

An Investigation of Noise Robustness for Flow-Matching-Based Zero-Shot TTS

Xiaofei Wang, Sefik Emre Eskimez, Manthan Thakker, Hemin Yang, Zirun Zhu, Min Tang, Yufei Xia, Jinzhu Li, Sheng Zhao, Jinyu Li, Naoyuki Kanda

Microsoft Corporation, USA

{Xiaofei.Wang, Sefik.Eskimez, Manthan.Thakker, Naoyuki.Kanda}@microsoft.com

Abstract

Recently, zero-shot text-to-speech (TTS) systems, capable of synthesizing any speaker’s voice from a short audio prompt, have made rapid advancements. However, the quality of the generated speech significantly deteriorates when the audio prompt contains noise, and limited research has been conducted to address this issue. In this paper, we explored various strategies to enhance the quality of audio generated from noisy audio prompts within the context of flow-matching-based zero-shot TTS. Our investigation includes comprehensive training strategies: unsupervised pre-training with masked speech denoising, multi-speaker detection and DNSMOS-based data filtering on the pre-training data, and fine-tuning with random noise mixing. The results of our experiments demonstrate significant improvements in intelligibility, speaker similarity, and overall audio quality compared to the approach of applying speech enhancement to the audio prompt.

Index Terms: zero-shot TTS, noise-robust TTS, conditional flow-matching, generative pre-training, multi-task fine-tuning

1. Introduction

In recent years, text-to-speech (TTS) technology has made significant advancements [1, 2, 3], achieving a level of naturalness comparable to human speech [4]. Further advancements have been made towards a zero-shot TTS system [5, 6, 7, 8, 9, 10, 11, 12, 13] that can generate any speaker’s voice with minimal enrolled recordings, or an audio prompt. Zero-shot TTS has a wide range of applications, including speech-to-speech translation, audio/video content creation, and personal assistant services. However, one of the challenges faced by such systems is handling noise in the audio prompt. Existing zero-shot TTS models tend to generate speech with a style of noise similar to that contained in the audio prompt. This property is undesirable for many applications that require clean speech. In this paper, we aim to develop a zero-shot TTS system that can generate high-quality clean speech from any speaker, regardless of the existence of background noise in the audio prompt. We refer to this property as the noise robustness of zero-shot TTS.

While there has been a surge of research interest in zero-shot TTS technology, research on noise robustness is limited. The most naive approach involves applying speech enhancement (SE) to the audio prompt before feeding it to a zero-shot TTS model. While this approach is simple, even the latest SE models inevitably cause processing artifacts (e.g., [14, 15]), which result in degraded speech quality from the zero-shot TTS model. Our preliminary experiment revealed that the application of SE causes degradation in both intelligibility and speaker characteristics of the generated audio. Recently, Fujita et al. [16] proposed enhancing the noise robustness of zero-

shot TTS by training a noise-robust speaker embedding extractor using a self-supervised learning model. While the authors reported promising results, their method is only applicable to a zero-shot TTS system based on speaker embeddings. Most recent zero-shot TTS models utilize in-context learning, such as neural-codec-based language modeling [5, 6, 10] or audio infilling [9, 12, 13], instead of representing the audio prompt as a speaker embedding. It is essential to study the noise robustness of zero-shot TTS in state-of-the-art model architectures.

In the spirit of advancing state-of-the-art technology, this paper presents our efforts to improve the noise robustness of flow-matching-based zero-shot TTS [9], one of the leading models in terms of intelligibility and speaker characteristics preservation. We explored a range of training strategies, including generative pre-training [17] with masked speech denoising, multi-speaker detection and DNSMOS [18]-based data filtering on the pre-training data, as well as the fine-tuning with random noise mixing. Through experiments with both clean and noisy audio prompt settings, we demonstrate that intelligibility, speaker similarity, and overall audio quality can be consistently improved compared to an approach that applies SE to the audio prompt. In addition, as a byproduct, we demonstrate for the first time that our zero-shot TTS model achieves better speaker similarity compared to the ground-truth audio in the widely used cross-utterance evaluation setting on LibriSpeech [19].

2. Flow-matching based zero-shot TTS

2.1. Overview

Our TTS system closely follows Voicebox [9], which consists of a flow-matching-based audio model and a regression-based duration model. This section covers the overview of each model.

The objective of the audio model is to generate a log mel spectrum $\tilde{x} \in \mathbb{R}^{D \times T}$ given a frame-wise phoneme index sequence $a \in \mathbb{Z}_+^T$ under the condition that the value of \tilde{x} is partially known as $x_{\text{ctx}} \in \mathbb{R}^{D \times T}$. Here, D represents the feature dimension, and T is the sequence length. x_{ctx} is also known as the audio context, and the known value of \tilde{x} is filled in; otherwise, the value is set to zero. In the inference, \tilde{x} is generated based on x_{ctx} and a where a part of x_{ctx} is filled by the log mel spectrum of the audio prompt. Based on the in-context learning capability of the model, the speaker characteristics of the generated part of \tilde{x} becomes similar to that of the audio prompt. The estimated \tilde{x} is then converted to the speech signal based on a vocoder.

The audio model needs to be trained to enable sampling from $P(\tilde{x}|a, x_{\text{ctx}})$. It is achieved based on the flow-matching framework. This technique morphs a simple initial distribution p_0 into a more complex distribution p_1 that closely matches the observed data distribution. The model is trained based

on the conditional flow-matching objective [20]. Specifically, the model is trained to estimate a time-dependent vector field $v_t, t \in [0, 1]$, which is used to construct a flow ϕ_t that pushes the initial distribution towards the target distribution. The sampling process of \tilde{x} is achieved by solving the ordinary differential equation with the estimated vector field v_t and initial random value sampled from p_0 . Refer [20] for more details.

The duration model follows the regression-based approach detailed in [9]. This model takes a phoneme sequence $p \in \mathbb{Z}_+^N$, where N represents the number of phonemes. The model is trained to predict the duration for each phoneme $\tilde{l} \in \mathbb{R}_+^N$ under the condition that the value of \tilde{l} is partially known as $l_{\text{ctx}} \in \mathbb{Z}_+^N$. Similar to the audio model, l_{ctx} is filled by the known value of \tilde{l} , and the unknown part is filled by zero. The model is trained based on the mean square error loss on the predicted duration. Refer [9] for more details.

2.2. Unsupervised pre-training of audio model

Liu et al. [17] proposed to pre-train the flow-matching-based audio model with a large amount of unlabeled training data. They reported superior audio model quality after fine-tuning. During pre-training, the phoneme sequence a is dropped, and the model is trained to predict the distribution of $P(\tilde{x}|x_{\text{ctx}})$. For each training sample, n non-consecutive random segments are selected with a constraint of the minimum number of frames, Min_F , of each masked segment. In this work, we set $\text{Min}_F = 5$ for all our exploration based on our preliminary experiment.

3. Approach toward noise robustness

This section explains our approaches to enhance noise robustness through pre-training and fine-tuning of the flow-matching-based audio model.

3.1. Data filtering in pre-training

We want to utilize a large amount of unlabeled data for pre-training to further improve the performance of the audio models. However, real-world data is often low-quality and noisy, and using such data without proper filtering can negatively impact the model performance. Therefore, to ensure the quality of our models, we explore data filtering techniques that can effectively identify and prioritize high-quality, noise-free samples for pre-training. Consequently, we employ the following two strategies to filter the pre-training data.

Our first strategy involves filtering out the samples with more than one speaker. To detect the multiple speakers in an audio sample, we employ an in-house speaker change detection model, and discard a sample whenever the speaker change is detected. Our second strategy involves assessing the speech quality of the samples. We employ the DNSMOS [18], a neural network-based mean opinion score estimator¹, to evaluate the speech quality. We then discard samples that fall below a certain DNSMOS value threshold DNSMOS_T . In the experiments section, we explore the impact of different threshold values for our second strategy.

3.2. Masked speech denoising in pre-training

Masked speech denoising, introduced in WavLM [21], is an approach to enhance the model's ability to focus on relevant speech signals amid noise. It involves estimating clean audio for the masked part from the noisy audio input. Inspired by the

success of WavLM, we investigate a similar approach for flow-matching-based model pre-training.

During pre-training, in a probability of P_n^{pre} , we simulate noisy speech by mixing training samples with randomly selected noise, which yields pairs of noisy speech and clean speech. We use the noisy speech to extract the context input x_{ctx} , and the original training sample as the training target. In the noise mixing phase, we randomly sample the noise from the DNS challenge corpus [22], crop it, and mixed it with the signal-to-noise ratio (SNR) ranging from 0dB to 20 dB. We ensure that the duration of the noise does not exceed 50% of that of the training audio. We also explore the mixing of a secondary speaker into the audio with a probability P_s^{pre} , drawing parallels to WavLM. The secondary speaker is picked from the same training batch of the primary speaker. All the mixing settings are the same as the noise mixing one, except that the SNR ranges between [0, 10] dB.

3.3. Fine-tuning with random noise mixing

We also explore the fine-tuning strategy of the audio model. Conventionally, the audio model is fine-tuned with clean training data [17]. On the other hand, Fujita et al. [16] concurrently² proposed to fine-tune their zero-shot TTS model by including noise to the audio prompt in a 50% ratio to improve the noise robustness. In our work, we also explore the similar approach in the context of flow-matching-based zero-shot TTS. Specifically, we randomly add noise in a probability P_n^{ft} to the audio to extract the audio context x_{ctx} , while the training target remains the original clean audio. Noise samples from the DNS challenge corpus [22] are randomly selected and mixed at SNRs between -5 dB and 20 dB.

4. Experimental results

4.1. Training data

The pre-training data of the audio model consisted of 200,000 hours of in-house unlabeled anonymized English audio, without undergoing any form of preprocessing. The audio occasionally included background noise, with significant variations in quality.

For fine-tuning of the audio and duration model, we used the LibriLight [23], which consists of approximately 60,000 hours of untranscribed English audio from over 7,000 speakers [23]. Since LibriLight does not provide reference transcriptions, we transcribed the audio based on the off-the-shelf Kaldi automatic speech recognition (ASR)³, and used the phoneme sequences from the ASR hypothesis to fine-tune the audio model, as similar to [9].

4.2. Training and inference configurations

In our experiment, the architecture of the audio model closely followed the configurations in [9]. Specifically, we used Transformer with 24 layers, features 16 attention heads, and an embedding dimension of 1024. It also included a feed-forward layer dimension of 4096, alongside 1024 dimensions for phone embeddings. The model underwent pre-training over 25.6 million iterations. Linear-decay learning rate (lr) schedulers were used for both pre-training and fine-tuning, with a warm-up having 1/10 of the total number of updates and a peak lr at 7.5e-5.

²The paper [16] was published on Jan 10th, 2024 when we were preparing our paper.

³<https://kaldi-asr.org/models/m13>

¹<https://github.com/microsoft/DNS-Challenge/tree/master/DNSMOS>

Table 1: Results on zero-shot TTS for cross-utterance settings with LibriSpeech test-clean. SE: speech enhancement, DF: data filtering, Hu: HuBERT-L, Ne: NeMo, Wa: WavLM

| Model | Zero-shot TTS system | | | | Clean prompt | | | Noisy prompt | | |
|---------------------|----------------------|----|--------------------|--------------------|--|---|-------------------|--|------------------------------------|-------------------|
| | SE | DF | P_n^{pre} | P_n^{fit} | WER (%) \downarrow Avg. (Hu / Ne) | SIM-o \uparrow Avg. (Wa / Ne) | DNSMOS \uparrow | WER (%) \downarrow Avg. (Hu / Ne) | SIM-o \uparrow Avg. (Wa / Ne) | DNSMOS \uparrow |
| Ground truth | - | - | - | - | 1.9 (2.1 / 1.7) | 0.74 (0.71 / 0.77) | 3.30 | 4.2 (5.1 / 3.3) | 0.71 (0.68 / 0.73) | 2.57 |
| Ground truth | ✓ | - | - | - | 2.0 (2.1 / 1.8) | 0.74 (0.71 / 0.76) | 3.36 | 3.6 (3.8 / 3.3) | 0.68 (0.67 / 0.68) | 3.24 |
| VALL-E [5] | - | - | - | - | - (5.9 / -) | - | - | - | - | - |
| Naturalspeech 2 [7] | - | - | - | - | - (2.3 / -) | - (0.62 / -) | - | - | - | - |
| Voicebox [9] | - | - | - | - | - (1.9 / -) | - (0.66 / -) | - | - | - | - |
| SpeechFlow [17] | - | - | - | - | - (2.1 / -) | - (0.70 / -) | - | - | - | - |
| (B1) Our TTS model | - | - | - | - | 2.7 (2.3 / 3.0) | 0.71 (0.67 / 0.75) | 3.33 | 2.5 (2.3 / 2.6) | 0.60 (0.56 / 0.63) | 3.01 |
| (B2) Our TTS model | ✓ | - | - | - | 2.8 (2.3 / 3.2) | 0.69 (0.66 / 0.72) | 3.37 | 2.8 (2.3 / 3.3) | 0.60 (0.58 / 0.61) | 3.28 |
| (P1) Our TTS model | - | ✓ | - | - | 2.6 (2.2 / 3.0) | 0.75 (0.72 / 0.78) | 3.35 | 2.6 (2.4 / 2.7) | 0.65 (0.62 / 0.68) | 2.99 |
| (P2) Our TTS model | ✓ | ✓ | - | - | 2.8 (2.3 / 3.2) | 0.74 (0.71 / 0.76) | 3.39 | 2.9 (2.4 / 3.3) | 0.64 (0.63 / 0.65) | 3.29 |
| (P3) Our TTS model | - | ✓ | - | 1.0 | 2.8 (2.3 / 3.2) | 0.74 (0.70 / 0.77) | 3.35 | 2.7 (2.3 / 3.1) | 0.64 (0.61 / 0.67) | 3.32 |
| (P4) Our TTS model | - | ✓ | - | 0.5 | 2.7 (2.2 / 3.1) | 0.74 (0.71 / 0.77) | 3.34 | 2.7 (2.3 / 3.1) | 0.64 (0.61 / 0.66) | 3.31 |
| (P5) Our TTS model | - | ✓ | 0.5 | 0.5 | 2.6 (2.2 / 3.0) | 0.75 (0.71 / 0.78) | 3.35 | 2.6 (2.2 / 2.9) | 0.65 (0.62 / 0.67) | 3.32 |

Other investigations of hyperparameters will be discussed in the next session. During the inference, we used classifier-guidance-free with a guidance strength of 1.0, and the number of function evaluations (NFE) was 32. A BigVGAN [24]-based vocoder was used to convert the mel spectrum into waveforms.

As for the duration model, we used a regression-based masked duration model by closely following the configuration in [12]. Specifically, we used the following configuration: 8 layers, 8 attention heads, 512 embedding, and 2048 feed-forward dimensions. The model was trained with an effective mini-batch size of 120K frames for 600K mini-batch updates. We closely followed the training parameters in [12].

4.3. Evaluation data

In this study, we designed two test settings to evaluate the performance of zero-shot TTS models: a clean-prompt setting and a noisy prompt setting.

Clean-prompt setting: To assess the zero-shot TTS capabilities of our model under neutral speech conditions, we conducted evaluations using the ‘test-clean’ subset from the LibriSpeech dataset [19]. Adhering to the prior works [9, 5, 7], we selected audio samples with durations ranging from 4 to 10 seconds. For each sample, zero-shot TTS was performed using the transcription of the sample as a text prompt, and a 3-second audio clip from another randomly selected audio of the same speaker as an audio prompt. Following [9], we used the final 3 seconds of the randomly selected audio as the audio prompt.

Noisy-prompt setting: We prepared each test sample from the clean-prompt set by blending its audio prompt with a noise sample. The noise sample was randomly selected from the MUSAN dataset [25]. The signal-to-noise ratio (SNR) for the mixture was determined randomly, falling within a range of 0 dB to 20 dB. We then trimmed the final 3 seconds of the mixed sample and used it as the audio prompts for zero-shot TTS. The audio prompt selection mirrored the clean-prompt setting, ensuring the only difference between clean and noisy settings was the presence of noise in the audio prompt.

4.4. Evaluation metrics

We evaluated the generated speech based on the following metrics. Note that we synthesized the audio with three random seeds and reported the average of them for all our experiments.

Word error rate (WER): We used the WER as a metric to evaluate the intelligibility of the generated audio. For our exper-

iments, we employed two ASR systems: hubert-large-ls960-ft model⁴, which was used in most prior publications, and NeMo’s stt_en_conformer_transducer_large model⁵, known for its superior stability and robustness against noise and processing artifacts. We report both the average WER and the individual WERs to credibly assess intelligibility.

Speaker similarity score (SIM-o): A speaker verification model was used to evaluate how closely the generated speech resembles the voice characteristics of the audio prompt. Specifically, SIM-o was computed as the cosine similarity between the speaker embeddings between the generated speech and the original clean audio prompt, for both clean-prompt and noisy-prompt settings. We utilized two speaker embedding extraction models—the WavLM Large⁶, which was used in most prior works, and the NeMo’s TitaNet-Large⁷ as another speaker verification model. We report both the average SIM-o and the individual SIMs to credibly assess the speaker similarity.

DNSMOS: For measuring the cleanliness of the generated audio, we utilized the DNSMOS score. Specifically, we employed the OVRL score from the DNSMOS P.835 model [18].

4.5. Result

4.5.1. Pre-analysis on the ground-truth audio

The first row of Table. 1 presents WER, SIM-o, and DNSMOS of the ground-truth audio. Unsurprisingly, the ground-truth audio in the clean-prompt condition provided us with the low WER, high SIM-o, and high DNSMOS scores, and the addition of the noise significantly deteriorate all metrics. We employed the AlignCruse [26] model, which was trained for both SE and personalized SE in a multi-task setting, using the training configuration described in [27]. This hybrid model operates in SE mode when the speaker embedding is a zero vector and in personalized SE mode when the speaker embedding is a non-zero vector. We applied this model in SE mode to our noisy prompts to remove background noise. As expected, the SE significantly improved the DNSMOS score in the noisy-prompt condition. However, it came with the cost of the notable degradation on the SIM-o in the noisy-prompt condition, and also the slight degradation of WER in the clean-prompt condition. This result

⁴<https://huggingface.co/facebook/hubert-large-ls960-ft>

⁵https://huggingface.co/nvidia/stt_en_conformer_transducer_xlarge

⁶https://github.com/microsoft/Unispeech/tree/main/downstreams/speaker_verification

⁷https://huggingface.co/nvidia/speakerverification_en_titanet_large

Table 2: Impact of pre-training data filtering and fine-tuning steps. MS: multi-speaker detection-based filtering.

| Pre-training | Fine-tuning | Clean Prompt | | Noisy Prompt | | | |
|--------------|-------------|---------------------|-------|--------------|--------------|-------------|--------------|
| | | DNSMOS _T | MS | Steps | WER (%)↓ | SIM-o↑ | WER (%)↓ |
| 0.0 | - | 0.64M | | 2.65 | 0.707 | 2.46 | 0.593 |
| 0.0 | ✓ | 0.64M | | 2.66 | 0.719 | 2.48 | 0.612 |
| 2.6 | ✓ | 0.64M | | 2.72 | 0.746 | 2.51 | 0.641 |
| 2.8 | ✓ | 0.64M | | 2.62 | 0.749 | 2.51 | 0.647 |
| 3.0 | ✓ | 0.64M | | 2.64 | 0.746 | 2.43 | 0.638 |
| | | ✓ | 0.16M | 2.70 | 0.756 | 2.62 | 0.659 |
| | | ✓ | 0.32M | 2.64 | 0.752 | 2.54 | 0.650 |
| 2.8 | ✓ | 0.64M | | 2.62 | 0.749 | 2.51 | 0.647 |
| | | ✓ | 1.6M | 2.64 | 0.745 | 2.98 | 0.633 |

demonstrated the inherent difficulty to improve all metrics in both clean and noisy conditions.

4.5.2. Main results

The results of various zero-shot TTS models are presented from the 3rd row to the last row of Table 1. Firstly, our baseline TTS model (model B1) showed decent WER and SIM-o compared to prior works [5, 7, 9, 17] in the clean prompt setting.⁸ However, the SIM-o and DNSMOS significantly dropped in the noisy prompt setting, indicating that the noise in the audio prompt was transferred into the generated speech. Unexpectedly, we observed an improvement in the WER from NeMo ASR in the noisy prompt, which resulted in a minor average WER improvement. This might be because NeMo ASR is sensitive to speech artifacts rather than noise, and small amounts of noise might conceal the artifacts within the speech portion.

Secondly, when we applied SE on the audio prompt (model B2), we observed a significant improvement in the DNSMOS score ($3.01 \rightarrow 3.28$) in the noisy prompt setting. However, it came with noticeable degradation of SIM-o in the clean prompt setting ($0.71 \rightarrow 0.69$) and degradation of WER in both clean ($2.7\% \rightarrow 2.8\%$) and noisy prompt settings ($2.5\% \rightarrow 2.8\%$).

We then applied our proposed pre-training data filtering based on multi-talker detection and DNSMOS-based filtering with $\text{DNSMOS}_T = 2.8$ (model P1). It significantly improved WER and SIM-o for both clean and noisy prompt settings. However, in this condition, the application of SE on the audio prompt (model P2) still introduced significant degradation of WER and SIM-o as a trade-off for the DNSMOS improvement.

Next, we investigated the random application of noise in the fine-tuning as described in 3.3, where we set $P_n^{\text{ft}} = 1.0$ (model P3) or $P_n^{\text{ft}} = 0.5$ (model P4). Compared to model P1, both models P3 and P4 achieved a significant improvement in the DNSMOS score ($2.99 \rightarrow 3.32$ or 3.31), while showing a marginal degradation of WER and SIM-o.

We then applied noise in the pre-training with $P_n^{\text{pre}} = 0.5$ (model P5), which resulted in small but consistent improvements in all metrics in both the clean and noisy prompts. Compared to our baseline model with SE (model B2), the final model P5 showed significant improvement in WER and SIM-o while keeping the DNSMOS score similarly high. We conducted paired t-tests on the metric values from models B2 and P5, and confirmed that all improvements on WER and SIM-o were statistically significant, with p-values less than 0.05.

Finally, unexpectedly, we observed that our SIM-o of 0.75 for the clean prompt setting is even higher than that of the ground truth (0.74). Upon examination, we found that some

⁸We did not list Audiobox [12] due to the difference of the experimental configuration.

Table 3: Impact of pre-training hyper parameters. In this experiment, we pre-trained the TTS model with only 1.6M steps for rapid exploration.

| P_n^{pre} | P_s^{pre} | Clean Prompt | | Noisy Prompt | |
|--------------------|--------------------|--------------|--------------|--------------|--------------|
| | | WER (%)↓ | SIM-o↑ | WER (%)↓ | SIM-o↑ |
| 0.00 | 0.00 | 2.64 | 0.698 | 2.65 | 0.577 |
| 0.25 | 0.00 | 2.68 | 0.702 | 2.61 | 0.580 |
| 0.50 | 0.00 | 2.64 | 0.706 | 2.58 | 0.585 |
| 0.75 | 0.00 | 2.65 | 0.693 | 2.61 | 0.577 |
| 1.00 | 0.00 | 2.72 | 0.689 | 2.66 | 0.572 |
| 0.25 | 0.25 | 2.61 | 0.696 | 2.62 | 0.579 |
| 0.50 | 0.25 | 2.67 | 0.694 | 2.62 | 0.571 |

speakers in the LibriSpeech use different voice characteristics to represent different characters in the book. This sometimes resulted in a low SIM-o for the ground truth in cross-utterance settings. To the best of our knowledge, this is the first time that a zero-shot TTS model has achieved a better SIM-o in the cross-utterance evaluation setting with LibriSpeech test-clean.

4.5.3. Ablation studies

Table 2 shows the impact of pre-training data filtering and fine-tuning steps. We first found that the multi-speaker detection-based data filtering, which discarded 23.3% of the pre-training data, significantly improved SIM-o for both clean and noisy prompt settings. We also found that the $\text{DNSMOS}_T = 2.8$ achieved the highest SIM-o for both clean prompt and noisy prompt settings while achieving the lowest WER for the clean prompt. These results suggest that our pre-training data filtering effectively filtered out low-quality audio samples while keeping the diverse speaker data as much as possible. We also found the gradual degradation in SIM-o scores as the number of fine-tuning steps increased from 0.16M to 1.6M, as shown in the lower half of the table. This suggests that the diverse speaker characteristics learned by the large-scale unsupervised pre-training can be deteriorated by the excessive fine-tuning with less speaker diversity.

Table 3 shows the result of further investigations on the pre-training hyperparameters. In this experiment, we pre-trained the TTS model with only 1.6M steps for rapid exploration. We first observed that the random noise mixing during the pre-training effectively improved SIM-o score, and observed the best result with $P_n^{\text{pre}} = 0.5$. We also investigated the mixing of a secondary speaker’s speech with $P_s^{\text{pre}} = 0.25$. However, we did not observe any noticeable improvement from this trial. Overall, we observed that appropriately injecting noise during the pre-training stage could effectively improve the zero-shot TTS model, not only for the noisy prompt setting but also for the clean prompt setting.

5. Conclusions

This paper presented our efforts to enhance the noise robustness of flow-matching-based zero-shot TTS. Our investigation covered a range of strategies, encompassing unsupervised pre-training with DNSMOS-based data filtering and masked speech denoising, as well as multi-task fine-tuning with random noise mixing. The results of our experiments demonstrated significant improvements in intelligibility, speaker similarity, and overall audio quality compared to the approach of applying speech enhancement to the audio prompt.

6. References

- [1] P. Taylor, *Text-to-speech synthesis*. Cambridge university press, 2009.
- [2] X. Tan, T. Qin, F. Soong, and T.-Y. Liu, “A survey on neural speech synthesis,” *arXiv preprint arXiv:2106.15561*, 2021.
- [3] C. Zhang, C. Zhang, S. Zheng, M. Zhang, M. Qamar, S.-H. Bae, and I. S. Kweon, “A survey on audio diffusion models: Text to speech synthesis and enhancement in generative AI,” *arXiv preprint arXiv:2303.13336*, vol. 2, 2023.
- [4] X. Tan, J. Chen, H. Liu, J. Cong, C. Zhang, Y. Liu, X. Wang, Y. Leng, Y. Yi, L. He *et al.*, “NaturalSpeech: End-to-end text-to-speech synthesis with human-level quality,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [5] C. Wang, S. Chen, Y. Wu, Z. Zhang, L. Zhou, S. Liu, Z. Chen, Y. Liu, H. Wang, J. Li *et al.*, “Neural codec language models are zero-shot text to speech synthesizers,” *arXiv preprint arXiv:2301.02111*, 2023.
- [6] Z. Borsos, M. Sharifi, D. Vincent, E. Kharitonov, N. Zeghidour, and M. Tagliasacchi, “SoundStorm: Efficient parallel audio generation,” *arXiv preprint arXiv:2305.09636*, 2023.
- [7] K. Shen, Z. Ju, X. Tan, Y. Liu, Y. Leng, L. He, T. Qin, S. Zhao, and J. Bian, “NaturalSpeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers,” *arXiv preprint arXiv:2304.09116*, 2023.
- [8] Z. Jiang, Y. Ren, Z. Ye, J. Liu, C. Zhang, Q. Yang, S. Ji, R. Huang, C. Wang, X. Yin *et al.*, “Mega-TTS: Zero-shot text-to-speech at scale with intrinsic inductive bias,” *arXiv preprint arXiv:2306.03509*, 2023.
- [9] M. Le, A. Vyas, B. Shi, B. Karrer, L. Sari, R. Moritz, M. Williamson, V. Manohar, Y. Adi, J. Mahadeokar *et al.*, “Voicebox: Text-guided multilingual universal speech generation at scale,” *NeurIPS 2024*, vol. 36, 2024.
- [10] X. Wang, M. Thakker, Z. Chen, N. Kanda, S. E. Eskimez, S. Chen, M. Tang, S. Liu, J. Li, and T. Yoshioka, “SpeechX: Neural codec language model as a versatile speech transformer,” *arXiv preprint arXiv:2308.06873*, 2023.
- [11] Y. Gao, N. Morioka, Y. Zhang, and N. Chen, “E3 TTS: Easy end-to-end diffusion-based text to speech,” in *ASRU 2023*. IEEE, 2023, pp. 1–8.
- [12] A. Vyas, B. Shi, M. Le, A. Tjandra, Y.-C. Wu, B. Guo, J. Zhang, X. Zhang, R. Adkins, W. Ngan *et al.*, “Audiobox: Unified audio generation with natural language prompts,” *arXiv preprint arXiv:2312.15821*, 2023.
- [13] N. Kanda, X. Wang, S. E. Eskimez, M. Thakker, H. Yang, Z. Zhu, M. Tang, C. Li, S. Tsai, Z. Xiao *et al.*, “Making flow-matching-based zero-shot text-to-speech laugh as you like,” *arXiv preprint arXiv:2402.07383*, 2024.
- [14] K. Iwamoto, T. Ochiai, M. Delcroix, R. Ikeshita, H. Sato, S. Araki, and S. Katagiri, “How bad are artifacts?: Analyzing the impact of speech enhancement errors on ASR,” in *INTERSPEECH 2022*, 2022, pp. 5418–5422.
- [15] S. E. Eskimez, X. Wang, M. Tang, H. Yang, Z. Zhu, Z. Chen, H. Wang, and T. Yoshioka, “Human listening and live captioning: Multi-task training for speech enhancement,” in *INTERSPEECH 2021*, 2021, pp. 2686–2690.
- [16] K. Fujita, H. Sato, T. Ashihara, H. Kanagawa, M. Delcroix, T. Moriya, and Y. Iijima, “Noise-robust zero-shot text-to-speech synthesis conditioned on self-supervised speech-representation model with adapters,” *arXiv preprint arXiv:2401.05111*, 2024.
- [17] A. H. Liu, M. Le, A. Vyas, B. Shi, A. Tjandra, and W.-N. Hsu, “Generative pre-training for speech with flow matching,” *arXiv preprint arXiv:2310.16338*, 2023.
- [18] C. K. Reddy, V. Gopal, and R. Cutler, “DNSMOS: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors,” in *ICASSP 2021*, 2021, pp. 6493–6497.
- [19] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “LibriSpeech: An asr corpus based on public domain audio books,” in *ICASSP 2015*. IEEE, 2015, pp. 5206–5210.
- [20] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, “Flow matching for generative modeling,” in *Proc. ICLR*, 2023.
- [21] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, “WavLM: Large-scale self-supervised pre-training for full stack speech processing,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.
- [22] H. Dubey, A. Aazami, V. Gopal, B. Naderi, S. Braun, R. Cutler, A. Ju, M. Zohourian, M. Tang, H. Gamper, M. Golestaneh, and R. Aichner, “ICASSP 2023 deep noise suppression challenge,” *arXiv preprint arXiv:2303.11510*, 2023.
- [23] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayı, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux, “Libri-Light: A benchmark for ASR with limited or no supervision,” in *ICASSP 2020*, 2020, pp. 7669–7673.
- [24] S.-g. Lee, W. Ping, B. Ginsburg, B. Catanzaro, and S. Yoon, “BigVGAN: A universal neural vocoder with large-scale training,” in *ICLR 2022*, 2022.
- [25] D. Snyder, G. Chen, and D. Povey, “MUSAN: A music, speech, and noise corpus,” *arXiv preprint arXiv:1510.08484*, 2015.
- [26] E. Indenbom, N.-C. Ristea, A. Saabas, T. Pärnamaa, and J. Gužvin, “Deep model with built-in self-attention alignment for acoustic echo cancellation,” *arXiv preprint arXiv:2208.11308*, 2022.
- [27] S. E. Eskimez, T. Yoshioka, A. Ju, M. Tang, T. Pärnamaa, and H. Wang, “Real-time joint personalized speech enhancement and acoustic echo cancellation,” in *INTERSPEECH 2023*, 2023, pp. 1050–1054.