Interspeech 2024
1-5 September 2024, Kos, Greece

SingOMD: Singing Oriented Multi-resolution Discrete Representation Construction from Speech Models

Yuxun Tang[1], *Yuning Wu*[1], *Jiatong Shi*[2], *Qin Jin*[*1]

[1] Renmin University of China, [2] Carnegie Mellon University
{tangyuxun, yuningwu, qjin}@ruc.edu.cn, jiatongs@cs.cmu.edu

Abstract

Discrete representation has shown advantages in speech generation tasks, wherein discrete tokens are derived by discretizing hidden features from self-supervised learning (SSL) pre-trained models. However, the direct application of speech SSL models to singing generation encounters domain gaps between speech and singing. Furthermore, singing generation necessitates a more refined representation than typical speech. To address these challenges, we introduce SingOMD, a novel method to extract singing-oriented multi-resolution discrete representations from speech SSL models. Specifically, we first adapt the features from speech SSL through a resynthesis task and incorporate multi-resolution modules based on resampling to better serve singing generation. These adapted multi-resolution features are then discretized via clustering. Extensive experiments demonstrate the robustness, efficiency, and effectiveness of these representations in singing vocoders and singing voice synthesis.

Index Terms: singing voice synthesis, discrete representation, multi-resolution

1. Introduction

Singing Voice Synthesis (SVS) has attracted considerable attention for its capability to produce high-fidelity vocal renditions from musical scores (e.g., lyrics, pitch and tempo). A typical SVS framework is the cascaded system [1–4], where mel-spectrograms are initially generated in the acoustic model and subsequently employed by a vocoder to produce the singing waveform. Recently, in light of the advantages associated with discrete representations, such as reduced storage requirements, enhanced training efficiency, and potential compatibility with other modalities such as text, and coupled with related explorations in various speech-related tasks [5–15], the concept of discrete SVS has begun to gain traction. To further facilitate the exploration of discrete SVS, the Interspeech 2024 Challenge on speech processing using discrete units[1] has recently proposed the SVS track, aiming at utilizing discrete representations to construct the SVS system. Contestants are tasked with developing a cascaded discrete SVS system akin to those utilizing mel-spectrograms, including an acoustic model to convert the music score into discrete representation and a vocoder to transform the representation into waveform.

While the investigation into discrete representations within SVS is still in its nascent stages, notable advancements have been witnessed in speech generation domains, such as text-to-speech (TTS) [6, 8, 12, 14, 16], speech-to-speech translation (S2ST) [5, 10, 11, 17], speech enhancement (SE) [7, 15]. In these tasks, a prevalent method for obtaining discrete units involves conducting clustering over the intermediate features of speech self-supervised learning (SSL) pre-trained models [18]. Specifically, these pre-trained SSL models [19–23] typically operate as either speech encoders or frozen feature extractors, capturing latent features from the intermediate layers. Subsequently, clustering techniques such as K-means or Gumbel-Softmax are applied on these features to derive discrete representations, which are subsequently utilized in downstream tasks.

However, there are currently no SSL models specifically designed for singing-related tasks to extract representations suitable for singing, primarily due to constraints imposed by the scale of singing data [24, 25]. Leveraging speech SSL models could be a simple solution. However, considering the disparity between singing and speech domains, singing encompasses nuanced pitch variations, a broader spectrum of vocal frequencies, and longer durations, so singing synthesis requires richer and more expressive discrete representations. Therefore, direct application of discrete representations extracted from speech SSL models to singing-related tasks faces challenges of domain gap. Moreover, inspired by the findings from Shi et al. [26, 27] that a fixed resolution for speech signals is suboptimal, given that singing is more refined than speech, a single resolution is definitely not optimal for singing-related tasks.

To this end, we propose **SingOMD**, a novel method designed to extract **sing**ing **o**riented **m**ulti-resolution **d**iscrete representations by leveraging speech SSL models. SingOMD trains continuous features extracted from raw singing audios using speech SSL models in a resynthesis task, thereby bridging the domain gap between the speech and singing. Moreover, to capture richer singing-specific features, we introduce a Unet-based resampling module [26, 28] designed to incorporate multi-resolution features. Following the training of the resynthesis task, singing oriented multi-resolution continuous features are extracted from the intermediate layers of the resampling module, and subsequently clustered using K-means to obtain corresponding discrete representations. Extensive experiments demonstrate that these singing oriented multi-resolution discrete representations exhibit high robustness in singing resynthesis. Furthermore, when integrated with a discrete singing voice acoustic model, our approach yields notable enhancements in both efficiency and effectiveness of singing voice synthesis.

The main contributions of this work include: (1) we propose a new method SingOMD to construct singing oriented multi-resolution discrete representations form speech SSL models using the resynthesis task without modifying model parameters or structures of speech SSL models; (2) our method

*Corresponding Author.

[1] https://www.wavlab.org/activities/2024/Interspeech2024-Discrete-Speech-Unit-Challenge/

10.21437/Interspeech.2024-2291

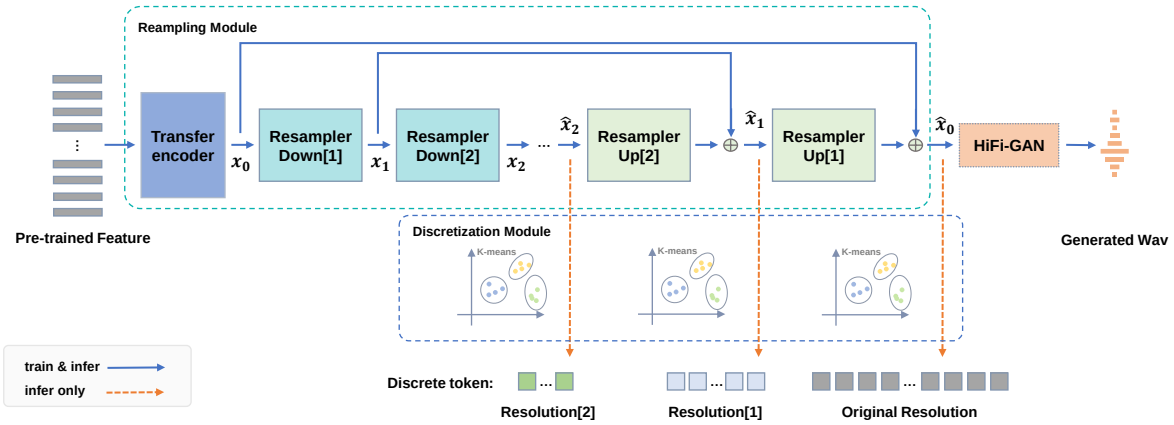

Figure 1: *Illustration of the overall workflow of our proposed SingOMD.*

achieves comparable results for singing voice synthesis with mel-spectrograms systems, and without using any auxiliary information, it outperforms the baseline which involves discrete representations from a larger model and requires pitch predictor. Our demo page can be accessed at `https://interspeech2024singomd.github.io`.

2. Method

Discrete representations have shown great potential in some speech related tasks but have not been well explored in SVS. As discussed in Section 1, there is currently no singing-related SSL model, and leveraging speech SSL models faces challenges due to the domain gap between speech and singing. Driven by issues above and inspiration from multi-resolution expolration in speech [26, 27], we propose a new method, SingOMD, to construct singing oriented multi-resolution discrete representations from speech SSL models.

Figure 1 illustrates the overall workflow of our proposed SingOMD, which involves two major components: a resampling module to form multi-resolution, and a discretization module to produce multi-resolution discrete representations through clustering. Initially, raw singing audio $\boldsymbol{y}$ is fed into a frozen speech SSL model (SSL) to obtain continuous speech features $\boldsymbol{s}$, which is then fed into the resampling module (Resampling) to produce features $\hat{\boldsymbol{x}}$ in different resolution. Finally, the output features $\hat{\boldsymbol{x}}$ from the resampling module are reconstructed to the waveform $\hat{\boldsymbol{y}}$ using the Vocoder.

$$\boldsymbol{s} = \text{SSL}\ (\boldsymbol{y}) \tag{1}$$

$$\hat{\boldsymbol{x}} = \text{Resampling}\ (\boldsymbol{s}) \tag{2}$$

$$\hat{\boldsymbol{y}} = \text{Vocoder}\ (\hat{\boldsymbol{x}}) \tag{3}$$

Such singing audio resynthesis process can serve as an adapter to better fit the speech continuous features for singing. We then construct discrete representations by clustering on the continuous features $\hat{\boldsymbol{x}}$. We elaborate the working details of SingOMD below.

Speech Self-supervised Learning Model. As the singing audio is fed into the Speech SSL model to obtain the continuous features, it is important to utilize as much of the available information as possible to augment the obtained features. In this context, following previous works [29, 30], we perform a weighted sum of the features extracted from all intermediate layers of the Speech SSL model. Let $s_1, s_2, s_3, \ldots, s_n$ represent these features from different hidden layers and n is the number of layers. The obtained speech features $\boldsymbol{s}$ can be expressed as: $\boldsymbol{s} = \sum_{i=1}^{n} w_i \cdot s_i$, where w_i denotes the trainable weight associated with the feature s_i.

Resampling Module. Inspired by findings from previous works [26, 27] that multi-resolution features are beneficial for speech processing tasks, we generate features at different resolutions based on the pre-trained speech features $\boldsymbol{s}$ in the resampling module, which involves a transfer encoder and downsampling-upsampling processing stages. The transfer encoder consists of a `Conv1d` layer with an equal number of input and output feature dimensions. Each resampler block in the downsampling-upsampling process is consistent with the sampling module described in [26], comprising a `Conv1d` for upsampling and a `ConvTranspose1d` for downsampling, cascaded in a residual manner. Specifically, the speech features s first undergo a transfer encoder to adapt from the speech domain to the singing domain, producing singing features $x^{(0)}$ in the original resolution. These original singing features are then fed into the Unet-based upsampling-downsampling module. Firstly, they traverse through t downsampling steps DOWN[i] sequentially to yield a sequence of downsampled features $x^{(0)}, x^{(1)}, \ldots, x^{(t)}$. Subsequently, these features go through t upsampling steps UP[i] that introduce residuals to obtain upsampled features $\hat{x}^{(t-1)} = w_{\text{res}} \cdot (\hat{x}^{(t)} + x^{(t-1)})$, where w_{res} is a hyperparameter. The downsampling and upsampling process is symmetrically repeated to obtain upsampled features $\hat{x}^{(t)}, \hat{x}^{(t-1)}, \ldots, \hat{x}^{(0)}$. The final features fed into the vocoder are $\hat{\boldsymbol{x}} = \hat{x}^{(0)}$.

Note that the scaling ratios of the upsampling and downsampling steps are symmetric. For instance, if DOWN[1] downsamples by a factor of 2, then UP[1] upsamples by a factor of 2, and this pattern continues. Additionally, the sampling ratios are determined by the adjacent sampling rates in the desired sampling rate sequence. For example, if the original audio resolution is 20ms and we aim to obtain features at [20ms, 40ms, 80ms], where 20ms represents the duration of each token (i.e., 50 features correspond to 1s of audio), then the downsampling ratios would be $[\frac{40}{20} = 2, \frac{80}{40} = 2]$. Furthermore, the upsampling ratios are symmetric to the downsampling ratios $[\frac{80}{40} = 2, \frac{40}{20} = 2]$.

Vocoder. Ultimately, the output features $\hat{\boldsymbol{x}}$ are fed into a

vocoder backbone, specifically HiFi-GAN [31] here, to reconstruct the features into the singing waveform $\hat{\boldsymbol{y}}$.

Loss Function. For the loss function of the entire SingOMD, we adopt the same settings as those used in [31], including GAN loss $L_{\text{Adv}}(G;D)$, feature matching loss $L_{\text{FM}}(G;D)$ and mel-spectrogtams loss L_{Mel}, where G and D represent the generator the discriminator respectively.

Discrete Representations. To obtain singing-adapted multi-resolution discrete representations, we apply K-means clustering individually on the continuous features $\hat{x}^{(t)}, \hat{x}^{(t-1)}, \ldots, \hat{x}^{(0)}$ from the resampling module corresponding to the desired resolutions for each $\hat{x}$ we want to obtain. This process results in obtaining corresponding discrete features for each desired resolution of $\hat{x}$.

3. Experiments

To assess the effectiveness of discrete representations constructed via SingOMD, we conduct experiments on two tasks: singing sesynthesis and singing voice synthesis, and evaluate the synthesized audio accordingly.

3.1. SingOMD Training

We first train SingOMD to construct discrete representations. The training datasets comprise ACE-Opencpop (130 hours) [32], OpenSinger (50 hours) [33], M4Singer (29.8 hours) [34], and Opencpop (5.2 hours) [35], collectively amounting to approximately 210 hours. Notably, ACE-Opencpop, OpenSinger, and M4Singer are multi-singer datasets. We follow the predefined data splits for ACE-Opencpop and Opencpop. For M4Singer and OpenSinger, we allocate the first 200 entries as the validation set, 201 and 250 entries as the test set respectively, and the remainder as the training set.

We choose HuBERT [19], one of the most prominent speech SSL models, as the pre-trained SSL model in our SingOMD. To better highlight the superiority of our approach, we opted for HuBERT base, featuring a 12-layer transformer with a hidden feature dimension of 768. The resampling module consists of a transfer encoder and multiple Resampler blocks. The transfer encoder is comprised of a `Conv1d` with both input and output channels set to 512, a kernel size of 7, and a stride of 1. The parameters for each Resampler are identical to those of the sampling module described in [26]. The kernel size and stride of `Conv1d` and `ConvTranspose1d` in Resampler are both 1. The residual coefficient w_{res} in the resampling module is set to $\sqrt{0.4}$. Three resolutions are set at most, similar to the setting in [26]. In the vocoder, we follow the settings of HiFi-GAN [31] and substitute spectrograms with features generated by resampling module. To obtain discrete representations, the cluster number in K-means is set to 1024. The training is performed using an NVIDIA 3090 GPU with a batch size of 16 for 250000 steps. We use the Adam optimizer with a learning rate of 2×10^{-4}. All experiments are conducted within the Parallel WaveGAN [36].[2]

3.2. Evaluation Dataset and Metrics

Dataset. The Opencpop dataset serves as the benchmark for evaluating the quality of SingOMD tokens and other discrete representations. In our quality assessment experiments concerning discrete tokens, we follow the default segmentation of the Opencpop dataset.

Metrics. The quality of discrete tokens is assessed based on the quality of audio segments generated in both singing resynthesis and singing voice synthesis tasks using discrete tokens. We employ both subjective and objective metrics to evaluate the quality of these audio segments. The objective metrics include Mel cepstral distortion (MCD), logarithmic F0 root mean square error (F0 RMSE), semitone accuracy (S. Acc.), and Voice/Unvoice Error (V/UV E.), consistent with previous works [37–39]. For subjective metrics, we utilize the Mean Opinion Score (MOS) approach, where 30 samples from each system are evaluated by 20 professional annotators on a 5-point scale, with 1 indicating an unreasonable synthesis and 5 signifying a synthesis indistinguishable from a real human voice.

3.3. Evaluation setup on singing resynthesis task

The singing resynthesis task directly converts the provided discrete tokens back into audio through a vocoder.

Baselines. The compared baseline models, including both single stream tokens and multi-stream tokens configurations and vocoders as unit HiFiGAN [5, 14, 40], are provided by the Interspeech2024 Challenge as follows:

- HuBERT-base/3: Single stream discrete tokens from the 3$^{\text{rd}}$ layer of HuBERT base.
- HuBERT-base/sum: Single stream discrete tokens from weighted sum features of all layers in HuBERT base.
- HuBERT-base/3+10+11: Multi-stream discrete tokens from 3$^{\text{rd}}$, 10$^{\text{th}}$, 11$^{\text{th}}$ layers of HuBERT base, top 3 weighted in weighted sum.

Experiment Setup. In the singing resynthesis experiments, the vocoder employed is unit HiFi-GAN, which includes an additional embedding layer for input discrete tokens compared to HiFi-GAN. The rest of its architecture and parameters remain consistent with the official setting. The embedding layer is configured with 512 channels. In scenarios involving multiple stream tokens input, we utilize embedding layers to embed each stream separately and then integrate them through a weighted sum. The training settings align with Section 3.1.

3.4. Experiment results on singing resynthesis

We conduct a comparison experiment on different discrete tokens and ablate on resolutions of SingOMD tokens.

Comparison of different discrete tokens. Table 1 shows the performance of different discrete tokens. "Resolution", refers to the resolution of the corresponding discrete representations. In experiments, our SingOMD tokens take only one stream for each resolution. Comparing results from rows 3, and 4, it's evident that employing only a transfer encoder without introducing multi-resolution significantly enhances the synthesized outcomes. It also confirms the effectiveness of the weighted sum approach in rows 1 and 3. Furthermore, it's observed that both utilize 3-stream discrete tokens in rows 2 and 6. However, the quantity of tokens extracted from the SSL model in row 2 is nearly double that of row 6 due to different resolutions. Despite this difference, our approach still significantly outperforms in all metrics. Additionally, a comparison between rows 1 and 2 illustrates that merely increasing the number of token streams does not lead to an improvement. In contrast, the inclusion of information from more resolutions results in substantial enhancements in rows 4-6. These results demonstrate the superiority of SingOMD in extracting singing-oriented discrete tokens.

[2] https://github.com/kan-bayashi/ParallelWaveGAN

Table 1: *Comparison of discrete singing resynthesis on Opencpop. 95% confidence intervals are reported in parentheses.*

	Method	SSL	Resolution	MCD ↓	F0 RMSE ↓	S. ACC.↑	VUV Error ↓	MOS ↑
1	Baseline	HuBERT-base/3	(20)	8.7103	0.2192	25.40%	9.93%	2.46 (± 0.06)
2	Baseline	HuBERT-base/3+10+11	(20)	8.8802	0.2922	27.42%	8.74%	2.34 (± 0.05)
3	Baseline	HuBERT-base/sum	(20)	7.6427	0.1847	38.90%	**7.66%**	2.78 (± 0.06)
4	SingOMD (ours)	HuBERT-base/sum	(20,)	6.9693	0.2167	60.32%	8.24%	3.39 (± 0.06)
5	SingOMD (ours)	HuBERT-base/sum	(20, 40)	6.6414	**0.1806**	64.02%	8.41%	3.48 (± 0.06)
6	SingOMD (ours)	HuBERT-base/sum	(20, 40, 80)	**6.5766**	0.1828	**64.83%**	8.16%	**3.55 (± 0.07)**
7	Ground Truth	-	-	-	-	-	-	4.66 ± 0.06

Ablation on resolution. To investigate the impact of resolution on SingOMD, experiments were conducted using discrete tokens of varying resolutions. The results from rows 4 and 6 indicate that when transitioning from a single resolution to multiple resolutions, all metrics show significant improvement. With the increase of resolutions in discrete tokens, as shown from rows 4-6, MCD, F0 RMSE, S. ACC., and MOS follow the improvement of resolutions or remain comparable results while VUV Error shows no apparent correlation. It suggests that incorporating information from more resolutions can enhance the informational content of discrete features. However, comparing rows 4-5 and rows 5-6, it seems that further increasing resolutions does not yield as pronounced effects as the initial shift. This finding suggests a trade-off between the quality of synthetic vocals and the quantity of tokens which represents the computational efficiency.

3.5. Evaluation setup on singing voice synthesis

We also evaluate the effectiveness of our singing-oriented discrete tokens on the SVS task. Specifically, We first train a vocoder using discrete tokens as input. Then, we train a discrete token-based acoustic model to directly predict discrete tokens from musical scores. Finally, the acoustic model and vocoder are cascaded as an SVS system.

Baselines. For all systems utilizing discrete tokens, the acoustic model employed for predicting discrete features is an RNN-based model [25] provided by the Interspeech2024 Challenge. To further explore the effectiveness of our method, we do not use any additional information in discrete tokens systems. For systems using Mel-spectrograms, we utilize XiaoiceSing [2], a classic transformer based model, as the acoustic model. All acoustic models above are available in ESPnet-Muskits [37].[3] All vocoders are either HiFi-GAN or unit HiFi-GAN, consistent with previous experiments.

The SVS baselines are as follows:

- Mel-septrograms: XiaoiceSing as acoustic model, HiFi-GAN as vocoder and mel-spetrograms as intermediate features.
- HuBERT-base/3: RNN with duration predictor, single stream from the 3^{rd} layer of HuBERT base.
- HuBERT-base/3+10+11: RNN with duration predictor, multi stream discrete tokens from the 3^{rd}, 10^{th} , and 11^{th} layers of HuBERT base.

Experiment Setup. In SVS experiments, all acoustic models including RNN and XiaoiceSing and the training parameters, adhere to the suggested settings specified in ESPnet Opencpop recipe.[4] For SingOMD systems, SingOMD tokens are chosen in resolution [20, 40, 80]. All acoustic models are trained using the Adam optimizer with a learning rate of 1×10^{-3} on a NVIDIA 3090 GPU with a batch size of 16 for 350 epochs. We choose the best model from the validation set. The configurations for vocoders are consistent with those detailed in 3.1, following the official HiFi-GAN settings. All experiments are conducted within the ESPnet framework.

[3] https://github.com/espnet/espnet

[4] https://github.com/espnet/espnet/tree/master/egs2/opencpop/svs1

Table 2: *Comparison of SVS performance on Opencpop. 95% confidence intervals are reported in parentheses.*

Model	MCD ↓	F0 RMSE ↓	MOS ↑
Mel spectrogram	**6.9283**	0.2610	3.04 ± 0.06
HuBERT-base/3	9.5528	0.2321	2.34 ± 0.06
HuBERT-base/3+10+11	9.7585	0.3200	2.34 ± 0.05
DiscreteSVS+SingOMD	7.7234	0.1941	**3.10 ± 0.06**
Ground Truth	-	-	4.66 ± 0.06

3.6. Experiment results on singing voice synthesis

The results in Table 2 demonstrate that SingOMD achieves the best grades among all systems using discrete tokens in all metrics. Furthermore, our approach achieves comparable MOS to the baseline system with Mel spectrograms and improves notably in F0 RMSE which validate our hypothesis that our model requires long-duration information in pitch. And the deterioration of MCD in SingOMD is reasonable, due to information loss during discretization. Although there is an undeniable gap between discrete systems and Ground-Truth, the results still underscore the efficacy and potential of SingOMD in enhancing the quality and effectiveness of singing voice synthesis.

4. Conclusion

This paper proposes SingOMD, a novel method to construct singing-oriented discrete representations for singing generation by leveraging speech SSL models. SingOMD first alleviates domain gaps between speech and singing by adapting the continuous features from hidden layers of speech SSL for singing through a singing audio resynthesis process. Moreover, a resampling module is incorporated to capture multi-resolution richer features. These adapted multi-resolution features are then discretized via K-means to form our singing-oriented discrete representations. Extensive experiments demonstrate the robustness of these representations in singing vocoders. They also enhance the efficiency and effectiveness of singing voice synthesis when integrated with a discrete singing acoustic model.

5. Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (No. 62072462) and the Beijing Natural Science Foundation (No. L233008).

6. References

[1] M. Blaauw and J. Bonada, "Sequence-to-sequence singing synthesis using the feed-forward transformer," in *Proc. ICASSP*, 2020.

[2] P. Lu, J. Wu, J. Luan, *et al.*, "XiaoiceSing: A high-quality and integrated singing voice synthesis system," *Proc. Interspeech*, 2020.

[3] J. Chen, X. Tan, J. Luan, *et al.*, "Hifisinger: Towards high-fidelity neural singing voice synthesis," *arXiv preprint arXiv:2009.01776*, 2020.

[4] J. Liu, C. Li, Y. Ren, *et al.*, "DiffSinger: Singing voice synthesis via shallow diffusion mechanism," in *Proc. AAAI*, 2022.

[5] A. Lee *et al.*, "Direct speech-to-speech translation with discrete units," in *Proc. ACL*, 2022.

[6] Y. Yang, F. Shen, C. Du, *et al.*, "Towards universal speech discrete tokens: A case study for ASR and TTS," in *Proc. ICASSP*, 2024.

[7] Z. Wang, X. Zhu, Z. Zhang, *et al.*, "SELM: Speech enhancement using discrete tokens and language models," *arXiv preprint arXiv:2312.09747*, 2023.

[8] R. Huang, C. Zhang, Y. Wang, *et al.*, "Make-a-voice: Unified voice synthesis with discrete representation," *arXiv preprint arXiv:2305.19269*, 2023.

[9] X. Chang, B. Yan, K. Choi, *et al.*, "Exploring speech recognition, translation, and understanding with discrete speech units: A comparative study," in *Proc. ICASSP*, 2024.

[10] L. Barrault, Y.-A. Chung, M. C. Meglioli, *et al.*, "Seamless: Multilingual expressive and streaming speech translation," *arXiv preprint arXiv:2312.05187*, 2023.

[11] J. Shi, Y. Tang, A. Lee, H. Inaguma, C. Wang, J. Pino, and S. Watanabe, "Enhancing speech-to-speech translation with multiple tts targets," in *Proc. ICASSP*, 2023.

[12] C. Wang, S. Chen, Y. Wu, *et al.*, "Neural codec language models are zero-shot text to speech synthesizers," *arXiv preprint arXiv:2301.02111*, 2023.

[13] J. Shi, C.-J. Hsu, H. Chung, D. Gao, P. Garcia, S. Watanabe, A. Lee, and H.-y. Lee, "Bridging speech and textual pre-trained models with unsupervised ASR," in *Proc. ICASSP*, 2023.

[14] T. Hayashi and S. Watanabe, "Discretalk: Text-to-speech as a machine translation problem," *arXiv preprint arXiv:2005.05525*, 2020.

[15] J. Shi, X. Chang, T. Hayashi, Y.-J. Lu, S. Watanabe, and B. Xu, "Discretization and re-synthesis: An alternative method to solve the cocktail party problem," *arXiv preprint arXiv:2112.09382*, 2021.

[16] Z. Borsos, R. Marinier, D. Vincent, *et al.*, "Audiolm: A language modeling approach to audio generation," *TASLP*, 2023.

[17] Y. Wang, J. Bai, R. Huang, R. Li, Z. Hong, and Z. Zhao, *Speech-to-speech translation with discrete-unit-based style transfer*, 2023.

[18] A. Mohamed *et al.*, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, 2022.

[19] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, *et al.*, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, vol. 29, 2021.

[20] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," *Proc. NeurIPS*, vol. 33, 2020.

[21] Y.-A. Chung, Y. Zhang, W. Han, *et al.*, "W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *Proc. ASRU*, 2021.

[22] S. Chen, C. Wang, Z. Chen, *et al.*, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IJSTSP*, vol. 16, no. 6, 2022.

[23] A. Babu, C. Wang, A. Tjandra, *et al.*, "XLS-R: Self-supervised cross-lingual speech representation learning at scale," *Proc. Interspeech*, 2022.

[24] S. Guo, J. Shi, T. Qian, S. Watanabe, and Q. Jin, "SingAug: Data Augmentation for Singing Voice Synthesis with Cycle-consistent Training Strategy," in *Proc. Interspeech*, 2022.

[25] J. Shi, S. Guo, N. Huo, Y. Zhang, and Q. Jin, "Sequence-to-sequence singing voice synthesis with perceptual entropy loss," in *Proc. ICASSP*, 2021.

[26] J. Shi, H. Inaguma, X. Ma, I. Kulikov, and A. Sun, "Multi-resolution HuBERT: Multi-resolution speech self-supervised learning with masked unit prediction," in *Proc. ICLR*, 2024.

[27] J. Shi, Y. Tang, H. Inaguma, H. Gong, J. Pino, and S. Watanabe, "Exploration on HuBERT with Multiple Resolution," in *Proc. Interspeech*, 2023.

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. MICCAI*, 2015.

[29] S.-W. Yang *et al.*, "SUPERB: Speech Processing Universal PERformance Benchmark," in *Proc. Interspeech*, 2021.

[30] J. Shi *et al.*, "ML-SUPERB: Multilingual Speech Universal PERformance Benchmark," in *Proc. Interspeech*, 2023.

[31] J. Kong, J. Kim, and J. Bae, "HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *Proc. NeurIPS*, 2020.

[32] J. Shi, Y. Lin, X. Bai, K. Zhang, Y. Wu, Y. Tang, Y. Yu, Q. Jin, and S. Watanabe, "Singing voice data scaling-up: An introduction to ace-opencpop and ace-kising," *arXiv preprint arXiv:2401.17619*, 2024.

[33] R. Huang, F. Chen, Y. Ren, J. Liu, C. Cui, and Z. Zhao, "Multi-singer: Fast multi-singer singing voice vocoder with a large-scale corpus," in *Proc. ACMMM*, 2021.

[34] L. Zhang, R. Li, S. Wang, *et al.*, "M4Singer: A multi-style, multi-singer and musical score provided mandarin singing corpus," in *Proc. NeurIPS*, 2022.

[35] Y. Wang, X. Wang, P. Zhu, J. Wu, H. Li, H. Xue, Y. Zhang, L. Xie, and M. Bi, "Opencpop: A High-Quality Open Source Chinese Popular Song Corpus for Singing Voice Synthesis," in *Proc. Interspeech*, 2022.

[36] T. Hayashi, R. Yamamoto, T. Yoshimura, P. Wu, J. Shi, T. Saeki, Y. Ju, Y. Yasuda, S. Takamichi, and S. Watanabe, "ESPnet2-TTS: Extending the edge of tts research," *ArXiv*, vol. abs/2110.07840, 2021.

[37] J. Shi, S. Guo, T. Qian, *et al.*, "Muskits: An end-to-end music processing toolkit for singing voice synthesis," in *Proc. Interspeech*, 2022.

[38] Y. Wu, J. Shi, T. Qian, D. Gao, and Q. Jin, "Phoneix: Acoustic feature processing strategy for enhanced singing pronunciation with phoneme distribution predictor," in *Proc. ICASSP*, 2023.

[39] Y. Wu, Y. Yu, J. Shi, T. Qian, and Q. Jin, "A systematic exploration of joint-training for singing voice synthesis," *arXiv preprint arXiv:2308.02867*, 2023.

[40] B. Yan *et al.*, "ESPnet-ST-v2: Multipurpose spoken language translation toolkit," in *Proc. ACL*.