

Rethinking Speech Foundation Model Fine-tuning: Better SFT or Better Match?

Wangjin Zhou¹, Yizhou Zhang¹, Yichi Wang¹, Tatsuya Kawahara¹

¹ Graduate School of Informatics, Kyoto University, Kyoto, Japan

{zhou}@sap.ist.i.kyoto-u.ac.jp

Abstract

Supervised fine-tuning (SFT) is widely used to adapt self-supervised speech representations to downstream classification tasks. Small gains observed under a single pretrained checkpoint are often interpreted as method-level improvements, i.e., a higher attainable performance ceiling. We show that such conclusions are not always reliable because SFT outcomes depend strongly on the specific pretrained instance. We conduct a systematic study on three SUPERB classification tasks, evaluating eight SFT variants across nine pretrained checkpoints from wav2vec 2.0, HuBERT, and WavLM, with multi-seed repetitions on representative base-scale models. We find that the identity of the statistically indistinguishable top-group SFT recipe is often checkpoint-dependent, with limited transferability across pretrained instances. These findings suggest that many reported downstream gains reflect instance and seed dependent *elicitation match*, rather than universally improving the attainable performance ceiling.

Index Terms: self-supervised learning, supervised fine-tuning, speech classification

1. Introduction

Supervised fine-tuning (SFT) is a dominant paradigm for adapting self-supervised pretrained speech models [1, 2, 3] to downstream classification tasks [4, 5, 6]. In common practice, researchers compare SFT variants by downstream performance and often treat small gains under a single pretrained checkpoint as evidence that a recipe is “better,” i.e., achieving a higher attainable performance ceiling [7, 8].

This practice implicitly assumes that the relative effectiveness of SFT methods is stable across pretrained model instances. In other words, a recipe that is “better” under one pretrained checkpoint is expected to remain competitive under other checkpoints of comparable capability [9, 10]. However, this assumption overlooks an important interaction in the adaptation process. Downstream performance is jointly shaped by the backbone architecture, the pretraining data, and the SFT recipe [11]. Yet existing evaluations often consider these factors in isolation. In particular, the interaction between a recipe and a specific pretrained instance is rarely examined [12, 13]. As a result, even when the architecture and the model size are held fixed, the apparent superiority of an SFT method may depend on which pretrained checkpoint it is applied to.

We propose to view SFT as a process of *capacity elicitation*: fine-tuning attempts to activate and reorganize latent knowledge already present in pretrained representations. Under this view, apparent SFT superiority can reflect *elicitation match*, meaning that a recipe more reliably activates the checkpoint’s usable capacity rather than increasing the capacity it-

self. Different SFT configurations impose different elicitation biases, which may match some pretrained instances better than others, leading to differences in activation reliability. Consequently, two SFT methods can exhibit ranking discrepancies across checkpoints: Method A may outperform Method B on one pretrained instance, while Method B surpasses Method A on another. Such discrepancies challenge the external validity of conclusions drawn from single-checkpoint comparisons.

To study this effect, we conduct a large-scale empirical investigation of SFT behavior in downstream speech classification. We evaluate eight SFT configurations along two axes: which intermediate representations are used for supervision, and how much of the pretrained network is frozen during adaptation. These configurations are applied to nine widely used self-supervised checkpoints from wav2vec 2.0, HuBERT, and WavLM. Experiments are conducted on three SUPERB classification tasks [14]: intent classification (IC) [15, 16], emotion recognition (ER) [17, 18], and speaker identification (SID) [19, 20]. To characterize run-to-run variability, we further repeat experiments with multiple random seeds on representative base-scale checkpoints.

Our results show that SFT comparisons can depend on the specific pretrained instance. Across tasks, switching checkpoints can change the relative ordering of SFT methods, including cases where architecture and model size are identical and only the pretraining data differs. Multi-seed repetitions further reveal bidirectional transitions between fully activated and under-activated outcomes for the same checkpoint and configuration, which can both expose and mask row-wise top-group differences. Overall, these findings suggest that reported improvements from SFT variants should be interpreted with caution: observed gains may reflect instance- and seed-dependent activation behavior, rather than a universally higher attainable performance ceiling.

This work makes two contributions.

- We provide systematic empirical evidence that the identity of the best-performing (or top-group) SFT configuration can vary across pretrained instances, even when the backbone architecture and the model size are fixed.
- We introduce the perspective of *elicitation match* to interpret apparent SFT superiority as differences in activation reliability across pretrained instances, explaining why top-group results on a single checkpoint may not transfer consistently.

Table 1: Pretrained SSL checkpoints used in our study, with model size and pretraining data.

Checkpoint	Family	#Params (M)	Pretraining data
facebook/wav2vec2-base-960h	wav2vec 2.0	95.00	LibriSpeech 960h
facebook/wav2vec2-base-100k-voxpopuli	wav2vec 2.0	95.00	VoxPopuli (100k unlabeled subset)
facebook/wav2vec2-large-960h	wav2vec 2.0	317.00	LibriSpeech 960h
facebook/wav2vec2-large-100k-voxpopuli	wav2vec 2.0	317.00	VoxPopuli (100k unlabeled subset)
facebook/hubert-base-ls960	HuBERT	95.00	LibriSpeech 960h
facebook/hubert-large-ll60k	HuBERT	317.00	Libri-Light 60k
microsoft/wavlm-base	WavLM	94.70	LibriSpeech 960h
microsoft/wavlm-base-plus	WavLM	94.70	Libri-Light 60k + GigaSpeech 10k + VoxPopuli 24k
microsoft/wavlm-large	WavLM	316.62	Libri-Light 60k + GigaSpeech 10k + VoxPopuli 24k

2. Problem Setup

2.1. Problem Statement

We study *pretraining-dependent instability* in choosing the top SFT configuration for downstream classification: swapping the pretrained SSL checkpoint (within the same backbone family and scale) can change which SFT configuration is best, or statistically indistinguishable from the best, with all other settings fixed.

2.2. Formal Setup

Let M be a pretrained SSL checkpoint, T a downstream task, and s a random seed. Each SFT configuration $a \in \mathcal{A}$ yields

$$y = f(a, M, T, s), \quad (1)$$

where y is the task metric reported by the evaluator.

2.3. SFT Configuration Space

We parameterize configurations by `FEATURE_MODE` $m_f \in \{1, 2, 3\}$ and `FREEZE_MODE` $m_z \in \{0, 1, 2\}$, denoted $a = (m_f, m_z)$. We evaluate

$$\mathcal{A} = \{(1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2), (3, 0), (3, 1)\}.$$

2.4. Top-Group Instability Across Checkpoints

For fixed (M, T, s) , let

$$\begin{aligned} y^*(M, T, s) &= \max_{a \in \mathcal{A}} f(a, M, T, s), \\ a^* &= \arg \max_{a \in \mathcal{A}} f(a, M, T, s). \end{aligned} \quad (2)$$

Define the *top group* as configurations not significantly worse than a^* under the paired exact McNemar test at level α :

$$\mathcal{G}(M, T, s; \alpha) = \{a \in \mathcal{A} : a \not\prec a^* \text{ at level } \alpha\}. \quad (3)$$

We can suspect *top-group instability* occurs between checkpoints M and M' if

$$\mathcal{G}(M, T, s; \alpha) \neq \mathcal{G}(M', T, s; \alpha). \quad (4)$$

3. Experiments

3.1. Datasets

We evaluate downstream speech classification on three tasks from the SUPERB benchmark: intent classification (IC), emotion recognition (ER), and speaker identification (SID). We follow the official SUPERB data splits and evaluation protocols for each task. Unless otherwise stated, we use the default evaluation metrics specified by SUPERB for the corresponding task.

3.2. Backbones

We consider nine widely used self-supervised speech checkpoints spanning three model families: wav2vec 2.0, HuBERT, and WavLM. Table 1 summarizes each checkpoint, including parameter count and pretraining data. The table includes same-architecture checkpoints pretrained on different data to enable controlled analysis of pretraining-instance dependence.

3.3. Configurations

Eight SFT configurations are defined by `FEATURE_MODE` $\in \{1, 2, 3\}$ and `FREEZE_MODE` $\in \{0, 1, 2\}$. Each configuration is denoted as Fm_f-Zm_z (e.g., $F1-Z2$ denotes `FEATURE_MODE=1` and `FREEZE_MODE=2`).

Feature modes. `FEATURE_MODE=1` uses the final-layer `last_hidden_state` as the supervised representation. `FEATURE_MODE=2` uses a single selected intermediate layer; we fix the selected layer to the fourth layer from the end, i.e., $\ell^* = L - 3$. `FEATURE_MODE=3` uses a weighted-sum fusion over transformer layers as the supervised representation.

Freeze modes. `FREEZE_MODE=0` fine-tunes the entire pre-trained network together with the task head. `FREEZE_MODE=1` freezes only the convolutional front-end and fine-tunes all transformer layers plus the task head. `FREEZE_MODE=2` freezes the convolutional front-end and the first N transformer layers, fine-tuning the rest and the task head ($N = 4$ in all runs).

3.4. Training Setup

Our training recipe follows the official SUPERB downstream configurations for IC, ER, and SID as closely as possible. All downstream fine-tuning and evaluation use the official SUPERB training pipeline and built-in evaluator, with identical scripts across runs; only the pretrained checkpoint and random seed are varied. For each task, we keep the entire pipeline fixed across all SSL checkpoints and SFT configurations, including data preprocessing, batching, and validation-based checkpoint selection. Task-specific hyperparameters (optimizer, learning-rate schedule, batch size, and training duration) are taken directly from the corresponding SUPERB configuration files and held constant within each task. Each experiment is run on a single NVIDIA H20 GPU, totaling approximately 10,000 GPU-hours across all runs.

3.5. Seed Protocol

We use a two-level seed design. For the full experimental matrix covering all nine SSL checkpoints, we run each (checkpoint, task, SFT configuration) setting with a fixed default seed of 1337. To characterize run-level stochasticity, we additionally perform multi-seed repetitions on three rep-

Table 2: Test accuracy computed from paired prediction and truth files for all experiments with seed 1337. Values are bold when they are not significantly different from the row maximum under a two-sided exact McNemar test at 95% confidence ($\alpha=0.05$); small values are marked as convergence anomalies with an asterisk (*).

Task	Model	F1-Z0	F1-Z1	F1-Z2	F2-Z0	F2-Z1	F2-Z2	F3-Z0	F3-Z1
SID	wav2vec2-base-960h	0.6590	0.7057	0.7833	0.0006*	0.0006*	0.0006*	0.6590	0.7036
	wav2vec2-base-100k-voxpupuli	0.6343	0.6769	0.7650	0.0006*	0.0006*	0.0006*	0.6343	0.6769
	wav2vec2-large-960h	0.4156	0.7229	0.8294	0.0006*	0.0006*	0.0006*	0.4156	0.7229
	wav2vec2-large-100k-voxpupuli	0.9066	0.9058	0.9015	0.8980	0.8811	0.9051	0.8981	0.9058
	hubert-base-ls960	0.6834	0.7335	0.8353	0.0006*	0.8753	0.0006*	0.6834	0.7294
	hubert-large-ll60k	0.9173	0.9387	0.9386	0.9034	0.9335	0.9167	0.9173	0.9387
	wavlm-base	0.1139*	0.1977*	0.5027	0.7237	0.7148	0.0006*	0.1190*	0.2462*
	wavlm-base-plus	0.0934*	0.1645*	0.5867	0.0006*	0.6376	0.0006*	0.0851*	0.1857*
	wavlm-large	0.9100	0.9161	0.9230	0.9018	0.8821	0.9189	0.9213	0.9114
IC	wav2vec2-base-960h	0.9902	0.9939	0.9963	0.9950	0.9955	0.1044*	0.9902	0.9939
	wav2vec2-base-100k-voxpupuli	0.9818	0.9913	0.9947	0.1044*	0.9942	0.9924	0.9818	0.9913
	wav2vec2-large-960h	0.9881	0.9934	0.9950	0.1044*	0.0910*	0.9842	0.9881	0.9934
	wav2vec2-large-100k-voxpupuli	0.9968	0.9955	0.9953	0.9966	0.9963	0.9945	0.9968	0.9955
	hubert-base-ls960	0.9831	0.9902	0.9958	0.9953	0.9960	0.1044*	0.9831	0.9902
	hubert-large-ll60k	0.9955	0.9958	0.9971	0.9947	0.9974	0.9953	0.9955	0.9958
	wavlm-base	0.9818	0.9892	0.9963	0.9958	0.9963	0.1044*	0.9868	0.9900
	wavlm-base-plus	0.9900	0.9929	0.9955	0.9966	0.9966	0.9852	0.9908	0.9908
	wavlm-large	0.9958	0.9974	0.9963	0.9963	0.9960	0.9953	0.9963	0.9939
ER	wav2vec2-base-960h	0.5539	0.6203	0.7005	0.6747	0.7060	0.4673	0.5539	0.6203
	wav2vec2-base-100k-voxpupuli	0.5687	0.6516	0.6931	0.6562	0.6691	0.6111	0.5687	0.6516
	wav2vec2-large-960h	0.5548	0.6258	0.6369	0.3539*	0.3539*	0.3539*	0.5548	0.6258
	wav2vec2-large-100k-voxpupuli	0.6811	0.6719	0.7244	0.6995	0.7253	0.7032	0.6811	0.6719
	hubert-base-ls960	0.6608	0.6525	0.6765	0.6581	0.7032	0.6747	0.6608	0.6525
	hubert-large-ll60k	0.7134	0.7235	0.7272	0.7161	0.7115	0.7244	0.7134	0.7235
	wavlm-base	0.5410	0.6018	0.6488	0.6664	0.6581	0.6101	0.5641	0.6065
	wavlm-base-plus	0.4922	0.5410	0.6535	0.6406	0.6912	0.6359	0.6018	0.5843
	wavlm-large	0.7355	0.7272	0.7014	0.7770	0.7493	0.7521	0.7456	0.7456

representative base-scale checkpoints: facebook/wav2vec2-base-960h, facebook/hubert-base-ls960, and microsoft/wavlm-base. For these three checkpoints, we repeat every (task, SFT configuration) setting with two extra seeds, 2048 and 7395, resulting in three seeds in total (1337, 2048, 7395) for each of the corresponding runs.

3.6. Evaluation Outputs and Analysis Targets

For each run, we report the task-defined metric using the official SUPERB evaluation scripts. The experimental matrix produces scores indexed by (checkpoint, task, SFT configuration, seed). All result aggregation, top-group identification via paired exact McNemar tests, and checkpoint- and seed-sensitivity analyses (including under-activation and pairwise sign flips) are presented in Section 4.

4. Results

Each row in Tables 2 and 3 corresponds to a fixed downstream task and a fixed pretrained checkpoint, under which we compare the eight SFT configurations in Section 3. We mark systems that are not significantly worse than the row-wise best as the *top group*, i.e., statistically indistinguishable from the best under the paired exact McNemar test [21] at $\alpha = 0.05$.

4.1. Why the differences are not ceiling improvements

At a glance, bold entries are not uniformly distributed across columns, and some SFT configurations enter the top group more frequently across pretrained checkpoints. A naive interpretation is that these configurations have an intrinsically higher attainable performance ceiling. However, several observations are inconsistent with a ceiling-improvement explanation.

First, configurations that frequently enter the top group can still exhibit *under-activation* on certain checkpoints, producing abnormally low performance on the same task. If the advantage primarily came from a systematically higher ceiling, one would expect the relative ordering to be more consistent and the “best” configurations to rarely collapse. Instead, extreme failures among apparent winners indicate that the observed gaps are not simply explained by different upper bounds.

Second, Table 3 shows that the effect is, to some extent, seed-dependent and *bidirectional*. For the same task, pretrained checkpoint, and SFT configuration, changing only the seed can occasionally turn an under-activated run into a fully activated one that reaches the row-wise top group statistically indistinguishable from the best. Conversely, a configuration that is in the top group under one seed can become under-activated (or markedly degraded) under another seed, without changing data, hyperparameters, or training mode. Such bidirectional transitions are difficult to reconcile with a stable ceiling-improvement story, but are consistent with sensitivity to the optimization tra-

Table 3: Test accuracy computed from paired prediction and truth files for experiments with seeds 2048 and 7395, plus seed 1337 for hubert-base-ls960, wav2vec2-base-960h, and wavlm-base, ordered to match the seed-1337 task and model layout. Values are bold when they are not significantly different from the row maximum under a two-sided exact McNemar test at 95% confidence ($\alpha=0.05$); small values are marked as convergence anomalies with an asterisk (*).

Task	Model	Seed	F1-Z0	F1-Z1	F1-Z2	F2-Z0	F2-Z1	F2-Z2	F3-Z0	F3-Z1
SID	wav2vec2-base-960h	1337	0.6590	0.7057	0.7833	0.0006*	0.0006*	0.0006*	0.6590	0.7036
		2048	0.6839	0.7292	0.7838	0.0006*	0.6236	0.0006*	0.6839	0.7427
		7395	0.6722	0.7061	0.7738	0.0006*	0.0006*	0.0006*	0.6722	0.7061
	hubert-base-ls960	1337	0.6834	0.7335	0.8353	0.0006*	0.8753	0.0006*	0.6834	0.7294
		2048	0.6759	0.6947	0.8313	0.0006*	0.8670	0.0007*	0.6759	0.6947
		7395	0.6798	0.7237	0.8474	0.8302	0.8738	0.0006*	0.6798	0.7237
	wavlm-base	1337	0.1139*	0.1977*	0.5027	0.7237	0.7148	0.0006*	0.1190*	0.2462*
		2048	0.1608*	0.2880*	0.4585	0.0006*	0.6931	0.0006*	0.0699*	0.2624*
		7395	0.1345*	0.2082*	0.4620	0.7101	0.6560	0.0006*	0.2053*	0.2718*
IC	wav2vec2-base-960h	1337	0.9902	0.9939	0.9963	0.9950	0.9955	0.1044*	0.9902	0.9939
		2048	0.9852	0.9937	0.9955	0.9955	0.9955	0.1044*	0.9852	0.9937
		7395	0.9884	0.9937	0.9955	0.1044*	0.9958	0.9945	0.9884	0.9937
	hubert-base-ls960	1337	0.9831	0.9902	0.9958	0.9953	0.9960	0.1044*	0.9831	0.9902
		2048	0.9871	0.9931	0.9960	0.0464*	0.9960	0.1044*	0.9871	0.9931
		7395	0.9831	0.9934	0.9963	0.0464*	0.9958	0.0464*	0.9831	0.9934
	wavlm-base	1337	0.9818	0.9892	0.9963	0.9958	0.9963	0.1044*	0.9868	0.9900
		2048	0.9839	0.9821	0.9960	0.9924	0.9955	0.1044*	0.9829	0.9871
		7395	0.9855	0.9900	0.9958	0.0000*	0.9953	0.1044*	0.9831	0.9889
ER	wav2vec2-base-960h	1337	0.5539	0.6203	0.7005	0.6747	0.7060	0.4673	0.5539	0.6203
		2048	0.5585	0.6175	0.6820	0.3548*	0.6940	0.2562*	0.5585	0.6175
		7395	0.5585	0.6424	0.6949	0.6590	0.7134	0.6175	0.5585	0.6424
	hubert-base-ls960	1337	0.6608	0.6525	0.6765	0.6581	0.7032	0.6747	0.6608	0.6525
		2048	0.6350	0.6654	0.6562	0.3539*	0.6728	0.6654	0.6350	0.6654
		7395	0.6258	0.6488	0.6857	0.6866	0.6793	0.6719	0.6258	0.6488
	wavlm-base	1337	0.5410	0.6018	0.6488	0.6664	0.6581	0.6101	0.5641	0.6065
		2048	0.5558	0.5991	0.6092	0.6719	0.7060	0.4931	0.4986	0.5364
		7395	0.6212	0.6055	0.6636	0.6645	0.7217	0.6194	0.5078	0.6138

jectory.

Finally, for the hubert-large-1160k checkpoint on the ER task, all eight configurations achieve statistically indistinguishable performance, placing the entire row in the top group. This limiting case indicates that, for some pretrained instances under our protocol, the choice among these SFT configurations has limited marginal effect on the final metric, further weakening the claim that certain configurations provide a generally higher ceiling.

4.2. Why the differences reflect activation matching

Taken together, the results are better explained by an *activation matching* perspective: different SFT configurations differ primarily in how reliably they can *activate* the latent capability of a given pretrained checkpoint under a particular training stochasticity, rather than in how high they can ultimately go.

The non-uniform bold-entry distribution then reflects differences in *activation success rate*. A configuration that appears in the top group more often is one that more frequently reaches a high-performing solution region for many checkpoints under the default recipe. Under-activation corresponds to a failure to enter that region for a specific checkpoint-seed combination, even for configurations that otherwise succeed often.

The seed results provide direct support for this view. The fact that activation can be flipped from failure to success (and vice versa) by changing only the seed suggests that the main

variability lies in whether optimization finds a suitable basin that aligns the SFT configuration with the pretrained representation. When activation succeeds, multiple configurations can reach comparable top-group performance for the same checkpoint, implying that the primary distinction is *how easy it is to get there* rather than a systematically higher attainable ceiling.

Notably, the abnormally low scores marked as convergence anomalies (*) do not stem from accidental training interruptions: all runs complete the full training pipeline under the same configuration. This further confirms that poor performance primarily reflects a failure of the SFT procedure to successfully activate the pretrained model’s usable capacity.

5. Conclusion

Across nine SSL checkpoints and eight SFT configurations, we show that selecting the statistically indistinguishable top-group SFT recipe is pretrained-instance dependent. Even with fixed architecture and scale, the best-performing configuration can change across checkpoints, and pairwise orderings may flip as a secondary effect. Multi-seed results further reveal this fragile activation, where the same recipe can alternate between fully activated and under-activated outcomes due to optimization stochasticity. Overall, apparent SFT gains often reflect instance and seed chdependent activation reliability rather than universally better recipes, motivating evaluation across multiple checkpoints and seeds.

6. Acknowledgments

This work was supported by JST BOOST, Grant Number JP-MJBS2407.

7. Generative AI Use Disclosure

Generative AI tools (Gemini and ChatGPT) were used for language editing and improving the phrasing of this manuscript.

8. References

- [1] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.
- [2] 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,” *IEEE/ACM transactions on audio, speech, and language processing*, vol. 29, pp. 3451–3460, 2021.
- [3] 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 Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.
- [4] Y. Gao, C. Chu, and T. Kawahara, “Two-stage finetuning of wav2vec 2.0 for speech emotion recognition with asr and gender pretraining,” in *Proc. Interspeech*, 2023, pp. 3637–3641.
- [5] Y. Gao, H. Shi, C. Chu, and T. Kawahara, “Enhancing two-stage finetuning for speech emotion recognition using adapters,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 11 316–11 320.
- [6] W. Zhou, Z. Yang, C. Chu, S. Li, R. Dabre, Y. Zhao, and K. Tatsuya, “Mos-fad: Improving fake audio detection via automatic mean opinion score prediction,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 876–880.
- [7] X. Shi, Y. Gao, J. He, J. Mi, X. Li, and T. Toda, “A study on multimodal fusion and layer adapter in emotion recognition,” in *Proc. Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2024, pp. 1–6.
- [8] J. Zhao, H. Shi, C. Cui, T. Wang, H. Liu, Z. Ni, L. Ye, and L. Wang, “Adapting whisper for code-switching through encoding refining and language-aware decoding,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.
- [9] X. Shi, J. He, X. Li, and T. Toda, “A comprehensive study on the effectiveness of asr representations for noise-robust speech emotion recognition,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 34, pp. 707–722, 2026.
- [10] M. Sang and J. H. Hansen, “Efficient adapter tuning of pre-trained speech models for automatic speaker verification,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 131–12 135.
- [11] Y. Zhang, Y. Gao, W. Zhou, Z. Yuan, K. Imoto, and T. Kawahara, “SONAR: Self-distilled continual pre-training for domain adaptive audio representation,” *ArXiv*, vol. abs/2509.15703, 2025.
- [12] A. Merchant, E. Rahimtooghi, E. Pavlick, and I. Tenney, “What happens to BERT embeddings during fine-tuning?” in *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, A. Alishahi, Y. Belinkov, G. Chrupala, D. Hupkes, Y. Pinter, and H. Sajjad, Eds. Association for Computational Linguistics, Nov. 2020, pp. 33–44.
- [13] C. Zhang, G. Neubig, and X. Yue, “On the interplay of pre-training, mid-training, and rl on reasoning language models,” *ArXiv*, vol. abs/2512.07783, 2025.
- [14] 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,” *Interspeech*, 2021.
- [15] Y. Tian and P. J. Gorinski, “Improving end-to-end speech-to-intent classification with reptile,” in *Proc. Interspeech*, 2020.
- [16] B. Sharma, M. Madhavi, and H. Li, “Leveraging acoustic and linguistic embeddings from pretrained speech and language models for intent classification,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 7498–7502.
- [17] Y. Gao, H. Shi, Y. Fu, C. Chu, and T. Kawahara, “Bridging Speech Emotion Recognition and Personality: Dataset and Temporal Interaction Condition Network,” *IEEE Transactions on Affective Computing*, vol. 17, no. 01, pp. 829–840, Jan. 2026.
- [18] E. Morais, R. Hoory, W. Zhu, I. Gat, M. Damasceno, and H. Aronowitz, “Speech emotion recognition using self-supervised features,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6922–6926.
- [19] B. Han, Z. Chen, and Y. Qian, “Exploring binary classification loss for speaker verification,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.
- [20] Z. Chen, S. Wu, X. Li, Z. Ai, and S. Xu, “Open-set speaker identification through efficient few-shot tuning with speaker reciprocal points and unknown samples,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 33, pp. 3347–3362, 2025.
- [21] Q. McNemar, “Note on the sampling error of the difference between correlated proportions or percentages,” *Psychometrika*, vol. 12, no. 2, p. 153–157, 1947.