INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Self-supervised Fine-tuning for Improved Content Representations by Speaker-invariant Clustering

Heng-Jui Chang, Alexander H. Liu, James Glass

MIT CSAIL, USA

{hengjui,alexhliu,glass}@mit.edu

Abstract

Self-supervised speech representation models have succeeded in various tasks, but improving them for content-related problems using unlabeled data is challenging. We propose speaker-invariant clustering (Spin), a novel self-supervised learning method that clusters speech representations and performs swapped prediction between the original and speaker-perturbed utterances. Spin disentangles speaker information and preserves content representations with just 45 minutes of fine-tuning on a single GPU. Spin improves pre-trained networks and outperforms prior methods in speech recognition and acoustic unit discovery.[1]

Index Terms: self-supervised learning, vector quantization, online clustering, speaker disentanglement, content representation

1. Introduction

Self-supervised learning (SSL) for speech representation using large neural networks and unlabeled data offers effective initialization and representations for downstream tasks [1–5]. Among prior methods, learning discrete units like K-means clusters benefits downstream performance [6–12]. While speech representation encompasses information from multiple aspects, most SSL methods lack explicit speaker disentanglement. Extracting speaker-invariant linguistic content can benefit downstream tasks like automatic speech recognition (ASR) and phoneme recognition (PR) [13–17]. In light of this, ContentVec [18] imposes speaker-invariant constraints to pre-trained HuBERT models [6] to improve content-related downstream tasks. However, ContentVec adds a substantial amount of computational cost, requiring 19 hours on 36 GPUs, on top of the pre-trained models, which are already expensive to compute.

In this paper, we first demonstrate the benefits of extracting features closer to the underlying phonetic content to motivate our work. Following this observation, we present speaker-invariant clustering (Spin), a novel and cost-effective self-supervised fine-tuning (SSFT)[2] method for SSL models that leverages vector quantization [21–23] and speaker disentanglement [18] to improve content representation. In short, Spin is trained to identify the unchanged spoken content from pairs of speaker-augmented utterances via quantized representation matching. Such design leads to a disentangled representation focusing on the spoken content, improving various downstream tasks, including content-related tasks in SUPERB [1] and ZeroSpeech [24]. In terms of efficiency, we show that Spin requires less than 45 minutes of training on a single GPU, costing less than 1% of ContentVec.

[1]Code: https://github.com/vectominist/spin

[2]We use the term SSFT to distinguish fine-tuning methods using only audio [18, 19] from supervised fine-tuning using labeled data [20].

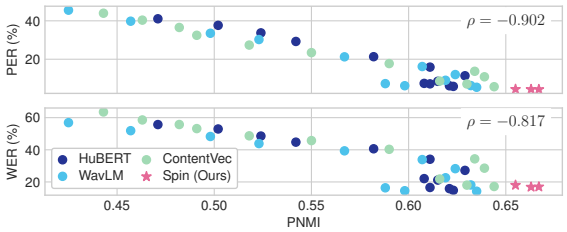


Figure 1: *Content representation quality (PNMI) vs. phoneme/word error rates (PER/WER) of SSL model hidden layer representations under a simplified setup in SUPERB [1]. ρ is Spearman's rank correlation coefficient.*

2. Method

2.1. Importance of Content Representation

This work assumes representations closer to the underlying phonetic content yield better performance for speaker-invariant downstream tasks like ASR. To verify this assumption, we extract speech representations from each layer of three pre-trained SSL models (HuBERT [6], WavLM [7], and ContentVec [18]) and compute two metrics: 1) the phone-normalized mutual information (PNMI; Sec. 3.5) that measures the similarity between phonemes and the discrete units derived by running K-means clustering on the extracted representation ($K = 256$); 2) the phone/word recognition error rate using the extracted features and a lightweight predictor as detailed in Sec. 3.3.

In Fig. 1, higher PNMI representations generally offer better recognition results across all models and layers. The Spearman's rank correlation coefficients for PNMI-PER (-0.902) and PNMI-WER (-0.817) verify the strong correlation between the content encoded and downstream performance, leading us to propose an SSFT method that learns from discrete acoustic units to focus on content encoding.

2.2. Proposed Method

Overview. An overview of the proposed Spin is illustrated in Fig. 2. Inspired by Swapping Assignments between Views (SwAV) [23] for image representation learning, our idea is to learn speaker-invariant clusters that capture the same content shared between perturbed speech and the original speech.

Speaker Perturbation. To alter the speaker identity without changing the spoken content, we adopt an algorithm proposed by Choi et al. [25] as ContentVec [18]. The algorithm randomly and uniformly scales formant frequencies and F0, and random equalization is applied. Because voice information resides in the formant frequencies and F0 [26], and the content is stored in the relative ratio between formant frequencies [27], this algorithm efficiently alters speakers with little content loss.

10.21437/Interspeech.2023-847

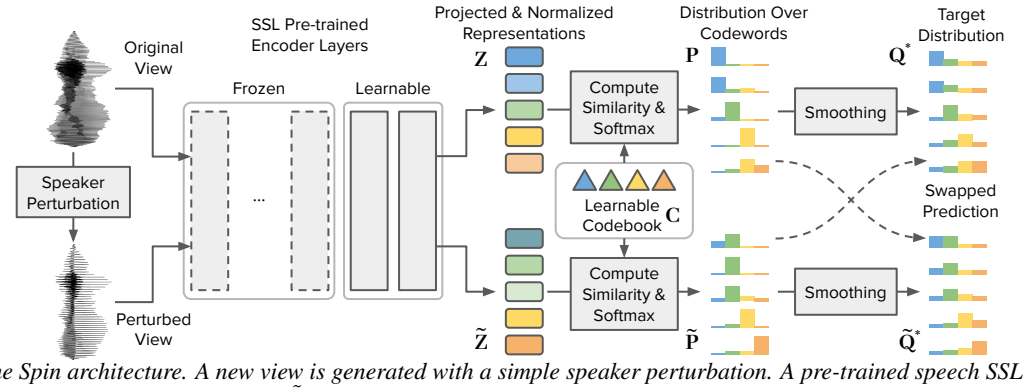


Figure 2: *The Spin architecture. A new view is generated with a simple speaker perturbation. A pre-trained speech SSL model extracts representations from both utterances (*$\mathbf{Z}$*/* $\tilde{\mathbf{Z}}$*). Representations are projected, normalized, and quantized with a learnable codebook into probability distributions (*$\mathbf{P}$*/* $\tilde{\mathbf{P}}$*). The distributions are smoothed to enforce full codebook usage (*$\mathbf{Q}^*$*/* $\tilde{\mathbf{Q}}^*$*). Finally, each frame's distribution is used to predict the target distribution produced by the other view (*$\mathbf{P} \rightarrow \tilde{\mathbf{Q}}^*$ *and* $\tilde{\mathbf{P}} \rightarrow \mathbf{Q}^*$*).*

Speaker-invariant Clustering. With the speaker-augmented and the original speech pair, we aim to discover the consistent underlying content via speaker-invariant clustering. As in Fig. 2, the output of the original view from the encoder is linearly projected and L2-normalized to representations $\mathbf{Z} = [\boldsymbol{z}_1 \dots \boldsymbol{z}_B]^\top \in \mathbb{R}^{B \times D}$, where D is the dimension of the representations, and B is the number of frames in a batch. A probability distribution is computed per frame by taking softmax over the scaled cosine similarity between $\mathbf{Z}$ and a learnable codebook of K codewords $\mathbf{C} = [\boldsymbol{c}_1 \dots \boldsymbol{c}_K]^\top \in \mathbb{R}^{K \times D}$ as

$$p\left(k|\boldsymbol{z}_b\right) = \frac{\exp\left(\boldsymbol{z}_b^\top \boldsymbol{c}_k / \tau\right)}{\sum_{k'} \exp\left(\boldsymbol{z}_b^\top \boldsymbol{c}_{k'} / \tau\right)},$$

for $k \in [K]$, $b \in [B]$,[3] $\|\boldsymbol{c}_k\|_2 = 1$, and $\tau > 0$ is a scaling temperature. We define $q\left(k|\tilde{\boldsymbol{z}}_b\right)$ the distribution over the same codebook using augmented speech. To learn speaker-invariant clusters that capture the unchanged content, distributions over the codebook should ideally be similar regardless of the speaker, i.e., minimizing the cross-entropy $-q\left(k|\tilde{\boldsymbol{z}}_b\right) \log p\left(k|\boldsymbol{z}_b\right)$.

Smoothing for Full Codebook Usage. In practice, minimizing the aforementioned cross-entropy term leads to a trivial solution where all representations are clustered into a single codeword if q is obtained similarly with p. To address the issue, we smooth the target distribution q to encourage higher utilization of the codewords. Following Asano et al. [22], q is obtained by

$$\mathbf{Q}^* \in \arg\max_{\mathbf{Q}} \operatorname{Tr}\left(\mathbf{Q}\mathbf{C}\mathbf{Z}^\top\right) + \varepsilon H\left(\mathbf{Q}\right), \qquad (1)$$

where $\mathbf{Q}^* \in [0,1]^{B \times K}$, $q\left(k|\boldsymbol{z}_b\right) = \mathbf{Q}^*_{b,k}$, and $H\left(\mathbf{Q}\right) = -\sum_{ij} \mathbf{Q}_{ij} \log \mathbf{Q}_{ij}$ is the entropy. The optimized variable $\mathbf{Q}$ is constrained so that each row is a probability distribution over the K codewords. When $\varepsilon = 0$, q is a categorical distribution and easily collapses to using only one codeword. When $\varepsilon > 0$, the entropy term smooths the distribution so that all codewords can be utilized more evenly, whereas a higher ε leads to a more uniform distribution. Eq. 1 can be efficiently solved by the Sinkhorn-Knopp algorithm on GPUs [28]. Note that no gradient is applied to q.

Speaker-invariant Swapped Prediction. With the smoothed target distribution q, the goal is to perform speaker-invariant swapped prediction by minimizing the cross-entropy loss

$$-\frac{1}{2B} \sum_b \sum_k \left[q\left(k|\tilde{\boldsymbol{z}}_b\right) \log p\left(k|\boldsymbol{z}_b\right) + q\left(k|\boldsymbol{z}_b\right) \log p\left(k|\tilde{\boldsymbol{z}}_b\right)\right],$$

where the second term emerges from the interchangeability of the role of the augmented and original speech.

[3] $[N]$ is defined as $\{1, 2, \dots, N\}$.

This objective encourages the model to produce similar representations at the same position between two different views by learning a codebook encoding speaker-invariant acoustic units. Since learning fewer parameters reduces computation, and top layers encode phonetic content [29–31], we propose fine-tuning some top layers to balance the tradeoff between downstream performance and training computation. Unlike previous methods, Spin does not require random masking, so all frames are utilized and contribute to updating the network. Spin is limited to pre-trained models because only the positional information is learned if the model is trained from scratch.

3. Experiments

3.1. Setup

Data. Spin is trained with the LibriSpeech train-clean 100 hours subset [33], and we found more data does not improve.

Implementation. We applied Spin to HuBERT [6] and WavLM [7], and only the last two layers are fine-tuned (7M parameters per layer).[4] We set $D = 256$, $\tau = 0.1$, $\varepsilon = 0.02$, and sweep the codebook sizes $K \in \{128, 256, 512, 1024, 2048\}$. Each view's mini-batch has at most 256 seconds of speech, corresponding to $B = 12.8$k frames. The learning rate is first linearly increased from 0 to 10^{-4} for 2.5k updates, then linearly decreased to 10^{-6} for 2.5k updates. The Sinkhorn-Knopp algorithm iterates three times to compute $\mathbf{Q}^*$ per view. Spin is trained on a single RTX A5000 GPU, each taking 45 minutes. We select models that are trained with all 5k updates.

3.2. Speech SSL Models

HuBERT and **WavLM** are pre-trained to predict cluster IDs of masked audio frames from clustering MFCC features or hidden representations of pre-trained models. These models serve as baselines for Spin. **data2vec** [32] is trained to masked-predicting hidden representations of the exponential moving average of the model itself. We avoid applying Spin to data2vec because the phonetic content resides at the bottom layers (Table 2), requiring fine-tuning many more top layers, and thus increasing computation costs. **ContentVec** is a stronger baseline as it is also trained to improve extracting content with speaker disentanglement. ContentVec learns to mask-predict a pre-trained HuBERT hidden representation K-means clusters. Based on the number of clusters in the target, there are two

[4] Checkpoints: `https://github.com/s3prl/s3prl`

Table 1: *SUPERB [1] phoneme recognition (PR), automatic speech recognition (ASR), keyword spotting (KS), query-by-example (QbE), intent classification (IC), and slot filling (SF). Metrics include accuracy (Acc%), phoneme error rate (PER%), word error rate (WER%), maximum term weighted value (MTWV), F1 score, and concept error rate (CER%). PT and SSFT denote pre-training and self-supervised fine-tuning. Top-3 best results are underlined. The number of hours of processed speech is computed with Eq. 2.*

	Training Processed Speech in Hours		Content				Semantic		
			PR	ASR	KS	QbE	IC	SF	
Method	PT	SSFT	PER↓	WER↓	Acc↑	MTWV↑	Acc↑	F1↑	CER↓
wav2vec 2.0 [20]♠	640k	0	5.74	6.43	96.23	0.0233	92.35	88.30	24.77
HuBERT [6]♠	506k	0	5.41	6.42	96.30	0.0736	98.34	88.53	25.20
WavLM [7]♠	1439k	0	4.84	6.31	96.79	0.0870	98.63	89.38	22.86
data2vec [32]♠	420k	0	4.69	4.94	96.56	0.0576	97.63	88.59	25.27
ContentVec_{500} [18]♣	506k	76k	4.54◇	5.70	96.40	0.0590	99.10	89.60	23.60
HuBERT + Spin_{256}	506k	356	4.39	6.34	96.53	0.0912	98.34	89.00	24.32
WavLM + Spin_{256}	1439k	356	4.18	5.88	96.20	0.0879	98.52	88.84	24.06

♠ Source: `https://superbbenchmark.org/leaderboard` (as of 3/7/2023). ♣ Reported in: [18]
◇ Re-implement for a fair comparison (original: 4.90).

	ContentVec_{500}	Spin_{128}	Spin_{256}	Spin_{512}	Spin_{1024}	Spin_{2048}
HuBERT	0.00017	0.00000	0.00000	0.00000	0.00000	0.00000
WavLM	0.07486	0.00284	0.00222	0.00300	0.00012	0.00000
ContentVec_{500}		0.25110	0.22849	0.25472	0.04632	0.00837

Figure 3: *t-test p-values of SUPERB [1] phoneme recognition error rates. All Spin models here are based on HuBERT.*

versions: ContentVec_{100} and ContentVec_{500}. These SSL models share a similar architecture: a 7-layer CNN feature extractor followed by a 12-layer transformer encoder [34], approximately having 95M parameters each. All models are frozen in evaluation tasks, and continuous transformer encoder hidden representations are used unless otherwise specified.

3.3. SUPERB

This section evaluates Spin on content and semantic tasks in the Speech processing Universal PERformance Benchmark (SUPERB) [1]. Each task and SSL model uses a set of learnable weights to weighted-sum representations across hidden layers of the frozen SSL model. The aggregated features are then fed to a prediction head for supervised training. We report phoneme recognition (PR), automatic speech recognition (ASR), keyword spotting (KS), query-by-example spoken term discovery (QbE), intent classification (IC), and slot filling (SF). We choose $K = 256$ for Spin as it offers the best overall results.

In Table 1, Spin benefits learning content representations because HuBERT and WavLM are improved in content-related tasks (PR, ASR, and QbE) while reducing the performance gap with ContentVec. According to the significance test on PR in Fig. 3, Spin passes t-test compared with HuBERT and WavLM. Increasing the codebook size ($K = 1024$ and 2048) outperforms ContentVec with a $p < 0.05$. Next, we show the hours of processed speech during training

$$\text{processed speech} = \text{training steps} \times \text{effective batch size} \quad (2)$$

to quantify machine-independent training costs. Based on these data, Spin requires less than 0.5% computation of ContentVec to outperform in PR and QbE while offering similar performance in other tasks. Moreover, most models perform similarly in KS and IC, and we found these tasks sensitive to hyperparameters, making them less suitable for comparison. Overall, Spin improves SSL models with a meager budget.

Table 2: *ABX error rates (%) on the ZeroSpeech 2021 phonetic dev set [24]. W and C before "–" denote within and across speakers. C and O after "–" denote clean and other corpus partitions. Only the layer with the lowest average score is reported for each model and is specified in column L.*

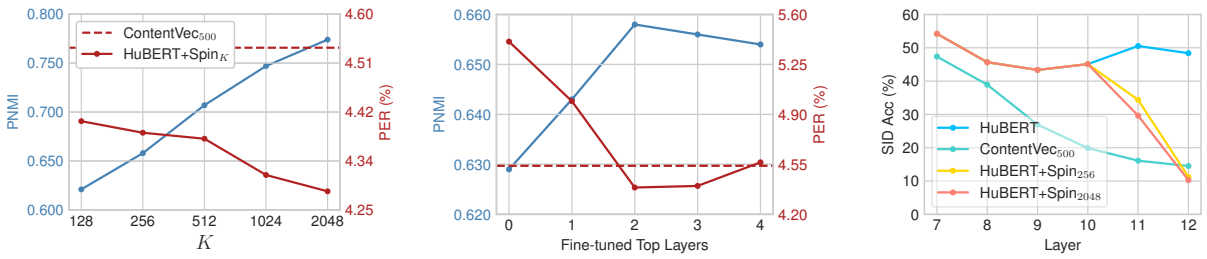
Method	L	W-C	W-O	C-C	C-O	Avg
Nguyen et al. [35]	–	3.26	3.81	4.00	5.91	4.25
Chorowski et al. [36]	–	2.95	3.54	4.50	7.05	4.51
HuBERT	11	3.07	3.90	3.71	6.19	4.22
WavLM	11	2.73	3.41	3.21	4.95	3.58
data2vec	4	4.03	5.09	4.72	6.97	5.20
ContentVec_{100}	12	2.98	3.70	3.44	5.17	3.82
ContentVec_{500}	12	3.91	4.37	4.46	5.80	4.64
HuBERT + Spin_{2048}	12	**2.44**	**3.00**	**2.81**	**3.76**	**3.00**
WavLM + Spin_{2048}	12	2.75	3.33	3.24	4.17	3.37

3.4. Acoustic Unit Discovery

This section inspects linguistic units captured in representations with Zero Resource Speech Benchmark (ZeroSpeech) 2021 [24]. The phonetic task measures how well speech representations distinguish between different phonemes via the ABX discrimination test [37]. We report $K = 2048$ since it performs the best in this task. As shown in Table 2, Spin boosts both models and surpasses the baselines, especially for HuBERT, surpassing prior art and reducing the average ABX error rate by a relative 29%. Although the performance gain for WavLM is minor, error rates of other corpus partitions are reduced, indicating that Spin helps WavLM in a noisier scenario. The results directly demonstrate that Spin improves extracting phonemes.

3.5. Discrete Unit Quality

This section analyzes discrete acoustic unit quality to reveal the relationship between speech representations and phonemes. We adopt three metrics proposed in HuBERT [6]: 1) cluster purity measures the purity of each phoneme's associated discrete units; 2) phone purity measures the average phoneme purity within one class of discrete units; 3) phone-normalized mutual information (PNMI) measures the uncertainty reduction for the underlying phone when observing the codeword of a frame. Higher values imply better performance. K-means clustering is



(a) (b) (c)

Figure 4: *PNMI and PER of HuBERT + Spin with different (a) codebook sizes and (b) fine-tuning layers. Fine-tuning zero layers in (b) denotes the HuBERT baseline. Results in (b) use $K = 256$. (c) shows layer-wise speaker identification accuracy.*

Table 3: *Discrete unit quality. Cls Pur, Phn Pur, and PNMI denote cluster purity, phone purity, and phone-normalized mutual information [6]. Only the layer with the highest PNMI is reported for each model and is specified in column L.*

Method	L	Cls Pur	Phn Pur	PNMI
K-means Clustering ($K = 256$)				
HuBERT	7	0.154	0.639	0.630
WavLM	11	**0.178**	0.624	0.640
data2vec	4	0.173	0.652	0.630
ContentVec$_{100}$	12	0.169	0.650	0.643
ContentVec$_{500}$	8	0.154	0.639	0.629
HuBERT + Spin$_{256}$	12	0.150	0.641	0.655
HuBERT + Spin$_{2048}$	12	0.151	**0.654**	**0.666**
WavLM + Spin$_{256}$	12	0.137	0.644	0.658
WavLM + Spin$_{2048}$	12	0.153	0.650	**0.666**
Online Clustering (**Codebook**)				
VQ-APC [38]♠	–	0.078	0.240	0.189
Co-training APC [39]♣	–	0.089	0.308	0.294
HuBERT + Spin$_{256}$ ◇	–	**0.138**	0.642	0.658
WavLM + Spin$_{256}$ ◇	–	0.133	**0.646**	**0.659**

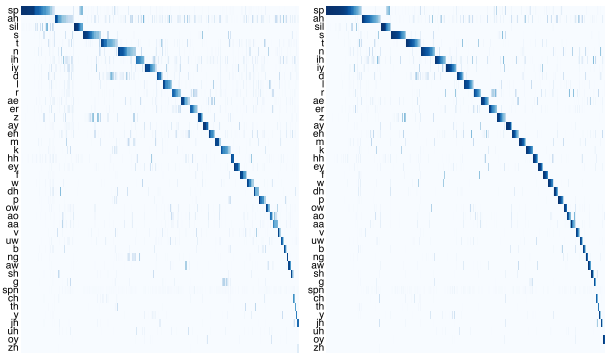
♠ 98 out of 512 codewords are utilized.
♣ 164 out of 256 codewords are utilized.
◇ All 256 codewords are utilized.

performed on a random 10 hours subset of LibriSpeech train-clean-100 split. The discrete units are evaluated on the combination of LibriSpeech dev-clean and dev-other splits. The offline clustering scores are averaged over three runs.

First, we cluster continuous representations into 256 clusters and report the layer with the highest PNMI, as shown in the upper part of Table 3. Independent of codebook sizes and pre-trained models, Spin outperforms all baselines in PNMI. Increasing the codebook size in Spin improves all three metrics (Spin$_{256}$ vs. Spin$_{2048}$), indicating that a larger codebook learns more fine-grained phoneme representations.

For online clustering (codebook learning), we compare the codebook in Spin$_{256}$ with VQ-APC [38] and Co-training APC [39], where the latter two methods leverage codebook learning to improve content modeling. We produce discrete units for Spin by taking $\arg\max$ over p per frame. In the lower part of Table 3, codebooks in Spin achieve high PNMI compared with prior works. Unlike prior methods, because of the constraint in Eq. 1, all learned codewords are utilized in Spin.

Next, we visualize $P(\text{phone}|\text{code})$ in Fig. 5 to demonstrate the relation between learned codewords and phonemes. Since the vertical axes are sorted by phoneme occurrence frequency in human speech, the figures show that Spin assigns more codewords to represent high-frequency phonemes. Furthermore, because off-diagonal values of $K = 2048$ are lower than those of



(a) $K = 256$ (b) $K = 2048$

Figure 5: $P(\text{phone}|\text{code})$ *for HuBERT + Spin. The vertical axes represent the phones sorted from high to low frequencies.*

$K = 256$ (Fig. 5b vs. 5a), a larger codebook helps each code to focus on encoding one phoneme, consistent with phone purity in Table 3. Overall, Spin learns good discrete acoustic units and improves continuous representations in SSL models.

3.6. Analysis

Codebook Size. In Fig. 4a, a larger codebook size improves discrete unit quality and PER, consistent with Sec. 3.4 and 3.5. Even when K is only 128, Spin outperforms ContentVec.

Fine-tuning Strategy. In Fig. 4b, Spin surpasses HuBERT and ContentVec when fine-tuning two or three layers. Moreover, fine-tuning two or three layers perform similarly, indicating that the choice of fine-tuning layers is flexible and robust.

Speaker Invariance. We show each layer's performance in SUPERB speaker identification, where prediction heads are trained only with 50k updates. In Fig. 4c, independent of codebook sizes, Spin reduces identification accuracy to 10% in the last layer, slightly lower than ContentVec, successfully removing speaker identity.

4. Conclusion

This paper proposes Spin, a self-supervised fine-tuning method that improves content representations motivated by speaker disentanglement and the strong relationship between discrete unit quality and downstream performance. We offer empirical evidence that the proposed method benefits various content-related tasks. Although only applying to HuBERT and WavLM, Spin paves a new way to enhance speech representation models after pre-training at a very low cost. Future works include applying Spin to other speech SSL models, introducing more complex data augmentation to improve robustness, and extending to pre-train networks from scratch.

5. References

[1] S.-w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin *et al.*, "SUPERB: Speech processing universal performance benchmark," in *Interspeech*, 2021.

[2] S. Evain, H. Nguyen, H. Le, M. Z. Boito, S. Mdhaffar, S. Alisamir, Z. Tong, N. Tomashenko, M. Dinarelli, T. Parcollet, A. Allauzen, Y. Estève, B. Lecouteux, F. Portet, S. Rossato, F. Ringeval, D. Schwab, and L. Besacier, "LeBenchmark: A reproducible framework for assessing self-supervised representation learning from speech," in *Interspeech*, 2021.

[3] X. Chang, T. Maekaku, P. Guo, J. Shi, Y.-J. Lu, A. S. Subramanian, T. Wang, S. wen Yang, Y. Tsao, H. yi Lee, and S. Watanabe, "An exploration of self-supervised pretrained representations for end-to-end speech recognition," in *ASRU*, 2021.

[4] H.-S. Tsai, H.-J. Chang, W.-C. Huang, Z. Huang, K. Lakhotia, S.-w. Yang, S. Dong, A. Liu, C.-I. Lai, J. Shi, X. Chang, P. Hall, H.-J. Chen, S.-W. Li, S. Watanabe, A. Mohamed, and H.-y. Lee, "SUPERB-SG: Enhanced speech processing universal PERformance benchmark for semantic and generative capabilities," in *ACL*, 2022.

[5] A. Mohamed, H.-y. Lee, L. Borgholt, J. D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S.-W. Li, K. Livescu, L. Maaløe *et al.*, "Self-supervised speech representation learning: A review," *IEEE JSTSP*, 2022.

[6] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, vol. 29, 2021.

[7] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE JSTSP*, vol. 16, no. 6, 2022.

[8] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu, "W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *ASRU*, 2021.

[9] T. Maekaku, X. Chang, Y. Fujita, and S. Watanabe, "An exploration of hubert with large number of cluster units and model assessment using bayesian information criterion," in *ICASSP*, 2022.

[10] C.-C. Chiu, J. Qin, Y. Zhang, J. Yu, and Y. Wu, "Self-supervised learning with random-projection quantizer for speech recognition," in *ICML*, 2022.

[11] S. Ren, S. Liu, Y. Wu, L. Zhou, and F. Wei, "Speech pre-training with acoustic piece," *Interspeech*, 2022.

[12] D. Wells, H. Tang, and K. Richmond, "Phonetic analysis of self-supervised representations of english speech," in *Interspeech*, 2022.

[13] W.-N. Hsu, Y. Zhang, and J. Glass, "Unsupervised learning of disentangled and interpretable representations from sequential data," *NIPS*, 2017.

[14] A. Tjandra, R. Pang, Y. Zhang, and S. Karita, "Unsupervised learning of disentangled speech content and style representation," *Interspeech*, 2021.

[15] D. M. Chan and S. Ghosh, "Content-context factorized representations for automated speech recognition," *Interspeech*, 2022.

[16] C. Peyser, W. R. Huang, A. Rosenberg, T. Sainath, M. Picheny, and K. Cho, "Towards disentangled speech representations," *Interspeech*, 2022.

[17] J. Williams, "Learning disentangled speech representations," Ph.D. dissertation, The University of Edinburgh, 2022.

[18] K. Qian, Y. Zhang, H. Gao, J. Ni, C.-I. Lai, D. Cox, M. Hasegawa-Johnson, and S. Chang, "Contentvec: An improved self-supervised speech representation by disentangling speakers," in *ICML*, 2022.

[19] K. P. Huang, Y.-K. Fu, Y. Zhang, and H.-y. Lee, "Improving distortion robustness of self-supervised speech processing tasks with domain adaptation," *Interspeech*, 2022.

[20] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[21] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *ECCV*, 2018.

[22] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," *ICLR*, 2020.

[23] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *NeurIPS*, 2020.

[24] T. A. Nguyen, M. de Seyssel, P. Rozé, M. Rivière, E. Kharitonov, A. Baevski, E. Dunbar, and E. Dupoux, "The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling," *arXiv*, 2020.

[25] H.-S. Choi, J. Lee, W. Kim, J. Lee, H. Heo, and K. Lee, "Neural analysis and synthesis: Reconstructing speech from self-supervised representations," *NeurIPS*, 2021.

[26] E. Eide and H. Gish, "A parametric approach to vocal tract length normalization," in *ICASSP*, 1996.

[27] K. N. Stevens, "Relational properties as perceptual correlates of phonetic features," in *International Conference of Phonetic Sciences*, 1987.

[28] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *NIPS*, 2013.

[29] A. Pasad, J.-C. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," *ASRU*, 2021.

[30] H.-J. Chang, S.-w. Yang, and H.-y. Lee, "DistilHuBERT: Speech representation learning by layer-wise distillation of hidden-unit bert," in *ICASSP*, 2022.

[31] L.-H. Tseng, Y.-K. Fu, H.-J. Chang, and H.-y. Lee, "Mandarin-english code-switching speech recognition with self-supervised speech representation models," *AAAI SAS Workshop*, 2022.

[32] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, "data2vec: A general framework for self-supervised learning in speech, vision and language," in *ICML*, 2022.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *ICASSP*, 2015.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017.

[35] T. A. Nguyen, B. Sagot, and E. Dupoux, "Are discrete units necessary for spoken language modeling?" *IEEE JSTSP*, 2022.

[36] J. Chorowski, G. Ciesielski, J. Dzikowski, A. Łańcucki, R. Marxer, M. Opala, P. Pusz, P. Rychlikowski, and M. Stypułkowski, "Information Retrieval for ZeroSpeech 2021: The Submission by University of Wroclaw," in *Interspeech*, 2021.

[37] T. Schatz, "Abx-discriminability measures and applications," Ph.D. dissertation, Université Paris 6 (UPMC), 2016.

[38] Y.-A. Chung, H. Tang, and J. Glass, "Vector-quantized autoregressive predictive coding," in *Interspeech*, 2020.

[39] S.-L. Yeh and H. Tang, "Autoregressive Co-Training for Learning Discrete Speech Representation," in *Interspeech*, 2022.