



DC-Spin: A Speaker-invariant Speech Tokenizer for Spoken Language Models

Heng-Jui Chang¹, Hongyu Gong², Changhan Wang², James Glass¹, Yu-An Chung²

¹MIT CSAIL, USA

²Meta AI, USA

hengjui@mit.edu, andyyuan@meta.com

Abstract

Spoken language models (SLMs) have gained increasing attention with advancements in text-based, decoder-only language models. SLMs process text and speech, enabling simultaneous speech understanding and generation. This paper presents SpinHuBERT and Double-Codebook Speaker-invariant Clustering (DC-Spin) to improve speech tokenization for bridging audio signals and SLM tokens. DC-Spin extracts speaker-invariant tokens rich in phonetic information and resilient to input variations, enhancing zero-shot SLM tasks and speech resynthesis. Comparisons of tokenization methods and downstream task proxies show that tokens easily modeled by an n-gram LM or aligned with phonemes offer strong performance, offering insights for designing speech tokenizers for SLMs.

Index Terms: speech tokenizer, self-supervised learning, spoken language model, speech resynthesis

1. Introduction

Spoken language models (SLMs) have gained more interest with the advancements of large language models (LLM) and audio tokenization techniques [1]. SLMs resemble causal LMs in natural language processing, but SLMs take speech and, optionally, text as input and generate speech or text. Hence, SLMs can perform tasks like speech continuation [2], automatic speech recognition (ASR) [3, 4], text-to-speech synthesis (TTS) [5], and the more complicated spoken language understanding (SLU) [6–8]. SLM has two main research directions: 1) LM architecture and training and 2) speech tokenization techniques, the latter of which is the focus of this paper.

Since directly taking raw audio waveform as input to an SLM is infeasible, prior studies leverage adaptors or discrete tokens.¹ Adaptors bridge speech encoders and text LMs [7, 9, 10], enabling speech understanding and ASR but complicating speech generation [11]. Token-based SLMs convert speech into discrete units for input and output [2, 4, 12, 13], allowing joint speech-text processing [8] and synthesis from tokens [14].

Self-supervised Learning (SSL) and neural audio codecs are common approaches for speech tokenization. First, SSL pre-trains speech encoders with unlabeled data, reducing reliance on human annotation [15]. These models extract phonetic representations [16–18], making K-means quantization of hidden features a common tokenization method [2, 4, 12]. Fine-tuning approaches further improve SSL encoders for speech tokenization [19, 20]. Next, neural codecs compress speech into compact units and reconstruct high-fidelity signals through residual vector quantization (RVQ) [21–23]. E.g., SpeechTokenizer distills from an SSL teacher model to enforce phonetic

¹Terms “token” and “unit” are used interchangeably in this paper, indicating discrete speech units.

representations but is limited by the teacher [24, 25]. Moreover, closed-source models like USM [3] claim strong performance but lack reproducibility. Therefore, we aim to develop effective tokenizers and provide detailed insights into tokenizer design.

We define two key qualifications for a good speech tokenizer inspired by prior studies. First, the tokens should contain phonetic or semantic information for recognition and understanding tasks [2]. Second, the tokens should retain acoustic details for being resynthesized into speech for generative tasks [5, 24, 26]. Hence, this paper tries to answer the following question: *how to build and evaluate a speech tokenizer for spoken language models that satisfies these qualifications?*

We simplify the setup by training a unit-based speech LM (uLM) [2] and a Hifi-GAN unit-to-speech synthesizer [14, 27]. This setup is commonly used in SLM studies and applications [4, 12, 20], which is an ideal proxy for more advanced SLMs. uLMs are causal transformer LMs [28] trained with the next-token prediction objective on speech tokens. uLMs perform zero-shot tasks like detecting spoken words and syntactic structures by estimating the probability of utterances [29]. We train Hifi-GANs to convert tokens to audio and quantify the intelligibility of the resynthesized speech to simulate speech generation with SLMs. Hence, we can examine whether speech tokenizers satisfy the desired qualities.

This paper proposes Double-Codebook Spin (**DC-Spin**) by extending speaker-invariant clustering (Spin) with an auxiliary codebook to extract strong phonetic speech units [17]. To boost robustness and token quality, we propose pre-training the Hidden-unit BERT (HuBERT) SSL speech encoder with Spin codeword units as a better initialization for DC-Spin [30], denoted as **SpinHuBERT**. First, the proposed tokenizer produces high-quality speaker-invariant speech tokens, achieving state-of-the-art spoken language modeling and speech resynthesis compared to open-source tokenizers on multiple benchmarks with limited resources. Second, we analyze multiple proxy tasks to understand the relation between speech tokenizer and SLM performance. We find that phoneme and character-normalized mutual information and the proposed n-gram predictability are good proxies for downstream tasks.

2. Method

2.1. Background

Hidden-unit BERT (HuBERT) [30] is a speech SSL method pre-trained with a mask prediction objective for multiple iterations, using pseudo labels derived from K-means clustered audio representations. In the first iteration, the labels are K-means cluster IDs of Mel-frequency cepstral coefficients (MFCCs). The second and beyond iteration models predict K-means clusters from the previous model’s hidden embeddings. Be-

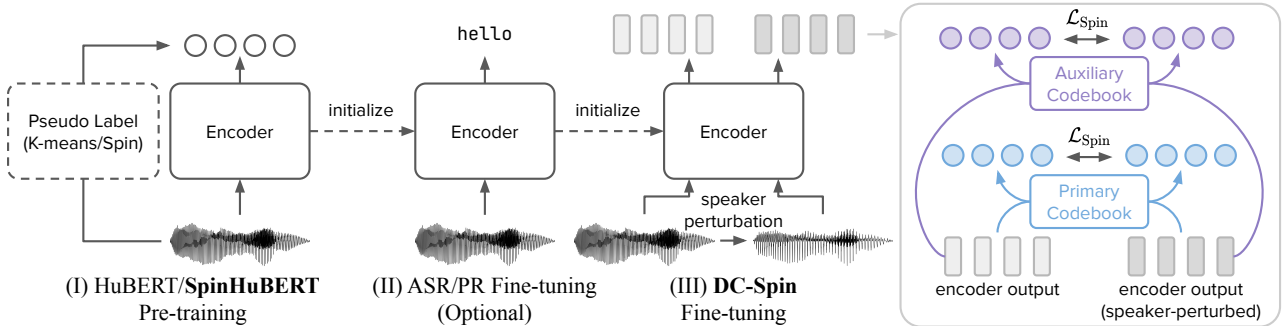


Figure 1: The proposed speech tokenizer training. Stage (I) pre-trains a speech encoder with pseudo labels from K-means or Spin units, where the latter is the proposed SpinHuBERT (Section 2.2). The optional stage (II) fine-tunes the encoder with CTC-based ASR or phoneme recognition (PR). In stage (III), the encoder is fine-tuned with the DC-Spin objective (Section 2.3).

sides pre-training, K-means units preserve phonetic information, making ideal tokens for SLMs [2].

Inspired by [31], **Speaker-invariant Clustering (Spin)** is a fine-tuning approach for improving pre-trained speech encoders in capturing speaker-invariant content in speech through online clustering [17, 32]. During training, each utterance is perturbed to sound like a different speaker. Both utterances are fed to an SSL encoder like HuBERT, and the frame-level output of each utterance is transformed into a sequence of probability distributions over a learnable codebook. The distributions are smoothed to enforce full codebook usage and serve as the learning target. Finally, the model minimizes the cross-entropy loss between the original codeword distribution and the smoothed targets from the perturbed output and vice versa. With minimal computing, Spin removes speaker information and improves SSL encoders in content-related problems like ASR and phoneme recognition (PR). Although the frame-wise codeword IDs of Spin align with phonemes and characters [32], the applications of these units remain undiscovered.

2.2. SpinHuBERT: HuBERT with Better Targets

Spin can be applied to any SSL speech encoder, but the fine-tuned performance depends on the encoder’s quality. Because of the speaker-invariant nature of Spin, discrete units derived from Spin codebooks are closer to phonemes than HuBERT K-means units [17, 32]. Hence, we propose **SpinHuBERT** by training HuBERT with Spin codeword IDs, which is expected to extract better content and phonetic representations.

Furthermore, inspired by [3, 11, 33], we introduce **Supervised Fine-tuning (SFT)** to boost SpinHuBERT. Specifically, we consider CTC-based [34] ASR and PR as the supervised tasks because 1) the data are relatively easy to collect compared to frame-level labels and 2) both tasks force the model to neglect redundant information and extract the content in speech. With SpinHuBERT and SFT, we expect a strong initial encoder model for Spin fine-tuning.

2.3. DC-Spin: Spin as Speech Tokenizer

Since Spin codebooks capture phonetic information and are speaker-invariant, the tokens extracted from Spin satisfy the first qualification in Section 1. These properties are beneficial for speech generation because the vocoder can condition on different speakers, allowing more flexible speech synthesis. Meanwhile, the Spin codebook is optimized with gradient descent and proven highly scalable compared with K-means [35]. Thus, this paper explores tokenizing speech with Spin for SLMs.

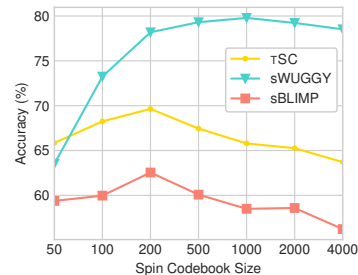


Figure 2: HuBERT + Spin on zero-shot SLM (see Sections 3.1 and 3.2 for the setup).

First, we fine-tune HuBERT with different Spin codebook sizes and use the codeword IDs as discrete units to perform zero-shot SLM tasks, where the setup can be found in Sections 3.1 and 3.2. In Figure 2, 200 and 500 codewords perform relatively better. However, large codebooks help speech resynthesis and improve phonetic representations in Spin [17]. The contradiction motivates us to develop **Double-Codebook Spin (DC-Spin)** to obtain a small but high-quality codebook.

DC-Spin extends Spin to two learnable codebooks as illustrated in the far right part of Figure 1. The primary codebook extracts discrete units for downstream tasks. The auxiliary codebook is a large codebook that enhances the encoder to capture fine-grained phonetic units. Because both codebooks share the same encoder, the auxiliary codebook indirectly helps the primary codebook encode high-quality units.

The training pipeline is shown in Figure 1. In stage (I), we pre-train a SpinHuBERT with pseudo labels generated with Spin. Optionally, SFT with ASR or PR is applied to stage (II). Stage (III) fine-tunes the encoder with the DC-Spin objective to obtain the discrete tokens from the primary codebook.

3. Experiment

3.1. Setup

Baselines We adopt EnCodec [22] and SpeechTokenizer [24] as the neural codec baselines. For SSL-based methods, we consider K-means clustering, augmentation invariant discrete representation [19], and Noise Aware Speech Tokenization (NAST) [20], where the latter two methods are designed for SLM via perturbation-invariant objectives. K-means clustering with K centroids is denoted as “K-means $_K$.”

SSL Pre-training We follow the architecture and training setup for HuBERT Base to train the 3rd-iteration HuBERT (it3) and SpinHuBERT, but we use a 124k-hour English corpus. Hu-

Table 1: Zero-shot SLM evaluation for unsupervised speech tokenizers based on HuBERT Base and the LibriSpeech dataset. All SLMs share the same architecture (150M parameters).

Units	Method	TSC \uparrow		sWUGGY \uparrow		sBLIMP \uparrow
		all	in-vocab	all	in-vocab	
50	K-means [20]	66.27	–	67.48	–	52.42
	[19] \blacklozenge	–	–	67.42	–	57.04
	NAST ₅₀ [20]	64.51	–	67.14	–	54.34
	Spin ₅₀	65.85	58.90	63.52	–	59.38
	DC-Spin _{50,4096}	69.91	65.05	73.51	–	60.15
	100	K-means [20]	67.18	–	67.75	–
[19] \blacklozenge	–	–	68.20	–	56.99	
NAST ₁₀₀ [20]	64.13	–	73.35	–	55.86	
Spin ₁₀₀	68.25	65.28	73.25	–	59.97	
DC-Spin _{100,4096}	70.18	68.04	78.47	–	61.35	
200	K-means [20]	67.55	–	71.88	–	52.43
	[19] \blacklozenge	–	–	70.68	–	56.26
	NAST ₂₀₀ [20]	66.70	–	76.42	–	55.62
	Spin ₂₀₀	69.64	68.95	78.19	–	62.55
	DC-Spin _{200,4096}	69.21	70.79	80.59	–	62.13
	500	K-means	63.23	66.74	74.72	–
[19] \blacklozenge	–	–	69.33	–	56.93	
Spin ₅₀₀	67.45	70.03	79.31	–	60.08	
DC-Spin _{500,4096}	67.50	71.48	81.38	–	60.84	

\blacklozenge The authors could not confirm the subset of sWUGGY, but it is more likely to be the in-vocab set according to [20].

BERT it3 learns to predict 500-unit K-means clusters of the 9th layer of HuBERT Base. SpinHuBERT predicts tokens extracted from a Spin model with a codebook size of 4096.

Supervised Fine-tuning We fine-tune SpinHuBERT with ASR and PR using a 3k-hour English speech dataset including LibriSpeech [36], denoted as SpinHuBERT-ASR and SpinHuBERT-PR, respectively.

Spin & DC-Spin Fine-tuning Following [17], we fine-tune SSL models with unlabeled data from LibriSpeech on a V100 GPU. “Spin_K” denotes Spin with a codebook size of K . DC-Spin with primary and auxiliary codebook sizes of K_1 and K_2 is denoted as “DC-Spin _{K_1, K_2} .”

Spoken Language Models We adopt uLM as the SLM for a fair comparison with prior works [2]. Each uLM is a 150M-parameter transformer decoder [28] trained with the tokenized 6k hours clean subset of Libri-Light [37]. uLM estimates the log probability of speech utterance normalized by length for zero-shot SLM tasks.

Speech Resynthesis We use the Espresso dataset [38] to train and evaluate unit-to-speech Hifi-GAN vocoders [14, 27]. The input includes a sequence of tokenized speech units, a speaker ID, and a style ID. Speaker and style information is reintroduced during speech resynthesis using external embeddings, so the speaker-invariant property will not compromise the intonation and expressiveness of the generated speech. This design leads to more robust, generalizable SLMs while still supporting rich generation. We resynthesize all utterances in the dev and test sets with the original speaker and speaking style IDs.

3.2. Zero-shot Spoken Language Modeling

This section discusses the impact of tokenizers on SLM. Results are reported in accuracy by comparing the SLM-estimated probabilities of pairs of utterances in the following tasks.

TSC We use the Topic Spoken StoryCloze to evaluate an SLM’s ability to capture continuation coherence and fine-grained textual nuances [12]. Each sample comprises two similar spoken stories with different endings. The SLM is expected

Table 2: Unconstrained resources zero-shot SLM evaluation.

Method	SLM Params	Data (hours)	TSC \uparrow	sWUGGY \uparrow all	sBLIMP \uparrow in-vocab	
High-resource Speech LM						
AudioLM [39]	300M	60k	–	71.5	83.7	64.7
VoxLM [4] \blacklozenge	1.3B	60k	–	65.6	–	57.1
TWIST [12] \blacklozenge	13B	150k	76.4	<u>74.5</u>	84.1	59.2
SPIRIT LM [8] \blacklozenge	7B	460k	82.9	69.0	–	58.3
Moshi [13]	7B	7M	<u>80.9</u>	74.8	–	59.9
Sylber [40]	125M	66k	–	–	78.0	60.8
Baselines						
EnCodec [22]	150M	6k	56.1	52.2	53.1	50.1
SpeechTokenizer [24]	150M	6k	63.7	64.9	72.1	53.9
HuBERT + K-means ₅₀₀	150M	6k	63.2	66.7	74.7	55.5
SpinHuBERT + DC-Spin_{500,4096}						
SpinHuBERT	150M	6k	70.7	72.3	82.2	62.8
SpinHuBERT-ASR	150M	6k	70.2	73.7	<u>84.5</u>	<u>65.7</u>
SpinHuBERT-PR	150M	6k	70.2	74.1	85.0	65.9
Cascaded Topline						
ASR + Llama2 [8]	7B	–	94.8	79.2	–	71.6

\blacklozenge LM pre-trained with text or paired speech-text data.

to find the utterance to have a consistent ending.

sWUGGY We adopt the spot-the-word task from ZeroSpeech [29]. Each sample has two spoken words with similar pronunciations, with one of the words absent from the English vocabulary. The “all” subset combines the “in-vocab” and out-of-vocabulary words not in the LibriSpeech training set.

sBLIMP Like sWUGGY, an SLM is expected to find the grammatically correct sample from two similar utterances.

Table 1 shows the results of unsupervised speech tokenization techniques based on HuBERT Base and LibriSpeech for a fair comparison. DC-Spin surpasses previous methods and consistently improves over Spin across different unit sizes, but the gap is narrowed when the codebook size is 500. Among all tasks, DC-Spin improves sWUGGY most significantly because this task is closely related to how well speech tokens represent pronunciation, which is related to phonetic information.

To compare the proposed methods with state-of-the-art SLMs, we report results with unconstrained resources in Table 2. First, EnCodec tokens perform worst because no explicit constraints are imposed on the encoder or quantizer to extract phonetic or semantic representations. SpeechTokenizer performs similarly to HuBERT with K-means because this model distills from a HuBERT teacher. Next, the proposed SpinHuBERT with DC-Spin offers the best performance on sWUGGY and sBLIMP, even using a relatively small SLM and training data size. DC-Spin is improved using ASR or PR SFT with similar performance gains. Nevertheless, large SLMs excel in TSC, showing that this task might correlate more with LM scaling. Results suggest speech tokenizers greatly impact SLMs, and SpinHuBERT and DC-Spin achieve state-of-the-art SLMs on several tasks with limited resources.

3.3. Speech Resynthesis

This section focuses on speech resynthesis from discrete units. **ASR-WER** uses an ASR model to transcribe the resynthesized speech and computes the word error rate to quantify the intelligibility of the audio [38]. Following [41, 42], we adopt **UTMOS**, a neural network-based mean opinion score (MOS) prediction, to assess the quality of the resynthesized speech because this metric highly correlates with human-rated MOS [43].

As shown in Table 3, HuBERT + DC-Spin reduces more than 10% relative WER compared with K-means, but the K-

Table 3: *Speech resynthesis ASR-WER and UTMOS on Espresso dev and test sets.*

Method	Bitrate	ASR-WER↓		UTMOS↑	
		dev	test	dev	test
Ground Truth	256k	15.2	14.3	3.24	3.28
Neural Codecs					
EnCodec RVQ1:2 [22]	1.5k	28.4	27.5	1.35	1.31
SpeechTokenizer RVQ1 [24]	500	30.7	32.9	1.27	1.27
HuBERT					
K-means ₅₀₀	448	24.0	24.4	2.93	2.76
DC-Spin _{500,4096}	448	21.3	22.4	2.96	2.93
SpinHuBERT					
K-means ₅₀₀	448	20.0	21.2	3.05	2.94
DC-Spin _{500,4096}	448	20.5	21.7	3.11	3.04
+ ASR	448	18.9	20.0	3.08	3.05
+ PR	448	18.8	18.7	3.02	2.92

Table 4: *HuBERT with different learning targets on SUPERB.*

Method	Data (hours)	Content				Semantics			
		PR PER↓	ASR WER↓	KS Acc↑	QbE MTWV↑	IC Acc↑	SF F1↑	ST CER↓	BLEU↑
HuBERT it2	960	5.41	6.42	96.30	0.0736	98.34	88.53	25.20	15.53
HuBERT it3	124k	4.84	7.13	96.01	0.1016	98.37	89.66	23.96	18.00
SpinHuBERT	124k	3.69	6.16	97.14	0.0903	99.24	90.06	22.21	19.62

means and DC-Spin are similar in SpinHuBERT, showing that SpinHuBERT extracts better representations for resynthesis even without DC-Spin. With SFT, the ASR-WERs are further reduced, especially when PR is introduced. Compared with codec-based approaches, DC-Spin tokens can be synthesized to produce high-intelligibility and quality speech at a relatively low bitrate since codecs require multiple RVQ codebooks to perform well. We notice that UTMOS among SSL-based methods are similar, possibly indicating that the resynthesis quality is less relevant to the tokens than the vocoder. Note that the ASR-WERs are greater than 10% because Espresso contains expressive speech like laughter and whispering. To summarize, the effectiveness of the proposed SSL-based tokenizers on speech resynthesis corroborates with the findings in [44].

3.4. Ablation Studies

3.4.1. HuBERT Learning Targets

Table 4 compares the effects of learning targets for HuBERT via the content and semantic-related tasks in the SUPERB benchmark [45, 46]. With more data and an additional iteration, the HuBERT it3 surpasses it2 in most tasks. Next, by introducing a speaker-invariant learning objective via Spin codewords, SpinHuBERT outperforms HuBERT it3 in almost all tasks, indicating the effectiveness of the proposed approach.

3.4.2. DC-Spin Auxiliary Codebook

We inspect the effect of the auxiliary codebook size in DC-Spin by showing the relation between the codebook sizes and zero-shot SLM in Figure 3a. Comparing Spin and DC-Spin (dashed vs. solid lines), DC-Spin helps downstream tasks in almost all cases. Moreover, the performance gain of larger auxiliary codebooks is more prominent in sWUGGY, corroborating with the findings in [17] and Section 3.2 that larger codebooks help the speech encoder capture phonetic representations. The results indicate the necessity of including a large auxiliary codebook to help with the primary Spin codebook for SLM.

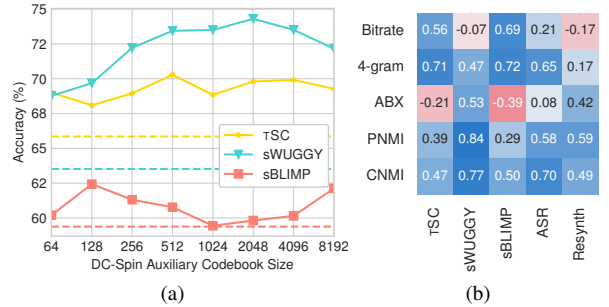


Figure 3: (a) *DC-Spin₅₀₀*, with different auxiliary codebook sizes vs. zero-shot SLM. Dashed lines indicate *Spin₅₀₀*. (b) Pearson correlation coefficients between proxy and downstream tasks.

3.5. Finding Proxy Tasks for Spoken Language Modeling

This section inspects the correlation between tasks to find proper proxies for SLM tasks. We compute the **bitrate** of tokens by considering the distribution of tokens via entropy. We propose **N-gram Predictability** by training a 4-gram LM with speech tokens and reporting the average perplexity on LibriSpeech to measure the difficulty of modeling speech tokens. **Phonetic ABX** error rate quantifies how well a tokenizer can distinguish phonemes [29, 47]. **Phone Normalized Mutual Information (PNMI)** computes the mutual information between the speech tokens and phonemes [30], where higher values imply better alignment with the underlying phoneme distribution. Similarly, **Character Normalized Mutual Information (CNMI)** compares tokens with character alignments obtained by Unity2 [32, 48].

Using 33 tokenizers with 500 units and 50Hz framerate, we compute the Pearson correlation coefficients between proxy and downstream metrics in Figure 3b, where ASR is uLMs finetuned with transcribed speech and evaluated on LibriSpeech. We make the values negative before calculating the coefficients for lower-better metrics (bitrate, 4-gram, ABX, ASR, and resynthesis). First, bitrate positively correlates with TSC and sBLIMP, implying short and compact tokens are more suitable for capturing the long context of speech. Low 4-gram perplexity correlates with SLM tasks, so repeating patterns in tokens improves SLM. The high correlation between PNMI, ABX, and sWUGGY verifies that sWUGGY relies on well-aligned phonetic units (Section 3.2). Similarly, CNMI quantifies the textual alignment quality, making this task more related to sBLIMP and ASR. Speech resynthesis correlates with phoneme alignment metrics (ABX and PNMI), suggesting that this task relies on the phonetic representations captured by the tokens to synthesize intelligible speech. Nevertheless, the ABX error rate negatively correlates with TSC and sBLIMP, implying this commonly used metric might fail as a proxy. Thus, n-gram predictability, PNMI, and CNMI are ideal proxies for developing speech tokenizers.

4. Conclusion

This paper builds and evaluates effective speech tokenizers for SLM and speech resynthesis. Through experiments, we show that the proposed SpinHuBERT and DC-Spin satisfy the two qualifications of an ideal tokenizer: captures phonetic information and acoustic details. We found that n-gram predictability, PNMI, and CNMI strongly correlate with downstream performance, making these tasks ideal proxies and offering future development guidelines. Scaling, multilinguality, and applications like ASR and TTS are potential follow-ups.

5. References

- [1] H. Wu, X. Chen, Y.-C. Lin, K.-w. Chang, H.-L. Chung, A. H. Liu, and H.-y. Lee, "Towards audio language modeling-an overview," *arXiv*, 2024.
- [2] K. Lakhota *et al.*, "On generative spoken language modeling from raw audio," *TACL*, 2021.
- [3] P. K. Rubenstein *et al.*, "Audiopalm: A large language model that can speak and listen," *arXiv*, 2023.
- [4] S. Maiti, Y. Peng, S. Choi, J.-w. Jung, X. Chang, and S. Watanabe, "Voxlm: Unified decoder-only models for consolidating speech recognition, synthesis and speech, text continuation tasks," in *ICASSP*, 2024.
- [5] C. Wang *et al.*, "Neural codec language models are zero-shot text to speech synthesizers," *arXiv*, 2023.
- [6] Y. Gong, A. H. Liu, H. Luo, L. Karlinsky, and J. Glass, "Joint audio and speech understanding," in *ASRU*, 2023.
- [7] Y. Chu, J. Xu, X. Zhou, Q. Yang, S. Zhang, Z. Yan, C. Zhou, and J. Zhou, "Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models," *arXiv*, 2023.
- [8] T. A. Nguyen *et al.*, "Spirit-lm: Interleaved spoken and written language model," *arXiv*, 2024.
- [9] Z. Ma *et al.*, "An embarrassingly simple approach for llm with strong asr capacity," *arXiv*, 2024.
- [10] C. Tang, W. Yu, G. Sun, X. Chen, T. Tan, W. Li, L. Lu, Z. MA, and C. Zhang, "SALMONN: Towards generic hearing abilities for large language models," in *ICLR*, 2024.
- [11] A. Dubey *et al.*, "The llama 3 herd of models," *arXiv*, 2024.
- [12] M. Hassid *et al.*, "Textually pretrained speech language models," *NeurIPS*, 2024.
- [13] A. Défossez, L. Mazaré, M. Orsini, A. Royer, P. Pérez, H. Jégou, E. Grave, and N. Zeghidour, "Moshi: a speech-text foundation model for real-time dialogue," *arXiv*, 2024.
- [14] A. Polyak, Y. Adi, J. Copet, E. Kharitonov, K. Lakhota, W.-N. Hsu, A. Mohamed, and E. Dupoux, "Speech resynthesis from discrete disentangled self-supervised representations," in *Inter-speech*, 2021.
- [15] A. Mohamed *et al.*, "Self-supervised speech representation learning: A review," *IEEE JSTSP*, 2022.
- [16] A. Pasad, J.-C. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *ASRU*, 2021.
- [17] H.-J. Chang, A. H. Liu, and J. Glass, "Self-supervised fine-tuning for improved content representations by speaker-invariant clustering," in *Interspeech*, 2023.
- [18] K. Choi, A. Pasad, T. Nakamura, S. Fukayama, K. Livescu, and S. Watanabe, "Self-supervised speech representations are more phonetic than semantic," in *Interspeech*, 2024.
- [19] I. Gat, F. Kreuk, T. Anh Nguyen, A. Lee, J. Copet, G. Synnaeve, E. Dupoux, and Y. Adi, "Augmentation invariant discrete representation for generative spoken language modeling," in *IWSLT*, 2023.
- [20] S. Messica and Y. Adi, "Nast: Noise aware speech tokenization for speech language models," in *Interspeech*, 2024.
- [21] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *TASLP*, 2021.
- [22] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *TMLR*, 2023.
- [23] Y.-C. Wu, I. D. Gebru, D. Marković, and A. Richard, "Audiodec: An open-source streaming high-fidelity neural audio codec," in *ICASSP*, 2023.
- [24] X. Zhang, D. Zhang, S. Li, Y. Zhou, and X. Qiu, "Spechtok-enizer: Unified speech tokenizer for speech language models," in *ICLR*, 2024.
- [25] 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.
- [26] A. Lee *et al.*, "Direct speech-to-speech translation with discrete units," in *ACL*, 2022.
- [27] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *NeurIPS*, 2020.
- [28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.
- [29] 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," in *NeurIPS SAS*, 2020.
- [30] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhota, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, 2021.
- [31] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *NeurIPS*, 2020.
- [32] H.-J. Chang and J. Glass, "R-spin: Efficient speaker and noise-invariant representation learning with acoustic pieces," in *NAACL*, 2024.
- [33] G. Gemini Team, "Gemini: a family of highly capable multimodal models," *arXiv*, 2023.
- [34] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.
- [35] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*, 2002.
- [36] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *ICASSP*, 2015.
- [37] J. Kahn *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP*, 2020.
- [38] T. A. Nguyen *et al.*, "Expresso: A benchmark and analysis of discrete expressive speech resynthesis," in *Interspeech*, 2023.
- [39] Z. Borsos *et al.*, "Audiolm: A language modeling approach to audio generation," *TASLP*, 2023.
- [40] C. J. Cho, N. Lee, A. Gupta, D. Agarwal, E. Chen, A. Black, and G. Anumanchipalli, "Sylber: Syllabic embedding representation of speech from raw audio," in *ICLR*, 2025.
- [41] P. Mousavi, L. Della Libera, J. Duret, A. Ploujnikov, C. Subakan, and M. Ravanelli, "Dasb-discrete audio and speech benchmark," *arXiv*, 2024.
- [42] X. Chang, J. Shi, J. Tian, Y. Wu, Y. Tang, Y. Wu, S. Watanabe, Y. Adi, X. Chen, and Q. Jin, "The interspeech 2024 challenge on speech processing using discrete units," in *Interspeech*, 2024.
- [43] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "UTMOS: Utokyo-sarulab system for voicemos challenge 2022," in *Interspeech*, 2022.
- [44] 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.
- [45] S. Yang *et al.*, "Superb: Speech processing universal performance benchmark," in *Interspeech*, 2021.
- [46] H.-S. Tsai *et al.*, "SUPERB-SG: Enhanced speech processing universal PERFORMANCE benchmark for semantic and generative capabilities," in *ACL*, 2022.
- [47] T. Schatz, "Abx-discriminability measures and applications," Ph.D. dissertation, Université Paris 6 (UPMC), 2016.
- [48] Seamless Communication, "Seamless: Multilingual expressive and streaming speech translation," *arXiv*, 2023.