# ZERO-SHOT SINGING STYLE TRANSFER VIA A STRUCTURED LYRICS–MELODY BOTTLENECK

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Zero-shot singing style transfer aims to reproduce a reference singer's vocal identity and expressive delivery while preserving a song's lyrics and melody. Existing conversion systems can retain unwanted traces of the original singer's style: the pitch trajectories and frame-level features used to preserve the song also encode how it was performed. We introduce $S^3$, a *score-conditioned, singer-prompted singing* framework built around a structured lyrics–melody bottleneck. This symbolic score represents timed notes, their lyric assignments, and groups of notes sung on the same syllable (melismas), separating score-level constraints from acoustic realization. Joint lyrics–melody transcription recovers this structure from full-mix audio (vocals with accompaniment) and supplied lyrics, allowing lyric evidence to guide note selection. A reference-conditioned generator explicitly encodes the structure and uses reference audio to guide pronunciation timing and vocal expression. It supports both synthesis from supplied scores and conversion from recovered scores without singer-specific fine-tuning. Experiments show higher mean style-similarity and naturalness ratings for both tasks, alongside improved note transcription, note–lyric assignment, conversion intelligibility, and predicted singing quality. Audio examples are available at `https://scube2027iclr.github.io/demo/`.

## 1 INTRODUCTION

Zero-shot singing style transfer aims to preserve a song's lyrics and melody while adopting another singer's voice and expressive delivery, without singer-specific training or fine-tuning. In singing voice conversion (SVC), the original performance is the *source*; a *reference recording* demonstrates the desired singer's voice and style. Performance style extends beyond vocal identity to articulation, phonation, pronunciation timing, and pitch modulation (Zhang et al., 2024a;b; Guo et al., 2025a). We study SVC with supplied lyrics and singing voice synthesis (SVS), where lyrics and a musical score specify the song directly.

Conventional SVC systems can retain unwanted traces of the original singer's style because their content representations entangle musical content with expressive delivery. They often use source frame-level fundamental frequency (F0) and linguistic features as a content bottleneck (Liu et al., 2021; Chen et al., 2024; Ferreira et al., 2025). F0 carries vibrato and pitch transitions, while frame-synchronous linguistic features retain articulation timing. Consequently, conversion may reproduce the reference singer's timbre while retaining aspects of the original singer's delivery, limiting style transfer.

Learned bottlenecks, speaker-invariant features, and factorized representations can separate content from voice and expression (Qian et al., 2019; 2022; Choi et al., 2023; Zhang et al., 2025b). In singing, coarse melody representations allow reference-guided expression, but improved style imitation can come at the cost of melody fidelity (Zhang et al., 2025a). To address this entanglement, we replace source acoustic conditioning with a *structured lyrics–melody bottleneck*: a symbolic score specifying note pitches and times, their lyric assignments, and groups of notes sung on the same syllable (melismas). The generator receives this score instead of source acoustic features, preserving musical constraints while allowing the reference to guide within-note pitch modulation, phonation, and precise pronunciation timing.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

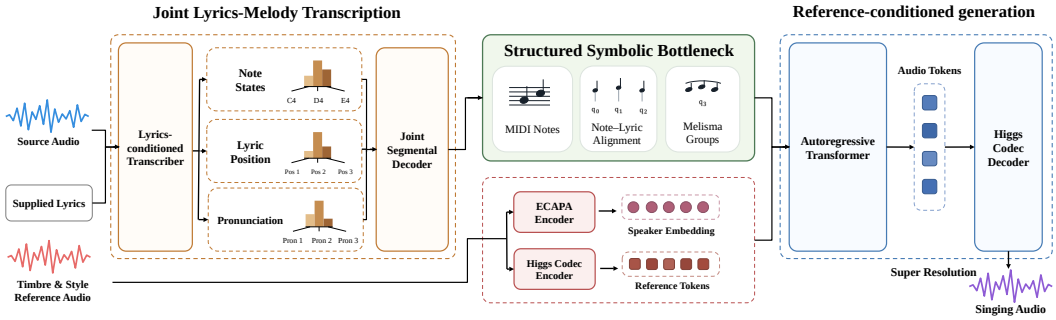


Figure 1: Overview of S$^3$. A structured lyrics–melody bottleneck represents timed notes, lyric assignments, and melisma groups. For SVC, joint transcription recovers this score from full-mix audio and supplied lyrics; for SVS, the score is supplied. The generator conditions on the score and a target-singer reference, encoded as codec tokens and a speaker embedding. Codec decoding and audio super-resolution produce the singing waveform.

To replace source acoustic features, the score must specify how the lyrics are distributed across notes. Adjacent notes may belong to different syllables or continue the same syllable, coupling note boundaries with lyric progression. Melody transcription alone does not resolve this correspondence (Donahue et al., 2022; OpenVPI, n.d.). Word-boundary guidance can help (Li et al., 2024), but fixing lyric alignment before note inference, even with joint training, constrains subsequent decisions and can propagate errors (Guo et al., 2025b). Joint decoding instead lets lyric evidence guide note selection and boundaries while note evidence resolves syllable assignments.

S$^3$, our *score-conditioned, singer-prompted singing* framework, uses this structured bottleneck to connect joint lyrics–melody transcription with reference-conditioned generation (Figure 1). Given full-mix audio (vocals with accompaniment) and supplied lyrics, the transcriber predicts note states, lyric positions, and pronunciation labels. Beat-aligned supervision encourages recovery of rhythmic structure, while lyric-position supervision distinguishes repeated pronunciations. Joint segmental decoding selects non-overlapping notes with monotonic lyric assignments, distinguishing syllable advancement from melismatic continuation. Recovered scores support automatic conversion without manual note editing or acoustic lyric timestamps at inference and provide score–audio pairs for generator training.

The generator explicitly encodes note–lyric correspondence and melisma groups in a shared symbolic format across training and inference. Note times and melisma indicators specify rhythm and syllable grouping without fixing acoustic syllable timestamps. Building on score-conditioned and reference-conditioned synthesis (Liu et al., 2022; He et al., 2023; Shen et al., 2024; Qian et al., 2026), our autoregressive model receives the score, reference codec tokens, and a speaker embedding to generate codec tokens with reference-guided pronunciation timing and expression. A common score interface connects synthesis and conversion (Lin et al., 2021; Wu et al., 2022; Dai et al., 2025): the same generator accepts supplied scores for SVS and transcribed scores for SVC, requiring neither technique labels nor singer-specific fine-tuning.

We evaluate transcription, synthesis from ground-truth scores, and automatic conversion on Mandarin singing. Listening tests yield higher mean style-similarity and naturalness ratings for both SVS and SVC. S$^3$ also improves note transcription, note–lyric assignment, conversion intelligibility, and predicted singing quality over evaluated baselines.

Our contributions are:

- A structured lyrics–melody bottleneck that preserves musical structure while enabling reference-guided singing style transfer in both SVS and SVC.
- A lyrics-conditioned transcriber using joint segmental decoding to recover note boundaries, pitches, and lyric assignments, including melismas.
- A reference-conditioned singing generator encoding note–lyric correspondence and melismas, with higher mean style-similarity and naturalness ratings in both tasks.

## 2 Related Work

**Singing Voice Transcription.** Singing voice transcription recovers note pitches and boundaries from recordings. Existing research covers melody transcription from full-mix audio and automatic vocal note transcription (Donahue et al., 2022; OpenVPI, n.d.), with efforts to improve robustness for synthesis and expand training corpora through automatic annotation (Li et al., 2024). Recent frameworks jointly model transcription, lyric alignment, and performance-style annotation (Guo et al., 2025b). However, accurate note sequences and acoustic lyric alignment do not guarantee correct note–lyric assignments. Beat-aligned note intervals can differ from sung syllable intervals, so temporal overlap alone can yield incorrect assignments, especially in rapid passages. Melismas further require distinguishing syllable changes from continuation across notes. Jointly recovering notes and their lyric assignments therefore provides a structured score: it specifies not only the melody and lyrics, but also which notes share a syllable.

**Singing Voice Synthesis.** SVS generates singing from lyrics and musical scores. Earlier neural approaches primarily learned score-to-acoustic mappings from singer recordings, focusing on synthesis quality, pitch and duration modeling, and reducing fine-grained annotation requirements. These models generally supported singers represented in their training data; extending them to new singers often required additional recordings for training or adaptation (Liu et al., 2022; He et al., 2023). Advances in reference-conditioned generation and large-scale training have shifted attention toward zero-shot synthesis: generating a target singer's voice from a reference recording without singer-specific training or fine-tuning (Shen et al., 2024; Zhang et al., 2024b; Qian et al., 2026). The objective has also broadened from timbre similarity to performance-style transfer and control, covering both expression conveyed by reference recordings and singing techniques specified through explicit conditions (Zhang et al., 2024a;b; Guo et al., 2025a). Our generator follows this reference-conditioned setting, using explicit note–lyric correspondence and melisma groups as musical constraints. These constraints apply to both supplied and recovered scores, leaving acoustic syllable boundaries and vocal expression to the generator.

**Singing Voice Conversion.** SVC transforms a source recording into the voice of a target singer while preserving the lyrics and melody. Conventional systems often use source linguistic features and frame-level fundamental frequency (F0) as a content bottleneck, with target-singer conditioning for timbre conversion. Zero-shot systems use reference recordings to support conversion without singer-specific training (Liu et al., 2021; Chen et al., 2024; Ferreira et al., 2025). Related work in speech and voice generation explores learned bottlenecks, speaker-invariant features, and factorized representations to separate content from voice and expression (Qian et al., 2019; 2022; Choi et al., 2023), enabling separate control of timbre and style (Zhang et al., 2025b). For singing, coarse melody representations support reference-guided style transfer (Zhang et al., 2025a). Unified zero-shot SVS/SVC frameworks connect audio and symbolic inputs through shared latent representations (Wu et al., 2022) or use extracted performance attributes for conversion (Dai et al., 2025). However, retaining source F0 and frame-synchronous linguistic features can preserve vibrato, pitch transitions, and articulation timing, constraining how the target reference shapes performance style. Our approach connects transcription and generation through an explicit score, so synthesis and conversion share a conditioning format. This representation constrains musical content without passing source acoustic features to the generator.

## 3 Method

Figure 2 shows $S^3$, which connects a lyrics-conditioned transcriber and a reference-conditioned singing generator through a structured lyrics–melody bottleneck. The transcriber recovers timed notes, lyric assignments, and melisma groups from full-mix audio and supplied lyrics. This score specifies the musical content to preserve, while a singer reference guides vocal identity and expressive delivery. The generator uses the same score representation for SVC with recovered scores and SVS with supplied scores.

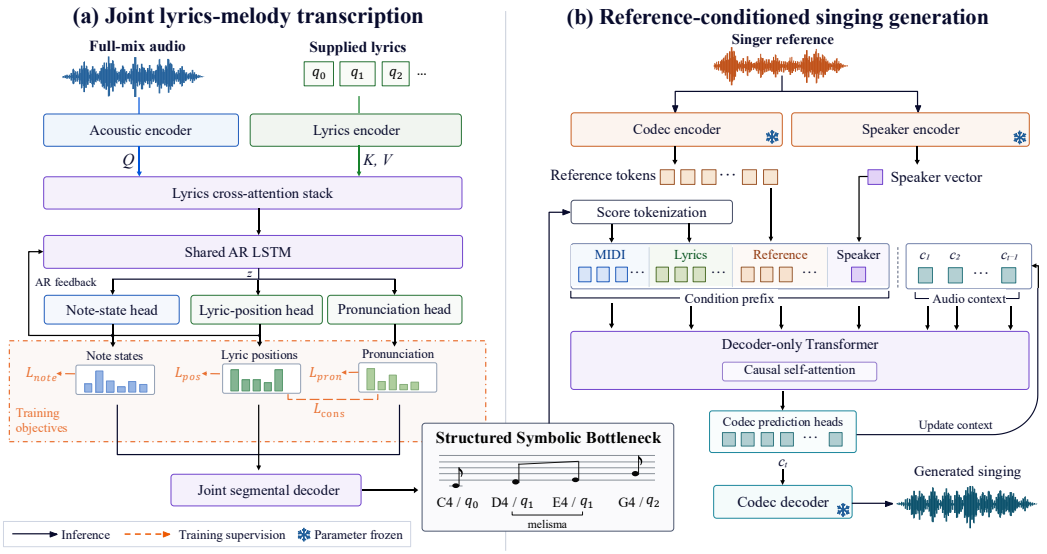


Figure 2: Architecture of S$^3$. (a) Audio–lyric cross-attention and a shared autoregressive LSTM support note-state, lyric-position, and pronunciation prediction. Joint segmental decoding recovers the structured score. Training uses note-aligned targets and a position–pronunciation consistency loss. (b) Melody, lyric, reference-token, and speaker embeddings form the conditioning prefix. A decoder-only Transformer predicts codec tokens autoregressively; codec decoding reconstructs the waveform.

### 3.1 Joint Lyrics–Melody Transcription

Given full-mix audio $x$ and supplied lyrics $\mathbf{q} = (q_0, \ldots, q_{L-1})$ represented as syllable-level pronunciation tokens, we recover a structured symbolic score:

$$(x, \mathbf{q}) \mapsto \hat{\mathcal{S}} = \big((a_i, b_i, p_i, j_i)\big)_{i=1}^{N}, \tag{1}$$

where note $i$ has onset and offset frames $a_i, b_i$, MIDI pitch $p_i$ (an integer pitch label), and lyric position $j_i \in \{0, \ldots, L-1\}$ specifying its pronunciation $q_{j_i}$. Together, the timed notes and their links to $\mathbf{q}$ define the structured lyrics–melody bottleneck. Consecutive notes sharing a lyric position form a melisma group, while identical pronunciations at different positions remain distinct. Joint segmental decoding (Section 3.2) recovers this structure.

**Lyrics-conditioned encoding.** To enable full-song inference without knowing which lyrics correspond to each clip, we pair each clip with the entire song's lyrics during training and inference. A CNN followed by a Conformer (Gulati et al., 2020) encodes the full-mix clip's log-mel spectrogram. We assign evenly spaced time anchors to lyric tokens over the song duration, preserving their order. Together with each frame's time within the song, these anchors provide coarse position cues for lyric assignment without acoustic lyric timestamps. A separate Transformer encodes the tokens, their sentence indices, and anchors. Cross-attention uses audio queries and lyric keys and values to produce lyrics-conditioned frame representations.

**Autoregressive multi-task prediction.** A shared autoregressive LSTM combines these representations with embeddings of the previous note state and lyric position. Its output $z_t$ at frame $t$ feeds three prediction heads. At inference, we feed back the most probable state and position to condition the next frame's predictions.

The note-state head predicts silence (no vocal note) or a pitch paired with onset, sustain, offset, re-onset, or slur-onset. Notes starting a new syllable use onset after a gap and re-onset otherwise. Notes continuing the same lyric position use slur-onset, with or without a gap. The lyric-position head scores each supplied lyric position and a null position using scaled dot products between projected $z_t$ and encoded lyric features. The pronunciation head predicts a pronunciation category or a null label. Repeated syllables share pronunciation labels but have distinct lyric positions.

**Note-aligned supervision.** Our note-level score annotations follow conventional manual transcription without exhaustively labeling vocal ornamentation. We quantize note timing using estimated beats. This supervision encourages the model to learn rhythmic structure from audio and predict rhythmically aligned notes. During data construction, each note's lyric position and pronunciation label are assigned to all frames in its interval. These labels identify the associated syllable independently of its precise acoustic timing. Slur-onset labels mark the starts of subsequent notes at the same lyric position. All three heads thus use note-aligned targets.

**Training objectives and schedule.** We jointly optimize the three prediction tasks with a position–pronunciation consistency loss:

$$\mathcal{L} = \lambda_{\text{note}}\mathcal{L}_{\text{note}} + \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{pron}}\mathcal{L}_{\text{pron}} + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}}. \tag{2}$$

All weights are nonnegative; the first three terms are classification losses for the corresponding heads. For $\mathcal{L}_{\text{cons}}$, we look up the pronunciation at the most probable lyric position and use it as an auxiliary target for the pronunciation head.

Using teacher forcing, we first train note-state prediction alone with lyric cross-attention disabled, then gradually increase $\lambda_{\text{pron}}$. Next, we enable cross-attention and lyric-encoder training, gradually increase $\lambda_{\text{pos}}$, and activate $\mathcal{L}_{\text{cons}}$. We also introduce scheduled sampling (Bengio et al., 2015), replacing some ground-truth feedback inputs with model predictions. Finally, we fine-tune with all four losses on manually annotated notes and lyric assignments. We report dataset sizes in Section 4.1 and model configurations in Appendix A.

### 3.2 Joint Segmental Decoding

Joint segmental decoding recovers the note attributes and lyric links in Equation 1 from the frame-wise note-state, lyric-position, and pronunciation distributions. These predictions are produced by autoregressive inference and held fixed throughout the search. Lyric-position and pronunciation evidence can help determine candidate note boundaries as well as lyric assignments. We combine these predictions in a semi-Markov path search that jointly selects note boundaries, pitches, and lyric assignments subject to note non-overlap and monotonic lyric progression.

**Candidate segments and joint evidence.** We first obtain initial notes using *Raw decoding*, a local beam search scored only by note-state predictions and structural priors. We construct a small set of candidates near their boundaries and pitches, retaining the original notes. A candidate $u = (a_u, b_u, p_u)$ specifies its onset frame, offset frame, and pitch. Joint search can adjust or discard the initial notes; the candidate set in our experiments does not include additional split or merge proposals.

Each candidate is paired with a lyric position $j$. Its score combines note-state, lyric-position, and pronunciation evidence with structural costs such as boundary deviations. Lyric evidence is a weighted combination of log-probability ratios for the lyric position and its pronunciation relative to their respective null classes, averaged over the segment. Position predictions distinguish occurrences with the same pronunciation, while pronunciation predictions provide acoustic support for the assignment.

The segment score also depends on the lyric transition. Advancing to a new position uses the combined onset and re-onset evidence; staying at the same position uses slur-onset evidence to represent another note on the same syllable. A candidate thus receives different scores along different lyric paths, coupling note selection with lyric assignment.

**Monotonic dynamic programming.** Let $D(u, j)$ be the best path score ending at candidate $u$ assigned to lyric position $j$. Let $s(u, j; k)$ denote its joint segment score when the preceding lyric position is $k$, and let $c(k, j)$ be the lyric-transition cost. The recurrence is

$$D(u, j) = \max_{(v,k)\in\mathcal{A}(u,j)} \left[D(v, k) + s(u, j; k) - c(k, j)\right], \tag{3}$$

where $\mathcal{A}(u, j)$ is the set of valid predecessors. The preceding note $v$ must have ended, $b_v \le a_u$, and the lyric position cannot move backward, $k \le j$. For $k < j$, the path advances and incurs a cost for

skipped lyric positions. For $k = j$, it continues the same syllable using slur-onset evidence, subject to the allowed inter-note gap.

A virtual start state accounts for leading skipped lyrics, and a terminal cost accounts for trailing skipped lyrics. Backtracking yields the best path within the candidate set, jointly determining non-overlapping notes and monotonic lyric assignments. One lyric position can span multiple notes, while unsupported notes or lyric positions may be skipped. The decoded score forms the generator's symbolic conditioning. Appendix D details candidate construction, joint scoring, dynamic programming, and output post-processing.

### 3.3 Reference-Conditioned Singing Generation

**Symbolic conditioning.** We encode supplied and recovered scores (Equation 1) as separate melody and lyric streams. Each note expands into repeated MIDI-pitch tokens on a 25 Hz grid. Zero denotes rests and ordinary note endings; a continuation token links notes belonging to the same syllable. The lyric stream contains one pronunciation token per lyric position represented in the score, in order. Both streams include sentence separators. For SVC, lyric tokens follow decoded positions, excluding skipped ones. This shared representation specifies score timing and lyric assignment without prescribing acoustic syllable boundaries.

**Reference-conditioned generation.** The Higgs Audio codec encoder (Boson AI, 2025) represents reference and target vocal waveforms as eight streams of discrete tokens, one per codebook, at 25 Hz. An ECAPA-TDNN encoder (Desplanques et al., 2020) extracts a speaker embedding. Both use released weights and remain frozen during training. The reference tokens provide acoustic context for vocal identity and expression, complemented by the speaker embedding. Special tokens identify the four input blocks. We concatenate melody embeddings $\mathbf{E}_{\mathrm{mel}}$, lyric embeddings $\mathbf{E}_{\mathrm{lyr}}$, reference-token embeddings $\mathbf{E}_{\mathrm{ref}}$, and the projected speaker embedding $\mathbf{e}_{\mathrm{spk}}$ along the sequence dimension to form the conditioning prefix:

$$\mathcal{P} = [\,\mathbf{E}_{\mathrm{mel}}; \mathbf{E}_{\mathrm{lyr}}; \mathbf{E}_{\mathrm{ref}}; \mathbf{e}_{\mathrm{spk}}\,]. \tag{4}$$

A decoder-only Transformer with the Qwen3 architecture (Yang et al., 2025) autoregressively predicts codec tokens conditioned on $\mathcal{P}$ and preceding codec tokens. Eight output heads predict the shifted codebook streams in parallel at each step using a delay pattern (Copet et al., 2023).

**Training.** We train the generator from random initialization on separated dry singing vocals paired with scores recovered by our transcriber. Each target segment is paired with a randomly sampled, non-overlapping reference segment from the same song; the speaker embedding comes from the full song's dry vocals. We randomly drop reference audio and speaker embeddings and optimize equally weighted next-token cross-entropy losses across the eight codebooks.

**Inference.** We generate successive score segments with classifier-free guidance, using codec tokens from previously generated audio as context for song continuation. The codec decoder reconstructs 24 kHz singing. We place decoded segments on the score timeline, crossfade them, and apply audio super-resolution to obtain the final 48 kHz waveform. Appendix A.2 details token representation, training, and inference.

## 4 Experiments

### 4.1 Dataset and Experimental Setup

**Training data.** Transcriber pretraining uses approximately 2,500 hours of Mandarin songs with MIDI annotations and automatic note–lyric assignments; fine-tuning uses a disjoint 80-hour manually annotated set. Generator training uses approximately 20,000 hours of separated Mandarin dry vocals with jointly decoded scores (Appendix A). We represent Mandarin syllables using toneless pinyin, mapping each complete syllable to one pronunciation token $q_j$.

**Evaluation data.** We evaluate score recovery on 57 Mandarin songs from HSD (Fu et al., 2022) with corrected annotations, all excluded from transcriber training. For SVS and SVC, we further exclude songs and singers used in generator training and retain one song per singer, yielding 36

songs. HSD is not used for checkpoint selection or model and decoder tuning. Reference and ground-truth recordings use processed dry vocals (Appendix C).

**Generation metrics.** We assess singer similarity using output–reference WavLM cosine similarity (Chen et al., 2022), intelligibility using FireRedASR2S character error rate (CER) against ground-truth lyrics (Xu et al., 2026), and singing quality using SingMOS (Tang et al., 2024). F0 frame error (FFE) (Chu & Alwan, 2009) measures pitch and voicing agreement with the ground-truth SVS target or SVC source recording.

**Subjective evaluation.** Thirty-two Mandarin-speaking listeners rate anonymized outputs for shared test cases on a five-point scale for timbre similarity, style similarity, naturalness, and singing accuracy (lyrics, melody, and rhythm). We report mean ratings and use two-sided listener-level sign-flip tests with Holm correction (Appendix B).

### 4.2 Joint Lyrics–Melody Transcription Evaluation

**Experimental setup.** We compare ROSVOT (Li et al., 2024), GAME (OpenVPI, n.d.), SheetSage-Jukebox (Donahue et al., 2022), and STARS (Guo et al., 2025b) on the 57 HSD recordings (HSD57), reporting mean per-song metrics. Among these baselines, only STARS uses supplied lyrics for transcription; we pair the others with SOFA (qiuqiao, n.d.) for note–lyric assignment. ROSVOT, GAME, STARS, and SOFA use separated vocals; SheetSage-Jukebox and $S^3$ use full mixes.

Three inference settings share $S^3$ model parameters: Raw decoding with or without lyrics, and Joint decoding with lyrics (Section 3.2). Decoder settings are selected on a separate development set and fixed before HSD57 evaluation.

**Note transcription metrics.** COn measures onset agreement within 100 ms using one-to-one note matching; COnP additionally requires pitch agreement within 50 cents. For evaluation, we select one global shift per system after decoding to maximize a note-matching F1 objective on HSD57, accounting for systematic timing differences between outputs and annotations. Each shift is shared across songs and never applied during generation or deployment. For score sequences, pitch error rate (PER), value error rate (VER), and note error rate (NER) measure edit errors in pitch, duration, and joint pitch–duration symbols. Durations use sixteenth-note units, with predictions quantized using a shared full-mix beat grid per song.

Table 1: Note transcription results on HSD57 after global time-shift calibration. Shifts are in milliseconds; all other values are percentages. P and R denote precision and recall. The three $S^3$ settings share model parameters and differ only in lyric input and decoding.

| System | Shift | COn ↑ | | | COnP ↑ | | | PER ↓ | VER ↓ | NER ↓ |
|---|---|---|---|---|---|---|---|---|---|---|
| | | P | R | F1 | P | R | F1 | | | |
| ROSVOT | +125 | 66.66 | 87.46 | 75.49 | 47.72 | 62.45 | 53.98 | 42.83 | 74.88 | 97.46 |
| GAME | +30 | 84.26 | **93.88** | 88.66 | 75.29 | 83.79 | 79.18 | 20.64 | 53.39 | 64.29 |
| SheetSage-Jukebox | +60 | 73.50 | 88.80 | 79.73 | 52.87 | 64.11 | 57.52 | 58.40 | 65.18 | 89.13 |
| STARS | +115 | 64.00 | 65.99 | 64.84 | 49.79 | 51.29 | 50.43 | 40.76 | 65.10 | 86.05 |
| $S^3$ Raw, no lyrics | +110 | 93.62 | 93.13 | 93.27 | 89.20 | **88.76** | 88.88 | 9.77 | 45.55 | 52.94 |
| $S^3$ Raw | +30 | 94.08 | 92.56 | 93.27 | 89.14 | 87.73 | 88.39 | 9.49 | 43.10 | 50.56 |
| $S^3$ Joint | +30 | **95.33** | 92.32 | **93.76** | **90.38** | 87.55 | **88.91** | **8.64** | **42.43** | **49.72** |

**Transcription results.** All three $S^3$ settings outperform the external baselines on both F1 scores and all symbolic error rates (Table 1). Under Raw decoding, adding lyrics reduces symbolic errors, leaves COn F1 unchanged, and slightly lowers COnP F1. Joint decoding achieves the highest F1 scores and lowest symbolic error rates. Its higher precision and slightly lower recall than Raw decoding with lyrics are consistent with more selective note retention.

**Note–lyric assignment metrics.** Assignment Accuracy measures pronunciation agreement among notes matched by onset and pitch; Joint Note–Lyric precision, recall, and F1 also account for unmatched notes. We use the same matching tolerances and normalized toneless pinyin labels. These labels do not distinguish lyric positions with identical pronunciation. Appendix C.1 details baseline configurations, calibration, and metric definitions.

Table 2: Note–lyric assignment results on HSD57 (%). SOFA + GT MIDI is included only as a reference for alignment with ground-truth notes.

| System | Assignment Accuracy ↑ | Joint P ↑ | Joint R ↑ | Joint F1 ↑ |
|---|---|---|---|---|
| SOFA + ROSVOT | 81.16 | 20.97 | 27.53 | 23.76 |
| SOFA + SheetSage-Jukebox | 89.89 | 45.96 | 55.89 | 50.07 |
| SOFA + GAME | 94.96 | 70.97 | 79.01 | 74.66 |
| STARS | 75.73 | 23.78 | 24.53 | 24.10 |
| $S^3$ Joint | **99.01** | **89.16** | **86.38** | **87.71** |
| SOFA + GT MIDI (reference) | 92.58 | — | — | — |

**Note–lyric assignment results.** $S^3$ Joint achieves the highest Assignment Accuracy (99.01%) and Joint F1 (87.71%; Table 2). Its advantage extends from pronunciation assignment among matched notes to recovering the notes and lyric content passed to the generator.

### 4.3 SVS Evaluation

**Experimental setup.** We evaluate reference-guided generation from HSD's ground-truth MIDI, lyrics, and note–lyric assignments, isolating the generator from transcription errors. For each of the 36 singers, the selected song's first lyric-aligned vocal segment serves as the reference for subsequent segments, yielding 36 references and 557 targets. All systems receive the same MIDI, lyrics, and reference for each case. Objective metrics are averaged over targets; Appendix C.2 details case construction and evaluation.

**Baselines.** We compare SoulX-Singer in score-conditioned mode (SoulX-Score) (Qian et al., 2026), TechSinger (Guo et al., 2025a), and TCSinger (Zhang et al., 2024b). All three baselines additionally receive phoneme timestamps; TechSinger receives no additional singing-technique labels.

Table 3: Objective SVS results on 557 segments.

| System | WavLM ↑ | FFE (%) ↓ | CER (%) ↓ | SingMOS ↑ |
|---|---|---|---|---|
| SoulX-Score | **0.908** | 31.905 | 1.533 | 4.046 |
| TechSinger | 0.757 | 31.581 | 2.525 | 3.516 |
| TCSinger | 0.820 | 28.359 | 3.174 | 3.258 |
| **$S^3$** | 0.900 | **25.562** | **1.529** | **4.070** |

Table 4: Mean SVS ratings (1–5).

| System | Timbre ↑ | Style ↑ | Nat. ↑ | Acc. ↑ |
|---|---|---|---|---|
| SoulX-Score | 3.621 | 3.360 | 3.395 | 3.954 |
| TechSinger | 1.750 | 1.841 | 2.295 | 2.618 |
| TCSinger | 1.851 | 1.724 | 1.667 | 2.148 |
| **$S^3$** | **4.057** | **4.126** | **4.471** | **4.455** |

**Results.** In the listening tests, $S^3$ receives the highest mean ratings in all four dimensions (Table 4). Style similarity and naturalness exceed all baselines (Holm-adjusted $p < 0.0001$), with gains of 0.766 and 1.076 points over SoulX-Score. Its timbre and accuracy advantages over SoulX-Score are also significant after correction.

$S^3$ also achieves the lowest FFE (25.562%), indicating the closest frame-level pitch and voicing agreement with the target recordings (Table 3). Its CER is marginally lower than SoulX-Score (1.529% vs. 1.533%), with slightly higher SingMOS and slightly lower WavLM similarity.

### 4.4 SVC Evaluation

**Experimental setup.** We pair each of the 36 singers (Section 4.1) with the other 35 as targets, yielding 1,260 directed source–target pairs. All systems share the retained cases after filtering to limit source-separation artifacts (Appendix C.3). Baselines receive dry lead vocals, while $S^3$ recovers its conditioning score from the original full mix.

**Baselines.** We compare SoulX-Melody and SoulX-SVC from SoulX-Singer (Qian et al., 2026), and Vevo2-Style and Vevo2-FM from Vevo2 (Zhang et al., 2025a). SoulX's pipeline extracts F0 and time-aligned phonemes for SoulX-Melody, and F0 and Whisper features for SoulX-SVC. Vevo2-Style performs style conversion with supplied lyrics; Vevo2-FM uses only flow matching. Appendix C.3 details each configuration.

Table 5: Objective SVC results.

| System | WavLM ↑ | FFE (%) ↓ | CER (%) ↓ | SingMOS ↑ |
|---|---|---|---|---|
| SoulX-Melody | 0.866 | 2.407 | 3.389 | 3.981 |
| Vevo2-Style | **0.888** | 27.681 | 4.696 | 3.992 |
| SoulX-SVC | 0.875 | **1.992** | 2.140 | 4.112 |
| Vevo2-FM | 0.864 | 11.125 | 3.622 | 4.011 |
| **$S^3$** | 0.887 | 20.433 | **1.707** | **4.121** |

Table 6: Mean SVC ratings (1–5).

| System | Timbre ↑ | Style ↑ | Nat. ↑ | Acc. ↑ |
|---|---|---|---|---|
| SoulX-Melody | 2.721 | 2.384 | 2.640 | 2.835 |
| Vevo2-Style | 2.384 | 2.035 | 1.930 | 1.624 |
| SoulX-SVC | 3.000 | 2.709 | 3.279 | 4.106 |
| Vevo2-FM | 2.860 | 2.655 | 3.314 | 3.506 |
| **$S^3$** | **3.733** | **3.686** | **4.174** | **4.212** |

**Results.** In the listening tests, $S^3$ receives the highest mean timbre, style-similarity, and naturalness ratings (Table 6). Style similarity exceeds all baselines (Holm-adjusted $p < 0.0001$); naturalness gains are significant against all baselines (adjusted $p < 0.001$). Mean singing accuracy is 4.212 versus 4.106 for SoulX-SVC, which retains source F0 and content representations; the difference is not significant (adjusted $p = 0.498$).

$S^3$ also achieves the lowest CER and highest SingMOS, with WavLM similarity close to Vevo2-Style (Table 5). Its FFE exceeds those of SoulX-Melody, SoulX-SVC, and Vevo2-FM. Under shared MIDI conditioning in SVS, $S^3$ has lower FFE than SoulX-Score (25.562% vs. 31.905%; Table 3), suggesting that source F0 may contribute to the SoulX variants' lower SVC FFE.

## 5 Discussion

The SVS results show that the generator can synthesize singing from supplied scores with high style-similarity ratings; the SVC results extend this finding to automatically recovered scores. In SVC, significant style-similarity gains, low CER, and high accuracy ratings show that the symbolic interface supports expressive generation without passing source F0 or frame-level linguistic features to the generator.

With shared model parameters, joint decoding improves note F1 and reduces symbolic errors relative to Raw decoding (Table 1). High note–lyric assignment accuracy (Table 2) further supports jointly recovering melody and lyric content for generation.

Retaining note timing and melisma groups makes musical constraints explicit, even where they reflect source performance choices. Within these constraints, the reference can guide a new acoustic realization. Source-relative FFE therefore complements style and accuracy ratings, but cannot distinguish expressive deviations from errors.

## 6 Conclusion and Future Work

$S^3$ supports zero-shot singing style transfer through a structured lyrics–melody bottleneck, combining joint transcription with reference-conditioned generation for SVS and SVC. Experiments on HSD show improved note transcription and assignment over evaluated baselines, alongside higher mean style-similarity and naturalness ratings in both tasks. Future work will examine how transcription errors affect generation and extend evaluation beyond Mandarin.

### AI USE STATEMENT

Generative AI tools were used to assist with implementing author-developed research ideas, discussing methodology, processing subjective evaluation data and conducting statistical analyses, interpreting experimental results, and drafting and editing the manuscript. Generative AI tools were not used to develop core research ideas, create synthetic training datasets or construct mathematical proofs. We have reviewed all AI-assisted work and take responsibility for the final content of this paper.

### ETHICS STATEMENT

All participants in the subjective listening evaluations provided informed consent. HSD recordings were used for research evaluation. Reference-conditioned singing generation can enable unauthorized voice imitation and misleading attribution; responsible use should respect performers' consent and the rights associated with the underlying recordings and musical works.

### REPRODUCIBILITY STATEMENT

The model architectures, training and inference settings, decoding algorithm, and evaluation protocols are described in the main paper and appendices. We plan to release selected evaluation pipelines to support independent evaluation and comparison.

## REFERENCES

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/e995f98d56967d946471af29d7bf99f1-Abstract.html.

Boson AI. Higgs Audio V2: Redefining expressiveness in audio generation. GitHub repository, 2025. URL https://github.com/boson-ai/higgs-audio. Audio tokenizer; accessed September 20, 2026.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, et al. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022. doi: 10.1109/JSTSP.2022.3188113. URL https://arxiv.org/abs/2110.13900.

Shihao Chen, Yu Gu, Jie Zhang, Na Li, Rilin Chen, Liping Chen, and Lirong Dai. LDM-SVC: Latent diffusion model based zero-shot any-to-any singing voice conversion with singer guidance. In *Proceedings of Interspeech 2024*, pp. 2770–2774, 2024. doi: 10.21437/Interspeech.2024-250. URL https://www.isca-archive.org/interspeech_2024/chen24e_interspeech.html.

Hyeong-Seok Choi, Jinhyeok Yang, Juheon Lee, and Hyeongju Kim. NANSY++: Unified voice synthesis with neural analysis and synthesis. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=elDEe8LYW7-.

Wei Chu and Abeer Alwan. Reducing F0 frame error of F0 tracking algorithms under noisy conditions with an unvoiced/voiced classification frontend. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3969–3972, 2009. doi: 10.1109/ICASSP.2009.4960497. URL https://www.seas.ucla.edu/spapl/paper/chu_icassp_09.pdf.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://arxiv.org/abs/2306.05284.

Shuqi Dai, Yunyun Wang, Roger B. Dannenberg, and Zeyu Jin. Everyone-Can-Sing: Zero-shot singing voice synthesis and conversion with speech reference, 2025. URL https://arxiv.org/abs/2501.13870.

Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Proceedings of Interspeech 2020*, pp. 3830–3834, 2020. doi: 10.21437/Interspeech.2020-2650. URL https://www.isca-archive.org/interspeech_2020/desplanques20_interspeech.html.

Chris Donahue, John Thickstun, and Percy Liang. Melody transcription via generative pre-training. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, 2022. URL https://arxiv.org/abs/2212.01884.

Alef Iury Siqueira Ferreira, Lucas Rafael Gris, Augusto Seben da Rosa, Frederico Santos de Oliveira, Edresson Casanova, Rafael Teixeira Sousa, Arnaldo Candido Junior, Anderson da Silva Soares, and Arlindo Galvão Filho. FreeSVC: Towards zero-shot multilingual singing voice conversion. In *2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2025. URL https://arxiv.org/abs/2501.05586.

Francesco Foscarin, Jan Schlüter, and Gerhard Widmer. Beat This! accurate beat tracking without DBN postprocessing. In *Proceedings of the 25th International Society for Music Information Retrieval Conference*, pp. 962–969, 2024. URL https://arxiv.org/abs/2407.21658.

Xiao Fu, Xin Yuan, and Jinglu Hu. HSD: A hierarchical singing annotation dataset. In *2022 IEEE International Symposium on Multimedia (ISM)*, pp. 245–246. IEEE, 2022. doi: 10.1109/ISM55400.2022.00054. URL https://doi.org/10.1109/ISM55400.2022.00054.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. In *Proceedings of Interspeech 2020*, pp. 5036–5040, 2020. doi: 10.21437/Interspeech.2020-3015. URL https://www.isca-archive.org/interspeech_2020/gulati20_interspeech.html.

Wenxiang Guo, Yu Zhang, Changhao Pan, Rongjie Huang, Li Tang, Ruiqi Li, Zhiqing Hong, Yongqi Wang, and Zhou Zhao. TechSinger: Technique controllable multilingual singing voice synthesis via flow matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23978–23986, 2025a. doi: 10.1609/aaai.v39i22.34571. URL https://ojs.aaai.org/index.php/AAAI/article/view/34571.

Wenxiang Guo, Yu Zhang, Changhao Pan, Zhiyuan Zhu, Ruiqi Li, Zhetao Chen, Wenhao Xu, Fei Wu, and Zhou Zhao. STARS: A unified framework for singing transcription, alignment, and refined style annotation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 15081–15093. Association for Computational Linguistics, 2025b. doi: 10.18653/v1/2025.findings-acl.781. URL https://aclanthology.org/2025.findings-acl.781/.

Jinzheng He, Jinglin Liu, Zhenhui Ye, Rongjie Huang, Chenye Cui, Huadai Liu, and Zhou Zhao. RMSSinger: Realistic-music-score based singing voice synthesis. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 236–248. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.16. URL https://aclanthology.org/2023.findings-acl.16/.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Leekyung Kim, Sungwook Jeon, Wan Heo, and Jonghun Park. Note-level singing melody transcription for time-aligned musical score generation. *IEEE Transactions on Audio, Speech and Language Processing*, 33:1088–1102, 2025. URL https://ieeexplore.ieee.org/document/10897320.

Ruiqi Li, Yu Zhang, Yongqi Wang, Zhiqing Hong, Rongjie Huang, and Zhou Zhao. Robust singing voice transcription serves synthesis. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9751–9766. Association for Computational Linguistics, 2024. URL https://aclanthology.org/2024.acl-long.526/.

Liwei Lin, Qiuqiang Kong, Junyan Jiang, and Gus Xia. A unified model for zero-shot music source separation, transcription and synthesis. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pp. 381–388, 2021. URL https://archives.ismir.net/ismir2021/paper/000047.pdf.

Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen, and Zhou Zhao. DiffSinger: Singing voice synthesis via shallow diffusion mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 11020–11028, 2022. doi: 10.1609/aaai.v36i10.21350. URL https://ojs.aaai.org/index.php/AAAI/article/view/21350.

Songxiang Liu, Yuewen Cao, Dan Su, and Helen Meng. DiffSVC: A diffusion probabilistic model for singing voice conversion. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021. URL https://arxiv.org/abs/2105.13871.

OpenVPI. GAME: Generative adaptive MIDI extractor. https://github.com/openvpi/GAME, n.d. Software repository, accessed September 17, 2026.

Jiale Qian, Hao Meng, Tian Zheng, Pengcheng Zhu, Haopeng Lin, Yuhang Dai, Hanke Xie, Wenxiao Cao, Ruixuan Shang, Jun Wu, Hongmei Liu, Hanlin Wen, Jian Zhao, Zhonglin Jiang, Yong Chen, Shunshun Yin, Ming Tao, Jianguo Wei, Lei Xie, and Xinsheng Wang. SoulX-Singer: Towards high-quality zero-shot singing voice synthesis, 2026. URL https://arxiv.org/abs/2602.07803.

Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. AutoVC: Zero-shot voice style transfer with only autoencoder loss. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 5210–5219. PMLR, 2019. URL https://proceedings.mlr.press/v97/qian19c.html.

Kaizhi Qian, Yang Zhang, Heting Gao, Junrui Ni, Cheng-I Lai, David Cox, Mark Hasegawa-Johnson, and Shiyu Chang. ContentVec: An improved self-supervised speech representation by disentangling speakers. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 18003–18017. PMLR, 2022. URL https://proceedings.mlr.press/v162/qian22b.html.

qiuqiao. SOFA: Singing-oriented forced aligner. https://github.com/qiuqiao/SOFA, n.d. Software repository, accessed September 23, 2026.

Kai Shen, Zeqian Ju, Xu Tan, Eric Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian. NaturalSpeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/035a73893121b4534bb3314e831050b1-Abstract-Conference.html.

Soul-AILab. SoulX-Singer: Official inference code and preprocessing documentation. https://github.com/Soul-AILab/SoulX-Singer, 2026. Accessed September 17, 2026.

Yuxun Tang, Jiatong Shi, Yuning Wu, and Qin Jin. SingMOS: An extensive open-source singing voice dataset for MOS prediction, 2024. URL https://arxiv.org/abs/2406.10911.

Jui-Te Wu, Jun-You Wang, Jyh-Shing Roger Jang, and Li Su. A unified model for zero-shot singing voice conversion and synthesis. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pp. 809–816, 2022. URL https://archives.ismir.net/ismir2022/paper/000097.pdf.

Kaituo Xu, Yan Jia, Kai Huang, Junjie Chen, Wenpeng Li, Kun Liu, Feng-Long Xie, Xu Tang, and Yao Hu. FireRedASR2S: A state-of-the-art industrial-grade all-in-one automatic speech recognition system, 2026. URL https://arxiv.org/abs/2603.10420.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

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, 2025a. URL https://arxiv.org/abs/2508.16332.

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 *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=anQDiQZhDP.

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, pp. 19597–19605, 2024a. doi: 10.1609/aaai.v38i17.29932. URL https://ojs.aaai.org/index.php/AAAI/article/view/29932.

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*, pp. 1960–1975. Association for Computational Linguistics, 2024b. doi: 10.18653/v1/2024.emnlp-main.117. URL https://aclanthology.org/2024.emnlp-main.117/.

# APPENDIX

## A IMPLEMENTATION DETAILS

### A.1 JOINT LYRICS–MELODY TRANSCRIBER

Table 7: Core configuration of the joint lyrics–melody transcriber.

| Component | Configuration |
|---|---|
| Parameters | 746.5 M |
| Audio input | 24 kHz; 14 s training clips |
| Features / frame rate | 128-bin log-mel / 50 Hz |
| Conformer | 22 layers; hidden size 896 |
| Lyric Transformer | 6 layers; hidden size 512 |
| Cross-attention | 8 layers; hidden size 896 |
| Autoregressive LSTM | 2 layers; hidden size 768 |

**Supervision construction.** The note annotations come from manually prepared MIDI scores, which are generally rhythmically aligned. We further estimate beats from the original full-mix recordings using Beat This! (Foscarin et al., 2024) and quantize MIDI note timing to a sixteenth-note grid derived from these estimates.

We construct automatic note–lyric assignments for approximately 2,500 hours of Mandarin songs. Forced alignment with FireRedASR2S (Xu et al., 2026) provides start and end times for each character in the supplied lyrics. We assign each note to the character whose aligned interval has the greatest temporal overlap with the note interval, retaining its position in the supplied lyrics. Notes with no overlapping character remain unassigned, and lyric characters not assigned to any note remain unmatched; we apply no further correction to these cases. Character timestamps are used only to construct note–lyric assignments; the position and pronunciation targets follow the note intervals, as described in Section 3.1.

For a disjoint 80-hour set of Mandarin songs, we obtain manually prepared MIDI scores and initialize note–lyric assignments with the same forced-alignment and overlap procedure. Human annotators then review and correct these assignments. We use this manually verified set to fine-tune the transcriber after its three-stage training on the automatically annotated corpus.

**Training and inference settings.** The transcriber (Table 7) is trained with AdamW on four NVIDIA H20 GPUs at a global batch size of 16. Stage 1 runs for 70,000 steps at learning rate $3 \times 10^{-5}$; Stages 2 and 3 each run for 3,000 steps, followed by 300 fine-tuning steps, all at $2 \times 10^{-6}$. Final Stage 3 loss weights $(\lambda_{\text{note}}, \lambda_{\text{pos}}, \lambda_{\text{pron}}, \lambda_{\text{cons}})$ are $(1.5, 1.0, 0.2, 1.0)$, and the final scheduled-sampling probability is 0.5. Whole-song inference uses 30 s windows with 5 s overlap, stitching matched notes and lyric-position sequences across windows.

### A.2 REFERENCE-CONDITIONED SINGING GENERATOR

Table 8: Core configuration of the singing generator.

| Component | Configuration |
|---|---|
| Trainable parameters | Approximately 1.43 B |
| Qwen3 backbone | 28 layers; hidden size 2,048 |
| Feed-forward dimension | 6,144 |
| Attention / KV heads | 16 / 8 |
| Maximum sequence length | 40,960 tokens |
| Audio representation | 8 codebooks at 25 Hz |
| Training segments | 15 s target and 15 s reference |

**Token representation.** We serialize the structured score in Equation 1 by quantizing note timing on a 25 Hz grid (40 ms per frame). Each note interval is filled with its MIDI pitch, and its final frame is replaced by an ending marker: zero for an ordinary ending, or a continuation token if the following note shares its lyric position. Rest frames also use zero. Replacing the final frame preserves the grid length. Sentence separators mark corresponding boundaries in the melody and lyric streams; training sentence boundaries are supplied by the dataset.

A note group consists of consecutive notes sharing a lyric position $j_i$ and contributes its pronunciation token $q_{j_i}$ once to the lyric stream. Groups follow score order and may contain a single note or a melisma; identical pronunciations at different lyric positions remain separate. These groups define score-based syllable spans. Acoustic pronunciation timing is learned from target vocals during training and guided by reference conditioning at inference. No acoustic syllable timestamps or explicit breath annotations are included in the symbolic input.

The following schematic example shows the melody stream $\mathbf{m}$, lyric stream $\mathbf{l}$, and conditioning layout $\mathcal{U}$ before codebook expansion and delay processing:

$$\begin{aligned}
\mathbf{m} &= (\underbrace{p_1, \ldots, p_1}_{d_1-1}, 0, \underbrace{0, \ldots, 0}_{r}, \underbrace{p_2, \ldots, p_2}_{d_2-1}, \langle\mathrm{CONT}\rangle, \underbrace{p_3, \ldots, p_3}_{d_3-1}, 0, \langle\mathrm{SEP}\rangle, \ldots), \\
\mathbf{l} &= (\ell_1, \ell_2, \langle\mathrm{SEP}\rangle, \ell_3, \ldots), \\
\mathcal{U} &= [\,\langle\mathrm{MIDI}\rangle, \mathbf{m}, \langle/\mathrm{MIDI}\rangle;\ \langle\mathrm{LYR}\rangle, \mathbf{l}, \langle/\mathrm{LYR}\rangle; \\
&\qquad \langle\mathrm{REF}\rangle, \mathbf{c}_{\mathrm{ref}}, \langle/\mathrm{REF}\rangle;\ \langle\mathrm{SPK}\rangle, \mathbf{e}_{\mathrm{spk}}\,].
\end{aligned} \tag{5}$$

Here, note $i$ has pitch $p_i$ and duration $d_i$ frames on the 25 Hz grid, including its ending marker; $r$ denotes the intervening rest duration in frames. The tokens $\ell_1, \ell_2, \ldots$ denote group pronunciations $q_{j_i}$ in score order. The first note corresponds to $\ell_1$, while the second and third share $\ell_2$ through $\langle\mathrm{CONT}\rangle$. The $\langle\mathrm{SEP}\rangle$ markers separate sentences in both streams. The reference codec tokens are denoted by $\mathbf{c}_{\mathrm{ref}}$. The speaker slot receives the projected continuous vector $\mathbf{e}_{\mathrm{spk}}$. Descriptive marker names stand for implementation-specific token IDs; Equation 4 omits these markers and shows only the four embedding blocks.

We use separate embeddings for the eight codebook channels and concatenate them along the feature dimension at each sequence position. Symbolic melody and pronunciation tokens are replicated across these channels. For audio tokens, codebook $k$ is shifted by $k-1$ steps to form the delay pattern; the shifts are reversed before codec decoding. The speaker embedding is projected into the Transformer's input space, and the conditioning blocks are concatenated along the sequence dimension.

**Training.** The generator configuration is given in Table 8. We train from scratch for 1,000,000 steps using AdamW, a peak learning rate of $10^{-4}$ with cosine decay, a global batch size of 512, and bfloat16 precision. The released codec and speaker encoder remain frozen. Training references are randomly sampled from regions of the same song that do not overlap the target segment. The speaker embedding is extracted from the full song's dry vocals. Reference audio is dropped with probability 0.15, and speaker embeddings are zeroed with probability 0.5.

The objective sums equally weighted next-token cross-entropy losses over the eight codebooks:

$$\mathcal{L}_{\mathrm{gen}} = -\sum_{k=1}^{8} \frac{1}{|\mathcal{V}_k|} \sum_{u \in \mathcal{V}_k} \log p_\theta(\tilde{c}_{u,k} \mid \mathcal{P}, \tilde{\mathbf{c}}_{<u})\,. \tag{6}$$

Here, $\tilde{c}_{u,k}$ is the target token at step $u$ in the shifted stream of codebook $k$, and $\tilde{\mathbf{c}}_{<u}$ contains the preceding shifted audio tokens. The set $\mathcal{V}_k$ contains valid prediction positions in target stream $k$; prefix and padding positions are excluded from the loss.

**Inference.** At inference, the supplied reference clip provides both the codec reference and the speaker embedding. Clips longer than 15 s are truncated to 15 s for codec encoding, while the speaker embedding is extracted from the entire supplied clip. Previous-phrase codec context is used only at inference for song continuation. These tokens follow $\mathcal{P}$ and precede newly generated tokens. Classifier-free guidance contrasts the full conditioning with a reduced branch that retains MIDI, omits lyrics and reference audio, and disables the speaker embedding. SVS evaluation uses

CFG scale 1.5 applied to log-softmax scores, temperature 0.8, top-$k = 4$, and top-$p = 0.9$. After reversing the codebook shifts, the codec decoder reconstructs 24 kHz segments. We place them on the score timeline and crossfade their overlaps, then apply audio super-resolution to produce the final 48 kHz waveform.

# B Subjective Evaluation

## B.1 Listening Protocol

**Participants and case selection.** For each task, we randomly sample a pool of 100 cases successfully generated by all systems from the objective evaluation set. Thirty-two Mandarin-speaking listeners evaluate subsets of these pools.

**Questionnaire structure.** The current questionnaire has three parts: timbre/style similarity, naturalness, and singing accuracy. Under this protocol, each listener is assigned three SVS and three SVC excerpts, reused across parts; allocation favors less-assigned excerpts. Part and excerpt order are randomized. All systems are anonymized, with labels counterbalanced across the pool.

**Rating instructions.** Listeners rate each dimension separately from 1 (worst) to 5 (best), using definitions and scoring anchors following Vevo2 (Zhang et al., 2025a). Table 9 specifies references and criteria. References are not rated; lyrics are shown for accuracy. Listeners are instructed to use headphones and may replay audio.

Table 9: References and rating criteria for the four listening dimensions.

| Dimension | Reference | Rating focus |
|---|---|---|
| Timbre similarity | SVS: singer prompt.<br>SVC: target singer's dry vocal. | Vocal identity similarity; lyric or pitch errors alone should not lower this score. |
| Style similarity | Same as timbre similarity. | Similarity of articulation, breathiness, vibrato, and emotion, assessed separately from timbre. |
| Naturalness | None. | Fluency and human-likeness, including audible artifacts; similarity and singing accuracy are not judged. |
| Singing accuracy | SVS: target ground-truth vocal.<br>SVC: source dry vocal. | Accuracy of lyrics, pitch, and rhythm, accounting for omissions and timing errors; ignore timbre differences. |

**Aggregation and sample counts.** Each system receives 85–89 ratings per task and dimension. Scores are pooled means of available ratings. Paired comparisons use listener–case pairs with ratings for both systems within a dimension.

## B.2 Paired Statistical Analysis

**Tests and confidence intervals.** We use two-sided sign-flip tests on paired $S^3$–baseline rating differences. All differences from each listener are flipped together; we enumerate all sign assignments using the pooled mean difference as the statistic. The tests assume independent listeners and sign symmetry under the null. Holm correction (Holm, 1979) controls the family-wise error rate at 0.05 across all baseline–dimension comparisons within each task.

Tables 10 and 11 report mean differences, Holm-adjusted $p$-values, and marginal 95% confidence intervals from inversion of the unadjusted tests under a common location-shift assumption. Significance is assessed at adjusted $p < 0.05$; the intervals are not multiplicity-adjusted.

Table 10: Paired SVS comparisons. $\hat{\Delta}$ is the mean rating difference ($S^3$ minus baseline); CIs are marginal 95% intervals. $p_{\mathrm{adj}}$ uses Holm correction across 12 comparisons.

| Baseline | Dimension | $\hat{\Delta}$ | 95% CI | $p_{\mathrm{adj}}$ |
|---|---|---|---|---|
| SoulX-Score | Timbre similarity | 0.437 | [0.286, 0.613] | $< 0.0001$ |
| | Style similarity | 0.791 | [0.625, 0.971] | $< 0.0001$ |
| | Naturalness | 1.081 | [0.892, 1.275] | $< 0.0001$ |
| | Singing accuracy | 0.506 | [0.315, 0.710] | $< 0.0001$ |
| TechSinger | Timbre similarity | 2.322 | [2.091, 2.568] | $< 0.0001$ |
| | Style similarity | 2.276 | [2.000, 2.579] | $< 0.0001$ |
| | Naturalness | 2.161 | [1.843, 2.500] | $< 0.0001$ |
| | Singing accuracy | 1.864 | [1.571, 2.211] | $< 0.0001$ |
| TCSinger | Timbre similarity | 2.207 | [2.034, 2.400] | $< 0.0001$ |
| | Style similarity | 2.402 | [2.140, 2.692] | $< 0.0001$ |
| | Naturalness | 2.805 | [2.575, 3.030] | $< 0.0001$ |
| | Singing accuracy | 2.307 | [2.063, 2.585] | $< 0.0001$ |

Table 11: Paired SVC comparisons. $\hat{\Delta}$ is the mean rating difference ($S^3$ minus baseline); CIs are marginal 95% intervals. $p_{\mathrm{adj}}$ uses Holm correction across 16 comparisons.

| Baseline | Dimension | $\hat{\Delta}$ | 95% CI | $p_{\mathrm{adj}}$ |
|---|---|---|---|---|
| SoulX-Melody | Timbre similarity | 1.012 | [0.743, 1.263] | $< 0.0001$ |
| | Style similarity | 1.302 | [0.979, 1.658] | $< 0.0001$ |
| | Naturalness | 1.535 | [1.258, 1.795] | $< 0.0001$ |
| | Singing accuracy | 1.376 | [1.029, 1.744] | $< 0.0001$ |
| Vevo2-Style | Timbre similarity | 1.349 | [0.935, 1.750] | $< 0.0001$ |
| | Style similarity | 1.651 | [1.278, 2.037] | $< 0.0001$ |
| | Naturalness | 2.244 | [1.882, 2.567] | $< 0.0001$ |
| | Singing accuracy | 2.588 | [2.300, 2.849] | $< 0.0001$ |
| SoulX-SVC | Timbre similarity | 0.733 | [0.531, 0.947] | $< 0.0001$ |
| | Style similarity | 0.977 | [0.700, 1.303] | $< 0.0001$ |
| | Naturalness | 0.895 | [0.586, 1.188] | 0.0001 |
| | Singing accuracy | 0.106 | [−0.170, 0.424] | 0.4982 |
| Vevo2-FM | Timbre similarity | 0.872 | [0.512, 1.194] | 0.0002 |
| | Style similarity | 1.023 | [0.690, 1.341] | $< 0.0001$ |
| | Naturalness | 0.860 | [0.561, 1.171] | 0.0001 |
| | Singing accuracy | 0.706 | [0.321, 1.150] | 0.0020 |

# C Evaluation and Baseline Details

**Vocal preparation.** For SVS and SVC, reference recordings and vocal ground truth are prepared through accompaniment separation, backing-vocal removal, and dereverberation to obtain dry lead vocals.

## C.1 Joint Lyrics–Melody Transcription

**ROSVOT and GAME.** ROSVOT (Li et al., 2024) is a robust vocal-note transcriber developed for constructing singing-synthesis data. GAME (OpenVPI, n.d.) uses generative boundary extraction and adaptive note and pitch prediction to recover a score from singing. In our evaluation, both receive separated vocals and predict notes without supplied lyrics. Their note outputs are evaluated directly for transcription and paired with SOFA for note–lyric assignment.

**SheetSage-Jukebox.** SheetSage (Donahue et al., 2022) uses representations from a pretrained music-generation model for lead-sheet transcription. We evaluate its Jukebox-based melody transcription on the original full mix, without supplied lyrics. Only its melody notes enter the transcription metrics and subsequent SOFA-based assignment.

**STARS.** STARS (Guo et al., 2025b) combines singing transcription, lyric alignment, and style annotation. We use its reference-lyric setting with separated vocals and the supplied lyrics, evaluating its predicted notes and note–lyric assignments. Style annotations are not part of the metrics reported here.

**SOFA-based assignment.** For note–lyric assignment, we use SOFA (qiuqiao, n.d.) with separated vocals and supplied lyrics to obtain assignments for the notes predicted by ROSVOT, GAME, and SheetSage-Jukebox. These combinations are reported as SOFA + the corresponding transcriber in Table 2. The SOFA + GT MIDI reference condition uses ground-truth notes to evaluate lyric assignment independently of note transcription errors.

**Timeline matching and calibration.** COn measures one-to-one onset matching with a 100 ms tolerance; COnP additionally requires pitches to agree within 50 cents. To account for systematic timing differences between predictions and annotations, we select one global time shift per system, shared by all 57 songs and used only for evaluation. We search $[-200, 200]$ ms in 1 ms increments, maximizing the mean of dataset-level COnP F1 and COnPOff F1, the latter additionally requiring an offset match. No per-song shift is applied. Reported metrics are averaged over songs.

**Symbolic-score metrics.** We follow the sequence edit-distance definitions used in T3MS (Kim et al., 2025). PER, VER, and NER measure errors in pitch, duration, and joint pitch–duration symbols, respectively, normalized by the number of reference notes. Reference durations are expressed in sixteenth-note units, and predicted durations are quantized using a beat grid extracted from the original full mix. All methods share the same beat grid for each song.

**Note–lyric assignment metrics.** We use normalized toneless pinyin as lyric labels. We first compute the maximum one-to-one note matching under the same onset and pitch tolerances, then add pronunciation agreement to compute the number of correct joint matches. Assignment Accuracy is the ratio of correct joint matches to the maximum number of onset–pitch matches, measuring lyric assignment conditional on note matching. Joint Note–Lyric Precision and Recall divide the correct joint count by all predicted notes and all reference notes with pinyin labels, respectively; F1 is their harmonic mean. Metrics are computed per song and averaged over HSD57.

### C.2 Singing Voice Synthesis

**Case construction.** SVS and SVC share the same excerpt construction: segments contain consecutive complete lyric lines and are capped at 15 s, with the opening vocal excerpt of the target singer's song serving as the reference. Excerpts and time boundaries are fixed across systems. For SVS, each of the 36 singers provides a reference for synthesizing subsequent segments of the same song, yielding 36 references and 557 targets. Reference and ground-truth audio use dry lead vocals. All systems receive the same ground-truth MIDI, lyrics, and reference; $S^3$ encodes the notes and note–lyric assignments as its structured score.

**Evaluation.** Objective metrics are averaged over the 557 target segments. WavLM cosine similarity compares each output with its input reference, and FFE compares it with the ground-truth target singing after temporal alignment. Both objective and subjective evaluations of $S^3$ use the final waveform after audio super-resolution.

**SoulX-Score.** This name denotes the official score-conditioned mode of SoulX-Singer (Qian et al., 2026; Soul-AILab, 2026). We supply ground-truth MIDI and a ground-truth, time-aligned phoneme sequence together with the shared target-singer reference recording. This differs from SoulX-Melody in SVC, whose F0 and time-aligned phonemes are automatically extracted by SoulX's own pipeline.

**TCSinger.** TCSinger (Zhang et al., 2024b) models reference singing style through a clustering style encoder, a style-and-duration language model, and a style-adaptive acoustic decoder. We evaluate reference-guided synthesis using ground-truth MIDI, lyrics, phoneme timestamps, and the shared reference recording.

**TechSinger.** TechSinger (Guo et al., 2025a) uses flow matching for singing synthesis and supports phoneme-level vocal-technique control. Our evaluation supplies ground-truth MIDI, lyrics, phoneme timestamps, and the shared reference recording, without additional singing-technique labels.

### C.3 Singing Voice Conversion

**Case construction.** Using the excerpt construction in Appendix C.2, SVC pairs each of the 36 source singers with the other 35 target singers, yielding 1,260 directed pairs. For each pair, we sample a source excerpt and use the target singer's reference. All systems share the cases retained after filtering to limit source-separation artifacts. $S^3$ recovers its conditioning score from the source full mix and supplied lyrics, then generates singing with the target reference.

**Baseline inputs.** The four baseline configurations use dry lead vocals as source input. Their conditioning differs as follows.

**SoulX-Melody.** We use SoulX-Singer's melody-conditioned synthesis mode (Qian et al., 2026; Soul-AILab, 2026). SoulX's own preprocessing pipeline automatically extracts the F0 contour and a time-aligned phoneme sequence from the source vocals. These extracted conditions, together with the target-singer reference, drive synthesis. No ground-truth MIDI, phoneme sequence, or phoneme timestamps are supplied for the source; both pronunciation and its timing come from the automatic pipeline.

**SoulX-SVC.** This is SoulX-Singer's audio-to-audio conversion configuration (Soul-AILab, 2026). We provide only the source vocals and target-singer reference as external inputs. SoulX's own pipeline extracts F0 and Whisper features from the source recording to condition conversion. No external lyrics, phoneme timestamps, or MIDI are supplied.

**Vevo2-Style.** We use Vevo2's style-conversion mode (Zhang et al., 2025a), comprising autoregressive content–style token generation followed by flow-matching acoustic generation. Source-derived prosody tokens provide melody and rhythm conditioning, while the target reference guides style and timbre. We supply verified lyrics for both the source and target reference, bypassing the system's automatic lyric-transcription step. The melody is still derived from source audio; no ground-truth MIDI is supplied.

**Vevo2-FM.** This configuration uses only Vevo2's flow-matching stage (Zhang et al., 2025a). Content–style tokens extracted directly from the source vocals condition acoustic generation with the target reference. It bypasses autoregressive content–style generation and requires no external lyrics or score. This mode is designed to preserve source content and style while converting timbre, providing a complementary comparison to Vevo2-Style.

## D Joint Segmental Decoding Details

### D.1 Candidate Construction

The decoder uses the model's note-state, lyric-position, and pronunciation probabilities. Following Section 3.2, a candidate $u = (a_u, b_u, p_u)$ specifies its onset frame, offset frame, and pitch; $j \in \{0, \ldots, L-1\}$ indexes the supplied lyrics of length $L$.

Raw decoding first produces initial notes through a local beam search scored by note-state probabilities and structural priors, without explicitly including lyric-position or pronunciation probabilities. Each initial note then anchors a small set of high-scoring nearby onset and offset positions. The pitch alternatives include the original pitch and pitches with strong sustain evidence within the segment.

We enumerate local combinations of these onsets, offsets, and pitches, discard segments outside the audio or allowed duration range, and pre-score the remaining segments using onset, sustain, and end-boundary evidence with a penalty for deviation from the original boundaries. Each anchor retains a small number of high-scoring variants together with the original note. No additional split or merge proposals are generated in this experiment; dynamic programming subsequently searches paths through this candidate lattice.

### D.2 Joint Scoring

Each candidate–lyric pair combines note-state, lyric-position, and pronunciation evidence. The note-state score includes onset, within-segment sustain, and end-boundary evidence. Sustain evidence is the mean log-probability ratio relative to silence over the interior frames, excluding the boundaries. End-boundary evidence combines offset, sustain, silence, and onset evidence at the next frame.

Lyric evidence likewise uses the respective null classes as references. Let $e_{\mathrm{pos}}(t, j)$ be the log-probability ratio of lyric position $j$ to the null position, and let $e_{\mathrm{pron}}(t, j)$ be the log-probability ratio of that position's pronunciation to the null pronunciation class. Their weighted average over the candidate interval is

$$E_{\mathrm{lyr}}(u, j) = \frac{1}{b_u - a_u} \sum_{t=a_u}^{b_u-1} \frac{w_{\mathrm{pos}} e_{\mathrm{pos}}(t, j) + w_{\mathrm{pron}} e_{\mathrm{pron}}(t, j)}{w_{\mathrm{pos}} + w_{\mathrm{pron}}}. \tag{7}$$

We combine this lyric evidence with note-state evidence, a boundary-deviation penalty, and a note-selection bias. For lyric advancement, the score uses the combined onset and re-onset evidence and is denoted $S_{\mathrm{HEAD}}(u, j)$. For continuation at the same lyric position, it instead uses slur-onset evidence and is denoted $S_{\mathrm{STAY}}(u, j)$. These are the two cases of $s(u, j; k)$ in Equation 3, for $k < j$ and $k = j$, respectively.

Changing candidate boundaries changes the note-state and lyric evidence included in the score. The same candidate also uses different onset evidence for lyric advancement and melismatic continuation. Lyric assignment and note selection therefore constrain each other through the joint score.

### D.3 Dynamic Programming and Output

Let $D(u, j)$ be the best path score ending at candidate $u$ and lyric position $j$. Candidates are processed by onset time, and only notes ending no later than the current onset can serve as predecessors. The lyric-advancement and same-position continuation branches are

$$\begin{aligned} H(u, j) &= S_{\mathrm{HEAD}}(u, j) + \max_{\substack{b_v \le a_u \\ k < j}} \left[ D(v, k) - c(k, j) \right], \\ M(u, j) &= S_{\mathrm{STAY}}(u, j) + \max_{v \in \mathcal{V}_{\mathrm{stay}}(u)} D(v, j) - w_{\mathrm{stay}}. \end{aligned} \tag{8}$$

Here, $c(k, j)$ charges $w_{\mathrm{skip}}$ for each skipped lyric position and an additional gap-open cost when any position is skipped; advancing to the immediately following position incurs no skip cost. The set $\mathcal{V}_{\mathrm{stay}}(u)$ contains completed predecessor notes satisfying the maximum-gap constraint, and $w_{\mathrm{stay}}$ is the same-position continuation cost. A maximum over an empty predecessor set is $-\infty$.

A path may also start directly at lyric position $j$, paying $j w_{\mathrm{skip}}$ for leading skipped positions. Comparing this start branch with the two transitions gives

$$D(u, j) = \max \left\{ S_{\mathrm{HEAD}}(u, j) - j w_{\mathrm{skip}},\ H(u, j),\ M(u, j) \right\}. \tag{9}$$

Each update stores the best predecessor. After all candidates have been processed, we select the terminal state by maximizing $D(u, j) - (L - 1 - j) w_{\mathrm{skip}}$, accounting for trailing skipped lyrics, and backtrack. The search can discard candidate notes or skip lyric positions, yielding the best non-overlapping, monotonic path within the fixed candidate lattice.

The implementation uses temporal prefix sums to compute segment-level lyric evidence efficiently. Per-position historical best scores and prefix maxima avoid repeatedly enumerating all predecessors, while STAY retains only predecessors within the allowed gap.

After backtracking, frame boundaries are converted to time, and lyric characters and pronunciation labels are read from the assigned positions. Adjacent notes are merged only when they share the same lyric position, pronunciation, and pitch and have a sufficiently short gap. Notes at the same lyric position then define melisma groups with head and continuation labels for the generator. Scoring weights and search settings are selected on a separate development set and frozen before final evaluation.

## E LIMITATIONS

Training and evaluation focus on Mandarin singing. Although the framework can be extended to other languages with suitable pronunciation representations and training data, its performance beyond Mandarin has not been systematically evaluated.