SCORE: SELF-SUPERVISED CORRESPONDENCE FINE-TUNING FOR IMPROVED CONTENT REPRESENTATIONS

Amit Meghanani, Thomas Hain

Speech and Hearing Research Group
Department of Computer Science, The University of Sheffield, United Kingdom
{ameghanani1,t.hain}@sheffield.ac.uk

ABSTRACT

There is a growing interest in cost-effective self-supervised fine-tuning (SSFT) of self-supervised learning (SSL)-based speech models to obtain task-specific representations. These task-specific representations are used for robust performance on various downstream tasks by fine-tuning on the labelled data. This work presents a cost-effective SSFT method named **S**elf-supervised **Corre**spondence (SCORE) fine-tuning to adapt the SSL speech representations for content-related tasks. The proposed method uses a correspondence training strategy, aiming to learn similar representations from perturbed speech and original speech. Commonly used data augmentation techniques for content-related tasks (ASR) are applied to obtain perturbed speech. SCORE fine-tuned HuBERT outperforms the vanilla HuBERT on SUPERB benchmark with only a few hours of fine-tuning (< 5 hrs) on a single GPU for automatic speech recognition, phoneme recognition, and query-by-example tasks, with relative improvements of 1.09%, 3.58%, and 12.65%, respectively. SCORE provides competitive results with the recently proposed SSFT method SPIN, using only 1/3 of the processed speech compared to SPIN.

***Index Terms*—** Self-supervised learning, Self-supervised fine-tuning, Correspondence training

1. INTRODUCTION

Self-supervised learning (SSL) based pre-trained speech models such as HuBERT [1], WavLM [2] are becoming popular for their state-of-the-art performance on almost all speech applications. These models extract latent features that capture underlying factors of speech, such as acoustic-phonetic information, speaker information, semantic information, and more [3]. These pre-trained representations are then fine-tuned for downstream application with labelled data. However, pre-trained SSL speech models may not be ideal for downstream tasks that do not align with the pre-trained objective (for example, handling overlapping speech [2]). One way to overcome this issue is to introduce a pre-training objective that relates to the downstream task, such as training with overlapping speech in WavLM [2]. However, this approach requires substantial amount of compute cost as the model is pre-trained from scratch. Another alternative falls within the realm of unsupervised or self-supervised fine-tuning (SSFT)[4]. SSFT is applied on top of pre-trained models to learn task-specific representations. Then the SSL models are fine-tuned with labelled data on the downstream tasks for robust performance. For example, ContentVec [5] employs content preserving strategies (by disentangling speakers) on top of pre-trained HuBERT model to learn content-specific representations. However, ContentVec is not very cost-effective as it requires 19 hrs on 36 GPUs on top of the pre-trained HuBERT [1] model. Another recent SSFT approach for content-related downstream task is speaker-invariant clustering (SPIN) [4], which requires a compute cost less than 1% of ContentVec. SPIN employs speaker invariant clustering to improve content representations. The term SSFT was proposed in [4] to distinguish fine-tuning methods using only audio [5, 6] from supervised fine-tuning using labelled data [7].

In this work, a simple and cost-effective SSFT method named **S**elf-supervised **Corre**spondence (SCORE) fine-tuning is proposed to preserve content. Correspondence training [8] is the task of learning similar representations from two different instances of the same spoken content. This technique has been successfully applied to extract high quality acoustic word embeddings (AWEs), where an auto-encoder takes input as a spoken word and the target output as the same word spoken by a different speaker [8, 9]. This technique ensures that the encoder learns only content and forget other unnecessary information such as speaker, duration, prosody, etc. Taking inspiration from this, for SCORE fine-tuning, a perturbed speech is generated from the original speech in such a way that the spoken content is preserved. Perturbed speech utterances are generated through the application of commonly employed data augmentation techniques

This work was supported by the Centre for Doctoral Training in Speech and Language Technologies (SLT) and their Applications funded by UK Research and Innovation [grant number EP/S023062/1]. This work was also funded in part by LivePerson, Inc.

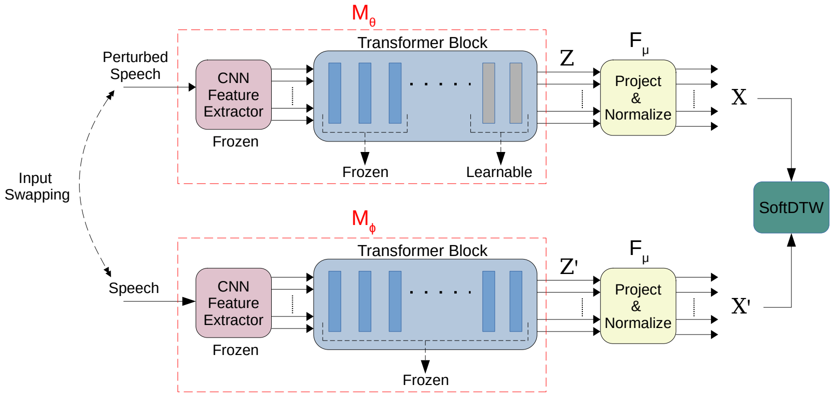

Fig. 1. SCORE fine-tuning method. SCORE takes a pair of original speech and perturbed speech as input. It then matches the output sequence X from the learnable model M_θ against the output sequence X' from the frozen model M_ϕ using soft-DTW loss.

in automatic speech recognition, such as speed perturbation [10] and pitch shifting. After obtaining perturbed speech, the objective is to learn similar speech representations from both the original speech and perturbed speech, making the representations pitch and duration invariant. Shifting pitch (fundamental frequency) alters the speaker information while keeping the content same. To match the representations from perturbed and original speech, soft-DTW [11, 12] is used as a loss function. Soft-DTW is a popular loss function for time-series data, and it has also been successfully used for the multi-pitch estimation task in music information retrieval [13]. The proposed method is tested on three content-related downstream tasks on SUPERB benchmark [14]: automatic speech recognition (ASR), phoneme recognition (PR), and query-by-example spoken term discovery (QbE). The results are compared against the performance of vanilla models (HuBERT and WavLM) and recently proposed content-preserving SSFT methods such as ContentVec and SPIN for SUPERB benchmark .

The main contributions of this work are as follows:

- A novel cost-effective self-supervised fine-tuning method named SCORE is proposed to improve the content representations.
- With just less than 5 hours of SCORE fine-tuning on a single V100 GPU, SCORE fine-tuned models outperform vanilla HuBERT and WavLM on the SUPERB benchmark for content-related tasks.

2. METHODOLOGY

Fig. 1 demonstrates the proposed SCORE fine-tuning method. SCORE involves two instances of the pre-trained model, one with frozen parameters (M_ϕ) and the other with learnable top layers (M_θ), both having same initial model weights. Top layers are chosen for fine-tuning, as they encode phonetic content for most of the SSL models [15, 16]. More implementation details are described in Sec. 3. The input to the SCORE is a pair of perturbed speech and original speech, randomly fed to either M_θ or M_ϕ, as shown in Fig. 1. Randomizing the input ensures that the model M_θ does not exclusively focus on the characteristics of perturbed speech, and found to be crucial to observe the benefits of the proposed method. To obtain the perturbed speech, data augmentations used in ASR [10] are employed, such as speech perturbation and pitch shift. Torchaudio [17] is used for these perturbations, with `SpeedPerturbation` and `PitchShift` functions under `torchaudio.transforms`[1]. The obtained representations from both the models M_θ and M_ϕ are projected to a lower dimension with linear feedforward layers and L2-normalized. Obtained sequences from both models (X and X') are different in lengths due to the perturbations. Therefore, a dynamic time warping based differentiable loss function soft-Dynamic Time Warping (soft-DTW) [2][11, 18, 19] is used to match sequences of unequal lengths (Eq. 1). This learning framework ensures that the model learns the speed and pitch invariant representations for the same spoken content. The soft-DTW replaces the "min" operation in the DTW with "soft-min" operation. Soft-DTW computes the soft-minimum of all alignment cost. The soft-DTW for two sequences $X = x_1, x_2, ...x_m$ and $X' = x'_1, x'_2, ..., x'_n$ is defined [19] as follows:

$$\text{soft-DTW}_\gamma(X, X') = \min_{\pi \in A(X,X')}{}^{\gamma} \sum_{(i,j)\in\pi} d(x_i, x'_j)^2 \qquad (1)$$

where $A(X, X')$ is the set of all possible paths. The $\min^\gamma$ is the soft-min operator with a smoothing factor γ and d is the distance function. The soft-min operator $\min^\gamma$ is defined as:

[1] `https://pytorch.org/audio/stable/transforms.html`

[2] `https://github.com/Maghoumi/pytorch-softdtw-cuda`

Algorithm 1 SCORE Fine-tuning

```
$M_\theta$ = Learnable model (Top 2 layers only)
$M_\phi$ = Frozen model
$M_\theta$, $M_\phi$ are initialized with the same SSL model weights.
$F_\mu$ = Linear projection layers + L-2 Normalisation
$S_i$ = i$^{\text{th}}$ speech utterance
while Not Converged do
    for i=1 to $N_{samp}$ do
        ${S_i}^p = SpeedPerturbation(S_i)$
        ${S_i}^p = PitchShift({S_i}^p)$
        k = random(0,1)
        if $k == 0$ then
            $Z = M_\theta({S_i}^p), Z' = M_\phi(S_i)$
        else
            $Z = M_\theta(S_i), Z' = M_\phi({S_i}^p)$
        $X = F_\mu(Z), X' = F_\mu(Z')$
        Compute Loss $L_{norm}(X, X')$.
        Compute Gradients $\frac{\partial L_{norm}}{\partial \theta}, \frac{\partial L_{norm}}{\partial \mu}$
        Update $\theta$ and $\mu$ to minimize $L_{norm}$.
```

$$\min{}^{\gamma}(a_1, \ldots, a_n) = -\gamma \log \sum_i e^{-a_i/\gamma} \tag{2}$$

In this work, soft-DTW$_\gamma$ is used as a loss function for SCORE fine-tuning as described in Eq. 3.

$$L(X, X') = \text{soft-DTW}_\gamma(X, X') \tag{3}$$

In all the experiments, we use a smoothing factor γ of 0.1. However, to address potential negative values in soft-DTW_γ loss, a normalized version described in Eq. 4 is employed. This normalization guarantees a minimum loss value of zero for identical sequences, i.e., $L_{norm}(X, X) = 0$, and ensures $L_{norm}(X, X') \geq 0$ for any pair of sequences. This approach guarantees a consistently positive loss [20, 21]. Further, the loss from Eq. 4 is normalized by dividing it with the total sequence length $m + n$. Algorithm 1 describes the entire SCORE fine-tuning method.

$$L_{norm}(X, X') = L(X, X') - \frac{1}{2}(L(X, X) + L(X', X')) \tag{4}$$

3. EXPERIMENTS

Experiments are conducted on two SSL speech models: HuBERT and WavLM (BASE versions). These SSL models are fine-tuned with the SCORE method. After the SCORE fine-tuning, obtained models are used for supervised training for the content-related downstream tasks on the SUPERB benchmark. Similar to SPIN [4], the top 2 layers (11th and 12th) of the SSL models are fine-tuned as it is cost-effective ($\approx$ 14M trainable parameters) and most of the SSL models encode phonetic content in top layers [15, 16]. In this study, Wav2vec2 [7] is omitted due to the fact that the linguistic content is less well represented in the final few layers [16], which is crucial for content-related tasks. Fine-tuning the entire model, from bottom layers to top layers, would result in increased computational expenses, contradicting the study's intended objectives. Furthermore, there is a concern that when the entire model is fine-tuned, the fine-tuning objective could potentially lead to a collapse of the original representations [22] learned during pre-training. The details about the data, SCORE fine-tuning, and evaluation on the SUPERB benchmark are described as following:

Data: In line with prior research [4] and to ensure a fair comparison, experiments are performed on LibriSpeech's [23] train-clean-100 hours of data for SCORE fine-tuning. Consistent with earlier discoveries [4], training more layers or additional data does not enhance results.

SCORE Fine-tuning Details: The representations obtained from the final Transformer layer (12th) of the models M_θ and M_ϕ are sequences of 768-dimensional vectors. These vectors are projected into 256-dimensional vectors with linear projection layers and then L2-normalized. The SCORE fine-tuning method is trained for 3.6k updates ($\approx$ 1 epoch with effective batch size of 8). The model converged in just one epoch, and additional training did not yield any improvements. AdamW [24] optimizer is used with a learning rate of $2.0e-5$ with 1k warm-up updates. One epoch roughly takes < 5 hours on V100 GPU. More details are available at GitHub[3].

SUPERB Benchmark: S3PRL toolkit [4] is used for all the SUPERB benchmark tasks. For ASR and PR, features from all the layers are aggregated with learnable weights. These aggregated features are then fed to the prediction head for each downstream task and fine-tuned with labelled data. For ASR, the prediction head consists of 2-layer 1024-unit Bi-LSTM network with CTC loss on characters [14]. The ASR model is evaluated without any external language model. For PR, the prediction head is a frame-wise linear transformation with CTC loss. More details can be found at SUPERB benchmark [14]. Adam optimizer is used for both ASR and PR with learning rate of $1.0e-4$ and $5.0e-4$, respectively. We conducted experiments for each ASR and PR model five times and have provided the results, including the means and standard deviations, for both the vanilla models (HuBERT and WavLM) and their SCORE fine-tuned versions. For QbE, conventional supervised phoneme posteriorgram are replaced with SSL representations [14]. For QbE, no training is required, and the evaluation is performed by running DTW on all layers separately and obtain a score for each query-document pair. For the evaluation on test set, the best layer is selected based on performance on dev set from QUESST 2014 [25] data. In our case, we found that 12th layer provides best results for QbE for both HuBERT + SCORE and WavLM + SCORE.

[3] https://github.com/Trikaldarshi/SCORE_Finetuning

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

Model	Training Processed Speech (hours)		ASR (WER) ↓	PR (PER) ↓	QbE (MTWV) ↑
	Pre-training	SSFT			
HuBERT [1]☆	506K	0	6.42	5.41	7.36
WavLM [2]☆	1439K	0	6.21	4.84	8.70
ContentVec$_{500}$ [5]☆	506K	76K	5.70	4.54	5.90
HuBERT + SPIN$_{256}$ [4]☆	506K	356	6.34	4.39	9.12
WavLM + SPIN$_{256}$ [4]☆	1439K	356	5.88	4.18	8.79
HuBERT [1]✱	506K	0	6.42 ± 0.08	5.02 ± 0.00	7.19
WavLM [2]✱	1439K	0	6.17 ± 0.02	4.85 ± 0.00	9.15
HuBERT + SCORE	506K	100	6.35 ± 0.07	4.84 ± 0.00	8.10
WavLM + SCORE	1439K	100	6.15 ± 0.04	4.72 ± 0.00	9.22

☆ The reported numbers are from their respective papers and SUPERB benchmark leaderboard [14] as of 13/09/2023 (`https://superbbenchmark.org/leaderboard`).

✱ Our results when we run the SUPERB [14] baseline scripts for HuBERT and WavLM for fair comparison.

Table 1. Results of the proposed SCORE fine-tuning of HuBERT and WavLM models along with baseline methods on SUPERB benchmark. The baseline methods include the BASE version of HuBERT and WavLM models, along with SSFT based ContentVec$_{500}$ and SPIN models. The downstream tasks include ASR, PR, and QbE, which are evaluated on word error rate (WER in %), phoneme error rate (PER in %), and maximum term weighted value (MTWV in %), respectively.

4. RESULTS AND DISCUSSIONS

Table 1 shows the processed speech during training in "pre-training" stage and in "SSFT stage". Processed speech is defined as "training steps × effective batch duration" to quantify machine-independent training costs [4]. HuBERT + SCORE improves the HuBERT model on all three tasks with relative improvement of 1.09%, 3.58%, and 12.65% for ASR, PR, and QbE, respectively. WavLM + SCORE improves the WavLM model on ASR, PR and QbE with relative improvement of 0.32%, 2.68% and 0.76%, respectively. The results are also compared with a stronger baseline ContentVec$_{500}$ [5], which uses 76K hours of processed speech compared to the SCORE which uses only 100 hrs in SSFT stage. ContentVec$_{500}$ provides better results in ASR and PR when compared with SPIN and SCORE at the compute cost of 76K hrs. However, both HuBERT + SCORE and WavLM + SCORE outperform ContentVec$_{500}$ on QbE task. Like SPIN, the goal of this work is to strike a balance between improving the downstream task and the additional training (i.e. SSFT) required. SCORE only needs $<$ 0.5 % of processed speech when compared with ContentVec$_{500}$ in SSFT stage. SCORE provides competitive results with the SPIN models. WavLM + SCORE outperforms WavLM + SPIN$_{256}$ in QbE task. Performance of HuBERT + SCORE is close to HuBERT + SPIN$_{256}$ on ASR. Among all the SSFT method, SCORE uses the least amount of processed speech ($\approx$ 100 hrs) in SSFT stage.

4.1. Layerwise Analysis for Speaker Identification (SID)

One of the data augmentation techniques used in this work is pitch shift, which alters the speaker information. To assess the degradation of speaker information, experiments are conducted for SID task of SUPERB benchmark on VoxCeleb1 [26]. The same configurations are used as provided in the SUPERB [14]. Only the fine-tuned layers (i.e. 11th and 12th) were used for training and evaluating the SID system. The results are presented in Table 2. From Table 2, we can observe a drop in SID accuracy for both HuBERT and WavLM, in both layers. This suggests that the SCORE fine-tuned HuBERT + SCORE and WavLM + SCORE models have representations that are relatively more speaker-invariant than the original models, benefiting content-related tasks.

Model	Layer 11	Layer 12
HuBERT	67.73	64.80
WavLM	52.30	49.16
HuBERT + SCORE	66.70	62.61
WavLM + SCORE	52.00	48.30

Table 2. Layerwise SID accuracy (in %) on SUPERB benchmark for original and SCORE fine-tuned SSL models.

5. CONCLUSION AND FUTURE WORKS

A simple and cost-effective SSFT method named SCORE is proposed to improve content representations of the pre-trained SSL speech models. For both the HuBERT and WavLM models, their respective SCORE fine-tuned models outperformed the original models on the SUPERB benchmark for ASR, PR, and QbE. Compared to other existing approaches of SSFT, SCORE requires the least amount of processed speech (less than 0.5% of processed speech compared to ContentVec$_{500}$). SCORE provides competitive results with SPIN using 1/3 of the processed speech used by SPIN. While we observed relatively fewer improvements in ASR compared to PR and QbE, we speculate that a stronger data augmentation technique directly applicable on speech waveforms could provide better gains. We consider this research direction for our future work.

6. REFERENCES

[1] W. Hsu, B. Bolte, Y. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 29, pp. 3451–3460, oct 2021.

[2] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, "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, July 2022.

[3] A. Mohamed, H. Lee, L. Borgholt, J.D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S. Li, K. Livescu, L. Maaløe, T. N. Sainath, and S. Watanabe, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, pp. 1179–1210, 2022.

[4] H. Chang, A. H. Liu, and J. Glass, "Self-supervised Fine-tuning for Improved Content Representations by Speaker-invariant Clustering," in *Proc. INTERSPEECH 2023*, 2023, pp. 2983–2987.

[5] K. Qian, Y. Zhang, H. Gao, J. Ni, C. Lai, D. Cox, M. Hasegawa-Johnson, and S. Chang, "ContentVec: An improved self-supervised speech representation by disentangling speakers," in *Proceedings of the 39th International Conference on Machine Learning*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, Eds. 17–23 Jul 2022, vol. 162 of *Proceedings of Machine Learning Research*, pp. 18003–18017, PMLR.

[6] K. P. Huang, Y. Fu, Y. Zhang, and H. Lee, "Improving Distortion Robustness of Self-supervised Speech Processing Tasks with Domain Adaptation," in *Proc. Interspeech 2022*, 2022, pp. 2193–2197.

[7] 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.F. Balcan, and H. Lin, Eds. 2020, vol. 33, pp. 12449–12460, Curran Associates, Inc.

[8] H. Kamper, "Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6535–3539, 2019.

[9] A. Meghanani and T. Hain, "Deriving translational acoustic sub-word embeddings," in *Proc. of ASRU*, 2023.

[10] T. Ko, V. Peddinti, Povey D, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. Interspeech 2015*, 2015, pp. 3586–3589.

[11] M Cuturi and M Blondel, "Soft-DTW: a differentiable loss function for time-series," in *Proceedings of the 34th International Conference on Machine Learning*, Doina Precup and Yee Whye Teh, Eds. 06–11 Aug 2017, vol. 70 of *Proceedings of Machine Learning Research*, pp. 894–903, PMLR.

[12] M. Maghoumi, *Deep Recurrent Networks for Gesture Recognition and Synthesis*, Ph.D. thesis, University of Central Florida Orlando, Florida, 2020.

[13] M. Krause, C. Weiß, and M. Müller, "Soft dynamic time warping for multi-pitch estimation and beyond," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[14] S. Yang, P. Chi, Y. Chuang, C. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G. Lin, T. Huang, W. Tseng, K. Lee, D. Liu, Z. Huang, S. Dong, S. Li, S. Watanabe, A. Mohamed, and H. Lee, "SUPERB: Speech Processing Universal PERformance Benchmark," in *Proc. Interspeech 2021*, 2021, pp. 1194–1198.

[15] H. Chang, S. Yang, and H. Lee, "Distilhubert: Speech representation learning by layer-wise distillation of hidden-unit bert," in *Prof. of ICASSP 2022)*, 2022, pp. 7087–7091.

[16] A. Pasad, J. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021, pp. 914–921.

[17] Y. Yang, M. Hira, Z. Ni, A. Astafurov, C. Chen, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang, J. Lian, J. Hwang, J. Chen, P. Goldsborough, S. Narenthiran, S. Watanabe, S. Chintala, and V. Quenneville-Bélair, "Torchaudio: Building blocks for audio and speech processing," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6982–6986.

[18] M. Maghoumi, E. M. Taranta, and J. LaViola, "Deepnag: Deep non-adversarial gesture generation," in *26th International Conference on Intelligent User Interfaces*, 2021, pp. 213–223.

[19] R. Tavenard, "Machine learning for time series – notes from lectures at ensai," 2021.

[20] M. Blondel, A. Mensch, and J. Vert, "Differentiable divergences between time series," in *Proc. of AIStat*. 13–15 Apr 2021, vol. 130 of *Proceedings of Machine Learning Research*, pp. 3853–3861, PMLR.

[21] R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar, and E. Woods, "Tslearn, a machine learning toolkit for time series data," *Journal of Machine Learning Research*, vol. 21, no. 118, pp. 1–6, 2020.

[22] A. Aghajanyan, A. Shrivastava, A. Gupta, N. Goyal, L. Zettlemoyer, and S. Gupta, "Better fine-tuning by reducing representational collapse," *CoRR*, vol. abs/2008.03156, 2020.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[24] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[25] X. Anguera, L. Rodriguez-Fuentes, A. Buzo, F. Metze, I. Szöke, and M. Penagarikano, "Quesst2014: Evaluating query-by-example speech search in a zero-resource setting with real-life queries," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5833–5837.

[26] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, pp. 101027, 2020.