INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Investigating wav2vec2 context representations and the effects of fine-tuning, a case-study of a Finnish model

Tamás Grósz, Yaroslav Getman, Ragheb Al-Ghezi, Aku Rouhe, Mikko Kurimo

Department of lnformation and Communications Engineering, Aalto University, Finland
firstname.lastname@aalto.fi

Abstract

Self-supervised speech models, such as the wav2vec2, have become extremely popular in the past few years. Their main appeal is that after their pre-training on a large amount of audio, they require only a small amount of supervised, finetuning data to achieve outstanding results. Despite their immense success, very little is understood about the pre-trained models and how finetuning changes them. In this work, we take the first steps towards a better understanding of wav2vec2 systems using model interpretation tools such as visualization and latent embedding clustering. Through our analysis, we gain new insights into the abilities of the pre-trained networks and the effect that finetuning has on them. We demonstrate that the clusters learned by the pre-trained model are just as important a factor as the supervised training data distribution in determining the accuracy of the finetuned system, which could aid us in selecting the most suitable pre-trained model for the supervised data.
Index Terms: wav2vec2, ASR, contextual embeddings, interpretation

1. Introduction

The latest version of the self-supervised deep learning models called wav2vec2 [1] enjoys wide popularity in the field of Automatic Speech Recognition (ASR). Their main advantage is the self-supervised pre-training which produces a model that can be easily fine-tuned with only a limited amount of supervised data. Thanks to the numerous, publicly available pre-trained models (which were often trained with more than one hundred thousand hours of speech data [1]) and the various tools like HuggingFace Transformers [2] that enable an easy way to fine-tune them, there are already hundreds of fine-tuned models available for various languages.

Furthermore, several works have empirically proved that the so-called context embedding produced by the Transformer component of the pre-trained wav2vec2 contains valuable information not just for speech recognition, but also for a wide range of speech tasks [3], including emotion recognition [4, 5], pronunciation evaluation [6] and other paralinguistic tasks. Despite its success and popularity, most research focuses on what wav2vec2 can be used for, and very little effort is spent on understanding the context embeddings produced by them. We should mention two recent works that aim to investigate self-supervised models. In [7], the authors studied how wav2vec2 learns phonetics. Unfortunately, their primary method was principal component analysis (PCA) [8], which is known to be a sub-optimal tool for the visualization of neural embeddings as it focuses on maintaining the global structure instead of preserving the local structure of the data. Another study [9] probed unsupervised speech models to see if they preserve phoneme, language and speaker information using t-Stochastic Neighbor Embedding (t-SNE) [10] and K-means clustering [11]. To emphasize the importance of understanding the models, Chen et. al. [12] remarks that interpretation could aid architecture optimization, although the authors purely focus on attention maps to understand their models. In contrast, we investigate the context embeddings produced by the Transformer component and specifically focus on the changes that occur due to the fine-tuning procedure. We employ multiple model interpretation techniques to compare the embeddings of a pre-trained model with the fine-tuned versions in order to better understand the capabilities of the pre-trained model and the effects of the fine-tuning procedure. Our main objective is to figure out what knowledge the model already possessed before fine-tuning and what new things did it discover from the supervised data.

Beyond common visual interpretation methods like PCA, t-SNE [10] and UMAP [13], we also employ clustering techniques like the standard K-means and its variant that uses the cosine distance called spherical K-means [14] to validate what concepts the wav2vec2 have discovered via self-supervision and how these adapted to the supervised data. In our experiments, we use a part of the colloquial Finnish speech corpus called Lahjoita Puhetta (Donate Speech) [15], and the Finnish wav2vec2 model fine-tuned using that data, which proved to be an excellent solution for this dataset. We selected the multi annotator test set as our primary data, which contains annotations from 4 different annotator groups. This allowed us to investigate how the model behaves when the data becomes challenging, i.e. when human annotators disagree. Additionally, we were interested in if we could see signs of model uncertainty or mistakes in the context vectors.

2. Methods

Wav2vec2 is a self-supervised framework for extracting deep acoustic representations from speech. The system architecture involves a convolutional feature encoder, followed by a stack of Transformer encoder blocks. During the pre-training phase, the model weights are optimized by solving a contrastive task over quantized masked latent audio representations. After learning to extract general audio representations, the model is fine-tuned for a specific downstream task such as speech recognition with Connectionist Temporal Classification (CTC) loss or speech classification with cross-entropy (CE) loss. The pre-training phase involves training on substantial amount of raw data audio, typically thousands of hours of speech, while a relatively small amount of labeled data is required for fine-tuning.

In this work, we explore the wav2vec2 Large (300M parameters) system pre-trained on 100.000 hours of European Parliament plenary session recordings, namely VoxPopuli dataset

10.21437/Interspeech.2023-837

[16]. This subset covers in total 23 languages and includes 4.400 hours of Finnish. The fine-tuned version of this model is trained on 100 hours of colloquial Finnish from the Lahjoita puhetta (Donate Speech) corpus [15]. During fine-tuning the weights of the CNN part are kept frozen, and only the Transformer part is updated.

2.1. Visual investigation and clustering

We utilized both visualization approaches based on PCA, t-SNE [10, 17] and UMAP [13], and clustering techniques to investigate our models. The main goal of visual interpretation is to reduce the dimension of the latent context representations into a low-dimensional space so that humans can inspect them. PCA is a standard tool for this; unfortunately, it merely focuses on maintaining global information, thus the crucial local structures could be lost. On the other hand, both t-SNE and UMAP employ optimization steps to maintain the local structures making them better suited for this task, and they are routinely used for visualizing hidden neural representation [18]. We should note that simply relying on visual inspections could be misleading, as a considerable amount of information could be lost during the transformation. Our primary aim in this paper is to establish a procedure for comparing pre-trained and finetuned models and raise awareness that we simply cannot blindly trust visual interpretations without vigorously validating them first. To ensure that our findings based on the generated images are well-founded, we employ k-Means clustering before and after the dimension reduction step of the visualization algorithms. By comparing the clusters of the original data and its visual form we can estimate the amount of information we lost due to the transformation. Note, that the original K-means algorithm uses Euclidean distances, which might not be well-suited for our high dimensional data, so we decided to test the Spherical K-Means variant [14], which uses cosine distance, a more commonly used metric in dealing with neural embeddings.

3. Validating our approaches

Since our investigations are largely dependent on visualization and clustering tools, we must rigorously validate them to ensure that we do not lose too much information during the transformation. The issue is that there is no single metric that can reliably measure how faithful these transformations are. Here we choose a pipeline approach, where we first apply clustering to the context representations of the fine-tuned model and compare the clusters with the predicted labels. The context embeddings were extracted from the last, 24th layer in the Transformer part. Our choice fell on this layer as the commonly used ASR training algorithm connects the CTC head directly to this layer, thus it generates the final embeddings before classification. Our motivation is that the clusters of a good method should be very similar to the pseudo labels of the fine-tuned model, which is the foundation of the Deep Clustering algorithm [19]. To this end, we performed K-means clustering and set the desired number of clusters to be the number of CTC outputs plus one, hoping that the blank parts could be split into two clusters: one for the silence and one for the confusing parts between characters (preliminary experiments showed that this approach is better than having only one cluster for the blank label). We used multiple metrics, including Adjusted Mutual Information (*AMI*) [20], Homogeneity (*Hom.*) and Completeness (*Comp.*) [21] to show the effectiveness of the clustering.

Table 1 summarizes the results of this analysis. We can see that the generated clusters had a high homogeneity and relatively good AMI for the fine-tuned vectors. The relatively lower completeness score is mainly caused by the fact that the clusterings failed to split the blank part into only two subclusters. Overall, we could say that K-means clustering, although it lost some information, still provided relatively good clusters. The low scores obtained with the pre-trained data demonstrate that during fine-tuning the embeddings changed considerably, furthermore, the large cluster of the blank label was not formed before supervised training. Interestingly using cosine distances during the clustering (spherical clusters) did not yield better results for the fine-tuned embeddings. This suggests that the Euclidean distance is equally good for discovering latent clusters.

Table 1: *Evaluation of the K-means clusters of the pre-trained and fine-tuned context embeddings.*

Metric	AMI	Hom.	Comp.
Clusters of pre-trained emb.	0.15	0.25	0.10
Clusters of fine-tuned emb.	0.70	0.94	0.56
Spherical clust. of fine-tuned emb.	0.69	0.93	0.55

Table 2: *Comparison of clusters found using the context embeddings and the visualized coordinates produced for them.*

Method	AMI	Hom.	Comp.
PCA (pre-trained)	0.45	0.44	0.45
t-SNE (pre-trained)	0.45	0.46	0.44
UMAP (pre-trained)	0.49	0.50	0.49
PCA (fine-tuned)	0.63	0.68	0.58
t-SNE (fine-tuned)	0.62	0.76	0.52
UMAP (fine-tuned)	0.69	0.83	0.59

Next, we turned our attention to the visualization methods. To determine how much information we lose by employing them, we compared the K-means clusters we got on the context embeddings with those that we found by applying K-means to the produced 2D coordinates. Our expectation was that the two clusterings should be quite similar, with minimal distortion. Table 2 contains the comparison results. We choose to evaluate the visualization approaches on both the pre-trained and fine-tuned data to have a complete picture. We can see that UMAP proved to be the most trustworthy method. The clusters formed on the 2D coordinates generated by UMAP have the highest similarity to the ones found in the raw data, according to all three metrics. This observation is in line with our previous work, where we compared UMAP and t-SNE during the interpretation of an acoustic model [18]. T-SNE proved to be a close second, performing just slightly worse than UMAP. PCA achieved similar results as t-SNE, but it homogeneity was the lowest, especially in the case of the fine-tuned data. That being said, we should also note that a considerable amount of information is lost during the transformation of context vectors into a low-dimensional space. This means that we should not trust the visualizations completely, and whatever is discovered through them should be validated via some other method like probing [22] or clustering. Interestingly, the fine-tuned embeddings proved to be easier to visualize, maintaining considerably more information than in the case of the pre-trained ones, possibly due to the clustering effects of the supervised training procedure.

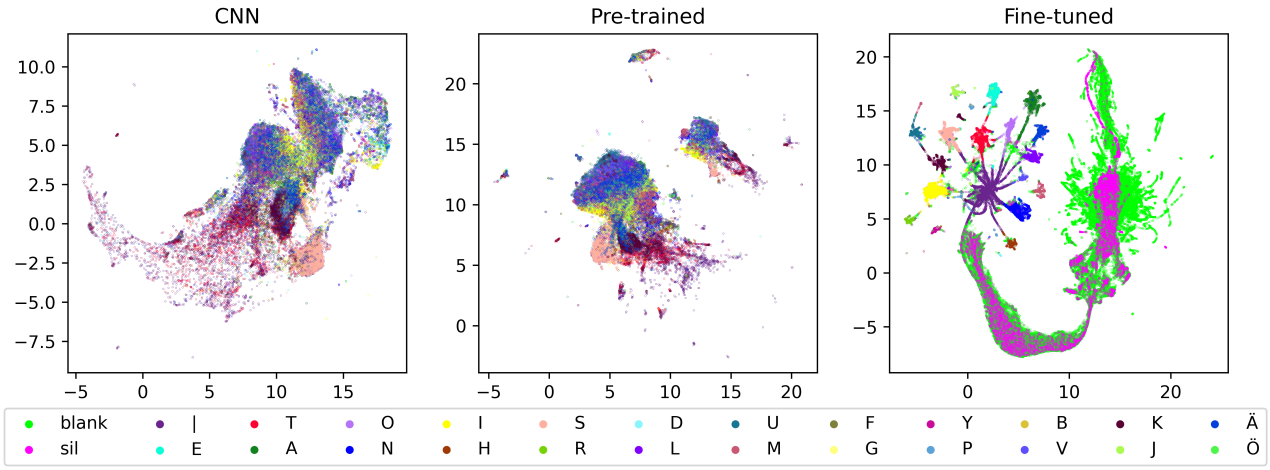

Figure 1: *Visualization of the embedded vectors extracted from the output of the CNN, the pre-trained Transformer and the fine-tuned model using the UMAP algorithm. We split the blanks into two sub-groups (silence and non-silent blank) using forced alignment. In case of the CNN and pre-trained visualizations we remove the blank and silence datapoints to enable easier inspection.*

4. Comparing the pre-trained and fine-tuned models

After the validation of our tools, we now turn our attention towards comparing the pre-trained embeddings with the fine-tuned ones. Based on Table 1, we can expect that there are major changes caused by the fine-tuning process, which is confirmed by the visualizations in Figure 1. Perhaps the first observation that we made is the strong prevalence of the blank category, which is caused by the CTC algorithm. We can see that all three methods discovered the large cluster of blanks, and t-SNE and UMAP even separated it into two sub-clusters[1]. Our initial hypothesis that the model separated the real silent parts from the non-silent blank parts, proved to be partially true, as one of the two clusters usually contained considerably more silences than the other. However, both clusters contained data labeled as silence by the force alignment of the transcripts, meaning that probably there are other factors behind this separation, such as environmental noise.

Next, we observed that the fine-tuned model was able to separate the character groups quite well, while the pre-trained context embeddings only categorized some of them and naturally had no knowledge about the blank label. Please note that the pre-trained model is a multilingual one, which only processed a small amount ($< 5\%$ of the speech corpus) of Finnish data during the training process and consequently struggled with the Finnish characters like 'Ä', 'Ö', and 'Y' ($[y:]$).

4.1. Knowledge encoded during pre-training

To further investigate the knowledge already present in the pre-trained model, we took a closer look at its clusters. First, we compared the CNN, pre-trained and fine-tuned clusters pairwise. We noticed that the highest similarity and mutual information was between the convolutional feature encoder embeddings and the pre-trained Transformer outputs, see Table 3. At the same time, the fine-tuned data seems to differ from the others considerably, still they have more in common with the pre-trained embeddings than with the CNN ones.

[1]See additional pictures and our codes at `https://github.com/aalto-speech/Wav2vec2Interpretation`

Table 3: *Comparison between the CNN, pre-trained and fine-tuned clusters.*

Emb.	Pre			Fine		
	AMI	Hom.	Comp.	AMI	Hom.	Comp.
CNN	0.43	0.44	0.43	0.12	0.15	0.10
Pre	–	–	–	0.23	0.22	0.23

Next, we investigated the pre-trained clusters to see what kind of information they contained about the graphemes. In this step, we evaluated each cluster individually and assigned a character label to them based on their granularity by selecting the most represented one. Based on this assignment, we discovered several groups of graphemes which were often appearing in the same cluster. Additionally, we saw that some characters could be "recognized" with relatively good accuracy, while others are merged with the blank cluster. The unknown characters are mainly non-Finnish characters, which are used very rarely, mainly in foreign words. Interestingly, the multilingual model already had a good idea about some Finnish characters like 'Ä' and 'Ö', but it excelled at recognizing the more common characters like 'S' and 'U', sometimes with close to 50% accuracy. This observation means that we do not necessarily need a supervised fine-tuning step, as the pre-trained model already obtained some information about the characters, and even with our simplistic approach we can achieve an overall 16.6% character recognition rate (83.4% CER). This explains why a more sophisticated clustering with Generative Adversarial Networks can be used to train ASR models completely unsupervised [23, 24].

Lastly, we investigated whether the fine-tuned model excelled and struggled with the same characters as the pre-trained system. For this experiment, we chose the best transcriber (annotator T2 in [15]) and calculated the error rate of each character. In Table 4, we list the character groups per accuracy interval and in Table 5 the CER groups of the fine-tuned model can be observed. We can observe large overlaps between the two groupings suggesting that the pre-trained models performance on characters is somewhat indicative of how well the fine-tuned

Table 4: *Discovered knowledge about characters present in the pre-trained model according to the K-means clustering.*

Category	Characters
Merged with blank	C, Q, W, X, Z, Å
Often confused character pairs	[E, J], [F, T], [M, N]
Characters accuracy <20%	A, C, D, H, J, L, O, T, Y
Characters accuracy 20 – 30%	E, I, M, N, Ö, R, Ä
Characters accuracy 30+%	B, G, K, P, S, U, V

Table 5: *Character groups based on their fine-tuned recognition accuracy (boundaries are the 33rd and 66th percentile).*

Interval	Characters
44 – 94%	B, C, D, G, H, N, Ä, Ö
94 – 96%	E, I, J, L, M, O, Y
96 – 98%	A, K, P, R, S, T, U, V

model will recognize them. At the same time, a correlation of 0.35 between the pre-trained and finetuned model's character level accuracy suggest that in many cases the fine-tuning process can compensate for the weaknesses of the pre-training. We should note that the correlation between the character frequencies in the training data and the final CER is very similar (0.38), and combining the pre-trained model's accuracy with the frequency statistics yielded a correlation of 0.52, meaning that the initial pre-trained models capabilities is almost as informative about the final performance as the composition of the training data.

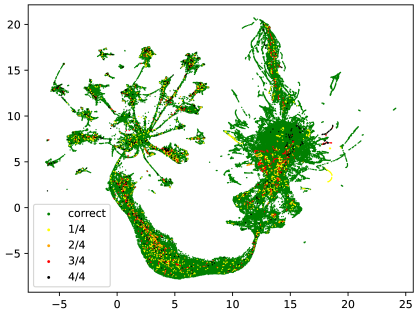

Figure 2: *Visualization of the five error categories based on how many annotators agreed about the error.*

After the comparison, we also took a closer look at the errors of the fine-tuned model. The fact that we had transcripts from four different annotators gave us a unique opportunity to investigate their distribution. In Figure 2, we can see the fine-tuned embeddings visualized and colored based on the number of annotators whose transcript marked a predicted character as a mistake. Our observation based on the visualizations is that the most serious mistake (4/4) generally happens closer to the edge of the character clusters than to the centroid. One exception is the deletion mistakes caused by the blank label, where most of the mistakes are close to the centre of the cluster, signalling that the model is quite confident in predicting the blank instead of the human-annotated characters.

Naturally, we were interested in testing and confirming this visual observation. In this case, we opted to use a probing approach and trained a simple classifier using the context embeddings to see if our theory based on the visualization holds

Figure 3: *Confusion matrix of the mistake classifier.*

in the original higher dimensional space. We should note that our approach was inspired by recent confidence estimation approaches, which employ a separate neural model as a mistake/correct classifier to estimate the confidence of end-to-end ASR models. In our case, we used only the embeddings, which could be considered an analogue to the output of the encoder in other works. In figure 3, we can see the confusion matrix of the classifier that was trained to separate correct samples from the different error categories. We can see that our original observation was most likely just an artefact of the visualization technique, as the classifier struggles to distinguish between correct and incorrect samples. Interestingly, disregarding the large percentage of mistakes recognized as correct output, we can see very little confusion between the different levels of annotator agreements and the most severe mistake category (4/4) is recognizable with the highest accuracy. Based on these findings, we can conclude that the context embeddings of fine-tuned wav2vec2 models, albeit contain some information about the confidence of the model, but it is not enough to build an accurate confidence estimator.

5. Conclusions

In this work, we investigated the capabilities of the pre-trained wav2vec2 models and compared them to the fine-tuned model. Our analysis showed that even pre-trained models have some information about various graphemes, and their initial ability to recognize the characters is just as important a factor in determining the final accuracy of the fine-tuned model as the composition of training data, and this could be used to select the most suited pre-trained model for a given task. We also saw that the embedded vectors carry some information about the confidence of the model, but further investigation is needed in order to determine what other information is needed to estimate model confidence properly. Our work is a case study focused on a single fine-tuned model, but we consider it a crucial future task to repeat the experiments on other languages using other models. These experiments would enable us to draw more general conclusions about the effects of fine-tuning and would get us one step closer to adequately understanding these neural models.

6. Acknowledgements

The authors are grateful for the founding recieved from the Academy of Finland project 345790 in ICT 2023 programme's project "Understanding speech and scene with ears and eyes".

7. References

[1] 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*, vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460.

[2] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on EMNLP: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[3] S. wen 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, T.-H. Huang, W.-C. Tseng, K. tik Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H. yi Lee, "SUPERB: Speech Processing Universal PERformance Benchmark," in *Proc. Interspeech 2021*, 2021, pp. 1194–1198.

[4] L. Pepino, P. Riera, and L. Ferrer, "Emotion Recognition from Speech Using wav2vec 2.0 Embeddings," in *Proc. Interspeech 2021*, 2021, pp. 3400–3404.

[5] M. Sharma, "Multi-lingual multi-task speech emotion recognition using wav2vec 2.0," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6907–6911.

[6] Y. Getman, R. Al-Ghezi, K. Voskoboinik, T. Grósz, M. Kurimo, G. Salvi, T. Svendsen, and S. Strömbergsson, "wav2vec2-based Speech Rating System for Children with Speech Sound Disorder," in *Proc. Interspeech 2022*, 2022, pp. 3618–3622.

[7] T. tom Dieck, P. A. Pérez-Toro, T. Arias, E. Noeth, and P. Klumpp, "Wav2vec behind the Scenes: How end2end Models learn Phonetics," in *Proc. Interspeech 2022*, 2022, pp. 5130–5134.

[8] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[9] M. de Seyssel, M. Lavechin, Y. Adi, E. Dupoux, and G. Wisniewski, "Probing phoneme, language and speaker information in unsupervised speech representations," in *Proc. Interspeech 2022*, 2022, pp. 1402–1406.

[10] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[11] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *JSTOR: Applied Statistics*, vol. 28, no. 1, pp. 100–108, 1979.

[12] L. Chen and M. Asgari, "Interpreting a pre-trained model is a key for model architecture optimization: A case study on wav2vec 2.0," *ArXiv*, vol. abs/2104.02851, 2021.

[13] L. McInnes and J. Healy, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[14] K. Hornik, I. Feinerer, M. Kober, and C. Buchta, "Spherical k-means clustering," *Journal of Statistical Software*, vol. 50, no. 10, p. 1–22, 2012. [Online]. Available: https://www.jstatsoft.org/index.php/jss/article/view/v050i10

[15] A. Moisio, D. Porjazovski, A. Rouhe, Y. Getman, A. Virkkunen, T. Grósz, K. Lindén, and M. Kurimo, "Lahjoita puhetta: a large-scale corpus of spoken finnish with some benchmarks," *Language Resources and Evaluation*, pp. 1 – 33, 2022.

[16] C. Wang, M. Riviere, A. Lee, A. Wu, C. Talnikar, D. Haziza, M. Williamson, J. Pino, and E. Dupoux, "VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," in *Proceedings of the 59th Annual Meeting of the ACL and the 11th ICNLP (Volume 1: Long Papers)*, Aug. 2021, pp. 993–1003. [Online]. Available: https://aclanthology.org/2021.acl-long.80

[17] A. C. Belkina, C. O. Ciccolella, R. Anno, R. Halpert, J. Spidlen, and J. E. Snyder-Cappione, "Automated optimized parameters for t-distributed stochastic neighbor embedding improve visualization and analysis of large datasets," *Nature communications*, vol. 10, no. 1, pp. 1–12, 2019.

[18] T. Grósz and M. Kurimo, "Visual Interpretation of DNN-based Acoustic Models using Deep Autoencoders," in *Machine Learning Methods in Visualisation for Big Data*. The Eurographics Association, 2020.

[19] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 139–156.

[20] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, p. 2837–2854, dec 2010.

[21] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proceedings of the 2007 (EMNLP-CoNLL)*. Prague, Czech Republic: Association for Computational Linguistics, Jun. 2007, pp. 410–420. [Online]. Available: https://aclanthology.org/D07-1043

[22] J. Shah, Y. K. Singla, C. Chen, and R. R. Shah, "What all do audio transformer models hear? probing acoustic representations for language delivery and its structure," *ArXiv*, vol. abs/2101.00387, 2021.

[23] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli, "Unsupervised speech recognition," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 826–27 839, 2021.

[24] A. H. Liu, W.-N. Hsu, M. Auli, and A. Baevski, "Towards end-to-end unsupervised speech recognition," *arXiv preprint arXiv:2204.02492*, 2022.