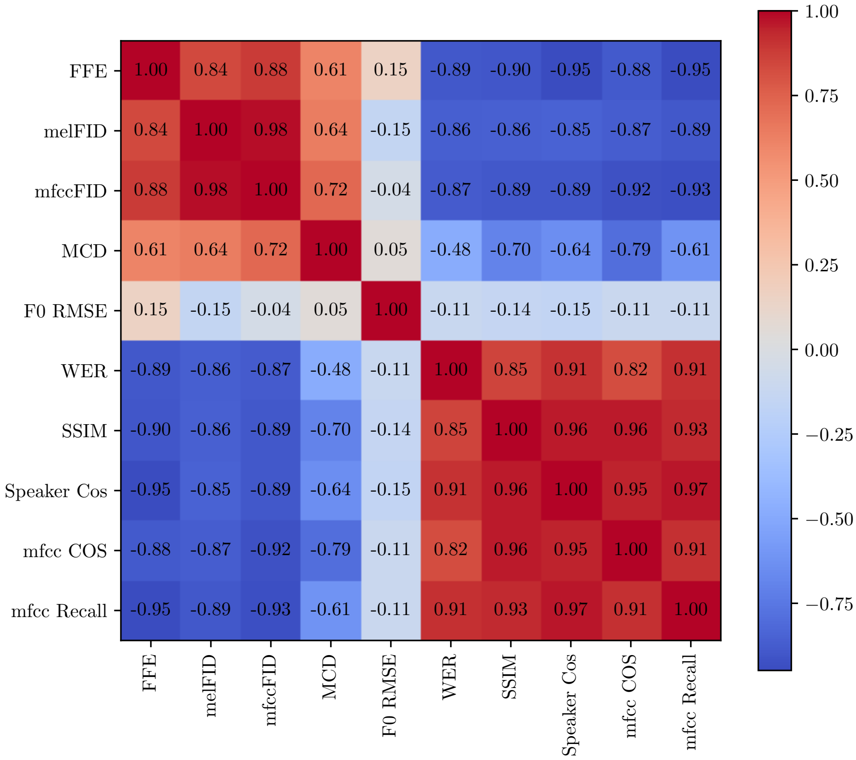

Figure 7: The Pearson correlation coefficient between different objective evaluation metrics.

We also explored relationships between various evaluation metrics, calculating trends' similarity using the Pearson coefficient and visualizing the results in Figure 7. Notably, significant correlations were observed among SSIM, Speaker Cos, mfccCOS, and mfcc Recall, indicating closely aligned trends. A strong correlation was also identified between the two types of FID. Conversely, MCD showed a weak relationship with metrics that perform better when lower. F0 RMSE displayed weak correlations with all other metrics, and FFE had a relatively modest relationship with metrics that are optimal when smaller. This study provides valuable insights for speech synthesis quality evaluation, suggesting that when testing only a few metrics, it's advisable to select those with lower correlations, as illustrated in the Figure 7, as evaluation indicators.