STYLEPITCHER: GENERATING STYLE-FOLLOWING AND EXPRESSIVE PITCH CURVES FOR VERSATILE SINGING TASKS

Jingyue Huang$^{\triangle\star}$ *Qihui Yang*$^{\triangle\star}$ *Fei-Yueh Chen*†* *Julian McAuley*$^{\triangle}$
Randal Leistikow$^{\circ}$ *Perry R. Cook*$^{\circ}$ *Yongyi Zang*$^{\circ}$

$^{\triangle}$ University of California San Diego † University of Rochester $^{\circ}$ Smule Labs

ABSTRACT

Existing pitch curve generators face two main challenges: they often neglect singer-specific expressiveness, reducing their ability to capture individual singing styles. And they are typically developed as auxiliary modules for specific tasks such as pitch correction, singing voice synthesis, or voice conversion, which restricts their generalization capability. We propose StylePitcher, a general-purpose pitch curve generator that learns singer style from reference audio while preserving alignment with the intended melody. Built upon a rectified flow matching architecture, StylePitcher flexibly incorporates symbolic music scores and pitch context as conditions for generation, and can seamlessly adapt to diverse singing tasks without retraining. Objective and subjective evaluations across various singing tasks demonstrate that StylePitcher improves style similarity and audio quality while maintaining pitch accuracy comparable to task-specific baselines.

Index Terms— F0 Estimation, Singing Voice Synthesis and Conversion, Pitch Correction, Rectified Flow Matching

1. INTRODUCTION

Pitch curves, or fundamental frequency (F0) curves, are the backbone of expressive singing. They encode not only the melody but also the subtle variations that define unique styles of different singers, such as their vibrato, ornaments, pitch bending, and others [1]. Therefore, pitch curves serve as critical intermediate representations across diverse singing generation and conversion tasks, such as automatic pitch correction (APC) [2–4], singing voice synthesis (SVS) [5–10], and singing voice conversion (SVC) [11–17].

Despite their importance, existing approaches face two main limitations. First, most of them overlook singer-specific styles encoded in pitch curves, treating pitch as a singer-agnostic feature and reusing the same curve across different singers [11, 12]. This limitation is critical: the same melody sung by different singers can produce distinct pitch patterns, which reflect individual singing techniques and styles. Losing these patterns will neglect their singer-specific expressiveness and the essence of their performance [8]. Second, while some recent approaches [4, 9] support style-informed pitch curve generation, they often develop the pitch curve generator as an auxiliary module for specific tasks, such as pitch correction [4] and singing voice synthesis [9]. This task-specific design constrains their generalization capability across different singing applications, as researchers have to retrain these modules with different hyperparameters, inputs and outputs for each adaptation. Therefore, a general-purpose model capable of generating style-following pitch curves is essential for diverse singing applications.

We propose **StylePitcher**, the first style-following pitch curve generation model for versatile singing tasks. We formulate pitch curve generation as a masked infilling problem: given surrounding pitch context and symbolic music scores, StylePitcher learns to generate missing pitch segments that continue the style patterns from context. This approach enables implicit style modeling without requiring explicit singer labels or embeddings, allowing generalization to unseen voices. We employ a rectified flow model [18] for stable, efficient, and high-quality generation process. In addition, we introduce a smoothing algorithm to construct reliable conditioning signals (i.e., symbolic music scores), removing the need for manual annotations. By separately modeling F0 and performing inpainting, StylePitcher generates pitch curves that follow the style of provided audio without any task-specific retraining. Once trained, it serves as a plug-and-play module for diverse applications, including pitch correction, zero-shot singing voice synthesis with style transfer and style-informed singing voice conversion.

Our contributions are threefold:

- StylePitcher[1] is the first general-purpose, style-following pitch curve generator supporting diverse singing tasks.
- We introduce a flow matching architecture to pitch curve generation, a smoothing algorithm for data annotation, and an inpainting mechanism for flexible task adaptation.
- Objective and subjective evaluations across multiple singing tasks show that StylePitcher achieves superior or compatible performance on style similarity, audio quality and pitch accuracy relative to previous baselines.

*Work done during internship at Smule Labs.

[1]Demos available at `https://stylepitcher.github.io/`

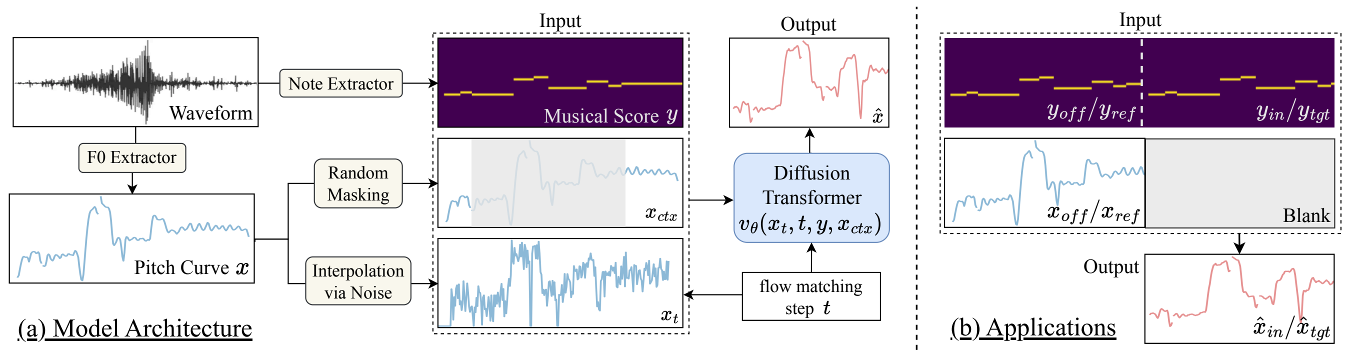

Fig. 1. Illustration of the methods. The unvoiced condition is omitted for clarity. Subscripts `off` and `in` denote features from off-key and in-key singing in the APC task; `ref` and `tgt` refer to reference and target content for SVS and SVC tasks.

2. RELATED WORK

Many singing generation frameworks explicitly predict pitch curves as an intermediate step. In automatic pitch correction, models generate pitch contours conditioned on target note sequences to correct out-of-tune vocals [2–4]. Singing voice synthesis aims to generate singing voice from lyrics and scores, with many approaches predicting F0 to model pitch variation [5–8, 10]. Specifically, StyleSinger [9] adopts style-specific and style-agnostic pitch predictors to capture singing styles from references. Singing voice conversion models transform the voice in a singing signal into that of a target singer while preserving content and melody. Most existing works [11–17] treat pitch as singer-agnostic, reusing the unchanged or key-shifted F0 sequences from the original singing, which converts timbre but overlooks singer-specific expressiveness. Unlike prior works that embed task-specific F0 predictors within complex frameworks, the proposed StylePitcher is a plug-and-play module for diverse singing tasks, supporting style preservation or transfer as needed.

3. METHODS

3.1. Rectified Flow

Rectified flow [18] defines generative modeling as learning a deterministic transport map from a prior distribution π_0 (e.g. Gaussian) to the data distribution π_1. It parameterizes a time-dependent velocity field $v_\theta(x_t, t, c)$ to push $x_0 \sim \pi_0$ towards $x_1 \sim \pi_1$ under the ordinary differential equation (ODE):

$$dx_t = v_\theta(x_t, t, c)dt \tag{1}$$

where c denotes conditions, $t \in [0, 1]$ is the scalar time parameter, and x_t follows a rectified linear interpolation:

$$x_t = (1 - t)x_0 + tx_1. \tag{2}$$

The model is trained with the flow matching objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0 \sim \pi_0, x_1 \sim \pi_1, t, c} \| v_\theta(x_t, t, c) - (x_1 - x_0)\|_2^2. \tag{3}$$

Once trained, starting from π_0, samples from π_1 are obtained by integrating the learned ODE from $t = 0$ to $t = 1$, producing high-quality results with few sampling steps.

3.2. Model Architecture

As illustrated in Fig. 1, we formulate pitch curve generation as a conditional infilling task, following Voicebox [19]. Given a fundamental frequency curve $x = (x^1, \cdots, x^N)$ from a singing voice, the corresponding note sequence $y = (y^1, \cdots, y^N)$, and a binary mask $m \in \{0, 1\}^N$, our model $p(x_{\text{mask}}|y, x_{\text{ctx}})$ predicts the masked segments $x_{\text{mask}} = m \odot x$ conditioned on the complete note sequence y and the context $x_{\text{ctx}} = (1 - m) \odot x$. Through in-context learning, the generated segments implicitly follow the singing style of the surrounding context remaining aligned with the target musical score.

We adopt rectified flow with a diffusion transformer [20, 21] to parameterize the velocity field $v_\theta(x_t, t, y, x_{\text{ctx}})$, where the full signal x is modeled instead of x_{mask} for simpler conditioning, and x_t is the linear interpolation between noise $\epsilon \sim \mathcal{N}(0, 1)$ and x at flow matching step t.

Specifically, the pitch curves x_t and x_{ctx} are linearly projected to embeddings of shape $(N, H_1 = 512)$, and notes $y \in [M]^N$ are projected to $(N, H_2 = 256)$, where M is the number of pitch classes. These three embeddings are concatenated along the frame dimension and projected to the embeddings for next layers. Additionally, an unvoiced indicator sequence $u \in \{0, 1\}^N$ is incorporated to align the generated F0 with singing phonemes. Finally, a sinusoidally positional-encoded flow step t modulates the representation to form the transformer input.

The training objective is:

$$\mathcal{L}_{pitch}(\theta) = \mathbb{E}_{\epsilon, p(x,y), t, m} \| m \odot [v_\theta(x_t, t, y, x_{\text{ctx}}) - (x - \epsilon)] \|_2^2 \tag{4}$$

where the loss is computed only on masked frames [19]. Classifier-free guidance (CFG) [22] is employed by randomly dropping conditions y, x_{ctx} and u with probability p_c for training. During inference, samples are generated by integrating the ODE for K steps with modified velocity field $\hat{v}_\theta$ and CFG scale α:

$$\hat{v}_\theta = v_\theta(x_t, t, \varnothing) + \alpha[v_\theta(x_t, t, y, x_{\text{ctx}}) - v_\theta(x_t, t, \varnothing)]. \tag{5}$$

3.3. Applications

As a standalone model, StylePitcher performs task-agnostic pitch curve generation via conditional inpainting, supporting style preservation or transfer across diverse singing applications without the need for re-training, as shown in Fig. 1(b).

Automatic Pitch Correction Given off-key singing with F0 x_{off}, notes y_{off}, unvoiced sequence u_{off}, and target notes y_{in}, we construct the input as $x = (x_{\text{off}}, 0)$ and generate $\hat{x}$ conditioned on $y = (y_{\text{off}}, y_{\text{in}})$ and $u = (u_{\text{off}}, u_{\text{off}})$. The corrected pitch $\hat{x}_{\text{in}}$ is obtained from the latter portion of $\hat{x}$, preserving the original singing style and matching the target notes.

Zero-Shot SVS with Style Transfer Given reference singing (x_{ref}, y_{ref}, u_{ref}) and target content from an SVS model (x_{tgt}, y_{tgt}, u_{tgt}), we concatenate the three sequences and mask the target segment x_{tgt}. StylePitcher then generates $\hat{x}_{\text{tgt}}$ that aligns with the target notes while following the reference style, which replaces x_{tgt} for SVS synthesis.

Style-informed SVC Unlike previous SVC models using unchanged F0, we modify pitch contours to capture singing style. Given reference and target audio features concatenated as $x = (x_{\text{ref}}, x_{\text{tgt}})$ with corresponding y and u, we mask and regenerate x_{tgt} to obtain $\hat{x}_{\text{tgt}}$, enabling the converted singing to transfer both timbre and pitch style while preserving content.

4. EXPERIMENTS

4.1. Datasets and Data Processing

For training, we employ two multi-speaker singing datasets, DAMP-VSEP [23] and DAMP-VPB [24], totaling 1916 hours of singing voice. For evaluation, different test sets are used depending on the task: (1) Samples from Diff-Pitcher [4] for pitch correction; (2) GTSinger [25] for singing voice synthesis and conversion; (3) VocalSet [26] for technique diversity.

We adopt RMVPE [27] for F0 estimation and unvoiced detection (16 kHz, 1024 frame size, 160 hop size); Basic Pitch [28] for MIDI extraction. Empirically, we replace the multi-pitch activation of Basic Pitch with that of RMVPE to obtain more accurate MIDI extraction results. The output F0 spans *C1* (32.7 Hz) to *B6* (1975.5 Hz), covering $M = 72$ note classes. Audio is pre-processed with vocal separation and denoising before F0 extraction. We observe that the extracted MIDI still contains style information expressed as short notes. To remedy, we apply Gaussian blur on the multi-pitch activation map to smooth expressive techniques (e.g., vibrato) and post-process by removing short rests and notes.

4.2. Experimental Setting

We use an 8-layer, 8-head diffusion transformer (DiT) with 512 hidden dim. and rotary position embeddings [29], totaling 49M parameters. The maximum sequence length is N=1024 frames (20.48 seconds at 50 Hz). We apply cosine schedule [30] to focus on lower t values, mask $r\%$ of sequences for infilling ($r \sim \mathcal{U}[70, 100]$), and set CFG drop probability $p_c = 0.5$. Pitch curves and notes are augmented by random shifts within $[-4, 4]$ semitones. The model is pretrained for 100k steps without unvoiced conditioning (learning rate 1e-4) and fine-tuned for 90k steps with it (1e-5), using 5k-step linear warm-up, AdamW [31] optimizer, cosine scheduler, and batch size 512. Training is done with FlashAttention-2 [32]. During inference, we use the midpoint solver of torchdiffeq [33] with K=16 steps and CFG scale α=1.25. Generated F0 curves can be interpolated to match the expected F0 sampling rates for downstream singing tasks.

Models	**RPA ↑**	**PCA ↑**	**OA ↑**	**Acc. ↓**
Diff-Pitcher	67.37	67.40	70.30	69.43
StyleSinger	-	-	-	71.48
StylePitcher	68.64	68.74	73.04	**51.85**
– w/o smo.	**69.49**	**69.61**	**73.61**	52.71
– w/o ctx.	66.71	66.82	71.34	52.12

Table 1. Objective evaluations (%) on GTSinger dataset.

4.3. Baselines and Metrics

We compare against three baselines to evaluate generated pitch curves under task-specific settings, focusing on style capture ability: (1) Diff-Pitcher [4] for APC; (2) StyleSinger [9] for zero-shot SVS with style transfer; (3) an in-house SVC model using unchanged F0. We evaluate only their pitch prediction modules where applicable. Two ablations are included to assess the proposed smoothing algorithm (w/o smo.) and the inpainting setting (w/o ctx., with mask $m = 0$).

For objective metrics, we measure pitch alignment [2] using Raw Pitch Accuracy (RPA, within half-semitone), Raw Chroma Accuracy (RCA, octave-invariant RPA), and Overall Accuracy (OA, including non-melody frames) [34]. To assess overall similarity, we train a 2-layer LSTM model on the curves and report classification accuracy (Acc.) between generated and ground-truth pitch, where **lower values indicate higher similarity**. These metrics are evaluated on the Chinese GTSinger set [25], unseen by all compared models.

For subjective evaluation, we conduct an online listening test on all three tasks. Participants first listened to the off-key voice (APC) or reference singing tracks (SVS/SVC), then rated generated samples from different models on 5-point Likert scales in three aspects: (1) pitch correction accuracy (APC only); (2) style preservation (APC/SVS) or capture (SVC); (3) overall quality. For SVS, we evaluate only cases where reference content remains unchanged. We collected 19 responses from participants with diverse musical backgrounds, yielding 76 ratings per task per model per aspect.

[2]StyleSinger is excluded because annotated scores are not perfectly aligned with the audio.

Models	**APC**			**SVS**		**SVC**	
	MOS-P	MOS-S	MOS-Q	MOS-S	MOS-Q	MOS-S	MOS-Q
*Baselines	Diff-Pitcher [4]			StyleSinger [9]		In-house SVC	
	4.18±0.21	3.38±0.20	3.09±0.18	3.21±0.22	3.07±0.19	2.62±0.23	**3.03**±0.22
StylePitcher	3.84±0.22	**3.64**±0.20	**3.26**±0.18	3.33±0.23	3.11±0.23	**2.95**±0.25	2.72±0.22
– w/o smo.	3.66±0.24	3.39±0.19	3.04±0.19	**3.45**±0.21	**3.18**±0.21	2.72±0.22	2.64±0.23

Table 2. Subjective evaluations on three singing tasks. *Baselines correspond to Diff-Pitcher, StyleSinger, and the in-house SVC for their respective tasks. MOS-P, MOS-S, and MOS-Q refer to mean opinion scores for Pitch, Style, and Quality aspects.

5. RESULTS AND DISCUSSIONS

5.1. Objective Evaluation

Table 1 shows that StylePitcher outperforms all baselines across pitch alignment and similarity metrics. Notably, the LSTM classifier achieves near-random accuracy (50%) when distinguishing our generated curves from real ones, demonstrating the effectiveness of rectified flow for modeling continuous signal. The ablation without smoothing achieves slightly better alignment metrics by adhering more strictly to musical scores, while removing context degrades performance, confirming the benefit of in-context learning.

5.2. Subjective Evaluation

Table 2 presents human evaluation results across three tasks.

Automatic Pitch Correction StylePitcher better preserves singing style and audio quality than Diff-Pitcher [4], though with lower pitch correction accuracy. As shown in Fig. 2(a), our method maintains expressive elements like pitch slides while correcting notes, producing more personalized corrections rather than enforcing strict alignment.

Zero-Shot SVS with Style Transfer Despite never training jointly with synthesis frameworks, StylePitcher achieves superior style capture compared to StyleSinger [9] and maintaining comparable audio quality. Fig. 2(b) also demonstrates that our method effectively captures vibrato and glissando characteristics that StyleSinger misses, validating its potential as a plug-and-play module for enhanced expressiveness.

Style-informed SVC Unlike the baseline using unchanged F0, StylePitcher successfully transfers both timbre and singing style. Fig. 2(c) shows the transformation of a flat target curve into one with strong vibrato from the reference. However, applying expressive techniques without content awareness can occasionally produce unnatural results, impacting audio quality. We leave resolving this limitation to future work.

These results validate StylePitcher as an effective general-purpose pitch generator that balances pitch accuracy with style capture across diverse tasks, enabling expressive singing applications without task-specific training.

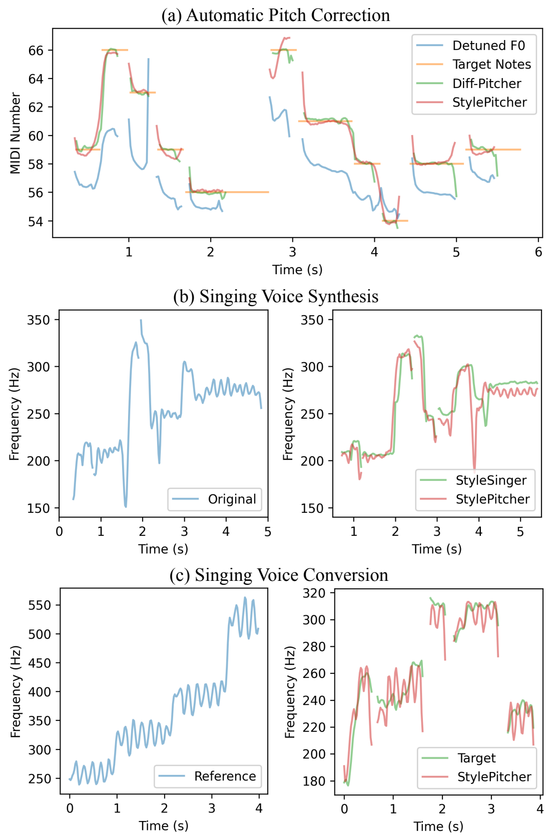

Fig. 2. Samples for three singing tasks. StylePitcher (red) captures singing styles better from input curves (blue) than baselines (green), such as pitch slides (a) and vibrato (b&c).

6. CONCLUSION

We presented StylePitcher, a general-purpose pitch generation framework that captures and transfers singing styles through masked infilling with rectified flow. Without task-specific training or manual annotations, our DiT-based model achieves superior performance across automatic pitch correction, singing voice synthesis, and voice conversion. Its plug-and-play design enables immediate deployment in existing systems. Future work will explore content-aware generation and extend to other performance parameters for comprehensive style modeling.

7. REFERENCES

[1] Shuqi Dai, Yuxuan Wu, Siqi Chen, Roy Huang, and Roger B. Dannenberg, "Singstyle111: A multilingual singing dataset with style transfer," in *Proc. ISMIR*, 2023.

[2] Olivier Perrotin and Christophe D'alessandro, "Target acquisition vs. expressive motion: Dynamic pitch warping for intonation correction," *ACM Trans. Comput.-Hum. Interact.*, 2016.

[3] Xiaobin Zhuang, Huiran Yu, Weifeng Zhao, Tao Jiang, and Peng Hu, "Karatuner: Towards end-to-end natural pitch correction for singing voice in karaoke," in *Proc. Interspeech*, 2022.

[4] Jiarui Hai and Mounya Elhilali, "Diff-pitcher: Diffusion-based singing voice pitch correction," in *Proc. WASPAA*, 2023.

[5] Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen, and Zhou Zhao, "Diffsinger: Singing voice synthesis via shallow diffusion mechanism," in *Proc. AAAI*, 2022.

[6] Yongmao Zhang, Jian Cong, Heyang Xue, Lei Xie, Pengcheng Zhu, and Mengxiao Bi, "Visinger: Variational inference with adversarial learning for end-to-end singing voice synthesis," in *Proc. ICASSP*, 2022.

[7] Jinzheng He, Jinglin Liu, Zhenhui Ye, Rongjie Huang, Chenye Cui, Huadai Liu, and Zhou Zhao, "Rmssinger: Realistic-music-score based singing voice synthesis," in *Proc. ACL(Findings)*, 2023.

[8] Shuqi Dai, Ming-Yu Liu, Rafael Valle, and Siddharth Gururani, "Expressivesinger: Multilingual and multi-style score-based singing voice synthesis with expressive performance control," in *Proc. ACM Multimed.*, 2024.

[9] 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 *Proc. AAAI*, 2024.

[10] 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 *Proc. EMNLP*, 2024.

[11] Xin Chen, Wei Chu, Jinxi Guo, and Ning Xu, "Singing voice conversion with non-parallel data," in *Proc. MIPR*, 2019.

[12] Chengqi Deng, Chengzhu Yu, Heng Lu, Chao Weng, and Dong Yu, "Pitchnet: Unsupervised singing voice conversion with pitch adversarial network," in *Proc. ICASSP*, 2020.

[13] Haohan Guo, Zhiping Zhou, Fanbo Meng, and Kai Liu, "Improving adversarial waveform generation based singing voice conversion with harmonic signals," in *Proc. ICASSP*, 2022.

[14] Xu Li, Shansong Liu, and Ying Shan, "A hierarchical speaker representation framework for one-shot singing voice conversion," in *Proc. Interspeech*, 2022.

[15] Adam Polyak, Lior Wolf, Yossi Adi, and Yaniv Taigman, "Unsupervised cross-domain singing voice conversion," in *Proc. Interspeech*, 2020.

[16] Bingsong Bai, Fengping Wang, Yingming Gao, and Ya Li, "SPA-SVC: self-supervised pitch augmentation for singing voice conversion," in *Proc. Interspeech*, 2024.

[17] Yiquan Zhou, Wenyu Wang, Hongwu Ding, Jiacheng Xu, Jihua Zhu, Xin Gao, and Shihao Li, "SYKI-SVC: advancing singing voice conversion with post-processing innovations and an open-source professional testset," in *Proc. ICASSP*, 2025.

[18] Xingchao Liu, Chengyue Gong, and Qiang Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," in *Proc. ICLR*, 2023.

[19] 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 *Proc. NeurIPS*, 2023.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017.

[21] William Peebles and Saining Xie, "Scalable diffusion models with transformers," in *Proc. ICCV*, 2023.

[22] Jonathan Ho and Tim Salimans, "Classifier-free diffusion guidance," *CoRR*, vol. abs/2207.12598, 2022.

[23] Inc Smule, "DAMP-VSEP: Smule digital archive of mobile performances - vocal separation," 2019, `https://zenodo.org/records/3553059` [Accessed: (October 28, 2019)].

[24] Inc Smule, "DAMP-VPB: Digital archive of mobile performances - smule vocal performances balanced," 2017, `https://zenodo.org/records/2616690` [Accessed: (November 3, 2017)].

[25] 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 *Proc. NeurIPS*, 2024.

[26] Julia Wilkins, Prem Seetharaman, Alison Wahl, and Bryan Pardo, "Vocalset: A singing voice dataset," in *Proc. ISMIR*, 2018.

[27] Haojie Wei, Xueke Cao, Tangpeng Dan, and Yueguo Chen, "RMVPE: A robust model for vocal pitch estimation in polyphonic music," in *Proc. Interspeech*, 2023.

[28] Rachel M. Bittner, Juan José Bosch, David Rubinstein, Gabriel Meseguer-Brocal, and Sebastian Ewert, "A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation," in *Proc. ICASSP*, 2022.

[29] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, 2024.

[30] Alexander Quinn Nichol and Prafulla Dhariwal, "Improved denoising diffusion probabilistic models," in *Proc. ICML*, 2021.

[31] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *Proc. ICLR*, 2019.

[32] Tri Dao, "Flashattention-2: Faster attention with better parallelism and work partitioning," in *Proc. ICLR*, 2024.

[33] Ricky T. Q. Chen, "torchdiffeq," 2018.

[34] Brian McFee, Matt McVicar, Daniel Faronbi, Iran Roman, Matan Gover, Stefan Balke, Scott Seyfarth, Ayoub Malek, Colin Raffel, Vincent Lostanlen, Benjamin van Niekirk, Dana Lee, Frank Cwitkowitz, Frank Zalkow, Oriol Nieto, Dan Ellis, Jack Mason, Kyungyun Lee, Bea Steers, David Südholt, et al., "librosa/librosa: 0.11.0," 2025.