ANYEDIT: A Unified Framework for Speech and Singing Voice Editing with Real-World Environmental Consistency

Yunjia Zhang[1]*, **Junan Zhang**[1]*, **Jing Yang**[2]*, **Xueyao Zhang**[1], **Fan Fan**[2], **Zhizheng Wu**[1,3,4]†
[1]The Chinese University of Hong Kong, Shenzhen
[2]Institute of Media Technology and Experience Design, Huawei
[3]City University of Macau
[4]Amphion Technology Co., Ltd.

Abstract

We present ANYEDIT, a unified, separation-free framework for speech and singing voice editing that preserves real-world acoustic environments, including noise, reverberation, and background music. Recent unified generative models already achieve strong speech and singing generation in clean conditions, but real-world vocal editing remains challenging because the model must revise content while simultaneously preserving speaker identity, prosody, and the surrounding acoustic scene. A straightforward solution is to directly model noisy edited audio end to end, yet this entangles semantic editing with environmental acoustics and can weaken the core vocal generation capability. We therefore adopt a two-stage design that preserves clean-domain editing ability while deferring acoustic rendering to a separate stage: an instruction-guided autoregressive editor first predicts clean prosody and content-style tokens, supported by a teacher-student noise-robust prosody tokenizer that distills clean prosodic tokens from degraded recordings; a flow-matching acoustic model then performs in-context rendering conditioned on the source acoustic context, so the edited segment inherits surrounding noise, reverberation, and background sound while maintaining consistent speaker timbre. To stabilize unedited regions, we introduce unmodified-area-aware supervision; after pre-training, we further apply direct preference optimization (DPO) on mined real-to-real singing pairs via Chromagram-DTW. We also release ANYEDITBENCH, a benchmark covering both speech and singing voice editing under clean, simulated, and in-the-wild acoustic conditions. Experiments on Chinese and English speech and singing datasets showed that ANYEDIT consistently outperforms strong baselines in content accuracy, subjective quality, and background consistency under noisy and reverberant conditions. Demo audios can be found at `https://any-edit.github.io/`. ANYEDITBENCH is released at `https://huggingface.co/datasets/Ririiii/anyeditbench`.

1 Introduction

Audio synthesis technologies, such as text-to-speech (TTS) and text-to-singing (TTSing), have achieved remarkable progress in generating high-quality vocals from scratch. Recent unified generative models, such as Vevo2 [1], can simultaneously handle speech and singing generation with impressive fidelity under clean conditions. However, in practical audio post-production for user-generated content (UGC), podcasts, and music, creators frequently need to perform zero-shot localized

*Equal contribution.
†Corresponding author. E-mail: yunjiazhang2@link.cuhk.edu.cn, wuzhizheng@cuhk.edu.cn

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

editing on an existing recording—such as inserting, deleting, or substituting words and lyrics (e.g., correcting mispronunciations or updating lyrics)—rather than synthesizing a new one entirely. In real-world scenarios, the original audio is rarely clean; vocals are inevitably entangled with complex background acoustics, including noise, reverberation, and background music.

Existing approaches to real-world vocal editing fall into two paradigms, neither of which fully addresses this challenge. **The first paradigm is direct end-to-end editing**, which couples semantic editing and acoustic-scene reproduction within a single generative stage. Speech editors such as VoiceCraft [2] and Step-Audio-EditX [3], singing editors such as EditSinger [4] and SongEditor [5], and general-purpose unified audio models such as Ming-UniAudio [6] and MiMo-Audio [7] all follow this approach. Since paired (noisy source, noisy target) supervision is essentially unobtainable at scale, these methods inherit a clean-domain prior and lose intelligibility, speaker fidelity, or background continuity on real-world inputs; the unified models additionally lack localized editing support. **The second paradigm is the "separation-edit-mix" pipeline**, which isolates the clean vocal via source separation, edits it with TTS/TTSing, and remixes it with the background. This pipeline introduces three intrinsic issues, one per stage: (1) **Separation artifacts**, especially severe for singing where the background is harmonically correlated music rather than uncorrelated noise [8]; (2) **Static background modeling**, as methods such as SeamlessEdit [9] (which directly sums the edited vocal with the separated background) and DAIEN-TTS [10] (which conditions synthesis on a single global SNR) cannot follow time-varying background dynamics such as fade-ins, fade-outs, or acoustic gaps; and (3) **Lack of true localized editing**, since the edit step is typically realized by TTS-style models that regenerate the entire utterance rather than modifying only the target span—SSR-Speech [11], for instance, regenerates unedited regions instead of keeping them, failing to maintain the fine-grained acoustic context of the original recording. As a result, unified speech and singing editing under realistic acoustics remains insufficiently explored; we provide a comprehensive survey in Appendix A.

This raises a central question: ***can a single framework achieve speech and singing voice editing that is unified across both domains, noise-robust on the input side, and environment-preserving on the output side?***

We propose ANYEDIT, a separation-free two-stage framework that addresses this question. **Our key insight is that semantic editing (what to say) and acoustic rendering (how it sounds in context) are fundamentally separable**: semantic editing benefits from clean-domain training, while acoustic rendering only requires faithful in-context reproduction. This decoupling lets ANYEDIT inherit the editing precision of clean-trained unified models while avoiding source separation entirely. The framework realizes this insight through three coordinated stages. **(1) Noise-Robust Prosody Tokenizer.** A teacher-student tokenizer distills clean prosody tokens from noisy chromagrams, providing stable prosodic conditions for the downstream editor. **(2) Instruction-Guided Autoregressive (AR) Editor.** The AR editor predicts clean prosody and content-style tokens from either target text or natural-language instructions, with *unmodified-area-aware supervision* that upweights the loss on prosody tokens within unchanged spans to prevent prosody drift. We further refine singing quality via Direct Preference Optimization (DPO) on mined *real-to-real* pairs from music structural repetitions (e.g., different verses sharing the same prosody but different lyrics), avoiding the synthesis artifacts that SVS-generated targets would otherwise inject into the editor. **(3) Flow-Matching Acoustic Model (FMT).** A flow-matching transformer renders the target mel-spectrogram conditioned on the source acoustic context, so the edited segment inherits surrounding noise, reverberation, and background music while preserving speaker timbre.

To support realistic evaluation, we introduce ANYEDITBENCH, a benchmark for real-world vocal editing comprising 300 speech and 300 singing entries across diverse acoustic scenarios (clean, simulated, and in-the-wild recordings). Extensive subjective and objective experiments on Chinese and English datasets demonstrate that ANYEDIT consistently outperforms strong baselines in content accuracy, subjective quality, and background continuity. Our main contributions are summarized as follows:

- We propose ANYEDIT, the first unified framework for speech and singing voice editing in real-world acoustic environments. By decoupling semantic editing from acoustic rendering, ANYEDIT combines a noise-robust semantic stage with an environment-aware in-context rendering stage, eliminating the need for source separation.
- We introduce a suite of separation-free techniques: (i) a noise-robust prosody tokenizer via teacher-student distillation; (ii) an instruction-guided AR editor with *unmodified-area-aware*

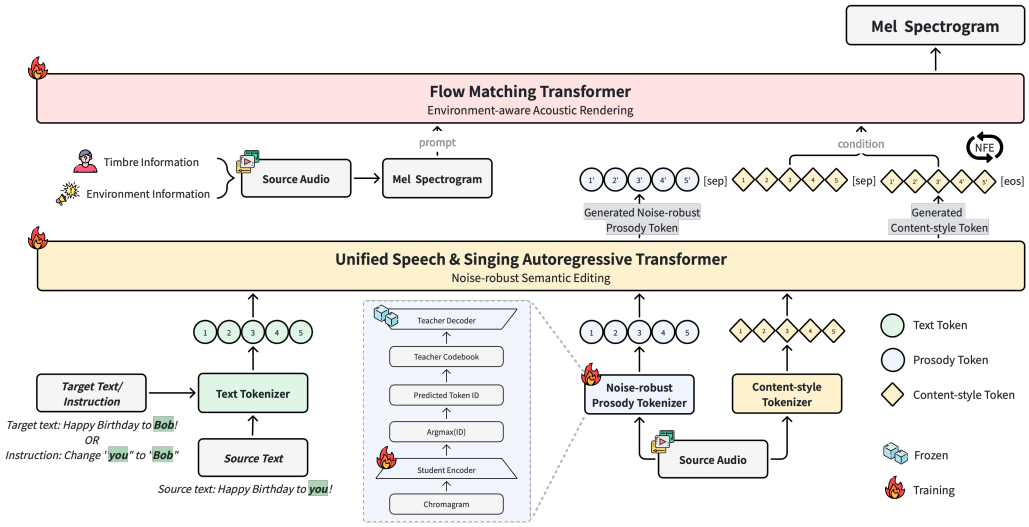

Figure 1: Overall architecture of the ANYEDIT framework. NFE indicates the Number of Function Evaluations (ODE integration steps) used during flow matching inference.

supervision; (iii) DPO alignment on *mined real-to-real singing pairs* via Chromagram-DTW matching of repeated song structures; and (iv) an in-context flow-matching acoustic renderer.

- We establish ANYEDITBENCH, the first benchmark for real-world vocal editing in Chinese and English across both speech and singing under clean, simulated, and in-the-wild conditions, on which ANYEDIT achieves state-of-the-art performance against existing baselines under noisy and reverberant settings in content accuracy, speaker fidelity, and background consistency

2 Method

We formulated real-world speech and singing editing as a conditional generation problem: given a source recording, its transcript, and either a target transcript or an instruction, the system synthesized target vocals that precisely incorporated the semantic edits while preserving the unedited vocal context and original acoustic environment. As shown in Figure 1, ANYEDIT achieved this via a separation-free, cascaded architecture: (1) an AR model first generated discrete prosody tokens and content-style tokens in a clean latent space, (2) followed by a FMT that rendered the target mel-spectrogram conditioned on the original timbre and acoustic context.

2.1 Noise-Robust Prosody Tokenizer

Real-world recordings often contain noise, reverberation, and background music, making prosody extraction unreliable. To provide stable prosodic conditions for both speech and singing editing, we built a noise-robust prosody tokenizer. Rather than relying on explicit source separation, which typically introduces artifacts and disrupts the acoustic correlation between vocals and background, our model directly extracts the prosody of the clean vocal content from the input recordings.

Representation Backbone. We took chromagrams as the tokenizer input. Chromagrams preserve melodic and rhythmic structures while being relatively insensitive to timbre variation and background interference, making them highly suitable for the unified modeling of speech and singing prosody under degraded conditions.

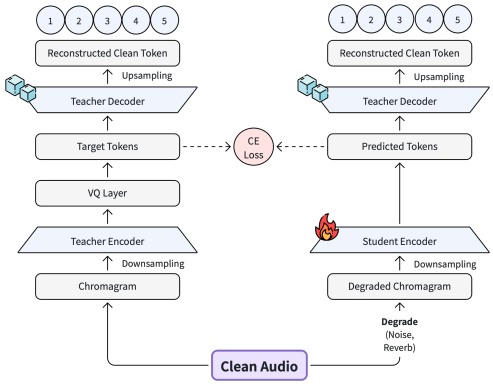

Figure 2: Training pipeline of the Noise-Robust Prosody Tokenizer via Teacher-Student Distillation.

Teacher-Student Architecture. We trained the tokenizer using teacher-student distillation (Figure 2). The teacher tokenizer Q_{tea} was a VQ-VAE trained on clean audio. Given clean chromagram features $\mathbf{c}_{\text{clean}}$, it produced discrete prosody tokens z^{tea}. The student model f_ϕ shared the same architecture but took degraded chromagrams $\mathbf{c}_{\text{degraded}}$ as input and was trained to recover the teacher's clean tokens. The degradation process followed the pipeline of AnyEnhance [12], introducing simulated corruptions of noise and reverberation.

The teacher followed a standard VQ-VAE design [1]: it included a downsampling layer, a convolutional encoder, a vector quantization (VQ) module with codebook $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_K]$, a decoder, and an upsampling layer. The VQ operation mapped encoded features $\mathbf{z}_e(\mathbf{x})$ to the nearest codebook vector $\mathbf{z}_q(\mathbf{x}) = \arg\min_j \|\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_j\|_2$. We extracted 50 Hz chromagram features from raw audio and applied $8\times$ downsampling, yielding a 6.25 Hz discrete sequence.

To ensure robustness, we trained the student model f_ϕ via a cross-entropy objective $\mathcal{L}_{\text{tokenizer}} = \sum_{t=1}^{T} \text{CE}(\hat{z}_t, z_t^{\text{tea}})$, where $z^{\text{tea}} = Q_{\text{tea}}(\mathbf{c}_{\text{clean}})$ and $\hat{z} = f_\phi(\mathbf{c}_{\text{degraded}})$. This distillation objective forced the student to recover clean prosody tokens from degraded inputs. While the teacher tokenizer applied to degraded inputs achieved a token match rate of only 0.6081, our student improved this to 0.7037, demonstrating improved noise-robust prosody recovery.

2.2 Autoregressive Editing Model (AR)

The AR model served as the clean semantic stage of ANYEDIT. It predicted two streams of discrete tokens: prosody tokens for encoding rhythm and temporal variations, and content-style tokens for encoding linguistic content and vocal styles.

For the content-style tokenizer, we adopt the RepCodec structure [1], which quantizes continuous self-supervised speech representations into discrete tokens that capture fine-grained phonetic and stylistic nuances while stripping out speaker timbre.

Training Strategy. The AR model was trained in three stages: Phase I pre-training, Phase II pre-training, and Direct Preference Optimization (DPO). Since directly learning to predict prosody without time alignment from scratch prevented convergence, we adopted a two-phase curriculum before DPO:

- **Phase I (Content-Style only):** The model predicted only content-style tokens, learning a stable time alignment with the input audio and the mapping from text to vocal content and style.
- **Phase II (Content-Style + Prosody):** Building upon the alignment from Phase I, the model jointly predicted content-style and prosody tokens, enabling successful prosody prediction without being fully bound to the original time alignment.

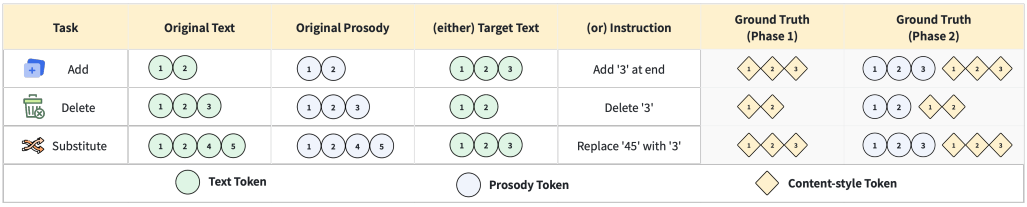

Figure 3: AR model pretraining data construction and training process

Pre-training Data Construction. To scale training without manual annotation, we constructed paired editing data using CTC forced aligner [3]. We utilized the Emilia dataset [13] for speech and SingNet [14] for singing. For each source utterance, we retrieved a reference sample from the same speaker following the pipeline of MimicLM [15]. We then synthesized insertion, deletion, and substitution edits at the beginning, middle, or end of the utterance. In total, we processed approximately 1.2 million same-speaker speech pairs for Chinese and English, yielding about 3,000 hours per language, and roughly 2,000 hours of singing paired data per language. The training process is showed in Figure 3.

[3] https://github.com/MahmoudAshraf97/ctc-forced-aligner

Model Architecture and Conditioning During training, the target text was used in 50% of cases, with a natural-language instruction used in the remaining 50%, allowing the model to perform editing flexibly from either explicit target text or free-form instructions. Detailed Implementations are in Appendix D.

2.2.1 Unmodified-Area-Aware Supervision

A common error in AR-based vocal editing is prosody drift, where unedited words around the edited span change in timing or intonation. To reduce this effect, we introduced *unmodified-area-aware supervision*.

During training, we computed the Longest Common Subsequence (LCS) between the source and target texts to identify unchanged spans. For contiguous unchanged spans longer than $L_{\min} = 4$, we mapped the corresponding text spans to their temporal positions in the source prosody-token sequence. Then we upweighted the negative log-likelihood of tokens in these mapped regions with a penalty factor $\lambda_{\text{keep}} = 5.0$:

$$\mathcal{L}_{\text{AR}} = -\sum_{t=1}^{T} w_t \log P_\theta(y_t | y_{<t}, \mathbf{X}), \tag{1}$$

where $w_t = \lambda_{\text{keep}}$ if step t belonged to a kept span, and $w_t = 1.0$ otherwise. This encouraged the model to preserve the prosody of unedited content and significantly improved boundary stability during singing editing.

2.2.2 Preference Alignment via mined Real-to-Real pairs

A key difficulty in singing voice editing is obtaining paired data in which the source and target share timbre, rhythm, but differ in lyrics. Existing approaches construct such pairs using singing voice synthesis (SVS) models, but synthetic targets inevitably introduce artifacts, biasing the editing model toward synthetic acoustic patterns. See Table 1 for a comparison with data construction pipelines for singing voice editing.

Table 1: Comparison of data construction pipelines for singing voice editing. **Target Realness**: Whether the ground-truth target audio is a natural human recording. **Melodic Match**: Whether the target strictly follows the source melody contour.

Method	Construction Protocol	Target Realness	Melodic Match	Scalability	Primary Limitation
Web Crawled Hand-crafted SVS	Content creators manually tune synthesis software parameters to align specific timbres with different lyrics.	✗	✓	✗	Extremely labor-intensive; dependent on human tuning skills; limited data volume.
SVS Model Generation	Synthesizes target audio from lyrics using a trained SVS model (Synthetic Target).	✗	✓	✓	Model learns *synthetic artifacts* and robotic prosody; upper bound limited by the SVS teacher.
Random Truncation	Randomly masks/cuts audio segments for reconstruction (pre-training style).	✓	✗	✓	Lacks specific lyrical control; cannot guarantee melodic preservation when lyrics change.
Mined Real-to-Real Pairing (Ours)	**Mines intra-song repetitions (Segment A/B) via Chromagram DTW alignment.**	✓	✓	✓	Requires songs with repetitive structures (common in pop music).

To overcome the limitations of synthetic targets, our goal is to acquire Real-to-Real paired data. We draw inspiration from the inherent compositional structure of popular music, which frequently features repeated sections—such as different verses—where the same singer performs an identical melody with varying lyrics. Leveraging this property, we automatically mine pairs of real vocal segments from the same song that share the identical singer, timbre, and melody, but differ only in textual content. Since both the source and target in these pairs are natural recordings, they better match the target distribution of practical singing editing. The specific extraction and alignment pipeline is detailed as follows:

Segment Extraction. Vocal segments are extracted using structural metadata (e.g., verse and chorus tags) and utterance-level timestamps available in music streaming corpora, discarding segments shorter than 2.0 seconds. This structural alignment eliminates the need for external music segmentation models while providing scalable paired data. Given a song with timestamped lyrics, we extract vocal segments $S = \{s_1, s_2, \ldots, s_N\}$ and discard segments shorter than 2.0 seconds.

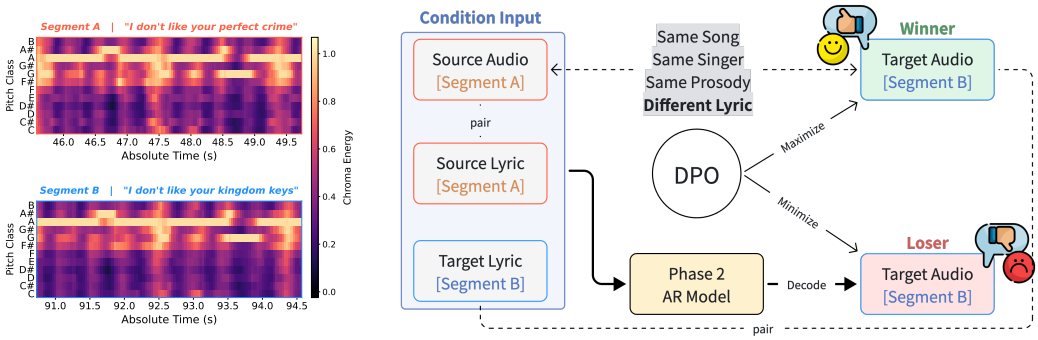

Figure 4: Example of a Real-to-Real mined singing pair (left) and the DPO training process (right). Different lyrics ("I don't like your perfect crime" vs. "I don't like your kingdom keys")in the same song of Look what you made me do by Taylor Swift. The song pair is only used for visualization for research purpose.

Feature Alignment via DTW. For each pair (s_i, s_j) from the same song, we extract Chroma CQT features and compute a normalized DTW distance $D(s_i, s_j) = \mathrm{DTW}(\mathrm{CQT}(s_i), \mathrm{CQT}(s_j))/\max(|s_i|, |s_j|)$, where $|s|$ denotes the sequence length.

Filtering Criteria. A pair is retained only if: (1) Lyrical Difference: $T_i \neq T_j$; (2) Duration Consistency: duration ratio difference within 30%; (3) Melodic Similarity: normalized DTW distance below threshold $\tau < 0.025$.

During DPO training, we use s_i as the source condition and T_j as the target text, while the real segment s_j provides the preferred response y^+. The rejected response y^- is generated by the Phase 2 AR model. We optimize the policy model π_θ against a frozen reference model π_{ref}:

$$\mathcal{L}_{\text{DPO}} = -\log \sigma\Big(\beta\big[\log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\big]\Big). \tag{2}$$

We construct more than 300 hours of English and about 250 hours of Chinese Real-to-Real singing pairs, corresponding to roughly one-tenth of the original song collection.

2.3 Flow-Matching Acoustic Model

The second stage utilizes a Flow Matching Transformer (FMT) [16] to convert discrete tokens predicted by the AR model into continuous mel-spectrograms. We formulate the acoustic rendering process as a conditional flow-matching problem, learning a continuous-time probability path $p_t(\mathbf{x})$ from a Gaussian prior to the target data distribution. We provide more details about flow matching in Appendix D.3.

The FMT is conditioned on the target content-style tokens, the prompt audio's content-style tokens, and the prompt's mel-spectrogram via bidirectional self-attention. To ensure high-fidelity vocal rendering while maintaining acoustic-scene continuity, we employ *Mixed-Condition Training*: during training, the target mel-spectrogram is clean in 50% of batches and corrupted with background acoustics in the remaining 50%, while the conditioning tokens are consistently extracted from clean vocals. This conditioning asymmetry forces the acoustic model to reconstruct environmental noise and reverberation purely from the continuous mel prompt, rather than relying on discrete semantic shortcuts.

To construct noisy training conditions, we mixed clean vocals with background audio from DNS Challenge [17], WHAM! [18], and AudioSet [19], following the degradation pipeline of AnyEnhance [12]. The SNR was uniformly sampled from $[-5, 20]$ dB, and reverberation with RT60 up to 1.5 s was simulated using synthetic and real room impulse responses (RIRs).

3 Experiments

We first describe the experimental setup, benchmark, and evaluation metrics (Section 3.1). We then present the unified speech and singing editing results of ANYEDIT in Section 3.2, focusing on its generalization to real-world scenarios (Table 2), performance across diverse simulated acoustics (Table 3), and robustness under extreme acoustic conditions (Table 4). Results on clean audio are in Appendix F.1 Finally, we conduct ablation studies on our core designs (Section 3.3, Table 5) and report subjective evaluation results (Table 6).

3.1 Experimental Settings

Baselines We compare ANYEDIT with a wide range of baselines: **1) Speech editing models**: SSR-Speech [11], VoiceCraft [2] (evaluated on English only), and Step-Audio-EditX [3]; **2) Singing voice editing models**: StyleSinger [20], TCSinger [21], and SoulX-Singer [22] (evaluated on Chinese only); **3) Unified generation models**: Ming-UniAudio [6], Mimo-Audio [7], and Vevo2 [1]. Notably, Vevo2 is the only baseline addressing both speech and singing voice modalities, serving as our primary unified reference.

Benchmark To rigorously assess generation and generalization capabilities, we evaluate on ANYEDITBENCH, a benchmark we constructed covering clean, simulated, and real-world speech and singing editing in both English and Chinese. It is organized into 12 subsets (3 subsets $\times$ 2 modalities $\times$ 2 languages), with 50 samples per subset, totaling 600 evaluation items. Full construction details are provided in Appendix C.1.

Metrics We evaluate our approach from multiple objective and subjective perspectives: **1) Content accuracy** via WER/CER using Whisper-large-v3 [23]; **2) Speaker similarity** (SIM-spk) via WavLM cosine similarity; **3) Melodic fidelity** for singing via log-F0 Pearson correlation (F0 Corr); **4) Acoustic environment fidelity** via CLAP background similarity [24] (SIM-bg), RT60Error, and SNRError. Subjective evaluation includes N-MOS (naturalness), S-MOS (similarity), and Fooling Rate. Full metric definitions are in Appendix C.2.

3.2 Main Results

To demonstrate the superior generalization of ANYEDIT, we focus our main analysis on in-the-wild recordings and complex acoustic simulations.

1) Real-World Generalization As shown in Table 2, ANYEDIT demonstrates state-of-the-art robustness on in-the-wild real-word recordings. For speech, ANYEDIT achieves the lowest English WER (0.0616) and highest SIM-spk (0.9675). More importantly, it retains a decisive background preservation advantage over the SOTA Step-Audio-EditX (English SIM-bg: 0.7705 vs. 0.6418). This confirms that avoiding source-separation pitfalls translates to tangible acoustic continuity in unpredictable wild environments. For real-world singing, ANYEDIT comprehensively dominates in text accuracy (Chinese CER: 0.0587 vs. 0.8919 for SoulX-Singer) and melodic fidelity, confirming robust lyrical control regardless of complex background music presence.

2) Performance in Simulated Acoustics Table 3 summarizes results averaged over 15 simulated conditions. Under additive noise and reverberation, the structural advantages of ANYEDIT become prominent. Compared to Step-Audio-EditX, ANYEDIT exhibits consistently higher background continuity (e.g., Chinese SIM-bg: 0.8374 vs. 0.4896). This advantage stems directly from our separation-free architecture, avoiding mechanical artifacts and phase distortions. For singing, ANYEDIT constantly outperforms Vevo2 across nearly all metrics, validating that our simulated training paradigm generalizes robustly across diverse acoustic conditions.

3) Robustness in Extreme Conditions To further push the limits of generalization, we compare ANYEDIT with Vevo2 under severe acoustic conditions (Table 4). Even at an extreme SNR of 0 dB or heavy reverberation (RT60=1.5 s), ANYEDIT achieves state-of-the-art robustness overall. It reduces SNR estimation errors, from 13.321 to 3.368 in speech SNR=0 dB and maintains high speaker similarity and background consistency in scenarios where Vevo2 struggles to generate coherent audio. Additionally, the clean subset results, showing ANYEDIT surpasses Vevo2 without being compromised by environment-aware training, are detailed in Appendix F.1. Note that while SSR-Speech occasionally shows competitive objective scores (e.g., in clean metrics, detailed in

Appendix F.1), it operates essentially as a full-utterance text-to-speech model rather than a localized editor, fundamentally altering unedited acoustic characteristics.

Table 2: **Objective results on the ANYEDITBENCH real-world subset** English uses WER, Chinese uses CER; F0 Corr is singing-only. Bold indicates best per domain; underlined entries indicate that ANYEDIT outperforms the unified baseline Vevo2 [1].

	Speech Editing						Singing Voice Editing					
Metric	SSR-Speech	Voice-Craft	Step-EditX	Vevo2	Ming-Uni	**ANYEDIT**	Style-Singer	TCSinger	SoulX-Singer	Vevo2	Ming-Uni	**ANYEDIT**
WER (en)↓	0.0827	0.1081	0.1706	0.0778	0.5385	**0.0616**	–	0.9971	–	0.2809	1.0000	**0.1749**
CER (zh)↓	0.1816	–	**0.1296**	0.1660	0.3000	0.1433	5.4133	0.3171	0.8919	0.1780	0.1538	**0.0587**
SIM-spk (en)↑	0.9652	0.9604	0.9369	0.9650	0.9632	**0.9675**	–	0.7082	–	0.8785	0.3009	**0.9167**
SIM-spk (zh)↑	0.8958	–	0.9054	0.9027	0.8085	**0.9108**	0.6745	0.7437	0.8942	0.8970	**0.9502**	0.9198
F0 Corr (en)↑	–	–	–	–	–	–	–	0.0217	–	0.2030	-0.1780	**0.2080**
F0 Corr (zh)↑	–	–	–	–	–	–	0.0217	0.0328	**0.4197**	0.3326	-0.0697	0.4027
SIM-bg (en)↑	**0.8729**	0.7039	0.6418	0.7030	0.6425	0.7705	–	0.3446	–	0.5671	0.4241	**0.7384**
SIM-bg (zh)↑	**0.8472**	–	0.6459	0.6798	0.8191	0.6818	0.3527	0.3367	0.5413	0.5294	0.6301	**0.6963**
RT60Err (en)↓	0.0276	0.0043	**0.0006**	0.0654	0.0009	0.0007	–	0.1971	–	0.1834	2.9556	**0.0256**
RT60Err (zh)↓	0.0149	–	0.0072	0.0072	**0.0002**	0.0075	0.0091	**0.0006**	0.1265	0.0674	–	0.0094
SNRErr (en)↓	3.5536	9.6702	15.5598	13.8125	**0.7558**	8.8241	–	46.1529	–	15.0969	38.5435	**6.1996**
SNRErr (zh)↓	6.8708	–	17.4128	16.9422	**5.9399**	12.3207	45.6697	40.9728	16.4016	14.0181	7.1816	**5.6354**

Table 3: **Objective results on the ANYEDITBENCH simulated subset (averaged over 15 conditions)** Layout same as Table 2.

	Speech Editing						Singing Voice Editing					
Metric	SSR-Speech	Voice-Craft	Step-EditX	Vevo2	Ming-Uni	**ANYEDIT**	Style-Singer	TCSinger	SoulX-Singer	Vevo2	Ming-Uni	**ANYEDIT**
WER (en)↓	0.1085	0.2173	0.1618	0.0686	**0.0588**	0.0770	–	1.2480	–	0.2579	0.5500	**0.2368**
CER (zh)↓	1.0317	–	**0.0875**	0.1031	0.1163	0.1184	0.7595	0.6019	0.9028	0.3065	0.6250	**0.2975**
SIM-spk (en)↑	**0.9660**	0.9383	0.9363	0.9296	0.9625	0.9535	–	0.5530	–	0.9015	0.8395	**0.9223**
SIM-spk (zh)↑	0.9056	–	0.8840	0.9164	0.9136	**0.9328**	0.7093	0.7394	0.8723	0.8838	0.7996	**0.9014**
F0 Corr (en)↑	–	–	–	–	–	–	–	-0.0221	–	0.2932	0.2436	**0.3014**
F0 Corr (zh)↑	–	–	–	–	–	–	0.0047	0.0135	**0.4363**	0.3152	0.0189	0.3281
SIM-bg (en)↑	**0.8409**	0.6808	0.5059	0.5445	0.8292	0.7787	–	0.3301	–	0.5943	0.2380	**0.8190**
SIM-bg (zh)↑	0.8187	–	0.4896	0.6760	0.7379	**0.8374**	0.4439	0.4436	0.6997	0.6793	0.4068	**0.8341**
RT60Err (en)↓	0.1204	0.0008	**0.0006**	0.0142	0.0007	0.0021	–	0.0436	–	0.0262	–	**0.0256**
RT60Err (zh)↓	0.4526	–	**0.0854**	0.0908	0.0952	0.1637	0.0533	0.0007	0.1626	0.1398	–	**0.0004**
SNRErr (en)↓	**1.7351**	5.9261	20.8958	12.8442	2.4145	5.1518	–	38.1293	–	8.3918	6.0304	**3.1332**
SNRErr (zh)↓	3.7138	–	19.1547	7.4499	26.3580	**1.7669**	38.9281	33.2150	7.5001	8.8160	**3.2311**	4.1795

3.3 Ablation Study

In Table 5, we explore the impact of several core designs of ANYEDIT, evaluated on a mixed speech-and-singing subset at 10 dB SNR. **1) Noise-Robust Tokenizer:** Replacing this module with a standard VQ tokenizer trained only on clean data severely degrades intelligibility (WER jumps from 0.042 to 0.145). This confirms its necessity for extracting stable semantic features under noisy conditions. As analyzed in Appendix F.5, our teacher-student distillation achieves significantly higher token match rates across all SNR levels compared to direct continuous spectral supervision, ensuring reliable semantic conditioning under adverse acoustic environments. **2) Training Data Modality:** Relying exclusively on speech data destroys the model's ability to render melodies (F0 correlation drops to 0.181). Conversely, training only on singing data worsens pronunciation accuracy. This proves that our unified training strategy is essential for cross-domain generation. **3) Staged AR Training:** Omitting Phase 1 pre-training causes near-total model collapse in background similarity (SIM-bg: 0.378) and melodic fidelity. Skipping Phase 2 heavily degrades textual alignment. Furthermore, removing unmodified-area-aware supervision mechanism reduces the model's ability to perfectly preserve the original prosody in unedited regions. Detailed results on the Unmodified Area Fidelity, demonstrating how this mechanism effectively prevents prosody drift and preserves original intonation, are provided in Appendix F.7. **4) Preference Alignment:** Removing DPO alignment and replacing our mined Real-to-Real pairs with synthesized targets yield similarly poor results. The no-DPO variant suffers from a clear drop in melodic fidelity, while the synthetic-DPO variant further degrades speech quality and intelligibility. This near-equivalence suggests that the acoustic artifacts in synthesized speech mislead preference optimization, rendering DPO almost as ineffective as having no alignment at all. It strongly supports our hypothesis that synthetic targets introduce detrimental artifacts, making natural human vocal pairs indispensable for effective preference learning. **5) Noisy Acoustic Rendering:** Training the Flow Matching Transformer solely on clean targets causes a failure in background consistency. This highlights that simulated noisy rendering is indispensable for seamless editing.

Table 4: **Comparison of ANYEDIT and Vevo2 [1] under extreme conditions** Bold indicates the better result in each pairwise comparison. Underlined entries highlight where ANYEDIT outperforms Vevo2.

Task	Condition	Model	SNRErr↓	RT60Err↓	WER/CER↓	SIM-spk↑	SIM-bg↑	N-MOS↑	S-MOS↑
Speech	SNR=0 dB	Vevo2	13.321	**0.011**	**0.196**	0.806	0.471	4.02	3.96
		ANYEDIT	**3.368**	0.165	0.386	**0.897**	**0.756**	**4.31**	**4.27**
	RT60=1.5 s	Vevo2	10.383	0.031	0.093	0.929	0.504	4.08	4.01
		ANYEDIT	**1.480**	**0.0003**	**0.067**	**0.940**	**0.820**	**4.36**	**4.33**
	Both (SNR0+RT60)	Vevo2	9.421	**0.0005**	0.156	0.884	0.458	3.95	3.90
		ANYEDIT	**2.017**	0.0006	**0.105**	**0.911**	**0.848**	**4.24**	**4.18**
Singing	SNR=0 dB	Vevo2	9.940	0.146	0.530	0.829	**0.783**	4.06	3.98
		ANYEDIT	**3.490**	**0.017**	**0.323**	**0.864**	0.205	**4.34**	**4.29**
	RT60=1.5 s	Vevo2	6.229	0.117	0.416	0.893	0.679	4.11	4.04
		ANYEDIT	**2.483**	**0.0003**	**0.142**	**0.932**	**0.850**	**4.39**	**4.35**
	Both (SNR0+RT60)	Vevo2	7.296	**0.0002**	1.154	0.821	0.573	3.92	3.86
		ANYEDIT	**1.809**	0.0006	**0.262**	**0.904**	**0.840**	**4.21**	**4.16**

Table 5: **Ablation study on unified editing under 10dB SNR** F0 Corr is computed specifically on singing samples.

Model Variation	WER/CER↓	SIM-s↑	F0 Corr↑	SIM-bg↑	N-MOS↑	S-MOS↑
ANYEDIT (Full Model)	**0.042**	**0.970**	**0.534**	**0.915**	**4.28**	**4.22**
Impact of Noise-Robust Tokenizer						
w/o Noise-Robust Tokenizer	0.145	0.900	0.412	0.720	3.08	4.18
Impact of AR Training Data Modality						
Speech Data Only	0.048	0.962	0.181	0.905	4.05	3.52
Singing Data Only	0.068	0.940	0.232	0.895	3.95	3.32
Impact of AR Training Process						
w/o Phase 1	0.049	0.961	0.042	0.378	3.21	3.39
w/o Phase 2	0.082	0.935	0.512	0.900	3.75	3.98
w/o Unmod.-Area-Aware Sup.	0.055	0.960	0.403	0.910	3.98	3.99
w/o Preference Alignment	0.055	0.958	0.421	0.912	3.78	3.98
w/ SVS Synth Alignment Data	0.057	0.959	0.414	0.913	3.76	4.01
Impact of FMT Training Data						
w/o Noisy Data Rendering	0.043	0.965	0.530	0.200	3.88	3.51

Table 6: **Subjective evaluation scores** We randomly choose 60 samples from ANYEDITBENCH. Comparisons can also be found on our demo page. More detailed statistics can be found in Table 15 .

Domain	Method	N-MOS↑	S-MOS↑	Fool Rate
Speech	Vevo2	4.15	4.02	31.5%
	SSR-Speech	4.48	3.99	43.2%
	VoiceCraft	4.04	4.32	35.7%
	Step-EditX	3.95	4.02	33.8%
	ANYEDIT	**4.48**	**4.45**	**46.5%**
Singing	Vevo2	4.22	4.15	28.6%
	SoulX-singer	3.51	3.34	0.0%
	ANYEDIT	**4.52**	**4.48**	**42.8%**

3.4 Case Study

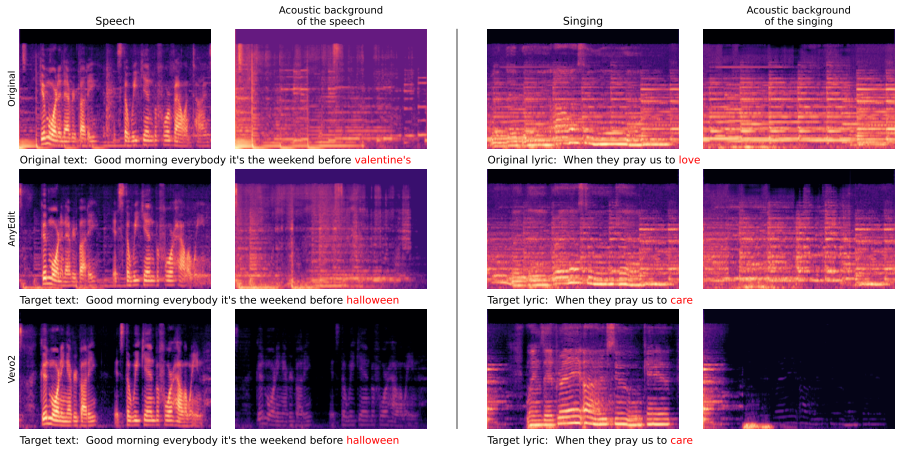

Figure 5: Case study on real-world speech and singing voice editing.

We visualize representative real-world speech and singing editing cases via Mel-spectrograms in Figure 5. When applying the baseline Vevo2 directly (Row 3), the background texture is severely discarded during the editing window, manifesting as stark, dark gaps in the background spectrograms. In contrast, ANYEDIT (Row 2) modifies the target text and remains practically identical to the original background spectrograms (Row 1). By natively modeling the unified acoustic scene, ANYEDIT bypasses the artifacts bottleneck of separation models, ensuring background continuity and realism. To further systematically compare our approach with explicit "separation-edit-mix"

pipelines, including direct mixing approaches analogous to SeamlessEdit [9] and SNR-aware mixing methods akin to DAIEN-TTS [10], we provide a comprehensive quantitative separation comparison in Appendix F.8

3.5 Subjective Evaluation

As shown in Table 6, we conducted a subjective evaluation with 10 raters. Compared with Vevo2, ANYEDIT yields consistently higher N-MOS and S-MOS across both domains. Notably, ANYEDIT achieves significant gains in Fooling Rate (where 50% indicates perfect perceptual indistinguishability from real audio). This demonstrates that listeners much more frequently perceived our edited outputs as authentic, unedited natural recordings.

4 Conclusion

In this paper, we presented ANYEDIT, a unified, separation-free framework for speech and singing voice editing in real-world acoustic environments. By disentangling clean-space semantic editing from context-aware acoustic rendering, ANYEDIT overcomes the limitations of traditional direct end-to-end modeling and separation-edit-mix pipelines. Our two-stage architecture leverages an instruction-guided autoregressive editor and a flow-matching acoustic model, supported by a noise-robust tokenizer for stable prosody extraction. Additionally, unmodified-area-aware supervision and DPO on mined Real-to-Real singing pairs enhance prosodic stability and naturalness.

Furthermore, we introduced ANYEDITBENCH, the first comprehensive benchmark for speech and singing editing across diverse acoustic conditions. Extensive evaluations confirmed that ANYEDIT achieves state-of-the-art performance, maintaining high content accuracy, timbre fidelity, and acoustic continuity under clean, simulated, and extreme real-world degradations.

Acknowledgments and Disclosure of Funding

The work of Zhizheng Wu is supported by the National Natural Science Foundation of China under Grant 62376237. This work is supported by the Program for Guangdong Introducing Innovative and Entrepreneurial Teams, Grant No. 2023ZT10X044.

References

[1] Xueyao Zhang, Junan Zhang, Yuancheng Wang, Chaoren Wang, Yuanzhe Chen, Dongya Jia, Zhuo Chen, and Zhizheng Wu. Vevo2: A Unified and Controllable Framework for Speech and Singing Voice Generation. *IEEE Transactions on Audio, Speech and Language Processing*, 34: 2262–2275, 2026. doi: 10.1109/TASLPRO.2026.3671970.

[2] 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. doi: 10.18653/v1/2024.acl-long.673.

[3] Chao Yan, Boyong Wu, Peng Yang, Pengfei Tan, Guoqiang Hu, Li Xie, Yuxin Zhang, Fei Tian, Xuerui Yang, Xiangyu Zhang, et al. Step-Audio-EditX Technical Report. *arXiv preprint arXiv:2511.03601*, 2025.

[4] Lichao Zhang, Zhou Zhao, Yi Ren, and Liqun Deng. EditSinger: Zero-Shot Text-Based Singing Voice Editing System with Diverse Prosody Modeling. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, pages 4503–4509, 2022. doi: 10.24963/ijcai.2022/625.

[5] Chenyu Yang, Shuai Wang, Hangting Chen, Jianwei Yu, Wei Tan, Rongzhi Gu, Yaoxun Xu, Yizhi Zhou, Haina Zhu, and Haizhou Li. SongEditor: Adapting Zero-Shot Song Generation Language Model as a Multi-Task Editor. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25597–25605, 2025. doi: 10.1609/aaai.v39i24.34750.

[6] Canxiang Yan, Chunxiang Jin, Dawei Huang, Haibing Yu, Han Peng, Hui Zhan, Jie Gao, Jing Peng, Jingdong Chen, Jun Zhou, et al. Ming-UniAudio: Speech LLM for Joint Understanding, Generation and Editing with Unified Representation. *arXiv preprint arXiv:2511.05516*, 2025.

[7] Dong Zhang, Gang Wang, Jinlong Xue, Kai Fang, Liang Zhao, Rui Ma, Shuhuai Ren, Shuo Liu, Tao Guo, Weiji Zhuang, et al. MiMo-Audio: Audio Language Models are Few-Shot Learners. *arXiv preprint arXiv:2512.23808*, 2025.

[8] Shaohan Jiang, Junan Zhang, Yunjia Zhang, Jing Yang, Fan Fan, and Zhizheng Wu. SingVERSE: A Diverse, Real-World Benchmark for Singing Voice Enhancement. *arXiv preprint arXiv:2509.20969*, 2025.

[9] Kuan-Yu Chen, Jeng-Lin Li, De-Yan Lu, and Jian-Jiun Ding. SeamlessEdit: Background Noise Aware Zero-Shot Speech Editing with in-Context Enhancement. In *European Signal Processing Conference (EUSIPCO)*, 2026. arXiv:2505.14066.

[10] Ye-Xin Lu, Yu Gu, Kun Wei, Hui-Peng Du, Yang Ai, and Zhen-Hua Ling. DAIEN-TTS: Disentangled Audio Infilling for Environment-Aware Text-to-Speech Synthesis. In *ICASSP 2026 – 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 17287–17291. IEEE, 2026. doi: 10.1109/ICASSP55912.2026.11460953.

[11] Helin Wang, Meng Yu, Jiarui Hai, Chen Chen, Yuchen Hu, Rilin Chen, Najim Dehak, and Dong Yu. SSR-Speech: Towards Stable, Safe and Robust Zero-shot Text-based Speech Editing and Synthesis. In *ICASSP 2025 – 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025. doi: 10.1109/ICASSP49660.2025.10889119.

[12] Junan Zhang, Jing Yang, Zihao Fang, Yuancheng Wang, Zehua Zhang, Zhuo Wang, Fan Fan, and Zhizheng Wu. AnyEnhance: A Unified Generative Model with Prompt-Guidance and Self-Critic for Voice Enhancement. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3085–3098, 2025. doi: 10.1109/TASLPRO.2025.3587393.

[13] Haorui He, Zengqiang Shang, Chaoren Wang, Xuyuan Li, Yicheng Gu, Hua Hua, Liwei Liu, Chen Yang, Jiaqi Li, Peiyang Shi, Yuancheng Wang, Kai Chen, Pengyuan Zhang, and Zhizheng Wu. Emilia: An Extensive, Multilingual, and Diverse Speech Dataset for Large-Scale Speech Generation. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 885–890. IEEE, 2024. doi: 10.1109/SLT61566.2024.10832365.

[14] Yicheng Gu, Chaoren Wang, Junan Zhang, Xueyao Zhang, Zihao Fang, Haorui He, and Zhizheng Wu. SingNet: Towards a Large-Scale, Diverse, and in-the-Wild Singing Voice Dataset. *arXiv preprint arXiv:2505.09325*, 2025.

[15] Tao Feng, Yuxiang Wang, Yuancheng Wang, Xueyao Zhang, Dekun Chen, Chaoren Wang, Xun Guan, and Zhizheng Wu. MimicLM: Zero-Shot Voice Imitation through Autoregressive Modeling of Pseudo-Parallel Speech Corpora. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 20392–20405, 2026. doi: 10.18653/v1/2026.findings-acl.1020.

[16] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[17] Chandan K. A. Reddy, Vishak Gopal, Ross Cutler, Ebrahim Beyrami, Roger Cheng, Harishchandra Dubey, Sergiy Matusevych, Robert Aichner, Ashkan Aazami, Sebastian Braun, Puneet Rana, Sriram Srinivasan, and Johannes Gehrke. The INTERSPEECH 2020 Deep Noise Suppression Challenge: Datasets, Subjective Testing Framework, and Challenge Results. In *Interspeech 2020*, pages 2492–2496, 2020. doi: 10.21437/Interspeech.2020-3038.

[18] Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux. WHAM!: Extending Speech Separation to Noisy Environments. In *Interspeech 2019*, pages 1368–1372, 2019. doi: 10.21437/Interspeech.2019-2821.

[19] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780. IEEE, 2017. doi: 10.1109/ICASSP.2017.7952261.

[20] Yu Zhang, Rongjie Huang, Ruiqi Li, JinZheng He, Yan Xia, Feiyang Chen, Xinyu Duan, Baoxing Huai, and Zhou Zhao. StyleSinger: Style Transfer for Out-of-Domain Singing Voice Synthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19597–19605, 2024. doi: 10.1609/aaai.v38i17.29932.

[21] Yu Zhang, Ziyue Jiang, Ruiqi Li, Changhao Pan, Jinzheng He, Rongjie Huang, Chuxin Wang, and Zhou Zhao. TCSinger: Zero-Shot Singing Voice Synthesis with Style Transfer and Multi-Level Style Control. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1960–1975, 2024. doi: 10.18653/v1/2024.emnlp-main.117.

[22] Jiale Qian, Hao Meng, Tian Zheng, Pengcheng Zhu, Haopeng Lin, Yuhang Dai, Hanke Xie, Wenxiao Cao, Ruixuan Shang, Jun Wu, et al. SoulX-Singer: Towards High-Quality Zero-Shot Singing Voice Synthesis. *arXiv preprint arXiv:2602.07803*, 2026.

[23] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust Speech Recognition via Large-Scale Weak Supervision. In *Proceedings of the 40th International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023.

[24] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-Scale Contrastive Language-Audio Pretraining with Feature Fusion and Keyword-to-Caption Augmentation. In *ICASSP 2023 – 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023. doi: 10.1109/ICASSP49357.2023.10095969.

[25] Tao Wang, Jiangyan Yi, Ruibo Fu, Jianhua Tao, and Zhengqi Wen. CampNet: Context-Aware Mask Prediction for End-to-End Text-Based Speech Editing. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:2241–2254, 2022. doi: 10.1109/TASLP.2022.3190717.

[26] 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. doi: 10.1109/ASRU51503.2021.9688051.

[27] 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. doi: 10.18653/v1/2023.findings-acl.741.

[28] Zongcai Du, Guilin Deng, Xiaofeng Guo, Xin Gao, Linke Li, Kaichang Cheng, Fubo Han, Siyu Yang, Peng Liu, Pan Zhong, and Qiang Fu. DiTSinger: Scaling Singing Voice Synthesis with Diffusion Transformer and Implicit Alignment. *arXiv preprint arXiv:2510.09016*, 2025.

[29] Xueyao Zhang, Xiaohui Zhang, Kainan Peng, Zhenyu Tang, Vimal Manohar, Yingru Liu, Jeff Hwang, Dangna Li, Yuhao Wang, Julian Chan, Yuan Huang, Zhizheng Wu, and Mingbo Ma. Vevo: Controllable Zero-Shot Voice Imitation with Self-Supervised Disentanglement. In *International Conference on Learning Representations*, 2025. URL `https://arxiv.org/abs/2502.07243`.

[30] Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. Voicebox: Text-Guided Multilingual Universal Speech Generation at Scale. In *Advances in Neural Information Processing Systems*, volume 36, pages 14005–14034, 2023. doi: 10.52202/075280-0618.

[31] Dongchao Yang, Jinchuan Tian, Xu Tan, Rongjie Huang, Songxiang Liu, Haohan Guo, Xuankai Chang, Jiatong Shi, Sheng Zhao, Jiang Bian, Zhou Zhao, Xixin Wu, and Helen M. Meng. UniAudio: Towards Universal Audio Generation with Large Language Models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 56422–56447. PMLR, 2024.

[32] Xinjian Li, Shinnosuke Takamichi, Takaaki Saeki, William Chen, Sayaka Shiota, and Shinji Watanabe. YODAS: YouTube-Oriented Dataset for Audio and Speech. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE, 2023. doi: 10.1109/ASRU57964.2023.10389689.

[33] 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. doi: 10.21437/Interspeech.2019-2441.

[34] Junichi Yamagishi, Christophe Veaux, and Kirsten MacDonald. CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit, 2019.

[35] Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. AISHELL-1: An Open-Source Mandarin Speech Corpus and a Speech Recognition Baseline. In *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*, pages 1–5. IEEE, 2017. doi: 10.1109/ICSDA.2017.8384449.

[36] Jiayu Du, Xingyu Na, Xuechen Liu, and Hui Bu. AISHELL-2: Transforming Mandarin ASR Research into Industrial Scale. *arXiv preprint arXiv:1808.10583*, 2018.

[37] Yao Shi, Hui Bu, Xin Xu, Shaoji Zhang, and Ming Li. AISHELL-3: A Multi-speaker Mandarin TTS Corpus and the Baselines. *arXiv preprint arXiv:2010.11567*, 2020.

[38] Wangyou Zhang, Robin Scheibler, Kohei Saijo, Samuele Cornell, Chenda Li, Zhaoheng Ni, Jan Pirklbauer, Marvin Sach, Shinji Watanabe, Tim Fingscheidt, and Yanmin Qian. URGENT Challenge: Universality, Robustness, and Generalizability for Speech Enhancement. In *Interspeech 2024*, pages 4868–4872, 2024. doi: 10.21437/Interspeech.2024-1239.

[39] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra. FSD50K: An Open Dataset of Human-Labeled Sound Events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852, 2022. doi: 10.1109/TASLP.2021.3133208.

[40] Nicolas Turpault, Romain Serizel, Ankit Parag Shah, and Justin Salamon. Sound Event Detection in Domestic Environments with Weakly Labeled Data and Soundscape Synthesis. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events*, 2019.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[42] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023.

[43] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

[44] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-Weighted Linear Units for Neural Network Function Approximation in Reinforcement Learning. *Neural Networks*, 107:3–11, 2018. doi: 10.1016/j.neunet.2017.12.012.

[45] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024.

[46] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. *arXiv preprint arXiv:2209.03003*, 2022.

[47] Simon3000. Genshin voice: A multi-lingual voice dataset from genshin impact. `https://huggingface.co/datasets/simon3000/genshin-voice`, 2025.

[48] Shuqi Dai, Yuxuan Wu, Siqi Chen, Roy Huang, and Roger B. Dannenberg. Singstyle111: A multilingual singing dataset with style transfer. In *ISMIR*, pages 765–773, 2023.

[49] Yu Zhang, Changhao Pan, Wenxiang Guo, Ruiqi Li, Zhiyuan Zhu, Jialei Wang, Wenhao Xu, Jingyu Lu, Zhiqing Hong, Chuxin Wang, Lichao Zhang, Jinzheng He, Ziyue Jiang, Yuxin Chen, Chen Yang, Jiecheng Zhou, Xinyu Cheng, and Zhou Zhao. Gtsinger: A global multi-technique singing corpus with realistic music scores for all singing tasks. In *NeurIPS*, 2024.

A Related Work

A.1 Speech Editing

Speech editing has evolved from waveform concatenation and mask-based reconstruction to neural generative infilling. Early work such as *CampNet* [25] framed editing as a localized text-to-speech (TTS) task with context-aware mask prediction. *EditSpeech* [26] introduced bidirectional context fusion, while *VoiceCraft* [2] leveraged token rearrangement in neural codec language models to enable autoregressive bidirectional infilling. *FluentSpeech* [27] further explored diffusion-based editing, and *SSR-Speech* [11] improved robustness and stability in zero-shot text-based speech editing. More recent systems such as *Step-Audio-EditX* [3] continued to extend audio editing capabilities. However, most existing methods were primarily developed for clean speech and did not explicitly preserve background acoustics in real-world recordings.

A.2 Singing Voice Editing and Synthesis

Singing voice editing is more challenging because lyric replacement must remain consistent with melody, rhythm, and vocal expression. *EditSinger* [4] addressed this by disentangling pitch from lyrics to preserve melodic contours. *StyleSinger* [20], *TCSinger* [21], and *SoulX-Singer* [22] demonstrated recent progress in controllable or zero-shot singing voice synthesis (SVS), while *DiTSinger* [28] and *SongEditor* [5] explored scalable singing generation and editing via implicit alignment or dual-track masking. Nevertheless, these approaches mainly targeted clean-condition synthesis or editing and typically ignored background-scene continuity. Furthermore, many singing editing pipelines relied on synthetic target construction, inadvertently introducing artifacts from the synthesis model.

A.3 Unified Speech and Singing Generation

Unified modeling across speech and singing enables shared representation learning across both modalities. *Vevo* [29] and *Vevo2* [1] showed that explicit prosody tokenization was effective for unified clean-condition generation and controllable rendering. However, these models were designed for generation rather than localized vocal editing, relied on externally provided prosodic conditions, and did not explicitly address acoustic scene preservation under noise, reverberation, or background music.

A.4 Environment-Aware and Unified Audio Generation

SeamlessEdit [9] followed a "separation-edit-mix" paradigm that explicitly separated vocals from the background before editing. However, such pipelines can be fragile when edited duration changes or source separation is imperfect. *DAIEN-TTS* [10] moved toward environment-aware generation by conditioning synthesis on background representations. More general models—*Voicebox* [30], *UniAudio* [31], *Ming-UniAudio* [6], and *MiMo-Audio* [7]—further broadened unified modeling across diverse audio tasks, yet none were specifically tailored for unified speech and singing voice editing in real-world environments.

B Definitions of Core Representations

To establish a clear and self-contained theoretical foundation for unified vocal editing, we factorize the continuous audio signal into three mutually orthogonal feature streams, following the disentanglement principles in [29, 1]:

- **Prosody Tokens (z^{prosody}):** Prosody tokens capture the temporal pitch contour (fundamental frequency F_0), melodic trajectory, and duration dynamics across speech and singing. They are derived from normalized chromagrams to ensure octave invariance and melodic compactness. Importantly, this representation explicitly isolates melodic movement and rhythm while discarding linguistic phonetics, expressive vocal nuances, speaker identity (timbre), and environmental background acoustics.
- **Content-Style Tokens (z^{CS}):** Content-style tokens encode linguistic information (phonetic transcriptions) alongside local expressive delivery habits, such as articulation manner, emotional tone, and vocal techniques (e.g., breathiness and vibrato). While content-style tokens capture fine-grained phonetic semantics and expressive dynamics, they are strictly invariant to speaker identity (timbre) and external acoustic environments (e.g., noise, reverberation).
- **Acoustic Context and Rendering:** Acoustic context refers to the continuous time-frequency prompt (mel-spectrogram) extracted from the unedited reference audio. It encapsulates speaker-specific timbre alongside environmental acoustic properties, including ambient noise, room impulse responses (reverberation), and background music. Correspondingly, *acoustic rendering* is the generative process handled by the Flow Matching Transformer (FMT), which synthesizes the post-edit mel-spectrogram by conditioning on the generated clean semantic tokens while strictly anchoring timbre and environmental textures to the reference acoustic context.
- **Semantic Editing:** In the scope of this work, *semantic editing* is defined specifically as localized textual and linguistic modifications—namely, inserting, deleting, or substituting words or lyrics within an existing utterance—while anchoring unedited content, prosody, and background continuity. Modifying high-level expressive performance styles (e.g., changing vocal emotion or singing technique) is considered orthogonal and deferred to future work.

Table 7: Taxonomy and orthogonal properties of core representations in ANYEDIT.

Representation	Encoded Attributes	Excluded Attributes	System Role
Prosody Tokens (z^{prosody})	Pitch contour, melody, rhythm	Phonetics, vocal style, timbre, acoustics	Condition for AR editor
Content-Style Tokens (z^{CS})	Phonemes, articulation, vibrato, emotion	Speaker timbre, room/noise acoustics	Output target of AR editor
Acoustic Context (Mel)	Speaker timbre, background noise, RT60	Semantic text content	In-context prompt for FMT

Decoupling Principle across Pipeline Stages: This factorization defines the functional boundary between our two generative stages. In Stage 1 (AR editor), degraded audio only affects the extraction of z^{prosody}. By equipping the prosody tokenizer with noise robustness, the entire semantic prediction process operates purely in the clean domain, generating clean z^{CS} conditioned on clean z^{prosody} and target text. In Stage 2 (FMT acoustic rendering), because z^{CS} carries neither speaker timbre nor background noise, the acoustic renderer is prevented from exploiting semantic shortcuts, forcing it to reconstruct the continuous acoustic scene purely from the source context.

C AnyEditBench and Evaluation Details

C.1 Benchmark Construction

ANYEDITBENCH is organized into 12 subsets (3 subsets $\times$ 2 modalities $\times$ 2 languages), with 50 samples per subset and 600 evaluation items in total. All subsets contain insertion, deletion, and substitution edits with random span lengths. Table 8 provides a detailed summary.

Clean subset: English speech subsets were drawn from LibriTTS [33] and VCTK [34]; Chinese speech subsets used AISHELL-1/2/3 [35–37]. Clean singing evaluation used the clean subset of SingVERSE [8]. Each subset contains both single-span and multi-span examples.

Simulated subset: Starting from clean-source speech and singing data, we synthesized degraded evaluation samples by combining noise and reverberation conditions. The noise setting was selected from SNR $\in \{0, 10, 20, \infty\}$ dB, and the reverberation setting from RT60 $\in \{0, 0.5, 1.0, 1.5\}$ s. Excluding the fully clean+dry case (SNR $= \infty$, RT60 $= 0$) yielded 15 acoustic conditions in total. Noise and RIR resources were collected from Urgent Challenge noise data [38], FSD50K [39], DESED [40], and a self-collected dataset.

Table 8: Summary of ANYEDITBENCH. Each subset contains 50 evaluation items. The clean subset includes both single-span and multi-span edits.

subset	Domain	Language	Source	Acoustic condition	Items
Clean	Speech	EN	LibriTTS, VCTK	Clean recordings; single-span and multi-span edits	50
	Speech	ZH	AISHELL-1/2/3	Clean recordings; single-span and multi-span edits	50
	Singing	EN	Clean SingVERSE test subset [8]	Clean recordings; single-span and multi-span edits	50
	Singing	ZH	Clean SingVERSE test subset [8]	Clean recordings; single-span and multi-span edits	50
Simulated	Speech	EN	Clean-source speech subsets	15 noise/reverberation conditions (SNR $\in \{0, 10, 20, \infty\}$, RT60 $\in \{0, 0.5, 1.0, 1.5\}$)	50
	Speech	ZH	Clean-source speech subsets	15 noise/reverberation conditions (excluding clean case)	50
	Singing	EN	Clean SingVERSE test subset [8]	15 noise/reverberation conditions (excluding clean case)	50
	Singing	ZH	Clean SingVERSE test subset [8]	15 noise/reverberation conditions (excluding clean case)	50
Real-world	Speech	EN	YODAS [32]	Naturally recorded environments	50
	Speech	ZH	YODAS [32]	Naturally recorded environments	50
	Singing	EN	SingVERSE [8]	Naturally recorded environments	50
	Singing	ZH	SingVERSE [8]	Naturally recorded environments	50

Real-world subset: Real-world speech data were taken from YODAS [32], and real-world singing data from SingVERSE [8]. These recordings were collected in naturally occurring acoustic environments.

C.2 Evaluation Metrics

C.2.1 Objective Evaluation Metrics

- **WER / CER (↓):** We used *Whisper-large-v3* [23] to compute English WER and Chinese CER for edited content.
- **SIM-spk (↑):** We computed the speaker cosine similarity between the source and edited audio using embeddings extracted with the pretrained WavLM [4].
- **F0 Corr (↑):** To assess the melodic quality of singing voice editing, we resampled audio to 16 kHz, extracted F0 with librosa YIN [5], and computed the Pearson correlation of log-F0 over vocal frames between the edited and reference audio.
- **SIM-bg (↑):** To assess the preservation of the acoustic environment, we used CLAP [24] [6] at 48 kHz and computed the cosine similarity between the background embeddings of the reference and edited audio.
- **RT60Error (↓):** We first separated each waveform into vocal and accompaniment using an optimized SingNet pipeline [14] (identical to the benchmark construction operation), estimated the room impulse characteristics from the separated vocal track, and then computed RT60 from the estimated decay curve. We reported the absolute RT60 error (seconds) between the edited result and the reference.
- **SNRError (↓):** Following the SingNet [14] pipeline, we used the separated vocal and background stems to compute the signal-to-noise ratio as the energy ratio between vocal and accompaniment, and reported the absolute SNR error (dB) with respect to the reference clip.

C.2.2 Subjective Evaluation Metrics

We conducted human evaluation with 10 subjects, sampling 15 items from each of the speech-English, speech-Chinese, singing-English, and singing-Chinese subsets. Subjects rated **N-MOS** for naturalness and **S-MOS** for similarity. **N-MOS (Naturalness Mean Opinion Score)** assesses the naturalness of the audio and consists of three sub-ratings:

- Naturalness of the human voice,
- Naturalness of the background sound,
- Naturalness of the mixture between background sound and human voice.

[4] `https://huggingface.co/microsoft/wavlm-base-plus-sv`

[5] `https://github.com/librosa/librosa`

[6] `https://huggingface.co/laion/clap-htsat-unfused`

S-MOS (Similarity Mean Opinion Score) assesses the similarity to a reference and consists of two sub-ratings:

- Similarity of the human voice,
- Similarity of the background sound.

We also conducted a **Fooling Rate** blind A/B test in which subjects identified the authentic human recording; a Fooling Rate closer to 50% indicates stronger perceptual indistinguishability.

D Implementation Details

D.1 Computational Budget and Model Parameters

To assess training and inference efficiency, we compare the parameter scale and language/domain coverage of ANYEDIT against existing state-of-the-art audio editing and generation models in Table 9. ANYEDIT is a unified model capable of handling both speech and singing editing across English and Chinese within a single 1.1B parameter framework.

Table 9: Comparison of model parameter counts and capability coverage.

Model	Supported Domain	Trained for Editing	Total for Inference
ANYEDIT (Ours)	Speech + Singing (EN + ZH)	872M	1.1B
Vevo2 [1]	Speech + Singing (EN + ZH)	872M	1.1B
Ming-UniAudio [6]	Speech (EN + ZH)	∼2.8B active	18.0B (∼2.8B active)
MiMo-Audio [7]	Speech (EN + ZH)	8.0B	9.3B
Step-Audio-EditX [3]	Speech (EN + ZH)	3.5B	3.7B
SSR-Speech [11]	Speech (EN + ZH)*	840M + 840M	1.8B
VoiceCraft [2]	Speech (EN only)	830M	0.9B
SoulX-Singer [22]	Singing (ZH only)	443M	1.8B
TCSinger [21]	Singing (EN + ZH)	330M	0.34B
StyleSinger [20]	Singing (ZH only)	64M	0.08B

*SSR-Speech requires two separate checkpoints for English and Chinese, respectively.

Hardware and Training Resources. All models were trained using the computational resources similar to Vevo2.

D.2 AR Model

All our models follow the standard Transformer [41, 42] architecture, employ RoPE positional encoding [43] and the SiLU [44] activation function. The encoder and decoder of the tokenizer and MGM models use bidirectional attention, while the AR models adopt causal attention.

ANYEDIT is initialized from the textual LLM `Qwen2.5-0.5B-Instruct` [45]: a 12-layer Transformer decoder with hidden size 1024, with vocabulary expanded to include prosody and content-style tokens (509M parameters total). We used the same text tokenizer as Vevo2 [1].

D.3 FMT Model

We provide additional details of the flow matching framework used to train the acoustic generation process of ANYEDIT. Flow matching [16] defines a continuous transformation from a prior distribution (*e.g.*, Gaussian noise) to a target data distribution (*e.g.*, mel-spectrograms) by learning a time-dependent velocity field along an interpolated trajectory $\boldsymbol{x}_t$.

While multiple interpolation strategies can be used to construct $\boldsymbol{x}_t$, we adopt the *optimal transport path* formulation [16, 46], instantiated in this work as simple linear interpolation. Specifically, given a clean mel-spectrogram $\boldsymbol{x} \in \mathbb{R}^{T\times d}$ and a noise sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we construct an intermediate sample as:

$$\boldsymbol{x}_t = t\boldsymbol{x} + (1-t)\boldsymbol{\epsilon}, \quad t \sim \text{Uniform}(0,1), \tag{3}$$

where t is sampled uniformly from $[0, 1]$, and $\boldsymbol{x}_t$ denotes the noisy input at time t. The corresponding ground-truth velocity is the temporal derivative of $\boldsymbol{x}_t$:

$$\boldsymbol{v} = \frac{d\boldsymbol{x}_t}{dt} = \boldsymbol{x} - \boldsymbol{\epsilon}. \tag{4}$$

The diffusion decoder $\mathcal{D}_\phi$ is trained to predict $\boldsymbol{v}$, conditioned on the token sequence $\boldsymbol{q} = \mathcal{Q}(\mathcal{E}_\theta(\boldsymbol{x}))$ and the associated text $\boldsymbol{x}_{text}$, using the following objective:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{x}_{text}),\boldsymbol{\epsilon},t}\left[\|(\boldsymbol{x} - \boldsymbol{\epsilon}) - \mathcal{D}_\phi(\boldsymbol{q}, \boldsymbol{x}_t, t, \boldsymbol{x}_{text})\|\right]. \tag{5}$$

Inference At inference time, we start with a noise sample $\boldsymbol{x}_0 = \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and solve the ordinary differential equation:

$$\frac{d\boldsymbol{x}_t}{dt} = \mathcal{D}_\phi(\boldsymbol{q}, \boldsymbol{x}_t, t, \boldsymbol{x}_{text}) \tag{6}$$

from $t = 0$ to $t = 1$ using a simple Euler ODE solver over a discretized set of N time steps.

Flow matching provides a stable and interpretable training signal by directly supervising the instantaneous direction in which a noisy sample $\boldsymbol{x}_t$ should evolve to match the clean target $\boldsymbol{x}$. In our setting, it enables effective training under real-world acoustic conditions, containing both speaker timbre information and background acoustic information.

FMT Architecture and Conditioning. The Flow Matching Transformer (FMT) is a non-autoregressive Transformer. Specifically, the model is constructed by stacking LLaMA-2 Decoder blocks, with the causal mask replaced by bidirectional full attention, allowing the model to attend to the sequence globally. The conditioning mechanisms are as follows:

- **Time Step Injection:** The continuous time step t is encoded using sinusoidal embeddings.
- **Condition Injection:** Both the time embeddings and the conditioning representations are injected into the Transformer blocks using Adaptive Layer Normalization (AdaLN).
- **Acoustic Prompting:** The reference audio prefix (capturing the target speaker's timbre and ambient background characteristics) is provided as an acoustic prompt to guide the flow-matching process in rendering the final mel-spectrogram with consistent speaker identity and acoustic conditions.

E Baselines

E.1 Speech Editing Baselines

VoiceCraft [2] A token-infilling neural codec language model designed for zero-shot speech editing and text-to-speech in the wild. It employs causal masking and delayed stacking to enable autoregressive generation within an existing sequence. We use the official code and checkpoints[7].

SSR-Speech [11] A Transformer decoder-based autoregressive neural codec model tailored for stable and robust zero-shot text-based speech editing. It incorporates frame-level watermarking and classifier-free guidance. We use the official codebase and pre-trained checkpoints[8].

Step-Audio-EditX [3] A state-of-the-art large language model-based tool developed by StepFun for precise, robust, and expressive speech editing. We use the official model checkpoints available on HuggingFace[9].

E.2 Unified Audio Models

Ming-UniAudio [6] A comprehensive unified audio generation model built upon a 16-billion parameter architecture. It supports a wide variety of audio-related tasks including generation and editing. We use the official pre-trained checkpoint[10].

[7] https://github.com/jasonppy/VoiceCraft

[8] https://github.com/WangHelin1997/SSR-Speech

[9] https://huggingface.co/stepfun-ai/Step-Audio-EditX

[10] https://huggingface.co/inclusionAI/Ming-UniAudio-16B-A3B

MiMo-Audio [7] A 7-billion parameter unified audio generation model developed by Xiaomi, designed to process and generate diverse audio content across different modalities. We utilize the official base checkpoint[11].

Vevo2 [1] A strong unified generative model capable of high-quality speech and singing voice generation. It acts as our primary unified baseline covering both modalities in clean conditions. We use the official implementation and checkpoints. **Vevo2** [1] serves as our primary unified baseline covering both speech and singing modalities. While Vevo2 is designed for zero-shot generation (TTS/SVS) and conversion (VC/SVC) in clean acoustic conditions, it exhibits two structural limitations in real-world localized editing: (1) it does not model or reconstruct background environmental acoustics, discarding ambient textures during editing; and (2) its autoregressive design couples sequence length with prosody preservation, making it challenging to simultaneously preserve original prosody while handling edits with variable duration. In contrast, ANYEDIT decouples semantic editing from acoustic rendering to resolve this trade-off. Furthermore, as evaluated in Appendix **??**, ANYEDIT naturally inherits strong zero-shot VC and SVC capabilities through its acoustic renderer without task-specific tuning.

E.3 Singing Voice Baselines

StyleSinger [20] A zero-shot singing voice synthesis model designed to generate high-fidelity and expressive vocals while capturing diverse singing styles. We use the official implementation and checkpoints[12].

TCSinger [21] A zero-shot singing voice synthesis model that provides highly controllable text-to-singing generation and timbre preservation. We use the official open-source repository and checkpoints[13].

SoulX-Singer [22] A high-quality, expressive singing voice synthesis model developed by Soul AILab. Although the model is claimed to support lyric editing, no public code is available for this functionality. Therefore, based on the model's inference process, we implemented our own inference code for lyric editing. The base model (weights) is used from the official repository[14], while the lyric editing inference is our own implementation.

F Extended Quantitative Analysis

F.1 Objective results on the ANYEDITBENCH clean split

Speech editing (left) and singing voice editing (right) are reported jointly. English uses WER, Chinese uses CER; F0 Corr is singing-only ("–" for speech). Bold: best per domain; underlined: ANYEDIT outperforms Vevo2 [1]. VoiceCraft [2]: English only; StyleSinger [20], SoulX-Singer [22]: Chinese only. Results are showed in Table 11.

F.2 Evaluation on Vevo2 [1]'s editing test set.

We tested ANYEDIT on the exact datasets used by Vevo2 for expressive speech [47] and singing voice [48, 49]editing task, which predominantly feature clean backgrounds. As shown below, ANYEDIT consistently outperforms Vevo2 across core metrics on its Vevo2 [1]'s Editing testsets. Results are showed in Table 10.

F.3 Evaluation on available RealEdit Benchmark

The RealEdit [2] benchmark has been tested by other editing models (SSR-Speech, VoiceCraft, etc.). However, one of its three subsets (Spotify) is currently unavailable. Furthermore, it only contains English speech with generally clean backgrounds, which does not fully align with our goal of real-world noisy editing. For a comprehensive comparison, we evaluated AnyEdit on the two

[11] https://huggingface.co/XiaomiMiMo/MiMo-Audio-7B-Base
[12] https://github.com/AaronZ345/StyleSinger
[13] https://github.com/AaronZ345/TCSinger
[14] https://github.com/Soul-AILab/SoulX-Singer

Table 10: Comparison between Vevo2 and ANYEDIT on Vevo 2's speech and singing voice editing test sets.

	Speech Editing		Singing Voice Editing	
Metric	Vevo2	**ANYEDIT**	Vevo2	**ANYEDIT**
WER (en)↓	0.1717	**0.1658**	0.2795	**0.2754**
CER (zh)↓	0.1950	**0.1841**	0.1986	**0.1883**
SIM-spk (en)↑	0.9582	**0.9583**	0.9496	**0.9519**
SIM-spk (zh)↑	0.9582	**0.9597**	**0.9574**	0.9553
F0 Corr (en)↑	0.1801	**0.1803**	0.3341	**0.3406**
F0 Corr (zh)↑	0.1989	**0.2079**	0.3295	**0.3485**

Table 11: Objective results on the ANYEDITBENCH clean split

	Speech Editing							Singing Voice Editing						
Metric	SSR-Speech	Voice-Craft	Step-EditX	Vevo2	Ming-Uni	Mimo-Audio	**ANYEDIT**	Style-Singer	TCSinger	SoulX-Singer	Vevo2	Ming-Uni	Mimo-Audio	**ANYEDIT**
WER (en)↓	**0.0246**	0.0428	0.0823	0.0600	0.1291	0.0423	0.0408	–	1.6463	–	0.1482	0.4123	0.1861	**0.1425**
CER (zh)↓	0.0877	–	0.0419	0.0561	0.1091	**0.0274**	0.0487	0.6124	0.3135	0.8572	0.2203	0.4444	0.1957	**0.0613**
SIM-spk (en)↑	**0.9876**	0.9714	0.9723	0.9739	0.9692	0.9448	0.9763	–	0.6569	–	0.9410	0.8314	0.8604	**0.9551**
SIM-spk (zh)↑	**0.9655**	–	0.9430	0.9582	0.9274	0.9344	0.9540	0.7865	0.8309	0.9182	0.9458	0.8444	0.8836	**0.9486**
F0 Corr (en)↑	–	–	–	–	–	–	–	–	-0.0538	–	0.1338	0.0120	0.0330	**0.1589**
F0 Corr (zh)↑	–	–	–	–	–	–	–	0.0081	0.0391	**0.3784**	0.2478	0.0390	0.0307	0.2411

accessible subsets. Results are showed in Table 12. ANYEDIT achieves SOTA results in the currently available RealEdit [2] benchmark.

Table 12: Objective results on speech editing

Metric	SSR-Speech	Voice-Craft	Step-EditX	Vevo2	Ming-Uni	Mimo-Audio	**ANYEDIT**
WER↓	0.0389	0.0531	0.1518	0.0266	0.1118	0.1029	**0.0255**
SIM-spk↑	0.9685	0.9657	0.9598	0.9657	0.9532	0.9465	**0.9689**
F0 Corr↑	0.1638	0.0076	0.2155	0.1966	0.2137	0.0248	**0.2330**

F.4 Evaluation on zero-shot voice conversion and singing voice conversion.

Vevo2 is explicitly designed and optimized as a unified model for zero-shot VC and SVC. In contrast, AnyEdit was never explicitly trained on VC/SVC tasks. However, thanks to our decoupled architecture, AnyEdit inherently possesses zero-shot VC and SVC capabilities. We evaluated AnyEdit on Vevo2's exact VC/SVC datasets, including the challenging cross-domain Speech-to-Singing transfer. Testsets are same as editing. The results are showed in Table 13.

Remarkably, without any task-specific training, AnyEdit achieves highly competitive performance against the dedicated Vevo2 model, even outperforming it in several textual accuracy and prosody preservation metrics, particularly in the difficult Speech-to-Singing scenario.

F.5 Ablation on Teacher-Student Prosody Distillation

To evaluate the effectiveness of the teacher-student distillation strategy, we compare our approach against a standard single-stage baseline, termed *Direct Clean Supervision*, where a single VQ-VAE takes noisy chromagrams as input and is trained to reconstruct clean chromagram targets directly in the spectral domain.

We evaluated both models on 500 vocal-noise mixtures across nine SNR levels ranging from -5 dB to 20 dB. Table 14 presents the token match rate between the predicted prosody tokens and the ground-truth clean tokens. Teacher-student distillation substantially outperforms direct clean reconstruction across all noise levels, demonstrating that distilling discrete code indices directly provides superior noise resilience compared to continuous spectral regression under heavy noise (e.g., 0.4587 vs. 0.2029 at 0 dB SNR).

Table 13: Zero-shot performance comparison between Vevo2 and AnyEdit across multiple conversion tasks.

	VC		SVC		Spk2Sing	
Metric	Vevo2	**AnyEdit**	Vevo2	**AnyEdit**	Vevo2	**AnyEdit**
WER (en)↓	0.0639	**0.0586**	**0.1884**	0.2262	0.2038	**0.1895**
CER (zh)↓	0.1028	**0.0998**	**0.3552**	0.3719	0.4302	**0.3380**
SIM-spk (en)↑	**0.9187**	0.9084	–	–	–	–
SIM-spk (zh)↑	**0.9291**	0.9134	–	–	–	–
F0 Corr (en)↑	–	–	0.1678	**0.1772**	0.1528	**0.1592**
F0 Corr (zh)↑	–	–	0.2161	**0.2240**	**0.1900**	0.1758

Table 14: Prosody token match rate between direct clean supervision and teacher-student distillation across varying SNR conditions.

Method	**-5 dB**	**0 dB**	**5 dB**	**10 dB**	**15 dB**	**20 dB**
Direct Clean Supervision	0.0896	0.2029	0.3367	0.4496	0.5497	0.6238
Teacher-Student Distillation (Ours)	**0.3360**	**0.4587**	**0.5586**	**0.6306**	**0.6968**	**0.7410**

F.6 Subjective Evaluation Protocol.

Ten professional evaluators assessed 60 randomized items across the speech and singing benchmarks. Listening tests were completed in sessions of less than 60 minutes to prevent listener fatigue. We performed two-sided paired Wilcoxon signed-rank tests between ANYEDIT and each baseline, and report 95% bootstrap confidence intervals for all mean opinion scores. Table 15 summarizes the comprehensive statistical results.

Table 15: Subjective evaluation scores with 95% confidence intervals (CI). Superscripts denote paired Wilcoxon signed-rank tests against ANYEDIT: * ($p < 0.05$), ** ($p < 0.01$), $^{\text{n.s.}}$ not significant ($p \geq 0.05$). Bold denotes optimal scores.

Domain	**Method**	**N-MOS (95% CI)↑**	**S-MOS (95% CI)↑**	**Fooling Rate (%)**
Speech	Vevo2	4.15 (4.06, 4.24)**	4.02 (3.92, 4.12)**	31.5
	SSR-Speech	4.48 (4.40, 4.56)$^{\text{n.s.}}$	3.99 (3.90, 4.08)**	43.2$^{\text{n.s.}}$
	VoiceCraft	4.04 (3.95, 4.13)**	4.32 (4.23, 4.41)*	35.7*
	Step-Audio-EditX	3.95 (3.86, 4.04)**	4.02 (3.93, 4.11)**	33.8**
	ANYEDIT (Ours)	**4.48 (4.41, 4.55)**	**4.45 (4.37, 4.53)**	**46.5**
Singing	Vevo2	4.22 (4.13, 4.31)**	4.15 (4.05, 4.25)**	28.6**
	SoulX-Singer	3.51 (3.40, 3.62)**	3.34 (3.22, 3.46)**	0.0**
	ANYEDIT (Ours)	**4.52 (4.44, 4.60)**	**4.48 (4.39, 4.57)**	**42.8**

F.7 Unmodified Area Fidelity

A critical challenge in localized audio editing is the unintentional alteration of the unedited context, often referred to as "prosody drift." To rigorously evaluate this, we introduce the Unmodified Area Fidelity evaluation.

SSR-Speech achieves better results than our ANYEDIT on several speech editing metrics. However, in both objective and subjective evaluations showed in Appendix F.7 and Table 16, we note that SSR-Speech does not perform true editing: instead of preserving the unmodified portions of the input, it regenerates the entire utterance from scratch, essentially behaving as a text-to-speech (TTS) model. Although the authors describe their system as "Speech Editing and Synthesis", this operational characteristic deviates from the task definition of speech editing, where unmodified segments should remain. Therefore, the superior numbers reported for SSR-Speech are not directly comparable to those of editing models like ANYEDIT.

Evaluation Protocol We first compare the source text and target text to extract the Longest Common Subsequence (LCS), effectively isolating the unchanged word/character spans. We then utilize a CTC forced aligner [15] to obtain the precise timestamps of these unchanged words in both the source audio and the generated edited audio. Evaluation metrics are exclusively computed within these matched unedited time windows. The specific definitions of the calculation methods for these metrics follow the unified standards detailed in Appendix C.2.

We calculate WER/CER to assess whether the original content is mistakenly modified, and SIM-spk to evaluate timbre consistency. Crucially, we utilize **F0 Corr** to evaluate prosody preservation. While F0 Corr is typically used for singing melody evaluation, we also employ it here for **speech evaluation**. In expressive speech, F0 effectively captures the original intonation contour, stress, and the rhythm and rate of speech. High F0 Corr in the unmodified region indicates that the model perfectly anchors the surrounding conversational or emotional context without causing unnatural prosodic shifts at the edit boundaries.

Results As shown in Table 16, SSR-Speech struggles significantly with F0 preservation in speech (F0 Corr: 0.468), primarily because it fails to capture the expressive intonation contours and stress of the original context. Vevo2 performs relatively better but still exhibits notable prosodic drift. Comparing AnyEdit (w/o unmodified-area-aware supervision) and the full AnyEdit model, we observe that the proposed supervision mechanism dramatically improves F0 Corr (from 0.815 to 0.963 for speech) and reduces WER to near zero. This confirms that our method strictly anchors the timing, stress, and pitch of the surrounding unedited vocals.

Table 16: Unmodified Area Fidelity (UAF) on the simulated subset. "–" denotes unavailable/inapplicable entries.

Method	WER/CER ↓	F0 Corr ↑		SIM-spk ↑
		Speech	Singing	
SSR-Speech	0.012	0.468	–	0.982
Vevo2	0.051	0.732	0.785	0.945
AnyEdit (w/o unmodified-area-aware sup.)	0.038	0.815	0.862	0.958
AnyEdit (Ours)	**0.009**	**0.963**	**0.975**	**0.988**

F.8 Comparison with Separation-Based Pipelines

To demonstrate the structural superiority of our separation-free architecture, we quantitatively compare AnyEdit against explicit "separation-edit-mix" pipelines.

Baseline Pipelines We utilize a open-source vocal separation model (BS-RoFormer) [16] to extract the clean vocal stem and the background accompaniment from the degraded source audio. We construct three baseline pipelines:

1. **MaskGCT + Sep (Direct Mix):** The separated clean vocal is processed by MaskGCT for localized masked infilling. The edited vocal is then directly mixed with the background accompaniment in the time domain.
2. **Vevo2 + Sep (Direct Mix):** Similar to the pipeline adopted by SeamlessEdit [9], the separated vocal is edited by Vevo2, followed by a direct time-domain mix.
3. **Vevo2 + Sep (SNR-aware Mix):** Inspired by DAIEN-TTS [10], this pipeline calculates the Signal-to-Noise Ratio (SNR) between the original separated vocal and the background. After Vevo2 edits the vocal, the background is scaled and mixed back to explicitly match the original SNR.
4. **AnyEdit (Separation-Free):** Our proposed method, which directly takes the noisy/mixed audio as input and renders the target features end-to-end.

[15] https://github.com/MahmoudAshraf97/ctc-forced-aligner

[16] https://github.com/TRvlvr/model_repo/releases/download/all_public_uvr_models/model_bs_roformer_ep_317_sdr_12.9755.ckpt

Table 17: Quantitative comparison of explicit separation-based pipelines versus our separation-free framework on sampled real-world subset

Pipeline Strategy	WER ↓	SIM-bg ↑	SNRErr (dB) ↓	N-MOS ↑
MaskGCT + Sep (Direct Mix)	0.158	0.348	19.12	3.61 ± 0.10
Vevo2 + Sep (Direct Mix)	0.165	0.352	18.54	3.55 ± 0.11
Vevo2 + Sep (SNR-aware Mix)	0.165	0.415	6.85	3.72 ± 0.08
ANYEDIT (Separation-Free, Ours)	**0.142**	**0.819**	**3.13**	$\mathbf{4.38 \pm 0.06}$

Results Table 17 summarizes the results on the singing subset under complex background conditions. While separation-based methods achieve reasonable WER (since the editing models operate on clean vocals), they fail catastrophically in preserving the acoustic environment (SIM-bg) and introduce massive SNR errors. Direct mixing causes phase mismatch and background discontinuity at the edit boundaries. The SNR-aware mix (DAIEN-TTS [10] style) improves the SNRErr compared to direct mixing but still suffers from low SIM-bg because the separation model inevitably introduces mechanical artifacts and frequency masking that cannot be recovered during the final fusion. In contrast, our separation-free ANYEDIT elegantly maintains high edit accuracy while achieving state-of-the-art background preservation and naturalness.

F.9 Expressive Capacity of Chromagram Prosody Representation

Singing prosody encompasses both rhythmic phrasing and melodic contour. While chromagrams collapse octave information into pitch classes, empirical analysis on vocal datasets indicates that this representation reliably preserves salient melodic motion.

We analyzed 3,000 vocal tracks randomly sampled from SingNet against ground-truth WORLD F0. Based on the 5th–95th percentiles of voiced frames, the mean local pitch range within typical vocal phrases spans 1.03 octaves. Furthermore, 99.77% of consecutive note intervals remain within a single octave, with 97.45% corresponding to step-wise motions (≤ 2 semitones). Table 18 summarizes the circular correlation and pitch-class error across bin resolutions. The 24-bin chromagram achieves a circular correlation of 0.730 with F0 pitch classes, confirming that it accurately captures melodic trajectories. The absolute vocal octave is subsequently reconstructed by the acoustic renderer conditioned on speaker style.

Table 18: Agreement between chromagram representations and F0 pitch contours on singing vocals.

Configuration	Circular Correlation ↑	Mean Pitch-Class Error (semitones) ↓	Onset Correlation ↑
12 bins	0.718	1.090	0.309
24 bins (Ours)	**0.730**	**1.040**	**0.318**

F.10 Evaluation on Varying-Length Editing

Unlike systems that rely on strict temporal infilling constraints, ANYEDIT natively supports edits with dynamic duration changes (such as word insertions and phrase expansions) because the autoregressive stage predicts variable token lengths.

We evaluated this capability on a length-changing subset comprising approximately 6% of ANYEDITBENCH across Clean, Real-world, and Simulated domains. Table 19 reports the quantitative comparison against baseline models. ANYEDIT outperforms the unified baseline Vevo2 and specialized speech editors across both editing accuracy and background preservation.

G Limitations and Future Work

The current framework use sufficient surrounding acoustic context (approximately 10 s) for reliable background rendering, limiting applicability to very long clips. For very long recordings, chunking the audio into segments of roughly 10 s is recommended for optimal rendering. The model does not yet support real-time editing. Future work includes extending this framework to support real-time and

Table 19: Quantitative results on the varying-length editing subset of ANYEDITBENCH.

Model	WER/CER ↓	SIM-spk ↑	SIM-bg ↑	F0 Corr ↑
SSR-Speech	0.0842	0.9521	0.7240	0.1824
VoiceCraft	0.1265	0.9418	0.6512	0.0150
Vevo2	0.0894	0.9510	0.6128	0.2014
ANYEDIT (Ours)	**0.0521**	**0.9654**	**0.7850**	**0.2480**

broader multi-lingual vocal editing. Current mined real-to-real's singing pair is available. At the same time, exploring advanced retrieval method to mine real to real speech pair is highly promising future directions. While ANYEDIT excels at preserving continuous background textures (e.g., stationary environmental noise, ambient babble, and general room reverberation), the continuous diffusion process in the FMT can occasionally smooth out highly transient sound events (e.g., sudden acoustic impulses) or complex long-tail echoes. Preserving transient and non-stationary acoustic events remains an important direction for future exploration.

H Broader Impacts

ANYEDIT advances the quality and accessibility of vocal post-production for user-generated content and music production. While beneficial, the technology also carries potential for misuse through voice spoofing or unauthorized content manipulation. These risks are particularly concerning in scenarios involving biometric authentication or deceptive media. To prevent misuse, we advocate for the development of reliable deepfake detection tools, watermarking methods for synthetic speech, and reporting mechanisms to flag suspected abuse.

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 claims made in the abstract and introduction are empirically verified in Section 3, in particular Section 3.2.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: We mention our limitations in Appendix G

   Guidelines:

   - The answer [N/A] 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: [Yes]

Justification: We provide full set of assumptions and a complete proof in Appendix D.3

Guidelines:

- The answer [N/A] 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: We disclose all information needed for reproduction in Appendix D

   Guidelines:

   - The answer [N/A] 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: [Yes]

Justification: We will provide them upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide experimental details in Appendix D.2

   Guidelines:

   - The answer [N/A] 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: We provided statistical significance of the experiments in Table **??**.

   Guidelines:

   - The answer [N/A] 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: [No]

   Justification: We do not provide information related to compute resources.

   Guidelines:

   - The answer [N/A] 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: We have read and understood the code of ethics and have made every effort to adhere to it.

   Guidelines:

   - The answer [N/A] 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: We discuss both potential positive societal impacts and negative societal impacts of the work performed in Appendix H.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: To mitigate the risk of misuse (such as unauthorized voice cloning or deception), we will integrate a post-hoc signal-level watermarking mechanism to ensure verifiable audio generation before releasing the model checkpoint. Furthermore, our code, model checkpoints, and AnyEditBench dataset will be released under a CC BY-NC license with an explicit Acceptable Use Policy (AUP).

    Guidelines:

    - The answer [N/A] 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:

    Guidelines: We use publicly available datasets and models all of which are properly cited in the paper.

    - The answer [N/A] 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: [Yes]

    Justification: We will release our codebase and benchmark dataset along with detailed README files.

    Guidelines:

    - The answer [N/A] 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: Details of the subjective listening test protocol, instructions are documented in Appendix F.6. Participant compensation 20 dollar per hour.

    Guidelines:

    - The answer [N/A] 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: [Yes].

    Justification: The human evaluation posed minimal risk to participants. The listening task involved neutral audio materials, fully informed consent, and strict fatigue prevention measures (session duration limited to under 60 minutes).

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We used an LLM solely for editing assistance.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.