Instance-Specific Test-Time Training for Speech Editing in the Wild

Taewoo Kim
Korea Electronics Technology Institute
kimtaewoo@keti.re.kr

Uijong Lee
Korea Electronics Technology Institute
jjong2201@keti.re.kr

Hayoung Park
Korea Electronics Technology Institute
hyformal@keti.re.kr

Choongsang Cho
Korea Electronics Technology Institute
ideafisher@keti.re.kr

Nam In Park
National Forensic Service
naminpark@korea.kr

Young Han Lee
Korea Electronics Technology Institute
yhlee@keti.re.kr

Abstract

Speech editing systems aim to naturally modify speech content while preserving acoustic consistency and speaker identity. However, previous studies often struggle to adapt to unseen and diverse acoustic conditions, resulting in degraded editing performance in real-world scenarios. To address this, we propose an instance-specific test-time training method for speech editing in the wild. Our approach employs direct supervision from ground-truth acoustic features in unedited regions and indirect supervision in edited regions via auxiliary losses based on duration constraints and phoneme prediction. This strategy mitigates the bandwidth discontinuity problem in speech editing, ensuring smooth acoustic transitions between unedited and edited regions. Additionally, it enables precise control over speech rate by adapting the model to target durations via mask length adjustment during test-time training. Experiments on in-the-wild benchmark datasets demonstrate that our method outperforms existing speech editing systems in both objective and subjective evaluations.

1 Introduction

Speech editing is a task that modifies speech content while preserving speaker identity and acoustic characteristics. It plays a pivotal role in speech applications such as disfluency removal, content creation, and speech de-identification. One key application is speech de-identification, which removes or replaces personally identifiable information such as names or credit card numbers, enabling the production of privacy-sensitive content without re-recording. However, achieving seamless integration between the edited and unedited regions remains challenging, particularly under diverse and unpredictable acoustic conditions. For practical deployment, it is essential that models maintain naturalness and speaker-identity consistency, ensure content fidelity, and remain robust to in-the-wild acoustic variability.

Recent advances in speech editing [1, 2, 3, 4, 5, 6, 7] have been largely enabled by the architectures and principles of neural text-to-speech [8, 9, 10, 11]. Tan et al. [1] proposed an autoregressive (AR) model that divides the audio into edited and unedited regions and merges forward and backward generation results to ensure smooth acoustic transitions. Bai et al. [2] introduced a speech editing system that demonstrated robust performance for unseen speakers by leveraging speech-text alignment embeddings. Furthermore, Jiang et al. [3] adopt a context-aware spectrogram denoiser

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: The First Workshop on Generative and Protective AI for Content Creation.

to achieve high-quality and expressive speech. Although these various approaches have achieved promising results in restricted environments such as audiobooks, their effectiveness in real-world speech scenarios remains largely unexplored.

Compared to controlled studio settings, speech editing in the wild is considerably more challenging, as it involves various complex factors such as background noise, reverberation, and bandwidth mismatch. Peng et al. [4] introduced VoiceCraft, a neural codec language model for speech editing, which improves context representation in an AR model through token rearrangement and refined causal masking. However, it depends on large-scale datasets and lacks fine-grained control over the prosody of the edited regions.

In this work, we propose an instance-specific test-time training (TTT) approach to address the challenges of speech editing in real-world scenarios. Our method fine-tunes a speech editing model for each test sample at inference time, leveraging direct supervision from unedited regions and indirect supervision from edited regions. To this end, we apply TTT in two stages, targeting the duration predictor and the spectrogram denoiser. The duration predictor is optimized using phoneme duration loss on unedited regions and auxiliary duration losses on the edited regions, enhancing prosodic consistency and enabling control over speech rate by adjusting the length of edited segments. The spectrogram denoiser is optimized with reconstruction loss and phoneme classification loss, which mitigates overfitting and improves speaker similarity and acoustic consistency under real-world conditions. Experimental evaluations demonstrate that, despite being pretrained on clean speech data, our method exhibits robust editing performance on acoustically challenging audio samples.

2 Related Work

2.1 Speech Editing

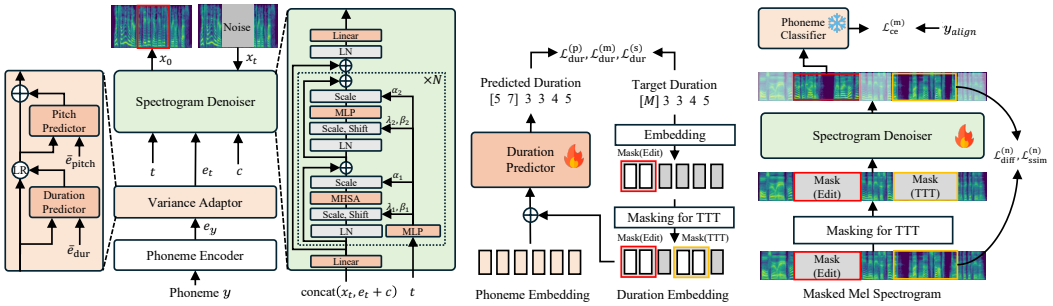
Early approaches to speech editing focused on modifying acoustic parameters rather than altering the linguistic content of speech. Traditional signal processing techniques such as PSOLA [12], MBROLA [13], and WORLD [14] enabled prosody modification, including pitch and duration adjustments, by directly manipulating the waveform. However, their reliance on direct waveform manipulation limited their ability to perform linguistic edits, such as inserting or replacing words.

Building on advances in automatic speech recognition (ASR) and neural text-to-speech (TTS) systems, research on speech editing has shifted toward detecting the target text segment to be modified and synthesizing replacement speech accordingly. Notable approaches include EditSpeech [1], which employs bidirectional fusion for smooth boundary transitions, and A^3T [2] with alignment-aware acoustic-text pretraining. However, these models still struggle to achieve robustness under diverse real-world conditions. More recently, VoiceCraft [4], a neural codec-based model, has been proposed to improve robustness in the wild, but it still lacks fine-grained controllability over prosody and duration in the edited regions. In this work, we explore methods to enhance robustness in real-world scenarios while enabling controllable prosody, without relying on large-scale speech datasets.

2.2 Test-Time Training for Speech Editing

Test-time training (TTT) [15] is a paradigm in which a model is adapted to each test instance during inference, typically by optimizing a self-supervised or auxiliary loss on the given input [15]. This allows the model to leverage instance-specific information, improving generalization to distribution shifts without retraining on large datasets. TTT has been successfully applied for domain adaptation [16, 17] and speech processing tasks such as speech recognition [18] and speech enhancement [19].

In the context of speech editing, TTT remains largely unexplored. The capability to fine-tune a speech editing model on each test utterance could mitigate mismatches between training and deployment conditions, particularly under diverse noise, reverberation, or bandwidth constraints. Our work extends this idea by introducing an instance-specific TTT framework that applies direct supervision on unedited regions and auxiliary constraints on edited regions, enabling prosodic control and improved acoustic consistency in real-world speech editing scenarios.



(a) Model Architecture (b) TTT for Duration Predictor (c) TTT for Spectrogram Denoiser

Figure 1: Overview of the proposed framework. In subfigure (a), LR denotes the length regulator. In subfigure (b), M indicates the length of the edit mask, which is required for test-time training (TTT) of the duration predictor. In subfigures (b) and (c), "Masking for TTT" refers to randomly masking unedited regions to compute the reconstruction loss during TTT. Red boxes indicate edit regions, and yellow boxes represent randomly masked regions for TTT. The flame icon denotes modules that are updated during TTT, whereas the snowflake icon indicates modules whose parameters remain frozen.

3 Method

In this section, we present our proposed method, which consists of three components: model architecture, train-time training, and test-time training. The overall framework is illustrated in Fig. 1. We first present a backbone model for speech editing and its train-time training procedure, followed by a detailed description of our test-time training strategy. Each component is discussed in the following subsections.

3.1 Model Architecture

Our backbone for speech editing is built upon the architecture of FluentSpeech [3], with a key modification: we replace the non-causal WaveNet [20] used in the spectrogram denoiser with a Diffusion Transformer (DiT) [21] to enhance context modeling and generation performance. The overall architecture of the model consists of a phoneme encoder, a variance adaptor, and a spectrogram denoiser, as illustrated in Fig. 1(a). The phoneme encoder converts a phoneme sequence $y \in \mathbb{Z}^{1\times N}$, where N is the length of the phoneme sequence, into D-dimensional phoneme representations $e_y \in \mathbb{R}^{D\times N}$. The variance adaptor, which includes a duration predictor and a pitch predictor, predicts the duration and pitch of the masked regions to transform e_y into aligned hidden representations $e_t \in \mathbb{R}^{D\times T}$, where T is the target length of the output sequence. In this process, both the duration predictor and the pitch predictor take e_y along with the masked contextual representations, e_{dur} and e_{pitch}, as inputs. Finally, the spectrogram denoiser takes as input the aligned hidden representation e_t, the noisy mel-spectrogram x_t at the diffusion timestep t, and the condition c, which consists of the speaker embedding and the masked mel-spectrogram embedding. It then predicts the clean target mel-spectrogram x_0 [3, 22] by performing the reverse process of the generator-based diffusion model, formulated as $f_\theta(x_t, c, e_t, t)$.

3.2 Train-Time Training

During training, following [3], the model is trained with reconstruction losses on duration, pitch, and mel-spectrogram prediction. First, the duration and pitch losses are computed using L2 loss as follows:

$$\mathcal{L}_{\text{dur}} = \|d - f_{\text{dp}}(e_y, \bar{e}_{\text{dur}})\|_2^2, \tag{1}$$

$$\mathcal{L}_{\text{pitch}} = \|p - f_{\text{pp}}(e_t, \bar{e}_{\text{pitch}})\|_2^2, \tag{2}$$

where f_{dp} and f_{pp} represent the duration predictor and the pitch predictor, respectively, and p and d are the target pitch and duration in the masked regions. $\bar{e}_{\text{dur}}$ and $\bar{e}_{\text{pitch}}$ denote masked embeddings

provided to each predictor. This encourages the predictors to infer prosodic patterns directly from corrupted or incomplete contextual cues. For mel-spectrogram loss, the output of the spectrogram denoiser is computed against the ground truth mel-spectrogram using both L1 loss and structural similarity index (SSIM) loss [23]:

$$\mathcal{L}_{\text{diff}} = \left\| f_\theta(x_t, c, e_t, t) - x_0 \right\|_1, \tag{3}$$

$$\mathcal{L}_{\text{ssim}} = 1 - \text{SSIM}\left(f_\theta(x_t, c, e_t, t), x_0 \right), \tag{4}$$

where x_0 denotes the ground-truth mel-spectrogram at the masked regions, and x_t is the noisy mel-spectrogram at timestep t, obtained through the forward diffusion process as formulated in [3].

Finally, the overall training objective is formulated as a weighted sum of the above components:

$$\mathcal{L}_{\text{train}} = \lambda_{\text{dur}}\mathcal{L}_{\text{dur}} + \lambda_{\text{pitch}}\mathcal{L}_{\text{pitch}} + \lambda_{\text{diff}}\mathcal{L}_{\text{diff}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}}, \tag{5}$$

where $\lambda_{\text{dur}}, \lambda_{\text{pitch}}, \lambda_{\text{diff}}$, and λ_{ssim} are coefficients that control the relative contributions of each loss term. This formulation ensures that the model jointly optimizes prosodic characteristics and spectral fidelity.

3.3 Test-Time Training

We propose a test-time training (TTT) strategy to enhance prosodic and acoustic consistency at inference time. This approach follows a commonly used instance-level TTT scheme [15], in which the model is adapted individually for each test sample. Our method consists of two stages that fine-tune the duration predictor and spectrogram denoiser.

3.3.1 TTT for Duration Predictor

In the first stage, TTT is applied to the duration predictor, a key module that predicts the durations within the edited region by capturing the prosodic context from the input text and surrounding unedited regions. To adapt to variations in speaking style across different test conditions, the duration predictor is fine-tuned at test time. To facilitate TTT, we apply additional random masking to the duration embeddings outside the edited region, as illustrated in Fig. 1(b). For each test sample, multiple input variants are created using different random masking patterns. These variants are grouped into a batch, increasing the batch size and enabling the model to adapt using a diverse set of masked inputs derived from the test sample. The model is then fine-tuned using a phoneme-level duration loss, denoted as $\mathcal{L}_{\text{dur}}^{(\text{p})}$, computed at these masked positions. A mask-level duration loss, $\mathcal{L}_{\text{dur}}^{(\text{m})}$, is defined with respect to the sum of the predicted phoneme-level durations within the masked region. A sentence duration loss, $\mathcal{L}_{\text{dur}}^{(\text{s})}$, is also introduced, based on the total predicted duration of the entire utterance. We define the total TTT loss for the duration predictor as a weighted combination of three L2 losses, where λ_{p}, λ_{m}, λ_{s} are the weights for phoneme-level, mask-level, and sentence-level duration losses, respectively:

$$\mathcal{L}_{\text{test}}^{\text{DP}} = \lambda_{\text{p}}\mathcal{L}_{\text{dur}}^{(\text{p})} + \lambda_{\text{m}}\mathcal{L}_{\text{dur}}^{(\text{m})} + \lambda_{\text{s}}\mathcal{L}_{\text{dur}}^{(\text{s})}. \tag{6}$$

3.3.2 TTT for Spectrogram Denoiser

In the second stage, TTT is applied to the spectrogram denoiser to enhance the naturalness and acoustic consistency of the generated speech. Similarly to the previous stage, an additional masking strategy is applied to regions of the mel-spectrogram outside the inference mask, as illustrated in Fig. 1(c). The spectrogram denoiser is fine-tuned by computing reconstruction losses over these newly masked regions. To maintain intelligibility, we employ a pretrained phoneme classifier to the predicted mel-spectrogram within the inference mask, computing a cross-entropy loss against the aligned phoneme sequence. The total TTT loss for the spectrogram denoiser is defined as a weighted sum of the following terms:

$$\mathcal{L}_{\text{test}}^{\text{SD}} = \lambda_{\text{diff}}\mathcal{L}_{\text{diff}}^{(\text{n})} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}}^{(\text{n})} + \lambda_{\text{ce}}\mathcal{L}_{\text{ce}}^{(\text{m})}, \tag{7}$$

where the superscripts (n) and (m) indicate the newly masked region used for reconstruction loss and the inference mask region used for phoneme classification, respectively. This joint optimization encourages the model to produce outputs that are both acoustically consistent and intelligible.

4 Experiments

4.1 Dataset and Preprocessing

We use the LibriTTS dataset [24], a multi-speaker English corpus containing approximately 585 hours of speech recorded at 24 kHz. For training on clean speech, we use only the `train-clean-100` and `train-clean-360` subsets, totaling about 245 hours from 1,151 speakers. We evaluated the model in both clean and in-the-wild conditions. The clean condition uses the `test-clean` subset of LibriTTS, while the in-the-wild condition is evaluated using the GigaSpeech test set [25], which consists of 16 kHz audio recordings from podcasts and YouTube videos. All audio is resampled to 22.05 kHz with 16-bit quantization.

In our evaluation setup, we randomly sample 400 utterances from each test set for objective evaluation, and 40 utterances for subjective evaluation. To align audio with transcripts, we use the Montreal Forced Aligner (MFA) [26]. For waveform synthesis from mel-spectrograms, we adopt the pretrained UNIVERSAL V1 HiFi-GAN vocoder[1] [27], which uses a 1024-point fast Fourier transform (FFT), a 256-sample hop size, a 1024-sample window length, and 80 mel-filterbanks covering the frequency range from 0 to 8 kHz. In addition, pitch contours are extracted using Parselmouth[2].

4.2 Experimental Setup

Our proposed speech editing model consists of three main components: a phoneme encoder, a variance adaptor, and a spectrogram denoiser. The spectrogram denoiser adopts Diffusion Transformer (DiT) blocks with zero-initialized adaptive Layer Normalization, configured with 12 Transformer layers, 6 attention heads, and a hidden dimension of 384. All other settings follow [3]. The total number of parameters is approximately 44M, increasing to 46M when including the phoneme classifier used for test-time training (TTT). The phoneme classifier follows the architecture introduced by [28], comprising two feed-forward Transformer blocks [9] followed by a linear projection layer. It is trained jointly with the baseline model, but optimized separately using cross-entropy loss to predict the aligned phoneme sequence. More detailed descriptions of the model configuration are provided in Appendix A.

For training, we use 8 diffusion steps ($T = 8$), a batch size of 32, and the Adam optimizer with a learning rate of 2×10^{-4}. The model is trained for 700K iterations on a single NVIDIA A40 GPU. Each TTT stage consists of 200 fine-tuning steps with the same batch size and optimizer. Only the duration predictor or the spectrogram denoiser is updated during TTT, with all other components remaining frozen. The learning rates for the masked duration predictor and the spectrogram denoiser are set to 2×10^{-4} and 5×10^{-5}, respectively. We apply 80% phoneme-level masking during both training and TTT. During TTT, however, masking is applied only to unedited regions. We set λ_{p}, λ_{m}, and λ_{s} all to 1.0, and λ_{dur}, λ_{pitch}, λ_{diff}, λ_{ssim}, and λ_{ce} to 1.0, 1.0, 0.5, 0.5, and 1.0, respectively.

4.3 Baselines

To evaluate the effectiveness of our proposed method, we compare it against two baseline systems: FluentSpeech [3] and VoiceCraft [4]. FluentSpeech is a non-autoregressive framework that incorporates a variance adaptor and a spectrogram denoiser for speech editing. Trained on the same LibriTTS clean subsets as our model, we directly use its official implementation[3]. In contrast, VoiceCraft adopts a large-scale autoregressive transformer architecture for speech editing. It is pretrained with 830M parameters on the GigaSpeech XL corpus [25], which contains approximately 10,000 hours of diverse audio data from podcasts and YouTube videos. We employ the released official model[4] for evaluation, representing a large-scale pretraining baseline that contrasts with our framework trained solely on LibriTTS clean subsets.

[1] `https://github.com/jik876/hifi-gan`

[2] `https://github.com/YannickJadoul/Parselmouth`

[3] `https://github.com/Zain-Jiang/Speech-Editing-Toolkit`

[4] `https://github.com/jasonppy/VoiceCraft`

Table 1: Speech editing performance on LibriTTS and GigaSpeech test sets. "Proposed" uses test-time training (TTT). "DP" and "SD" denote Duration Predictor and Spectrogram Denoiser, respectively.

System	Data (hr)	Params	WER ↓	SIM ↑	MCD ↓	MOS$_{\pm CI}$ ↑
Test Set: LibriTTS clean						
FluentSpeech	LibriTTS(245)	24M	4.35	0.765	4.38	3.94$_{\pm 0.04}$
VoiceCraft	GigaSpeech(10,000)	830M	5.22	0.778	4.39	3.99$_{\pm 0.04}$
Proposed	LibriTTS(245)	46M	**4.13**	**0.815**	<u>4.02</u>	**4.02**$_{\pm 0.04}$
w/o TTT for DP	LibriTTS(245)	46M	4.21	<u>0.811</u>	4.22	4.00$_{\pm 0.03}$
w/o TTT for SD	LibriTTS(245)	44M	<u>4.20</u>	0.789	**4.01**	<u>4.02</u>$_{\pm 0.04}$
w/o TTT for Both	LibriTTS(245)	44M	4.24	0.792	4.26	4.01$_{\pm 0.03}$
Ground Truth	-	-	4.24	-	-	4.11$_{\pm 0.03}$
Test Set: GigaSpeech						
FluentSpeech	LibriTTS(245)	24M	17.88	0.662	6.16	3.69$_{\pm 0.04}$
VoiceCraft	GigaSpeech(10,000)	830M	20.72	**0.758**	6.13	3.86$_{\pm 0.04}$
Proposed	LibriTTS(245)	46M	<u>16.88</u>	<u>0.725</u>	**5.60**	**3.87**$_{\pm 0.04}$
w/o TTT for DP	LibriTTS(245)	46M	**16.85**	0.711	5.92	<u>3.86</u>$_{\pm 0.04}$
w/o TTT for SD	LibriTTS(245)	44M	17.46	0.687	<u>5.64</u>	3.79$_{\pm 0.04}$
w/o TTT for Both	LibriTTS(245)	44M	17.15	0.688	5.98	3.75$_{\pm 0.04}$
Ground Truth	-	-	16.78	-	-	3.88$_{\pm 0.04}$

Table 2: Ablation study of test-time training components in the Duration Predictor on the GigaSpeech test set.

Method	WER ↓	SIM ↑	MCD ↓	CMOS ↑
TTT for Duration Predictor	**17.46**	**0.687**	**5.64**	**0**
w/o Mask Duration Loss (MDL)	17.49	0.681	5.80	−0.31
w/o MDL and Sentence Duration Loss	17.91	0.683	6.08	−0.34

4.4 Evaluation Metrics

For evaluation, we follow the setup of [2], where the middle third of each evaluation utterance is masked and reconstructed using the original transcript, allowing for comparison between the ground-truth and the reconstructed audio. Objective metrics include word error rate (WER)[5] [29], speaker similarity (SIM)[6] [30], and mel-cepstral distortion (MCD) [31]. For subjective evaluation, we use mean opinion score (MOS) and comparative MOS (CMOS) [32], while a detailed description of the evaluation protocol and results is provided in Appendix B.

5 Results

5.1 Performance Evaluation

Table 1 summarizes the performance of our proposed method, the baselines, and ablated variants across both LibriTTS and GigaSpeech test sets. We report results using the objective and subjective metrics described in Section 4.4.

On the LibriTTS clean test set, our model achieves the best overall performance compared to the baselines across all metrics. It records the lowest WER (4.13) and MCD (4.02), while attaining the highest SIM (0.815) and MOS (4.02), outperforming both FluentSpeech and VoiceCraft despite being trained only on 245 hours of clean data. This demonstrates the effectiveness of our framework under clean conditions.

[5] `https://huggingface.co/facebook/hubert-large-ls960-ft`

[6] `https://github.com/microsoft/UniSpeech/tree/main/downstreams/speaker_verification`

Table 3: Ablation study of test-time training components in the Spectrogram Denoiser on the GigaSpeech test set.

Method	WER ↓	SIM ↑	MCD ↓	CMOS ↑
TTT for Spectrogram Denoiser	**16.85**	**0.711**	**5.92**	**0**
Replacing CE Loss with CTC Loss	18.76	0.696	5.98	−0.54
w/o Phoneme Classifier	20.77	0.704	5.97	−0.59

On the more challenging GigaSpeech test set, our method consistently surpasses FluentSpeech across all metrics and remains competitive with VoiceCraft, which benefits from large-scale pretraining on 10,000 hours of data. Specifically, our model achieves lower WER (16.88 vs. 17.88) and MCD (5.60 vs. 6.13), as well as higher MOS (3.87 vs. 3.69) compared to FluentSpeech. Compared to VoiceCraft, our system performs better in WER, MCD, and MOS, with only a slightly lower SIM (0.725 vs. 0.758). These results highlight the competitiveness of our approach even without large-scale pretraining. To support subjective evaluation, corresponding audio samples are available online[7].

To confirm the effect of test-time training (TTT), we conducted ablation studies by removing TTT from the duration predictor (DP), the spectrogram denoiser (SD), or both modules. Disabling TTT for the duration predictor increases MCD, likely due to unnatural speech rhythm that in turn degrades spectral quality. Removing TTT for SD leads to reduced speaker similarity, indicating that TTT enhances acoustic coherence. When TTT is removed entirely, performance drops across all metrics, underscoring the importance of test-time adaptation for robust speech editing in the wild.

5.2 Ablation Studies on Test-Time Training Components

We conduct ablation experiments on the GigaSpeech test set to examine the contribution of individual components within our instance-specific test-time training (TTT) framework, focusing on the duration predictor and the spectrogram denoiser.

5.2.1 Duration Predictor

Table 2 presents the results of TTT applied to the duration predictor. With TTT enabled for the duration predictor, the system achieves the best overall performance within this ablation setting, striking a consistent balance among WER, SIM, MCD, and CMOS. Removing mask-level duration loss (MDL) leads to only marginal changes in WER and SIM, but noticeably increases MCD ($5.64 \rightarrow 5.80$) and decreases CMOS ($0 \rightarrow -0.31$), suggesting a decline in spectral fidelity and perceived naturalness. Further removing sentence-level duration loss results in additional degradation in WER ($17.49 \rightarrow 17.91$) and MCD ($5.80 \rightarrow 6.08$), indicating that sentence-level control helps stabilize temporal alignment at a global level. These findings highlight the complementary effects of both loss terms in enhancing prosodic stability during TTT.

5.2.2 Spectrogram Denoiser

As shown in Table 3, we further investigate the contributions of the spectrogram denoiser components in the TTT framework. Replacing the cross-entropy (CE) loss with connectionist temporal classification (CTC) loss [33] leads to considerable degradation in WER ($16.85 \rightarrow 18.76$), SIM ($0.711 \rightarrow 0.696$), and CMOS ($0 \rightarrow -0.54$). Furthermore, removing the phoneme classification branch entirely results in even worse performance, with WER increasing to 20.77 and CMOS dropping to -0.59. These results highlight the importance of phoneme classification for intelligibility and acoustic consistency in real-world conditions.

5.3 Visualizations

To qualitatively analyze the effect of test-time training, we provide spectrogram visualizations. Figure 2 demonstrates the controllability of speech rate achieved by applying TTT to the duration predictor. The middle row corresponds to the original sentence duration, while the top and bottom

[7] `https://rlataewoo.github.io/ttt-editor`

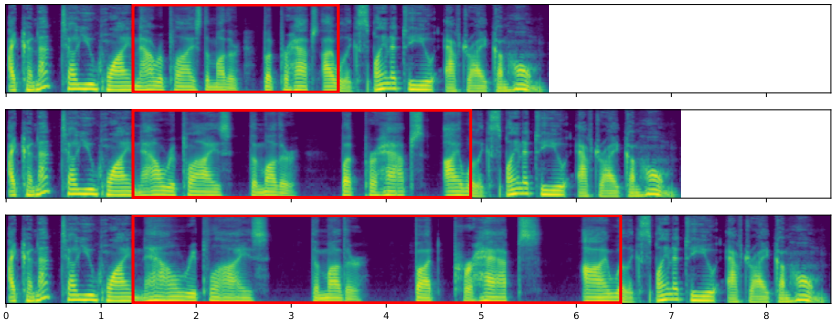
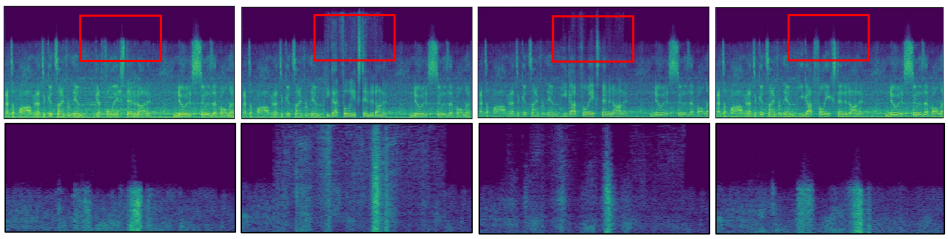


Figure 2: Mel-spectrograms of generated speech at different speech rates using TTT for the duration predictor. The middle row represents the original sentence duration, while the top and bottom rows show -20% and $+20\%$ adjustments, respectively. Red boxes indicate the edited regions.



(a) Ground Truth (b) FluentSpeech (c) Ours (w/o TTT) (d) Ours (w/ TTT)

Figure 3: Linear spectrograms of ground-truth and generated speech from different systems. The top panel shows the full spectrogram, while the bottom panel highlights the corresponding regions with red boxes.

rows represent -20% and $+20\%$ adjustments, respectively. We observe that TTT enables clear and consistent modifications of speech tempo, without requiring additional duration control modules as in prior approaches [34, 35]. These results confirm that TTT provides effective instance-specific prosodic control.

To further assess the effect of TTT, Fig. 3 illustrates its application to the spectrogram denoiser. In real-world scenarios, bandwidth mismatches between edited and unedited regions often result in perceptual discontinuities. Our method alleviates this issue by adapting the model to the acoustic conditions of the input, leading to smoother transitions and enhanced spectral coherence. As shown in Fig. 3(b), FluentSpeech exhibits prominent bandwidth mismatches at the edited regions, producing audible discontinuities. Ours without TTT (Fig. 3(c)) partially mitigates these artifacts, benefiting from the DiT-based denoiser which captures longer-range acoustic context compared to the WaveNet architecture used in FluentSpeech. Finally, with TTT applied (Fig. 3(d)), our model further adapts to input-specific conditions, resulting in the smoothest transitions and the most coherent spectral structure across the edited and unedited regions.

6 Conclusion

In this work, we introduce an instance-specific test-time training framework for speech editing under real-world conditions. Our method leverages ground-truth supervision from unedited regions together with auxiliary objectives in edited regions, enhancing both prosodic stability and acoustic consistency. The framework also enables fine-grained control of speech rate through duration adaptation, without requiring explicit duration control modules. Experiments on LibriTTS and GigaSpeech demonstrate that our approach consistently outperforms prior systems across objective and subjective metrics. These results highlight the effectiveness of test-time training for speech editing and demonstrate that, even with limited training data, our framework generalizes well to unseen and diverse conditions, indicating strong potential for real-world adoption.

Acknowledgments

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by Korea Government (MSIT) under Grant RS-2025-02215393 and No.2022-0-00963).

References

[1] Daxin Tan, Liqun Deng, Yu Ting Yeung, Xin Jiang, Xiao Chen, and Tan Lee. Editspeech: A text based speech editing system using partial inference and bidirectional fusion. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 626–633. IEEE, 2021.

[2] He Bai, Renjie Zheng, Junkun Chen, Mingbo Ma, Xintong Li, and Liang Huang. A^3T: Alignment-aware acoustic and text pretraining for speech synthesis and editing. In *Proceedings of the 39th International Conference on Machine Learning*, pages 1399–1411. PMLR, 2022.

[3] Ziyue Jiang, Qian Yang, Jialong Zuo, Zhenhui Ye, Rongjie Huang, Yi Ren, and Zhou Zhao. Fluentspeech: Stutter-oriented automatic speech editing with context-aware diffusion models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11655–11671, 2023.

[4] Puyuan Peng, Po-Yao Huang, Shang-Wen Li, Abdelrahman Mohamed, and David Harwath. Voicecraft: Zero-shot speech editing and text-to-speech in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12442–12462, 2024.

[5] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, et al. Voicebox: Text-guided multilingual universal speech generation at scale. *Advances in neural information processing systems*, 36:14005–14034, 2023.

[6] Xiaofei Wang, Manthan Thakker, Zhuo Chen, Naoyuki Kanda, Sefik Emre Eskimez, Sanyuan Chen, Min Tang, Shujie Liu, Jinyu Li, and Takuya Yoshioka. Speechx: Neural codec language model as a versatile speech transformer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.

[7] Guillermo Cámbara, Patrick Lumban Tobing, Mikolaj Babianski, Ravichander Vipperla, Duo Wang, Ron Shmelkin, Giuseppe Coccia, Orazio Angelini, Arnaud Joly, Mateusz Lajszczak, et al. Mapache: Masked parallel transformer for advanced speech editing and synthesis. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10691–10695. IEEE, 2024.

[8] Chengzhu Yu, Heng Lu, Na Hu, Meng Yu, Chao Weng, Kun Xu, Peng Liu, Deyi Tuo, Shiyin Kang, Guangzhi Lei, Dan Su, and Dong Yu. Durian: Duration informed attention network for speech synthesis. In *Interspeech 2020*, pages 2027–2031, 2020.

[9] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

[10] Myeonghun Jeong, Minchan Kim, Joun Yeop Lee, and Nam Soo Kim. Efficient parallel audio generation using group masked language modeling. *IEEE Signal Processing Letters*, 2024.

[11] Sanyuan Chen, Chengyi Wang, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. Neural codec language models are zero-shot text to speech synthesizers. *IEEE Transactions on Audio, Speech and Language Processing*, 2025.

[12] Francis Charpentier and M Stella. Diphone synthesis using an overlap-add technique for speech waveforms concatenation. In *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 11, pages 2015–2018. IEEE, 1986.

[13] Thierry Dutoit, Vincent Pagel, N. Pierret, F. Bataille, and O. Van der Vrecken. The mbrola project: towards a set of high quality speech synthesizers free of use for non commercial purposes. In *4th International Conference on Spoken Language Processing (ICSLP 1996)*, pages 1393–1396, 1996.

[14] Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE TRANSACTIONS on Information and Systems*, 99(7):1877–1884, 2016.

[15] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020.

[16] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021.

[17] David Osowiechi, Gustavo A Vargas Hakim, Mehrdad Noori, Milad Cheraghalikhani, Ismail Ben Ayed, and Christian Desrosiers. Tttflow: Unsupervised test-time training with normalizing flow. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2126–2134, 2023.

[18] Changhun Kim, Joonhyung Park, Hajin Shim, and Eunho Yang. Sgem: Test-time adaptation for automatic speech recognition via sequential-level generalized entropy minimization. In *Interspeech 2023*, pages 3367–3371, 2023.

[19] Sunwoo Kim and Minje Kim. Test-time adaptation toward personalized speech enhancement: Zero-shot learning with knowledge distillation. In *2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 176–180. IEEE, 2021.

[20] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*, page 125, 2016.

[21] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[22] Rongjie Huang, Zhou Zhao, Huadai Liu, Jinglin Liu, Chenye Cui, and Yi Ren. Prodiff: Progressive fast diffusion model for high-quality text-to-speech. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2595–2605, 2022.

[23] Yi Ren, Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Revisiting over-smoothness in text to speech. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8197–8213, 2022.

[24] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. Libritts: A corpus derived from librispeech for text-to-speech. In *Interspeech 2019*, pages 1526–1530, 2019.

[25] Guoguo Chen, Shuzhou Chai, Guan-Bo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, Mingjie Jin, Sanjeev Khudanpur, Shinji Watanabe, Shuaijiang Zhao, Wei Zou, Xiangang Li, Xuchen Yao, Yongqing Wang, Zhao You, and Zhiyong Yan. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. In *Interspeech 2021*, pages 3670–3674, 2021.

[26] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech 2017*, pages 498–502, 2017.

[27] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 33:17022–17033, 2020.

[28] Yongmao Zhang, Jian Cong, Heyang Xue, Lei Xie, Pengcheng Zhu, and Mengxiao Bi. Visinger: Variational inference with adversarial learning for end-to-end singing voice synthesis. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7237–7241. IEEE, 2022.

[29] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460, 2021.

[30] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[31] Robert Kubichek. Mel-cepstral distance measure for objective speech quality assessment. In *Proceedings of IEEE pacific rim conference on communications computers and signal processing*, volume 1, pages 125–128. IEEE, 1993.

[32] Philipos C Loizou. Speech quality assessment. In *Multimedia analysis, processing and communications*, pages 623–654. Springer, 2011.

[33] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[34] Taewoo Kim, Choongsang Cho, and Young Han Lee. Masked duration model for utterance duration-controllable text-to-speech. *IEEE Access*, 12:136313–136318, 2024.

[35] Johanes Effendi, Yogesh Virkar, Roberto Barra-Chicote, and Marcello Federico. Duration modeling of neural tts for automatic dubbing. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8037–8041. IEEE, 2022.

A Model Configuration

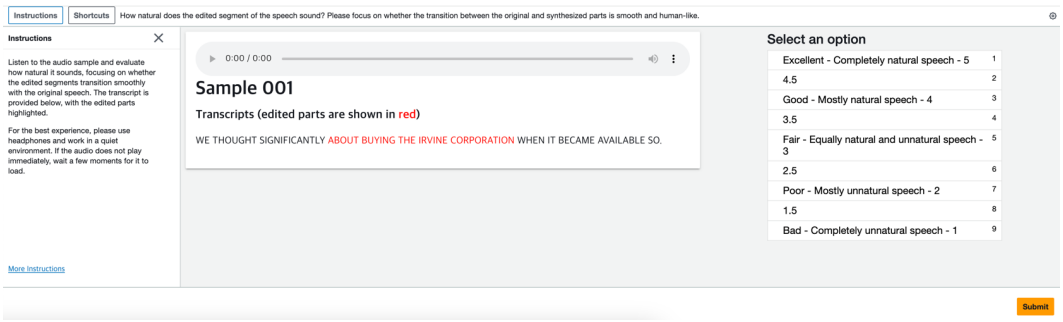
Table 4: Hyperparameters of the proposed model.

Layer	Hyperparameter	Setting
Text Encoder	Phoneme Embedding	192
	Encoder Layers	4
	Encoder Hidden	192
	Encoder Heads	2
	Encoder Conv1D Kernel	5
	Encoder Conv1D Filter Size	768
	Encoder Dropout	0.0
Duration Predictor	Predictor Conv1D Kernel	5
	Predictor Layers	3
	Predictor Conv1D Filter Size	192
	Predictor Dropout	0.2
Pitch Predictor	Predictor Conv1D Kernel	5
	Predictor Layers	5
	Predictor Conv1D Filter Size	192
	Predictor Dropout	0.2
Mel Encoder	Encoder Hidden	192
Spectrogram Denoiser	Diffusion Embedding	384
	DiT Blocks	12
	Denoiser Hidden	384
	Denoiser Heads	6
	Denoiser MLP Hidden	1536
	Denoiser Dropout	0.1
Phoneme Classifier	Classifier Layers	2
	Classifier Hidden	256
	Classifier Heads	2
	Classifier Conv1D Kernel	3
	Classifier Conv1D Filter Size	1024
	Classifier Dropout	0.5
Total Number of Parameters		45.9M

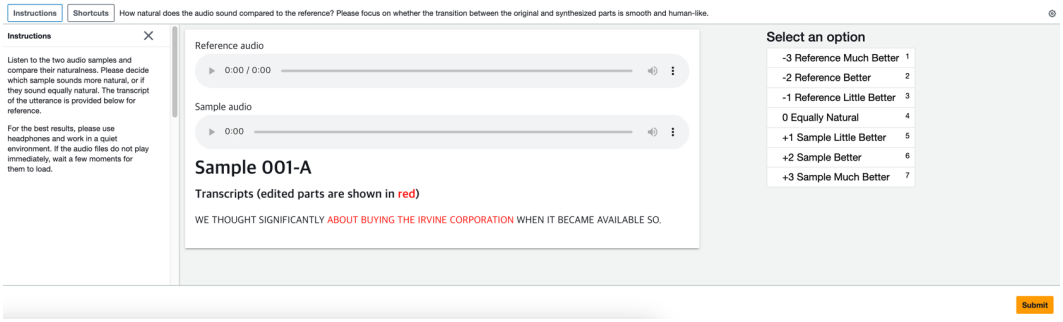
B Subjective Evaluation

We conducted subjective evaluation using Amazon Mechanical Turk (MTurk) [8], in terms of both mean opinion score (MOS) and comparative MOS (CMOS). All audio samples were resampled to 22.05 kHz for evaluation. We recruited 40 U.S.-based crowd workers for each test, ensuring that each participant was instructed to submit ratings independently in a quiet environment using headphones. For the MOS test, participants rated the naturalness of each audio sample on a five-point Likert scale with 0.5 increments ranging from 1 (completely unnatural) to 5 (completely natural). As illustrated in Figure 4(a), the transcript was provided alongside the audio, with edited segments highlighted in red, and workers were instructed to pay particular attention to the smoothness of transitions between original and synthesized segments. For the CMOS test, workers were presented with a pair of utterances: a reference audio and a corresponding sample audio, and they were asked to compare the naturalness of the two, focusing on the seamlessness of transitions between original and synthesized parts on a scale ranging from -3 (reference much better) to $+3$ (sample much better), with 0 indicating equal naturalness, as shown in Figure 4(b).

[8] `https://www.mturk.com/`



(a) Mean opinion score (MOS) instruction.



(b) Comparative mean opinion score (CMOS) instruction.

Figure 4: Instruction interfaces for subjective evaluation tasks.

C Limitations and Future Work

Despite the promising results, our framework has several limitations that warrant further investigation. First, test-time training (TTT) introduces additional computational overhead, which obstructs real-time deployment in latency-sensitive scenarios. Moreover, while our method demonstrates robustness across diverse conditions, performance degradation may still occur under extreme noise, highly divergent accents, or when only very limited unedited regions are available for supervision. Finally, controllability in our system is primarily restricted to speech rate, leaving other prosodic factors less explored.

Future work will focus on addressing these challenges. One promising direction is the development of parameter-efficient or meta-learning-based TTT approaches that enable faster adaptation suitable for real-time applications. Extending controllability beyond duration to aspects such as pitch, energy, and style is another important avenue. Furthermore, exploring cross-lingual and low-resource scenarios will broaden the applicability of our framework.

D Broader Impacts

The proposed test-time training framework for speech editing has the potential to substantially impact both research and real-world applications. By enhancing prosodic stability and acoustic consistency in edited speech, it enables more natural and controllable audio synthesis for content creation, personalized media production, and accessibility technologies. Moreover, its ability to adapt to unseen conditions without large-scale retraining makes it suitable for deployment in diverse scenarios, including low-resource settings where data collection is challenging.

On the other hand, the technology carries risks of misuse, particularly in creating deceptive or harmful audio such as deepfakes. To mitigate these concerns, it is essential to develop safeguards including provenance tracking, watermarking, and detection systems, and to communicate transparently about the framework's capabilities and limitations. Overall, this work highlights both the societal benefits of more robust and controllable speech editing and the importance of proactive measures to ensure its ethical and responsible deployment.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately describe the main contribution: improving speech editing in the wild through instance-specific test-time training. These claims are supported by the experimental results in Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper explicitly discusses limitations in Appendix C, including additional computational overhead for test-time training, performance degradation under extreme noise or heavy accents, and limited controllability to speech rate.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not provide formal theoretical results or proofs; it focuses on model design and empirical validation.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The datasets, model architecture, training settings, and evaluation protocols are described in detail (Sections 3–4, Appendix A). Public code repositories (HiFi-GAN, FluentSpeech) are also referenced.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The proposed model and code are not publicly released. Baseline implementations (HiFi-GAN, FluentSpeech) are used from official repositories, but our framework is not open-sourced at this stage.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed training and test configurations, including hyperparameters, optimizer settings, masking ratios, GPU type, and training iterations, are provided in Section 4 and Appendix A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Subjective MOS scores are reported with 95% confidence intervals.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The compute resources are specified, including a single NVIDIA A40 GPU, training iterations (700K), parameter counts (46M), and batch size (32).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research follows the NeurIPS Code of Ethics in all respects.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The broader societal impacts, including both positive and negative aspects, are discussed in Appendix D.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [Yes]

Justification: Safeguards such as watermarking, provenance tracking, and detection mechanisms are discussed as essential for responsible use.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets (LibriTTS, GigaSpeech) and external codebases (HiFi-GAN, VoiceCraft, FluentSpeech) are properly cited with references and usage conditions.

Guidelines:

- The answer NA means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new datasets are introduced.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: Appendix B provides full evaluation instructions, screenshots, and details on the Amazon Mechanical Turk setup for subjective evaluations.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: The listening test does not involve sensitive data collection and does not require IRB approval.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: No large language models were used as an essential or original part of the core methodology.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.