On the impact of the quality of pseudo-labels on the self-supervised speaker verification task

Abderrahim Fathan, Jahangir Alam, Woo Hyun Kang.
Computer Research Institute of Montreal, Montreal (Quebec) H3N 1M3, Canada
abderrahim.fathan@crim.ca, jahangir.alam@crim.ca, woohyun.kang@crim.ca

Abstract

One of the most widely used self-supervised speaker verification system training methods is to optimize the speaker embedding network in a discriminative fashion using clustering algorithm-driven pseudo-labels. Although the pseudo-label-based self-supervised training scheme showed impressive performance, recent studies have shown that label noise can significantly impact the performance. In this paper, we have explored various pseudo-labels driven by different clustering algorithms and conducted a fine-grained analysis of the relationship between the quality of the pseudo-labels and the speaker verification performance. From our experimental results, we shed light on several previously unexplored and overlooked aspects of the pseudo-labels that can have an impact on the speaker verification performance. Moreover, we could observe that the self-supervised speaker verification performance is heavily dependent on multiple qualitative aspects of the clustering algorithm that was used for generating the pseudo-labels. Furthermore, we show that the speaker verification performance can be severely degraded from overfitting to the noisy pseudo-labels and that the mixup strategy can mitigate the memorization effects of label noise.

1 Introduction

Speaker verification is the task of verifying the claimed speaker identity based on the given speech samples. In recent years, it has become a key technology for personnel authentication in numerous applications [17]. Typically, utterance-level fixed-dimensional embedding vectors are extracted from the enrollment and test speech samples and then fed into a scoring algorithm (e.g., cosine distance, probabilistic linear discriminant analysis) to measure their similarity or likelihood of being spoken by the same speaker. Classically, the i-vector framework has been one of the most dominant approaches for speech embedding [10, 24] thanks to its ability to summarize the distributive patterns of the speech in an unsupervised manner and with a relatively small amount of training data.

In recent years, various other deep learning-based architectures have been proposed to extract embedding vectors. They have shown better performance than the i-vector framework when a large amount of training data is available, in particular, with a sufficient number of speakers [38]. One widely employed architecture is ECAPA-TDNN [12], which has achieved state-of-the-art performance in text-independent speaker recognition. The ECAPA-TDNN uses squeeze-and-excitation (SE), employs channel- and context-dependent statistics pooling & multi-layer aggregation and applies self-attention pooling to obtain an utterance-level fixed dimensional embedding vector.

Most of the deep embedding models are trained in a fully supervised fashion and require large speaker-labeled datasets for optimization. However, well-annotated datasets can be time-consuming and expensive to obtain, which has lead to an increased interest in more affordable and larger but noisy/unlabeled datasets. One common way to solve this issue for speaker verification systems is to use clustering algorithms to generate pseudo-labels and train the speaker embedding network using

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

these labels in a discriminative fashion. The impressive performance of pseudo-labels-based self-supervised speaker verification schemes relies greatly on accurate pseudo-labels as label noise can significantly impact the performance. Most notably, due to the memorization effects [1], deep models (in particular, overparameterized networks), tend to fit easy (clean) patterns in the pseudo-labels first, and then overfit the hard and complex (noisy) patterns gradually. This leads to overfit the noise and corruptions in the training pseudo-labels and eventually the validation curve starts to drop gradually.

To boost the downstream speaker verification performance and mitigate these side effects, we employ mixup [46] as an efficient strategy to augment data by interpolating different data samples alongside their labels, which leads to better generalization to out-of-set samples. This mixup strategy has been applied and proven its strength in various tasks (e.g., image classification [46], anti-spoofing [41] and speech recognition [29]). Unlike the conventional augmentations, which simply augment the waveform or spectrogram of the speech by introducing adversaries via masking or additive noise, the mixup scheme aims to create a synthetic training sample with a new target identity. Indeed, [46] has shown that mixup not only reduces the memorization to adversarial samples, but also performs better than Empirical Risk Minimization (ERM) [42]. On the other hand, the pseudo-labels provided by the clustering algorithms are in general inaccurate and contain noise due to the discrepancy between the clustering objective(s) and the final speaker verification task which causes that mixup may not perform well [48]. Hence, in this paper we explore the effectiveness of mixup to reduce the memorization effects of noisy labels by studying two variants of mixup at both the instance input-level (i-mix) [28] and the latent space (l-mix) [22]. To this end, we explore the adaptation of the two variants to the self-supervised embedding learning for speaker verification to produce robust embeddings which can perform well on verifying out-of-set speakers.

The contributions of this paper are as follows:

- We experimented with various pseudo-labels created using a wide range of clustering algorithms and configurations (e.g., distance functions, grouping methods, architectures) for self-supervised speaker verification.
- We performed a fine-grained analysis of the quality and limitations of the obtained pseudo-label assignments from various and complementary perspectives to establish a relationship with the downstream self-supervised speaker verification performance.
- We analyzed the training behaviour of different self-supervised speaker verification systems (e.g., no regularization, i-mix, l-mix) using pseudo-labels to study the behaviour and effectiveness of mixup, at different levels, on the generalization of the learned embeddings.

2 Background and Related work

We can generally group the methods to learn from noisy data into two categories: approaches focusing on creating noise-robust algorithms to learn directly from noisy labels [2, 15, 35, 21, 30, 23], and label-cleansing approaches that aim to remove or correct mislabeled data [4, 40, 43]. This paper differs from these approaches by studying the behavior of speaker verification neural networks trained in settings with various realistic label noise generated by several clustering algorithms, which could help us to better understand which criteria/qualities in pseudo-labels are important to lead to better downstream performance in self-supervised learning.

Since the instance mix (i-mix) augmentation scheme [28] performs interpolation on the training samples and their pseudo-labels, the i-mix strategy can be applied to self-supervised learning tasks where no actual class labels are provided, and has shown potential in a number of self-supervised tasks including image classification and voice command recognition. On the other hand, the l-mix [22] strategy that applies i-mix on the latent space, instead of the raw data domain, may yield more diverse synthetic samples. In order to apply i-mix on the latent space of the speech, l-mix incorporates a variational autoencoder (VAE) encoder [25] to extract the latent variable of the given acoustic features. The resulting mixed latent variable is then fed into the VAE decoder to generate a new synthetic sample, which has different patterns from the samples generated via the the standard i-mix.

3 System Description and Clustering Metrics

To generate pseudo-labels, we explored diverse clustering algorithms including widely used classical algorithms (e.g., GMM, variational GMM [3], K-means [18], BIRCH [47], CURE [16], Agglomerative Hierarchical Clustering (AHC) [9], Divisive Hierarchical Clustering (DHC) [32]), and some

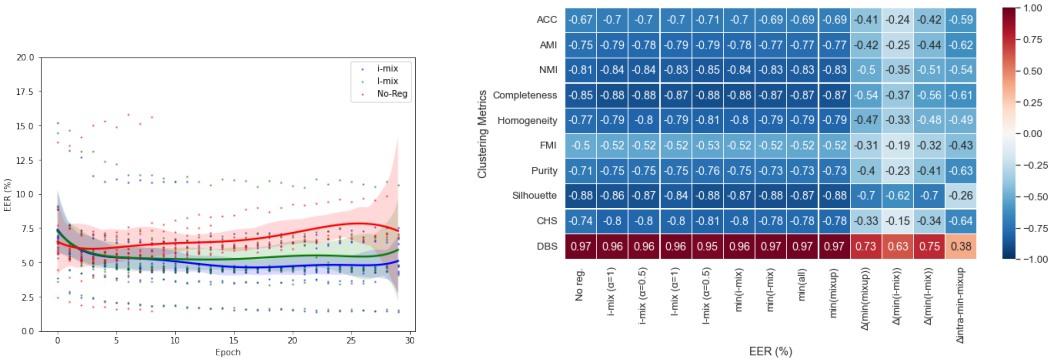
Table 1: EER (%) performance comparison between the three studied systems (no regularization, i-mix, l-mix) trained with different pseudo-labels of various qualities in terms of clustering metrics.

Model	Clustering Metrics											EER (%)				
	ACC	AMI	NMI	No. of clusters	Completeness	Homogeneity	FMI	Purity	Silhouette	CHS	DBS	No reg.	i-mix (α=1)	i-mix (α=0.5)	l-mix (α=1)	l-mix (α=0.5)
Supervised (True Labels)	**1.0**	**1.0**	**1.0**	5994	**1.0**	**1.0**	**1.0**	**1.0**	**-0.006**	31.708	4.692	**1.474**	**1.988**	**1.341**	**1.612**	**1.458**
GMM (Full cov.)	0.45	0.631	0.747	5000	0.767	0.728	0.312	0.566	-0.015	39.266	4.673	5.143	4.348	4.221	4.199	4.046
Bayesian GMM (γ=1e-5, μ=1, Full cov.)	0.45	0.629	0.746	5000	0.766	0.727	0.312	0.566	-0.015	39.257	4.673	5.143	4.136	4.348	4.311	4.284
DHC	0.097	0.204	0.477	5000	0.479	0.474	0.035	0.132	-0.060	18.044	9.068	13.531	13.012	10.816	10.498	10.997
KMeans	0.302	0.468	0.591	5000	0.645	0.546	0.194	0.311	-0.114	24.936	**2.714**	6.978	6.066	6.156	6.49	6.251
CURE	0.151	0.218	0.393	5000	0.466	0.34	0.011	0.216	-0.052	17.77	5.372	6.994	6.458	6.442	6.654	6.564
BIRCH	0.299	0.374	0.54	5000	0.725	0.43	0.013	0.353	-0.027	24.348	4.901	5.642	5.514	5.493	5.758	5.573
AHC (Ward linkage)	**0.587**	**0.74**	**0.825**	5000	**0.841**	**0.81**	0.311	**0.684**	**-0.010**	**39.561**	4.991	**3.685**	**3.478**	**3.51**	**3.377**	**3.409**
SOM	0.025	0.088	0.402	5041	0.404	0.4	0.01	0.037	-0.041	10.148	18.402	15.806	16.474	16.691	19.385	15.514
DeepCWRN	0.003	0.006	0.15	1008	0.179	0.129	0.001	0.003	-0.217	3.841	41.521	38.171	33.537	34.093	33.234	33.34
DEC	0.029	0.122	0.365	4911	0.386	0.345	0.007	0.036	-0.084	8.734	7.266	11.957	13.006	13.802	11.866	14.406
IMSAT	0.393	0.491	0.649	4987	0.668	0.63	0.297	0.426	-0.044	22.887	6.668	5.912	6.84	6.909	6.84	8.319

recent deep learning-based clustering models (IMSAT [19], DEC [44], DeepCWRN [7], SOM [26]), which allows us to generate diverse types of pseudo-labels depending on the optimization objective. Moreover, in order to thoroughly analyze the quality of pseudo-labels from different perspectives and the relationship with the downstream equal error rate (EER) performance, we use a list of 7 supervised metrics (Unsupervised Clustering Accuracy (ACC), Normalized Mutual Information (NMI) [13], Adjusted MI (AMI) [45], Completeness score [36], Homogeneity score [36], Purity score, and Fowlkes-Mallows index (FMI) [14]), and 3 unsupervised metrics (Silhouette score [37], Calinski-Harabasz score (CHS)[5], and Davies-Bouldin score (DBS) [8]). More details and discussion about the metrics and the different systems employed can be found in appendix B.

4 Results and Discussion

In Table 1, we provide the results for training the 3 speaker verification systems (ECAPA-TDNN without regularization, and with i-mix or l-mix regularizations) using the various pseudo-labels. According to the results, we can see that the quality of generated pseudo-labels, both in terms of the clustering metrics and the effectiveness for the downstream speaker verification task, widely depends on the used clustering algorithm and configuration, with some pseudo-labels achieving very impressive performance without access to true labels, narrowing the gap with the supervised models. The AHC pseudo-labels outperformed all other systems. The Gaussian Mixture Models (GMM) and their variational bayesian estimation also performed very well. From the values of clustering metrics, we can observe that the clustering algorithms have difficulty to achieve high accuracy for a dataset as large and complex as VoxCeleb2. The Silhouette scores near 0 indicate overlapping clusters, with the relatively low purity, FMI, and AMI scores suggesting clusters are highly noisy and not pure, hence the existence of discrepancies between the pseudo-labels and the speaker-identity ground truths.
However, we can observe from Table 1 and the Pearson correlation coefficients between the clustering metrics and the validation EER performance depicted in Figure 1-b that clustering metrics are highly predictive of the final downstream speaker-verification performance, with correlations for metrics such as DBS, Silhoutte, and completeness been very highly significant. This is especially interesting since unlike ACC, NMI, or the adjusted random index [20, 39] which are the main widely reported metrics in the literature, we find that notions such as completeness, dispersion and cohesion of generated clustering assignments are very important. These high degrees of linear correlation (close to 1 or -1) strongly suggest to us that monitoring clustering metrics during the generation of pseudo-labels, in particular the unsupervised Silhoutte, CHS, and DBS scores which do not need access to any ground-truth, can be very effective and practical to constantly ensure good downstream performance. Furthermore, we can notice that despite the high correlations, each of these metrics has its predictive limitations failing sometimes to predict how good or bad the downstream performance will be compared to other systems. We do observe however a complementarity between all of these metrics (e.g., GMM with full cov. vs. AHC or Bayesian GMM where DBS score alone wasn't enough to infer the relative EER performance).
Results overall show that our adopted ECAPA-TDNN-based embedding systems trained with AAM-Softmax objective [11] are robust and able to generalize even on massively noisy labels, instead of merely memorizing noise. Interestingly, they can perform well even at levels of accuracy below 40% and relatively low purity, completeness, and mutual information (e.g., KMeans, CURE, or BIRCH). Regarding mixup, as shown in Table 1, the i-mix and l-mix regularization strategies, tested with alpha values of 1 and 0.5, were both able to improve the performance in almost all the cases, further



(a) (b)

Figure 1: (a) EER (%) performance over time of speaker verification systems trained with pseudo-labels. (b) Heatmap plot of the Pearson product-moment correlation coefficients between the clustering metrics and the speaker verification EER validation performance for all pseudo-labels combined.

reducing the performance gap between the self-supervised systems and the supervised (i.e., true labels) ones. Accordingly, we could see that by favoring the smoothness of the output distribution, i-mix and l-mix are effective in mitigating the various label noise in pseudo-labels. Finally, Figure 1-a shows the speaker verification EER performance on VoxCeleb1 trials over epochs for the 3 types of systems (No reg., i-mix, and l-mix). Very importantly, we notice that non-regularized training overfits very quickly to the noisy labels and validation performance degrades dramatically over time, while mixup helps to generalize better and mitigate the memorization effects issue [1] by diluting the noise in labels and creating synthetic samples around the borders that lead to smoothing the data manifold and better class separation. Hence, helping to slow down the memorization of noisy labels and learn longly enough from the simple patterns available, which results in keeping the EER performance steadily improving. As shown in the figure, we find instance-level i-mix to be slightly more stable and robust during training compared to l-mix, which can be attributed to the high presence of noise in the pseudo-labels that can hinder the mixed up latent representations of the variational autoencoder of the l-mix system, especially with a strong discriminative objective such as AAMSoftmax, where overfitting to incorrect labels can lead to severe performance degradation. In light of this observation, in Figure 1-b we study on the one hand the differences in performance (Δ(min(i-mix)), Δ(min(l-mix)), and Δ(min(mixup))) between our systems without regularization (No. Reg) and our systems incorporating i-mix, l-mix, or mixup (best of both variants) respectively. On the other hand, we study the difference in EER performance between i-mix and l-mix (Δintra-min-mixup). From the high correlation coefficients, we can observe that (1) in general, mixup tends to become more helpful when the generated clusters are less compact or not well distanced. (2) instance i-mix often outperforms latent l-mix when clusters are less pure, less compact or not well distanced between each other.

5 Conclusion

In this paper, we analyzed the impact of the quality of pseudo-labels on the self-supervised speaker verification task. In particular, we investigated the performance of several clustering algorithms and configurations. To this end, we have conducted experiments on the Voxceleb dataset encompassing several classical and deep clustering algorithms, and three variants of the SOTA ECAPA-TDNN speaker verification model (without mixup, and with i-mix or l-mix regularization). Through our analysis, we find very high correlation between the various clustering metrics and the downstream task, where, in particular, beyond accuracy or mutual information, metrics such as completeness, separation and cohesion of clusters were found to be very helpful monitoring unsupervised metrics providing complementary indicators to yield better generalization in the downstream task. Our results showed that the pseudo-labels - based self-supervised speaker embedding systems can yield comparable performance to the supervised embedding systems without any access to the ground-truth labels during training, and demonstrated a high effectiveness of mixup, at both input and latent space levels, to mitigate the memorization effects of noisy pseudo-labels and prevent overfitting inaccurate pseudo-labels.

6 Acknowledgment

The authors wish to acknowledge the funding from the Government of Canada's New Frontiers in Research Fund (NFRF) through grant NFRFR-2021-00338 and Natural Sciences and Engineering Research Council of Canada (NSERC) through grant RGPIN-2019-05381. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the NFRF & NSERC.

References

[1] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017.

[2] E. Beigman and B. B. Klebanov. Learning with annotation noise. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 280–287, 2009.

[3] D. M. Blei and M. I. Jordan. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[4] C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of artificial intelligence research*, 11:131–167, 1999.

[5] T. Caliński and J. Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[6] J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In *INTERSPEECH*, 2018.

[7] P. Dahal. Learning embedding space for clustering from deep representations. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 3747–3755, 2018. doi: 10.1109/BigData.2018.8622629.

[8] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.

[9] W. H. E. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1:7–24, 1984.

[10] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4): 788–798, 2011. doi: 10.1109/TASL.2010.2064307.

[11] J. Deng, J. Guo, J. Yang, N. Xue, I. Cotsia, and S. P. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. doi: 10.1109/TPAMI.2021.3087709.

[12] B. Desplanques, J. Thienpondt, and K. Demuynck. ECAPA-TDNN: emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In H. Meng, B. Xu, and T. F. Zheng, editors, *Interspeech 2020*, pages 3830–3834. ISCA, 2020.

[13] P. A. Estévez, M. Tesmer, C. A. Perez, and J. M. Zurada. Normalized mutual information feature selection. *IEEE Transactions on neural networks*, 20(2):189–201, 2009.

[14] E. B. Fowlkes and C. L. Mallows. A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, 78(383):553–569, 1983.

[15] M. Guan, V. Gulshan, A. Dai, and G. Hinton. Who said what: Modeling individual labelers improves classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[16] S. Guha, R. Rastogi, and K. Shim. Cure: An efficient clustering algorithm for large databases. *SIGMOD Rec.*, 27(2):73–84, jun 1998. ISSN 0163-5808. doi: 10.1145/276305.276312. URL `https://doi.org/10.1145/276305.276312`.

[17] J. H. Hansen and T. Hasan. Speaker recognition by machines and humans: A tutorial review. *IEEE Signal Processing Magazine*, 32(6):74–99, 2015. doi: 10.1109/MSP.2015.2462851.

[18] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *JSTOR: Applied Statistics*, 28 (1):100–108, 1979.

[19] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International conference on machine learning*, pages 1558–1567. PMLR, 2017.

[20] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[21] A. Joulin, L. v. d. Maaten, A. Jabri, and N. Vasilache. Learning visual features from large weakly supervised data. In *European Conference on Computer Vision*, pages 67–84. Springer, 2016.

[22] W. H. Kang, J. Alam, and A. Fathan. l-mix: a latent-level instance mixup regularization for robust self-supervised speaker representation learning. *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[23] D. Karimi, H. Dou, S. K. Warfield, and A. Gholipour. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, 65:101759, 2020.

[24] P. Kenny. A Small Footprint I-vector Extractor. In *Odyssey*, pages 1–6, 2012.

[25] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6114`.

[26] T. Kohonen. *Self-organizing maps*, volume 30. Springer Science & Business Media, 2012.

[27] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 52(1):7–21, 2005.

[28] K. Lee, Y. Zhu, K. Sohn, C.-L. Li, J. Shin, and H. Lee. i-mix: A domain-agnostic strategy for contrastive representation learning. In *ICLR*, 2021.

[29] L. Meng, J. Xu, X. Tan, J. Wang, T. Qin, and B. Xu. Mixspeech: Data augmentation for low-resource automatic speech recognition. In *Proc. 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[30] I. Misra, C. Lawrence Zitnick, M. Mitchell, and R. Girshick. Seeing through the human reporting bias: Visual classifiers from noisy human-centric labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2939, 2016.

[31] A. Nagrani, J. S. Chung, and A. Zisserman. Voxceleb: a large-scale speaker identification dataset. In *INTERSPEECH*, 2017.

[32] F. Nielsen. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*, pages 195–211. Springer, 2016.

[33] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. Specaugment: A simple data augmentation method for automatic speech recognition. In *Interspeech 2019*, pages 2613–2617, 2019.

[34] D. Povey, A. Ghoshal, G. Boulianne, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, and G. Stemmer. The kaldi speech recognition toolkit. In *In IEEE 2011 workshop*, 2011.

[35] D. Rolnick, A. Veit, S. Belongie, and N. Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

[36] A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[37] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[38] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333, 2018. doi: 10.1109/ICASSP.2018.8461375.

[39] D. Steinley. Properties of the hubert-arable adjusted rand index. *Psychological methods*, 9(3): 386, 2004.

[40] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[41] A. Tomilov, A. Svishchev, M. Volkova, A. Chirkovskiy, A. Kondratev, and G. Lavrentyeva. STC Antispoofing Systems for the ASVspoof2021 Challenge. In *Proc. 2021 Edition of the Automatic Speaker Verification and Spoofing Countermeasures Challenge*, pages 61–67, 2021. doi: 10.21437/ASVSPOOF.2021-10.

[42] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015.

[43] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 839–847, 2017.

[44] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.

[45] N. Xuan, V. Julien, S. Wales, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. 2010.

[46] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[47] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, June 1997. ISSN 13845810. doi: 10.1023/a:1009783824328. URL `http://dx.doi.org/10.1023/a:1009783824328`.

[48] L. Zhong, Z. Fang, F. Liu, J. Lu, B. Yuan, and G. Zhang. How does the combined risk affect the performance of unsupervised domain adaptation approaches? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11079–11087, 2021.

A Clustering Evaluation Metrics

Following the commonly used evaluation metrics for clustering, we evaluate our clustering models by thoroughly assessing the quality of their generated pseudo-labels from different perspectives:

- **Unsupervised Clustering Accuracy (ACC)**: measures the consistency between the true labels and the generated pseudo-labels. $ACC = \max_m \frac{\sum_{i=1}^{N} \mathbb{1}\{y_i = m(c_i)\}}{N}$ where y_i is the ground-truth label, c_i is the model's generated cluster assignment, and m is a mapping function which ranges over all possible one-to-one mappings between true labels and assignments. The optimal mapping can be efficiently computed using the Hungarian algorithm [27].
- **Normalized Mutual Information (NMI)** [13]: $NMI(Y, C) = \frac{I(Y,C)}{\frac{1}{2}[H(Y)+H(C)]}$ where Y and C denote the ground-truth labels and the clustering assignments, respectively. H is the entropy function and I denotes the MI metric. NMI is the harmonic mean between below homogeneity and completeness scores.
- **Adjusted MI (AMI)** [45]: Since the NMI measure is not adjusted for chance, including the adjusted MI score might be preferred for comparison in some of our cases.
- **Completeness score** [36]: A clustering assignment satisfies completeness if all the data points that are members of a given class are elements of the same cluster. The scores are between 0 and 1, where 1 stands for perfectly complete assignment.
- **Homogeneity score** [36]: A clustering assignment satisfies homogeneity if all of its clusters contain only data points which are members of a single class. The score is between 0 and 1, where 1 stands for perfectly homogeneous assignment.
- **Purity score**: To compute purity, each cluster is assigned to the class which is most frequent in the cluster, and then the accuracy of this assignment is measured by counting the number of correctly assigned samples and dividing by number of samples N. Cluster purity measures how pure clusters are. If a cluster is composed of members of the same class, then it is completely pure.
- **Fowlkes-Mallows index (FMI)** [14]: Measures the similarity of two clusterings by computing the geometric mean between the precision and recall. A higher score indicates a good similarity between two clusters.
- **Silhouette score** [37]: The Silhouette score is calculated using (a) the mean intra-cluster distance and (b) the mean nearest-cluster distance for each sample. The Silhouette Coefficient for a sample is $\frac{(b-a)}{\max(a,b)}$.
- **Calinski-Harabasz score (CHS)** [5]: Taking only into account the data samples and the pseudo-labels (regardless of the original true labels), this score is defined as the ratio of the sum of between-cluster dispersion and of within-cluster dispersion. It is commonly used to compare assignments of different methods and number of clusters. The higher the value, the better is the assignment. In particular, this is very suitable when clusters are more or less spherical and compact in their middle.
- **Davies-Bouldin score (DBS)** [8]: The average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score. Lower values indicate better clustering.

B System description

B.1 Clustering-based pseudo-label generation

In order to be able to perform a complete and thorough study of the quality of the generated pseudo-labels and their relationship with the downstream speaker verification EER performance, we analyze these pseudo-labels from different perspectives through a list of 10 complementary clustering metrics. This list comprises metrics that are on the one hand based on both the pseudo-labels and true labels (e.g., Unsupervised Clustering Accuracy (ACC), Normalized Mutual Information (NMI)

[13], Adjusted MI (AMI) [45], Completeness score [36], Homogeneity score [36], Purity score, and Fowlkes-Mallows index (FMI) [14]). Among the criteria that these metrics assess, we can list the following: clustering accuracy and mutual information as measures of the consistency between the true labels and the generated pseudo-labels, homogeneity, completeness, and purity of clusters, and precision and recall. On the other hand, we include unsupervised metrics which are only based on the generated pseudo-labels and the data samples (e.g., Silhouette score [37], Calinski-Harabasz score (CHS)[5], and Davies-Bouldin score (DBS) [8]) without taking into account the original true labels. These metrics allow us to measure how compact or scattered are the clusters (e.g., intra-class dispersion, between-cluster distances, etc.). For more details about each metric, please check our appendix A.

To this end, we have extracted i-vector [10, 24] using the Kaldi toolkit [34], which is a statistical unsupervised fixed-dimensional representation from each training utterance and performed clustering on top of them. After training the clustering algorithms, we selected the aligned cluster for each utterance and used the cluster-id as pseudo-label. With the clustering-based pseudo-labels, we can train the embedding network via softmax-based objectives, analogous to supervised learning.

In order to evaluate our proposed clustering method and to evaluate the performance of the generated pseudo-labels for self-supervised speaker verification, we conducted a set of experiments based on the VoxCeleb2 dataset [6]. For training the embedding networks, we used the development subset of the VoxCeleb2 dataset, consisting of 1,092,009 utterances collected from 5,994 speakers. The evaluation was performed according to the original VoxCeleb1 trial list [31], which consists of 4,874 utterances spoken by 40 speakers.

The acoustic features used in the experiments were 40-dimensional Mel-frequency cepstral coefficients (MFCCs) extracted at every 10 ms, using a 25 ms Hamming window via Kaldi toolkit [34]. Moreover, we have used waveform-level data augmentations including additive noise and room impulse response (RIR) simulation [38]. In addition to the waveform-level augmentations, for the ECAPA-TDNN-based systems, we have also applied augmentation over the extracted MFCCs feature, analogous to the specaugment scheme [33].

For all of our clustering algorithms, we have set the number of clusters to be 5000 (except self-organizing maps (SOM) where number of clusters was set to be the size of the map 71*71=5041).

B.2 Instance-mixup (i-mix) for speaker verification

The i-mix [28] is an augmentation method for improving the generalization of the self-supervised system [28]. For an objective function $L_{pair}(x, y)$, where x is the input data and y is the corresponding pseudo-label, given two data instances (x_i, y_i) and (x_j, y_j), the i-mix loss is defined as follows:

$$\begin{aligned} &L_{pair}^{i-mix}((x_i, y_i), (x_j, y_j)) \\ &= L_{pair}(\lambda x_i + (1-\lambda)x_j, \lambda y_i + (1-\lambda)y_j), \end{aligned} \tag{1}$$

where $\lambda \sim Beta(\alpha, \alpha)$ is a mixing coefficient. For losses that are linear with respect to the labels, equation 1 can be rewritten as,

$$\begin{aligned} &L_{pair}^{i-mix}((x_i, y_i), (x_j, y_j)) \\ &= \lambda L_{pair}(\lambda x_i + (1-\lambda)x_j, y_i) \\ &+ (1-\lambda) L_{pair}(\lambda x_i + (1-\lambda)x_j, y_j). \end{aligned} \tag{2}$$

The i-mix aims to generate synthetic training sample $\lambda x_i + (1-\lambda)x_j$ with identity label $\lambda y_i + (1-\lambda)y_j$.

The i-mix strategy can be easily applied to the self-supervised speaker verification system training process [22]. For instance, let us think about a self-supervised speaker embedding network being trained with additive angular margin softmax (AAMSoftmax) objective which is formulated as follows:

$$L_{AAMSoftmax} = -\frac{1}{N}\sum_{i=1}^{N} log(\frac{e^{s(cos(\theta_{y_i,i}+m))}}{K_1}), \tag{3}$$

where $K_1 = e^{s(cos(\theta_{y_i,i}+m))} + \sum_{j=1,j\neq i}^{C} e^{scos\theta_{j,i}}$, N is the batch size, C is the number of classes, y_i corresponds to pseudo-label index, $\theta_{j,i}$ represents the angle between the column vector of weight

matrix W_j and the i-th embedding ω_i, where both W_j and ω_i are normalized. Then we can incorporate the i-mix strategy into the self-supervised AAMSoftmax as:

$$\begin{aligned} &L_{i-AAMSoftmax} = \\ &-\lambda\frac{1}{N}\sum_{i=1}^{N} log(\frac{exp(s(cos(\theta_{y_i,mix(i,r\neq i)}+m)))}{K^{AAM}_{mix,i}}) \\ &-(1-\lambda)\frac{1}{N}\sum_{i=1}^{N} log(\frac{exp(s(cos(\theta_{y_{r\neq i},mix(i,r\neq i)}+m)))}{K^{AAM}_{mix,r\neq i}}), \end{aligned} \tag{4}$$

$$\begin{aligned} K^{AAM}_{mix,i} &= exp(s(cos(\theta_{y_i,mix(i,r\neq i)}+m))) \\ &+\sum_{j=1,j\neq i}^{c} exp(s(cos(\theta_{y_j,mix(i,r\neq i)}))), \end{aligned} \tag{5}$$

where $\theta_{y_i,mix(i,r\neq i)}$ is the angle between the normalized W_j and $\omega_{mix(i,r\neq i)}$.

B.2.1 latent-level instance mixup (l-mix) for speaker verification

Although applying i-mix augmentation to the raw data has proven its strength in generalization in speaker verification, due to the nature of linear interpolation, the standard i-mix strategy can only generate synthetic samples between the original samples. Since such limitation may restrict the overall diversity of the synthetic samples generated by the i-mix method, a latent-level i-mix (l-mix) was proposed [22].

Before training the embedding system, given training MFCC x, the VAE is trained according to the following objective:

$$L_{VAE} = D_{KL}(q_\phi(z|x)||p_\theta(z)) - E_{q_\phi(z|x)}[log_\theta(x|z)], \tag{6}$$

where z is the latent variable, ϕ is the encoder parameter and θ is the decoder parameter. The encoder network takes the MFCC sample as input and generates the mean and log-variance of the posterior latent distribution $q_\phi(z|x)$. The decoder network takes a latent sample and reconstructs the MFCC. Detailed information on the VAE used for l-mix can be found in [22].

In the l-mix framework, a VAE is trained prior to training the embedding network. Once the VAE has been trained, the VAE is used to perform mixup on the latent space:

$$\begin{aligned} &z_{mix} = \lambda z_1 + (1-\lambda)z_2 \\ &\sim N(\lambda\mu_1 + (1-\lambda)\mu_2, \lambda^2\sigma_1^2 + (1-\lambda)^2\sigma_2^2), \end{aligned} \tag{7}$$

where $\lambda \sim Beta(\alpha, \alpha)$. The mean of the mixed up latent variable z_{mix} is fed into the decoder network to generate an MFCC sample x_{l-mix}.

Analogous to the i-mix method, we can apply the l-mix to the self-supervised AAMSoftmax objective as follows:

$$\begin{aligned} &L_{l-AAMSoftmax} = \\ &-\lambda\frac{1}{N}\sum_{i=1}^{N} log(\frac{exp(s(cos(\theta_{y_i,l-mix(i,r\neq i)}+m)))}{K^{AAM}_{l-mix,i}}) \\ &-(1-\lambda)\frac{1}{N}\sum_{i=1}^{N} log(\frac{exp(s(cos(\theta_{y_{r\neq i},l-mix(i,r\neq i)}+m)))}{K^{AAM}_{l-mix,r\neq i}}), \end{aligned} \tag{8}$$

$$\begin{aligned} K^{AAM}_{l-mix,i} &= exp(s(cos(\theta_{y_i,l-mix(i,r\neq i)}+m))) \\ &+\sum_{j=1,j\neq i}^{c} exp(s(cos(\theta_{y_j,l-mix(i,r\neq i)}))). \end{aligned} \tag{9}$$

Attributed to the non-linear nature of the VAE, the resulting samples are expected to be more diverse than the standard i-mix strategy.

C Full results with additional configurations

Table 2: EER (%) performance comparison between the three studied systems (no regularization, i-mix, l-mix) trained with different pseudo-labels of various qualities in terms of clustering metrics.

Model	Clustering Metrics											EER (%)				
	ACC	AMI	NMI	No. of clusters	Completeness	Homogeneity	FMI	Purity	Silhouette	CHS	DBS	No reg.	i-mix (α=1)	i-mix (α=0.5)	l-mix (α=1)	l-mix (α=0.5)
Supervised (True Labels)	**1.0**	**1.0**	**1.0**	5994	**1.0**	**1.0**	**1.0**	**1.0**	**-0.006**	31.708	4.692	**1.474**	**1.988**	**1.341**	**1.612**	**1.458**
GMM (Full cov.)	0.45	0.631	0.747	5000	0.767	0.728	0.312	0.566	-0.015	39.266	4.673	5.143	4.348	4.221	4.199	4.046
GMM (Spherical cov.)	0.427	0.587	0.711	5000	0.739	0.685	0.22	0.539	-0.037	38.665	4.864	5.265	4.47	4.3	4.512	4.544
GMM (Diagonal cov.)	0.425	0.6	0.721	5000	0.748	0.696	0.23	0.539	-0.033	38.455	4.874	5.451	4.544	4.671	4.698	4.459
GMM (Tied cov.)	0.457	0.66	0.767	5000	0.788	0.747	**0.317**	0.574	-0.016	38.922	4.726	5.164	4.274	4.465	4.454	4.348
Bayesian GMM (γ=1e-5, μ=1, Full cov.)	0.45	0.629	0.746	5000	0.766	0.727	0.312	0.566	-0.015	39.257	4.673	5.143	4.136	4.348	4.311	4.284
Bayesian GMM (γ=100, μ=0.01, Full cov.)	0.449	0.63	0.746	5000	0.766	0.727	0.311	0.566	-0.015	39.258	4.675	4.958	4.268	4.39	4.364	4.348
DHC	0.097	0.204	0.477	5000	0.479	0.474	0.035	0.132	-0.060	18.044	9.068	13.531	13.012	10.816	10.498	10.997
KMeans	0.302	0.468	0.591	5000	0.645	0.546	0.194	0.311	-0.114	24.936	**2.714**	6.978	6.066	6.156	6.49	6.251
CURE	0.151	0.218	0.393	5000	0.466	0.34	0.011	0.216	-0.052	17.77	5.372	6.994	6.458	6.442	6.654	6.564
BIRCH	0.299	0.374	0.54	5000	0.725	0.43	0.013	0.353	-0.027	24.348	4.901	5.642	5.514	5.493	5.758	5.573
AHC (Ward linkage)	**0.587**	**0.74**	**0.825**	5000	**0.841**	**0.81**	0.311	**0.684**	**-0.010**	**39.561**	4.991	**3.685**	**3.478**	**3.51**	**3.377**	**3.409**
SOM	0.025	0.088	0.402	5041	0.404	0.4	0.01	0.037	-0.041	10.148	18.402	15.806	16.474	16.691	19.385	15.514
DeepCWRN	0.003	0.006	0.15	1008	0.179	0.129	0.001	0.003	-0.217	3.841	41.521	38.171	33.537	34.093	33.234	33.34
DEC	0.029	0.122	0.365	4911	0.386	0.345	0.007	0.036	-0.084	8.734	7.266	11.957	13.006	13.802	11.866	14.406
IMSAT	0.393	0.491	0.649	4987	0.668	0.63	0.297	0.426	-0.044	22.887	6.668	5.912	6.84	6.909	6.84	8.319

D Relationship between the clustering metrics and the downstream speaker verfification EER performance.

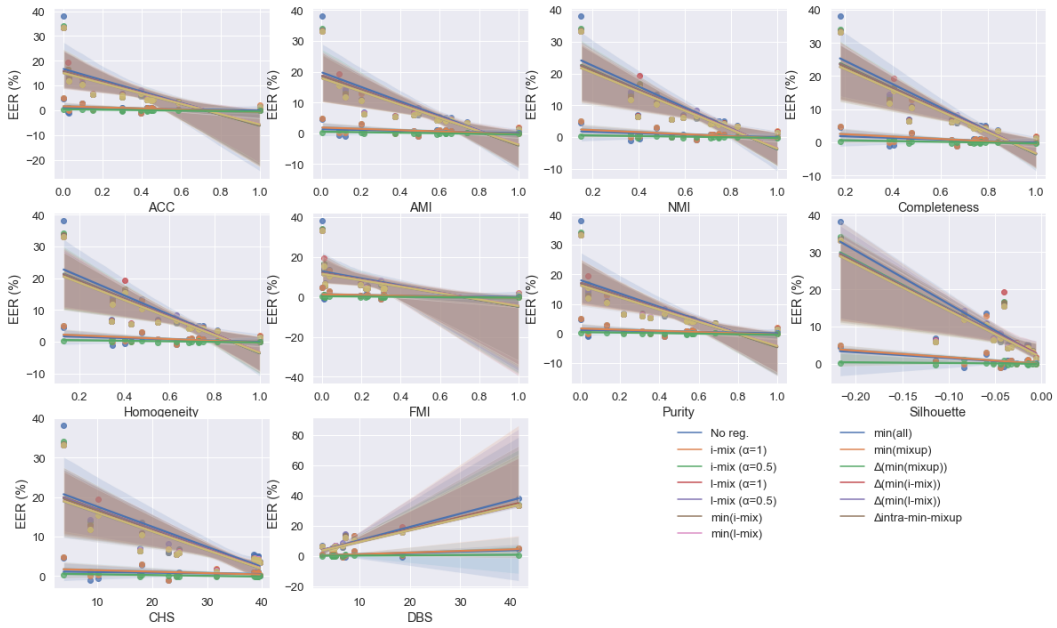


Figure 2: visualization of the estimation and plot of the linear regression models relating each of the 10 clustering metrics of pseudo-labels with the speaker verification EER performance.

E Self-supervised angular additive margin softmax (AAMSoftmax) objective

The angular additive margin softmax (AAMSoftmax) objective is one of the most popular methods for training a speaker embedding network [11]. The AAMSoftmax objective is formulated as follows:

$$L_{AAMSoftmax} = -\frac{1}{N}\sum_{i=1}^{N} log(\frac{e^{s(cos(\theta_{y_i,i}+m))}}{K_1}), \tag{10}$$

where $K_1 = e^{s(cos(\theta_{y_i,i}+m))} + \sum_{j=1,j\neq i}^{c} e^{scos\theta_{j,i}}$, N is the batch size, c is the number of classes, y_i corresponds to label index, $\theta_{j,i}$ represents the angle between the column vector of weight matrix W_j

and the i-th embedding ω_i, where both W_j and ω_i are normalized. The scale factor s is used to make sure the gradient is not too small during the training and m is a hyperparameter that encourages the similarity of correct classes to be greater than that of incorrect classes by a margin m.

The training of AAMSoftmax for self-supervised speaker embedding learning is made possible by the use of our generated pseudo-labels as the above objective requires speaker labels for training.

F Variational Autoencoder (VAE) used for extracting the latent variables

Table 3: Architecture for the variational autoencoder (VAE) used for extracting the latent variable from the MFCCs.

Layer #	Encoder	Decoder
1	3×3 2D-Conv, 32 ReLU, stride 3	64×32 FC
2	3×3 2D-Conv, 64 ReLU, stride 3	3×3 2D-TransposedConv, 32 ReLU, stride 3
3	3×3 2D-Conv, 32 ReLU, stride 3	3×3 2D-TransposedConv, 64 ReLU, stride 3
4	3×3 2D-Conv, 32 ReLU, stride 3	3×3 2D-TransposedConv, 32 ReLU, stride 3
5	32×64 FC for each μ and $log\sigma^2$	3×3 2D-TransposedConv, 1 ReLU, stride 3