Interspeech 2025
17-21 August 2025, Rotterdam, The Netherlands

Differentiable K-means for Fully-optimized Discrete Token-based ASR

Kentaro Onda[1,2], *Yosuke Kashiwagi*[2], *Emiru Tsunoo*[2], *Hayato Futami*[2], *Shinji Watanabe*[3]

[1]The University of Tokyo, Japan
[2]Sony Group Corporation, Japan
[3]Carnegie Mellon University, USA
ondakentaro@gavo.t.u-tokyo.ac.jp

Abstract

Recent studies have highlighted the potential of discrete tokens derived from self-supervised learning (SSL) models for various speech-related tasks. These tokens serve not only as substitutes for text in language modeling but also as intermediate representations for tasks such as automatic speech recognition (ASR). However, discrete tokens are typically obtained via k-means clustering of SSL features independently of downstream tasks, making them suboptimal for specific applications. This paper proposes the use of differentiable k-means, enabling the joint optimization of tokenization and downstream tasks. This approach enables the fine-tuning of the SSL parameters and learning weights for outputs from multiple SSL layers. Experiments were conducted with ASR as a downstream task. ASR accuracy successfully improved owing to the optimized tokens. The acquired tokens also exhibited greater purity of phonetic information, which were found to be useful even in speech resynthesis.
Index Terms: discrete tokens, automatic speech recognition, self-supervised learning, acoustic unit discovery

1. Introduction

The rapid advancement of Self-Supervised Learning (SSL) techniques has significantly influenced numerous speech processing tasks [1]. Pioneering SSL-based models such as HuBERT [2], wav2vec 2.0 [3], WavLM [4], and w2v-BERT [5] have demonstrated strong performance across diverse applications including automatic speech recognition (ASR), emotion analysis, and text-to-speech (TTS) synthesis [6]. One notable feature of these models is their ability to learn robust and generalizable representations of audio signals without extensive labeled data. Moreover, joint optimization strategies, in which the SSL model is fine-tuned on downstream objectives, have consistently yielded further performance gains. This underscores the critical importance of end-to-end training in speech-related tasks using SSL models [2, 4].

While continuous SSL embeddings are already widely used, there is a growing tendency to discretize these learned representations [7]. Typically obtained via methods such as k-means clustering, these discrete tokens have proven highly effective for tasks that benefit from a text-like representation. They are also integrated seamlessly with large language models (LLMs), opening new possibilities in speech and language research, notably in generative frameworks such as SpeechGPT [8], VoxtLM [9], and AudioPaLM [10].

A major limitation of relying on k-means is its non-differentiability. The conventional SSL pipeline commonly involves three steps: (1) SSL pre-training, (2) k-means clustering to obtain tokens, and (3) downstream training. Such a procedure restricts the ability to fine-tune the SSL model's parameters to the final task objectives [11–13]. Consequently, the resulting tokens may not capture task-relevant information optimally, leading to a performance gap compared with systems that leverage continuous SSL features and end-to-end optimization [14].

To address this shortcoming, we propose a framework that integrates differentiable k-means into a discrete token-based ASR system, thereby enabling joint optimization of both the SSL model and the clustering process. Our approach is inspired by previous work on differentiable k-means in autoencoders [15], but extends this idea to SSL-based speech representations. By making the tokenization stage differentiable, we can back-propagate error signals from the downstream ASR objective into the SSL feature extractor. While discretization techniques such as those in VQ-VAE [16] could be employed for this purpose, we adopt differentiable k-means as a more straightforward extension of standard k-means, which is widely used for tokenizing SSL representations. Moreover, the framework supports simultaneous optimization of layer-weighting factors when multiple SSL layers are used [13], which further refines the quality of the learned discrete tokens. This end-to-end design also offers greater flexibility in how clustering boundaries are formed, allowing the model to focus on phonetic distinctions that are most relevant for the final recognition task. As a result, the system effectively learns discrete tokens that discard extraneous speaker or noise information. Our experimental results indicate that this method narrows the performance gap relative to systems using continuous SSL features in ASR while also providing an efficient representation that benefits both ASR and generative tasks.

Our key contributions can be summarized as follows:

- **Differentiable Tokenization for ASR:** We illustrate how integrating differentiable k-means into a discrete token-based ASR system enables end-to-end training, allowing each component (from SSL parameters to clustering centroids) to be jointly optimized for the recognition task.
- **Enhanced Representational Flexibility:** We demonstrate that our method allows layer-weighting strategies for exploiting multiple SSL layers, thereby enabling more flexible and effective use of the hierarchical representations learned during SSL pre-training.
- **Analysis of Updated Tokenization:** Through an analytical investigation, we show that the discrete tokens learned under an ASR-centric objective become increasingly similar to phoneme-level representations. This property not only improves ASR accuracy but also holds potential for generative applications.

10.21437/Interspeech.2025-593

2. Discrete Token-based ASR with Differentiable K-means

2.1. Discrete token-based ASR

Discrete token-based ASR has recently attracted attention as an alternative to using continuous features extracted from SSL models [11–13, 17, 18]. In this paradigm, each frame of the SSL output is clustered into discrete tokens, which then serve as input to a conventional sequence-to-sequence (Seq2Seq) ASR model or a similar architecture [19–21]. Let X denote the input speech and Y denote the corresponding text token sequence. We denote the intermediate discrete token sequence by Z, and the ASR objective is typically formulated as minimizing a cross-entropy loss $\mathcal{L}^{\text{asr}}(Y, \text{ASR}(Z; \theta_{\text{asr}}))$, where θ_{asr} represents ASR model parameters.

Although continuous SSL features often yield higher recognition accuracy, discrete tokens offer several advantages. They can represent speech frames in a compact manner by clustering similar frames and possibly applying deduplication, which shortens the input sequence for the subsequent model. In addition, discrete tokens approximate a "text-like" representation that is highly compatible with LLMs. This property facilitates downstream tasks such as ASR and speech-to-text translation, where LLMs can be leveraged or fine-tuned from large-scale text-based pretraining [8–10, 22]. One of the main challenges, however, is that the discrete clustering step in most existing pipelines relies on k-means, a non-differentiable algorithm that prevents error backpropagation from the final ASR objective into the SSL parameters.

2.2. Differentiable k-means

Standard k-means clustering assigns each data point $\boldsymbol{s_i}$, the output from the SSL model, to the nearest centroid $\boldsymbol{\mu_j}$ based on Euclidean distance. Since this involves a discrete selection ($\arg\min$), it is inherently non-differentiable. To make k-means amenable to gradient-based optimization, [15] proposed an approach that uses the Gumbel-Softmax reparameterization trick [23]. Concretely, the probability of assigning $\boldsymbol{s_i}$ to cluster j is defined as:

$$p(j \mid \boldsymbol{s_i}) = \frac{\exp\big(-\sigma^2 \|\boldsymbol{s_i} - \boldsymbol{\mu_j}\|_2^2\big)}{\sum_{c=1}^{k} \exp\big(-\sigma^2 \|\boldsymbol{s_i} - \boldsymbol{\mu_c}\|_2^2\big)}, \tag{1}$$

where σ is a hyperparameter (or learnable parameter), and k is the number of clusters. This distribution $p(j \mid \boldsymbol{s_i})$ is then sampled via Gumbel-Softmax:

$$h_{i,j} = \frac{\exp\big(\tau^{-1}(\log p(j \mid \boldsymbol{s_i}) + G_j)\big)}{\sum_{c=1}^{k} \exp\big(\tau^{-1}(\log p(c \mid \boldsymbol{s_i}) + G_c)\big)}, \tag{2}$$

where G is noise drawn from the Gumbel distribution, and τ is a temperature parameter. As $\tau \to 0$, $\boldsymbol{h}_i = [h_{i,1}, ..., h_{i,k}]$ approaches a one-hot vector. Although Gumbel-Softmax produces discrete outputs, i.e. $\arg\max h_i$, it internally handles assignments in a probabilistic manner. Therefore it enables backpropagation to both the output from SSL model $\boldsymbol{s}_i$ and the centroids $M = \{\boldsymbol{\mu_1}, \ldots, \boldsymbol{\mu_k}\}$. The k-means loss for optimizing the centroids M is given by:

$$\mathcal{L}^{\text{km}}(S, M) = \sum_{i=1}^{N} \Big\| \boldsymbol{s_i} - \tilde{\boldsymbol{h}}_i \, M \Big\|_2^2, \tag{3}$$

where N is the total number of data points, and $\tilde{\boldsymbol{h}}_i$ is the discrete one-hot version of $\boldsymbol{h}_i$. S is a set of $\boldsymbol{s_i}$. Differentiable k-means was originally developed for autoencoders, but can be applied in a more general context by substituting the relevant downstream loss.

2.3. Error Backpropagation via differentiable k-means

We extend this differentiable k-means approach to discrete token-based ASR. Unlike conventional pipelines, which cluster SSL features into discrete tokens in a downstream-agnostic manner, our method enables joint optimization of clustering and recognition. Let θ_{ssl} be the parameters of the SSL model and θ_{asr} be those of the ASR model. The overall loss is:

$$\mathcal{L} = \mathcal{L}^{\text{asr}}\Big(Y,\, \text{ASR}\big(\text{DiffKM}\big(\text{SSL}(X;\, \theta_{\text{ssl}}),\, M\big);\, \theta_{\text{asr}}\big)\Big) + \alpha\, \mathcal{L}^{\text{km}}\big(\text{SSL}(X;\, \theta_{\text{ssl}}),\, M\big), \tag{4}$$

where $\text{DiffKM}(\cdot)$ denotes the differentiable k-means operator in Eqs. (1)–(2), and α is a scalar that balances the cross-entropy loss $\mathcal{L}^{\text{asr}}$ and the k-means loss $\mathcal{L}^{\text{km}}$. By backpropagating through both cluster assignments and SSL feature extractor, the discrete tokens become optimized with respect to the ASR objective. In practice, α can be tuned or set to zero if the ASR objective alone sufficiently drives discriminative clustering.

2.4. Joint optimization with multilayer weighting

Combining outputs from multiple SSL layers often leads to performance gains, particularly when working with continuous representations [6, 17, 18]. In that continuous domain, a widely adopted approach is to take a weighted sum of each layer's output before feeding it into a downstream model:

$$\text{SSL}_{\text{ws}}(X; \theta_{\text{ssl}}, \boldsymbol{w}) = \sum_{\ell=1}^{L} w_\ell \, \text{SSL}_\ell(X; \theta_{\text{ssl}}), \tag{5}$$

where L is the total number of SSL layers, $\text{SSL}_\ell(\cdot)$ is the output of the ℓth layer, and $\boldsymbol{w} = [w_1, \ldots, w_L]$ denotes the set of learnable layer weights.

Although extending this idea to discrete representations has been more challenging, our differentiable k-means-based framework resolves this issue. By substituting Equation (5) into Equation (4), we can obtain weighted-sum based differentiable loss function as:

$$\mathcal{L}^{\text{asr}}\Big(Y,\, \text{ASR}\big(\text{DiffKM}\big(\text{SSL}_{\text{ws}}(X; \theta_{\text{ssl}}, \boldsymbol{w}), M\big);\, \theta_{\text{asr}}\big)\Big) \tag{6}$$

Thus, we can jointly optimize $\boldsymbol{w}$, the cluster centroids M, and even the SSL model parameters θ_{ssl} for the final ASR objective. This design ensures a single token is generated for each frame of the fused representation, removing unnecessary redundancy. In addition, every component of the system can be trained cooperatively, which avoids ad hoc decisions about how to merge tokens from different layers and allows us to exploit the rich complementary information in a more principled manner.

3. Experiments

3.1. Experimental Setup

Our experiments were conducted using ESPnet [24], following the basic configurations described in [11, 12]. We used the joint CTC/attention-based encoder-decoder model [19] with WavLM-large[1] as the upstream SSL. To avoid additional complexity in handling gradients, we did not perform deduplication or subword modeling on the token sequences. Regarding

[1] https://huggingface.co/microsoft/wavlm-large

Table 1: *LibriSpeech-100 results with single-layer settings: WER(%) of test-{clean, other}*

cluster size	**baseline**	**freeze SSL**	**full finetune**
100	7.6/13.2	7.3/12.6	**5.4/9.2**
500	4.7/ 7.9	4.8/ 7.9	**4.3/7.1**
1000	4.3/ 7.5	4.2/ 7.4	**3.9/6.8**
2000	4.4/ 7.2	4.4/ 7.2	**4.2/6.8**

Table 2: *LibriSpeech-100 results with multi-layer settings: WER(%) of test-{clean, other}*

cluster size	**baseline**	**freeze SSL**	**full finetune**
100	8.4/18.3	7.5/15.1	**5.3/8.2**
2000	5.4/10.8	4.7/ 8.8	**4.2/6.8**
continuous	3.7/ 6.7	3.6/ 6.7	3.7/6.6

the hyperparameters, σ^2 in Equation (1) was fixed at 1.0. τ in Equation (2) was initialized to 2.0 and gradually decreased during training. α in Equation (4) was set to 0, as a smaller value led to better ASR performance in our preliminary experiments. During decoding, the beam size was set to 5, which may result in slightly lower performance compared to other studies. However, the same settings were applied in all experiments to ensure a consistent evaluation of the proposed method's effectiveness.

3.2. Single-layer settings with LibriSpeech100

We first conducted experiments on LibriSpeech-100 [25] using the 21st layer of the SSL model by following [12] for feature extraction. We compared ASR performance with various cluster sizes. In all cases, training was conducted for 90 epochs, using the loss shown in Equation (4). During the first 30 epochs, the SSL model (θ_{ssl}) and centroids (M) were frozen, and only the downstream ASR module (θ_{asr}) was trained. We compared the following three conditions for the parameter updates applied during the remaining 60 epochs: **baseline**) ASR model θ_{asr} only, **freeze SSL**) ASR model θ_{asr} and centroids M, and **full finetune**) ASR model θ_{asr}, centroids M, and SSL model θ_{ssl}. In all cases, M was initialized using standard k-means, pre-trained on the train_clean_100. Here, **baseline** functions equivalently to a conventional discrete token-based ASR.

Table 1 shows the results. In all cases, the proposed method (**full finetune**) demonstrated the highest performance. For cluster sizes, we obtained larger improvements, particularly in the small cluster size. Generally, increasing the number of clusters leads to higher recognition accuracy. However, with the proposed method, there was no significant difference in performance between cluster sizes of 500, 1000, and 2000. This indicates that differentiable k-means effectively enabled the representation of necessary linguistic information with a smaller number of units. This characteristic also offers an advantage from the perspective of data compression. For **freeze SSL**, no significant improvement was observed.

3.3. Multi-layer settings with LibriSpeech100

Next, we explored the cases where outputs from multiple layers of the SSL model are used with a weighted summation, as discussed in Section 2.4. In the experiments, the outputs of all layers are used. The training was also conducted for 90 epochs using the loss of Equation (6). As in single-layer settings, for the first 30 epochs, only θ_{asr} was updated, and other parameters, including the layer weights ($\boldsymbol{w}$) were frozen. Similar to the previous experiment, the update with the remaining 60 epochs was applied to **baseline**) θ_{asr}, **freeze SSL**) θ_{asr}, $\boldsymbol{w}$, and M, or **full finetune**) θ_{asr}, $\boldsymbol{w}$, M and θ_{ssl}. In the case of **full finetune**, the update of θ_{ssl} was performed only during the last 30 epochs. This is because $\boldsymbol{w}$ was effectively tuned in this case, resulting in better recognition accuracy compared to when SSL was updated from the beginning. $\boldsymbol{w}$ was initialized such that all of its elements to be the same. M was initialized using centroids obtained by applying standard k-means to a weighted sum computed with the initialized $\boldsymbol{w}$.

The results are shown in Table 2. Experiments were conducted with cluster sizes of 100 and 2000, and as a top line, results with the continuous SSL features are also presented. In the case of continuous features, all SSL layers were used as well, and the same strategy for parameter updating was applied. Similar to the single layer case in Section 3.2, for both of the discrete cases, **full finetune** achieved the best results. However, compared to Table1, **baseline** and **freeeze SSL** performed worse than the single-layer results, while **full finetune** was comparable. These results indicate that while multi-layer approaches effectively leverage frozen SSL features by aggregating information from various layers, full fine-tuning can condense this information into a single layer optimized for downstream ASR tasks. Notably, **full finetune** results approached top-line performance. This is a highly positive outcome, as it preserves both high performance and a compact discrete token representation simultaneously, without heuristic layer selection.

3.4. Evaluation for token properties

To investigate the impact of the ASR-centric fine-tuning on the tokenization process, we evaluated the clusters themselves. Clustering quality is commonly evaluated based on their relationship with phonemes using metrics such as phone normalized mutual information (PNMI) [2, 11, 26]. In addition to this, we explore three more evaluation metrics to examine the effect of fine-tuning more directly.

1) Normalized Quantization Error (NQE): The average Euclidean distance between the continuous SSL features of each frame and the corresponding cluster centroid. Since SSL feature spaces are changed by finetuning, they are divided by the averaged L2 norm for normalization. As we use the ASR objective (Equation (4) or (6)), which is different from the k-means objective (Equation (3)), this value is expected to be higher with our proposed methods.

2) Token Sequence Length (TSL): The length of token sequences after deduplication. If the tokens are robust to subtle pronunciation variations and represent more phoneme-like symbols, the sequence increases the repetition and TSL will be shorter. A lower value also indicates a higher efficiency in data compression.

3) Mean Token Error Rate (MTER): The edit distance between discrete token sequences from different utterances with the same transcriptions. A lower value indicates the robustness of discrete tokens to non-linguistic information. Unlike word error rate (WER) or character error rate (CER), which are calculated with the ground truth transcriptions, there is no definitive correct token sequence in this case. Therefore, we prepare multiple utterances with the same transcriptions and calculate token error rate (TER) for all possible pairs, then use the mean value for the metric.

The TIMIT dataset [27] was used for evaluation. Since each sentence is read by multiple speakers, it can be used for calculating MTER. Additionally, as the phoneme alignment is manually annotated, it also makes PNMI more precise. We used

Table 3: *Cluster evaluations using TIMIT corpus*

cluster size		PNMI (↑)	NQE	TSL (↓)	MTER [%] (↓)
100	baseline	0.4754	0.616	94.5	42.2
	full finetune	0.4561	0.801	**72.4**	28.8
	+ multi layer	**0.5492**	0.675	77.8	**28.5**
2000	baseline	0.7256	0.528	108.0	56.2
	full finetune	0.7274	0.602	**97.5**	48.7
	+ multi layer	**0.7615**	0.626	100.5	**44.7**

Table 4: *Resynthesis performance with acquired units: synthesis models were trained with LJSpeech and tested with in-domain (ID) and out-of-domain (OOD) data (VCTK).*

cluster size		MCD (↓)	F0 RMSE (↓)	WER [%] (↓)		UTMOS (↑)	
		ID	ID	ID	OOD	ID	OOD
100	baseline	6.61	0.296	7.4	31.4	3.78	3.82
	full finetune	6.68	0.297	4.0	12.4	3.78	3.76
	+ multi layer	**6.53**	**0.290**	**2.7**	**6.1**	**3.85**	**3.89**
2000	baseline	**6.35**	0.292	2.7	14.0	3.86	3.43
	full finetune	6.44	**0.282**	**2.6**	14.9	3.87	3.62
	+ multi layer	6.40	0.285	3.2	**7.3**	3.85	**3.77**
(75)	g2p	7.15	0.309	3.8	3.3	3.77	3.76

phonetically compact (SX) set, where each of the 450 sentences is read by seven different speakers. Table 3 shows the results for tokens obtained through **baseline** and **full finetune** with the custer sizes of 100 and 2000 from the previous sections. The results for **full finetune** with multiple layers are also presented.

For PNMI, the larger cluster size led to a better result, which was consistent with the trend in the recognition accuracy shown in Table 1. However, among the single-layer cases with the same cluster size, the values were almost the same regardless of the improvement in recognition accuracy. When using multiple layers, PNMI shows relatively greater improvements compared to recognition accuracies. These findings indicate that, when the cluster size was fixed, there was no strong consistency between PNMI and recognition accuracy. While SSL fine-tuning contributed more to improving recognition accuracy, the use of multiple layers helped obtain more phoneme-like tokens. For NQE, **baseline** achieved the lowest values in all cases. The large increase in NQE for **full finetune** is expected due to the absence of the k-means loss ($\mathcal{L}^{\text{km}}$) aimed at minimizing quantization error. This indicates that minimizing quantization error is not always required for optimizing downstream tasks. For TSL and MTER, significant improvements were observed with **full finetune**. The proposed method clearly enhanced the robustness to speaker variety and subtle pronunciation differences, indicating that the tokens have more phonetic properties.

3.5. Speech resynthesis with acquired tokens

We evaluated the performance of speech resynthesis using the tokens learned through fine-tuning for ASR to investigate their general applicability to other tasks. For the synthesis model, we employed Tacotron2 [28] with the deduplicated token sequences as input. The training was done using the ESPnet LJSpeech recipe [24, 29], and for decoding, a Parallel WaveGAN [30] model pretrained on LJSpeech[2] was used as the vocoder.

The evaluation was conducted from two perspectives: 1) acoustic reconstruction and 2) content reconstruction. For the former, we used mel-cepstral distortion (MCD) and f0 root mean square error (F0 RMSE) for in-domain (ID) LJSpeech data. More acoustic information is expected to improve these values. For the latter, WER by a pre-trained whisper large [31] model was used. This evaluation was performed also on out-of-domain (OOD) VCTK corpus [32] to evaluate the purity of linguistic information in the tokens. If acoustic information is not discarded and OOD speaker information remains in the tokens, a degradation may be caused by the mismatch between the synthesis model trained on LJSpeech data. UTMOS [33] was also evaluated for overall naturalness. As a reference, we also compared with human-defined phonemes, pure linguistic symbols with no acoustic information, which were obtained via g2p[3].

The results are shown in Table4. First, when comparing the performance of g2p and discrete token resynthesis, the latter outperformed in many cases except for intelligibility (WER) and UTMOS OOD metrics. This suggests that discrete tokens often capture richer acoustic information than phonemes for these metrics. Among various cluster sizes and training methods, the combination of a cluster size of 100 and **full finetune** with a multi-layer approach yields the most balanced results across all metrics. This indicates that, despite ASR being our primary training objective, using a smaller cluster with dedicated fine-tuning leads to a compact and efficient representation for generation purposes. These findings provide a useful guideline for designing discrete tokens that effectively support both recognition and generation tasks [9, 10].

Table 5: *LibriSpeech-960 (WER [%]) & ML-SUPERB (CER [%]) results with single-layer settings*

cluster size		LibriSpecch-960 test-{clean, other}	ML-SUPERB normal/few-shot
2000	baseline	3.0/5.9	28.2/41.3
	full finetune	**2.8/5.4**	**19.9/28.8**

3.6. Recognition performance with other corpora

Lastly, we evaluated the recognition performance on LibriSpeech960 and ML-SUPERB [34]. In both cases, by following the finding in Sections 3.2 and 3.3, a single layer with a cluster size of 2000 was used. For ML-SUPERB, a multilingual dataset, we utilized the 35th layer of XLSR-1B[4], following [12]. For LibriSpeech960, the 21st layer of WavLM-large was used as well. The results are shown in Table5. In both cases, the performance was well improved. This shows the validity of our method, even with larger or multi-lingual datasets.

4. Conclusion

In this study, we introduced differentiable k-means to discrete token-based ASR to enable the joint optimization of tokenization and the downstream task, along with SSL finetuning and weighted summation across SSL layers. As a result, ASR performance improved and the obtained tokens exhibited more phonetic characteristics with purer linguistic information. These tokens also showed strong performance even in speech generation tasks. Future work could extend this approach to other tasks such as speech translation and language modeling.

[2] `ljspeech_parallel_wavegan.v1` available at `https://github.com/kan-bayashi/ParallelWaveGAN`

[3] `https://github.com/Kyubyong/g2p`

[4] `https://huggingface.co/facebook/wav2vec2-xls-r-1b`

5. References

[1] A. Mohamed, H.-y. Lee, L. Borgholt *et al.*, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1179–1210, 2022.

[2] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai *et al.*, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[3] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460.

[4] S. Chen, C. Wang, Z. Chen *et al.*, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, pp. 1505–1518, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:239885872

[5] Y.-A. Chung, Y. Zhang, W. Han *et al.*, "w2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 244–250, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:237048255

[6] S. wen Yang, P.-H. Chi, Y.-S. Chuang *et al.*, "SUPERB: Speech processing universal performance benchmark," in *Interspeech 2021*, 2021, pp. 1194–1198.

[7] X. Chang, J. Shi, J. Tian *et al.*, "The Interspeech 2024 challenge on speech processing using discrete units," in *Interspeech 2024*, 2024, pp. 2559–2563.

[8] D. Zhang, S. Li, X. Zhang *et al.*, "Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023, pp. 15 757–15 773.

[9] S. Maiti, Y. Peng, S. Choi *et al.*, "Voxtlm: Unified decoder-only models for consolidating speech recognition, synthesis and speech, text continuation tasks," in *ICASSP 2024*. IEEE, 2024, pp. 13 326–13 330.

[10] P. K. Rubenstein, C. Asawaroengchai, D. D. Nguyen *et al.*, "Audiopalm: A large language model that can speak and listen," *arXiv preprint arXiv:2306.12925*, 2023.

[11] X. Chang, B. Yan, Y. Fujita, T. Maekaku, and S. Watanabe, "Exploration of efficient end-to-end ASR using discretized input from self-supervised learning," in *Interspeech 2023*, 2023, pp. 1399–1403.

[12] X. Chang, B. Yan, K. Choi *et al.*, "Exploring speech recognition, translation, and understanding with discrete speech units: A comparative study," in *ICASSP 2024*, 2024, pp. 11 481–11 485.

[13] Y. Yang, F. Shen, C. Du *et al.*, "Towards universal speech discrete tokens: A case study for ASR and TTS," in *ICASSP 2024*, 2024, pp. 10 401–10 405.

[14] S.-L. Yeh and H. Tang, "Estimating the completeness of discrete speech units," in *2024 IEEE Spoken Language Technology Workshop (SLT)*, 2024, pp. 415–422.

[15] B. Gao, Y. Yang, H. Gouk, and T. M. Hospedales, "Deep clustering with concrete k-means," in *ICASSP 2020*, 2020, pp. 4252–4256.

[16] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[17] P. Mousavi, J. Duret, S. Zaiem *et al.*, "How should we extract discrete audio tokens from self-supervised models?" in *Interspeech 2024*, 2024, pp. 2554–2558.

[18] J. Shi, X. Ma, H. Inaguma, A. Sun, and S. Watanabe, "MMM: Multi-layer multi-residual multi-stream discrete speech representation from self-supervised learning model," in *Interspeech 2024*, 2024, pp. 2569–2573.

[19] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *ICASSP 2017*, 2017, pp. 4835–4839.

[20] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[21] Y. Peng, J. Tian, W. Chen *et al.*, "Owsm v3.1: Better and faster open whisper-style speech models based on e-branchformer," in *Interspeech 2024*, 2024, pp. 352–356.

[22] Y. Xu, S.-X. Zhang, J. Yu, Z. Wu, and D. Yu, "Comparing discrete and continuous space llms for speech recognition," in *Interspeech 2024*, 2024, pp. 2509–2513.

[23] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=rkE3y85ee

[24] S. Watanabe, T. Hori, S. Karita *et al.*, "ESPnet: End-to-end speech processing toolkit," in *Interspeech 2018*, 2018, pp. 2207–2211.

[25] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *ICASSP 2015*, 2015, pp. 5206–5210.

[26] K. Lakhotia, E. Kharitonov, W.-N. Hsu *et al.*, "On generative spoken language modeling from raw audio," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 1336–1354, 2021. [Online]. Available: https://aclanthology.org/2021.tacl-1.79

[27] J. S. Garofolo, L. F. Lamel, W. M. Fisher *et al.*, "TIMIT acoustic-phonetic continuous speech corpus," 01 1993. [Online]. Available: https://cir.nii.ac.jp/crid/1881146593179904768

[28] J. Shen, R. Pang, R. J. Weiss *et al.*, "Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions," in *ICASSP 2018*, 2018, pp. 4779–4783.

[29] K. Ito and L. Johnson, "The LJ speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[30] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020*, 2020, pp. 6199–6203.

[31] A. Radford, J. W. Kim, T. Xu *et al.*, "Robust speech recognition via large-scale weak supervision," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho *et al.*, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 28 492–28 518. [Online]. Available: https://proceedings.mlr.press/v202/radford23a.html

[32] C. Veaux, J. Yamagishi, K. MacDonald *et al.*, "CSTR VCTK corpus: English multi-speaker corpus for cstr voice cloning toolkit," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, vol. 6, p. 15, 2017.

[33] T. Saeki, D. Xin, W. Nakata *et al.*, "UTMOS: UTokyo-SaruLab system for VoiceMOS challenge 2022," in *Interspeech 2022*, 2022, pp. 4521–4525.

[34] J. Shi, D. Berrebbi, W. Chen *et al.*, "ML-SUPERB: Multilingual speech universal performance benchmark," in *Interspeech 2023*, 2023, pp. 884–888.